Springer Tracts in Advanced Robotics 114

Masayuki Inaba
Peter Corke *Editors*

Robotics Research

The 16th International Symposium ISRR

star

Springer

Springer Tracts in Advanced Robotics 114

Masayuki Inaba · Peter Corke
Editors

Robotics Research

The 16th International Symposium ISRR

Springer

Editors
Masayuki Inaba
Department of Creative Informatics
The University of Tokyo
Tokyo
Japan

Peter Corke
School of Electrical Engineering
and Computer Science
Queensland University of Technology
Brisbane, QLD
Australia

ISSN 1610-7438 ISSN 1610-742X (electronic)
Springer Tracts in Advanced Robotics
ISBN 978-3-319-28870-3 ISBN 978-3-319-28872-7 (eBook)
DOI 10.1007/978-3-319-28872-7

Library of Congress Control Number: 2016930274

Printed on acid-free paper

This Springer imprint is published by SpringerNature
The registered company is Springer International Publishing AG Switzerland

Series Foreword

Robotics is undergoing a major transformation in scope and dimension. From a largely dominant industrial focus, robotics is rapidly expanding into human environments and is vigorously engaged in its new challenges. Interacting with, assisting, serving, and exploring with humans, the emerging robots will increasingly touch people and their lives.

Beyond its impact on physical robots, the body of knowledge robotics has produced is revealing a much wider range of applications reaching across diverse research areas and scientific disciplines, such as biomechanics, haptics, neurosciences, virtual simulation, animation, surgery, and sensor networks among others. In return, the challenges of the new emerging areas are proving an abundant source of stimulation and insights for the field of robotics. It is indeed at the intersection of disciplines that the most striking advances happen.

The *Springer Tracts in Advanced Robotics* (STAR) is devoted to bringing to the research community the latest advances in the robotics field on the basis of their significance and quality. Through a wide and timely dissemination of critical research developments in robotics, our objective with this series is to promote more exchanges and collaborations among the researchers in the community and contribute to further advancements in this rapidly growing field.

As one of robotics pioneering symposia, the International Symposium on Robotics Research (ISRR) has established over the past two decades some of the field's most fundamental and lasting contributions. Since the launching of STAR, ISRR and several other thematic symposia in robotics find an important platform for closer links and extended reach within the robotics community.

This 16th edition of "Robotics Research," edited by Masayuki Inaba and Peter Corke, brings a collection of a broad range of topics in robotics including control, design, intelligence and learning, manipulation, perception, and planning. The content of these contributions provides a wide coverage of the current state of robotics research: the advances and challenges in its theoretical foundation and technology basis, and the developments in its traditional and novel areas of applications.

The novelty and span of the work presented in this volume reveal the field's increased maturity and expanded scope. This 16th edition of ISRR culminates with this important reference on the current developments and new directions in the field of robotics—a true tribute to its contributors and organizers!

Stanford, California
November 2015

Oussama Khatib
STAR Editor

Preface

This volume contains the papers presented at the 16th International Symposium of Robotics Research (ISRR). This is a biennial meeting sponsored and organized by the International Foundation of Robotics Research (IFRR). It is the longest running series of robotics research meetings and dates back to the very earliest days of robotics as a research discipline—the first meeting was organized by Mike Brady and Richard Paul and took place in Bretton Woods (New Hampshire, USA) in August 1983. This most recent meeting was held in the 30th anniversary year of the very first meeting, and represents 30 years at the forefront of ideas in robotics research.

This particular meeting, the 16th in the series, took place in Singapore during 16–19 December 2013. There were similarities and differences compared to the very first ISRR. Importantly, we adopt a single track format, limit the number of participants, and design the program to maximize meaningful technical interactions. The first meeting was by invitation only but more recent meetings have had an open call for papers as well as invited speakers.

The invited papers were nominated by individual IFRR officers, who then collectively voted on those nominations. This meeting continued the reform of the speaker invitation process. This year only 16 of the 45 speakers were invited and for the first time selection was based on the votes of all IFRR officers rather than selection by geographic region as was the previous convention.

The open submission process resulted in 58 papers from 11 countries. Of these 29 were selected by the reviewers for presentation at the meeting, a 50% acceptance rate. These papers were presented in interactive format which enables real conversations between speakers and the audience.

In addition the program had, for the first time, a number of forums on topics such as emerging areas in robotics, learning and control human–robot interaction.

The program included a tribute to remember two very long serving officers of the Foundation, Dr. Georges Giralt and Prof. Ray Jarvis, who passed away during 2013.

In this volume we have collected the papers presented in regular sessions, interactive sessions, and topical forums and organized them into traditional ISRR categories: control; design; intelligence and learning; manipulation; perception; and planning.

Finally, the meeting would not have been possible without the brilliant organizing team led by Marcelo Ang, and the support of the sponsors. We acknowledge and thank them all.

Masayuki Inaba
Peter Corke

Organizing Committee

Sponsors

A*STAR Agency for Science, Technology and Research (Singapore)
Toyota
KUKA

The ISRR 2013 Committee

Co-chairs
Masayuki Inaba
Peter Corke

Local Arrangements
Marcelo Ang
Webmasters: Andras Kupcsik, Andrew English

The International Foundation of Robotics Research

President
Oussama Khatib

Executive Officers
Henrik Christensen
Gerd Hirzinger
Yoshiko Nakamura

Honorary Officers
Ruzena Bajcsy
Robert Bolles

Contents

Part I
Control

Is Active Impedance the Key to a Breakthrough for Legged Robots?

Claudio Semini, Victor Barasuol, Thiago Boaventura, Marco Frigerio and Jonas Buchli

Abstract This work addresses the question whether active impedance control is key to a breakthrough for legged robots. In this paper, we will talk about controlling the mechanical impedance of joints and legs with a focus on stiffness and damping control. In contrast to passive elements like springs, active impedance is achieved by torque-controlled joints allowing real-time adjustment of stiffness and damping. We argue that legged robots require a high degree of versatility and flexibility to execute a wide range of assistive tasks to be truly useful to humans and thus to lead to a breakthrough. Adjustable stiffness and damping in realtime is a fundamental building block towards versatility. Experiments with our 80 kg hydraulic quadruped robot HyQ demonstrate that active impedance alone (thus no springs in the structure) can successfully emulate passively compliant elements during highly-dynamic locomotion tasks (running and hopping); and, that no springs are needed to protect the actuation system. Here we present results of a flying trot, also referred to as running trot. To the authors' best knowledge this is the first time a flying trot was successfully implemented on a robot without passive elements such as springs. A critical discussion on the pros and cons of active impedance concludes the paper. An extended version of this paper has been published in IJRR in 2015 [43].

C. Semini (✉) · M. Frigerio
Department of Advanced Robotics, Istituto Italiano di Tecnologia (IIT), via Morego, 30, 16163 Genova, Italy
e-mail: claudio.semini@iit.it; claudio@semini.ch

M. Frigerio
e-mail: marco.frigerio@iit.it

V. Barasuol
PPGEAS - Department of Automation and Systems, Federal University of Santa Catarina (UFSC), Florianpolis, SC, Brazil
e-mail: victor.barasuol@posgrad.ufsc.br

T. Boaventura · J. Buchli
ETH Zurich, Agile & Dexterous Robotics Lab, John-von-Neumann-Weg 9, 8093 Zürich, Switzerland
e-mail: tboaventura@ethz.ch

J. Buchli
e-mail: buchlij@ethz.ch

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_1

1 Introduction

Robots with arms and legs have the potential to become true assistants to humans in everyday life and might replace them for dangerous, dull and dirty tasks. While the legs will allow these robots to move with agility in any kind of terrain accessible to humans and animals, their arms will allow them to execute tasks with human dexterity. However, today's most advanced robots are still very far from this goal. In fact, the majority of today's legged robots struggle to move in even slightly rough terrain. This inability presents a stark contrast to human capabilities. The discrepancy in performance has several reasons. Historically, robot arms—and later legs—were controlled with stiff position-controlled joints. Interactions with the environment had to be carefully planned in the kinematic domain since neither information about the contact dynamics and forces could easily be taken into account, nor force and torque control was available. While this may be sufficient for most tasks of today's industrial robots, an autonomous machine will never be able to obtain neither a perfect map of the environment nor a perfect robot state estimation. Thus, precise kinematic planning of footholds is not a feasible solution for tomorrow's robots that have to move and interact in challenging and dynamically changing environments. Handling collisions and non-smooth interactions has to be part of their list of specifications.

The physical laws governing interaction dynamics show that it is paramount to control also the joint torques and/or the contact forces during interactions with the environment [18], e.g. during locomotion on irregular terrain. Studies support the assumption that humans and animals are able to control joint torques thanks to antagonistically acting muscle pairs. The elasticity of the tendons in combination with muscle control allow to adjust both the passive and active joint impedance, respectively [17, 36, 40]. Active impedance for the hand or the foot is obtained by means of muscle control by co-contracting the antagonistic muscle pair [9, 12, 31]. This control naturally has a delay of few tens of milliseconds or more [14, 23]. During collisions, the passive compliance[1] and damping in the tendons helps to protect the actuation system during this delay. The smaller the delay, the less passive compliance/damping is needed to prevent damage.

In the last decades, researchers have proposed several possible ways on how to more properly cope with the interaction forces with the environment. Some approaches use the passive dynamics of mechanical and pneumatic springs in the leg structure to govern the interaction dynamics (e.g. Buehler et al. [8], Raibert et al. [29]). The resonant frequency of the resulting spring-mass system is then used to achieve a resonant hopping and running motion. Pratt et al. [27] proposed the series elastic actuator (SEA) where (usually stiffer) springs are put in series to the actuator. The main purposes of the spring in a SEA is to control joint forces, absorb impact peaks and temporarily store energy. Springs are especially popular for electrically actuated robots, as they can protect the gears[2] from getting damaged during collisions and non-smooth interactions.

[1]Compliance is the inverse of stiffness.

[2]Reduction gears are required to amplify the low output torque of electric motors.

These springs, however, introduce passive dynamics and low-frequency resonant modes into the system and therefore have to be tuned for a certain task. While this is fine for a single-purpose machine (e.g. a robot for highly efficient running), it drastically reduces the versatility and thus usefulness of a service robot in human environments. Even the normally stiffer springs of the SEA reduce the actuator bandwidth as a result of the resonant modes, and therefore make certain tasks where a stiff and precise motion is required impossible. This topic is further elaborated in Sect. 6.

To overcome this problem researchers have been working on variable stiffness actuators (VSA) [42] that can vary the stiffness of each joint with the help of a (generally smaller) second actuator. While recent progresses in this field have increased the range of adjustable stiffnesses [41], the actuators are still bulky, complex and often cannot absorb high energy impacts due to the limited size of the springs.

Active impedance is a promising alternative that does not require any physical springs, because the required stiffness and damping is controlled by software and torque-controlled joints (e.g. impedance control [19], operational space control [24], and virtual model control [28]). Any stiffness and damping (within the limitations of the actuation and control system) can be selected in realtime either for the endeffector or for each joint independently [5]. This approach has most advantages of VSA without the above mentioned limitations. Boaventura et al. [5] present an experimental comparison study of active versus passive compliance and show that active impedance systems can emulate passive elements in the dynamic range needed for locomotion and interaction with the environment in general. The performance of the emulation is such that there is no relevant difference between the dynamic behavior of the actively controlled system and its fully passive 'template' system.

In this work we will demonstrate that active impedance can enable a legged robot to potentially execute a wide range of different tasks in natural environments and thus increase its versatility and usefulness. We will present our previous work on our torque-controlled hydraulic quadruped robot HyQ [32, 33] in this context. And, we will demonstrate the advantages and the potential of active impedance and torque-controlled robots with two new experiments: a flying trot and resonant hopping. The flying trot demonstrates the robustness and performance of the impedance controller in a very demanding situation due to the high frequency impacts at the touch down moments. The resonant hopping demonstrates the flexibility and versatility of the control concept.

The major contribution of this work is the presentation of a flying trot with an 80 kg quadruped robot with purely impedance-controlled legs, thus without any springs in its mechanical structure. To the best knowledge of the authors no machine has achieved this before. In this paper we use the success of this experiment as an example to discuss the importance of active impedance in legged robots for real-world tasks.

This paper first discusses the state of the art in the field of purely impedance-controlled legged robots and machines that successfully demonstrated a flying trot. Section 3 then introduces the active impedance controller of our quadruped robot HyQ. The control required to implement a flying trot is explained in Sect. 4; and Sect. 5 presents the experimental results of a flying trot and resonant hopping motion with variable joint stiffness. Finally, Sect. 6 discusses if active impedance can help legged robots to break through and Sect. 7 concludes the paper with final remarks.

2 State of the Art

We will discuss the state of the art of active impedance on legged robots and work related to experimental implementations of flying trots on quadruped robots.

2.1 Legged Robots with Active Impedance

In this section we will focus on legged robots with active but no passive impedance, i.e. without any physical spring in their structure. For a more general and extensive review of impedance control in robotics, including fields like haptics and manipulation, please refer to Boaventura et al. [4, 5]. There exist only a few examples of purely impedance-controlled legs with internal torque control loop in the literature. Ott et al. [26] presented a bipedal walking robot with actuators based on the modular drives of the DLR-Lightweight-Robot-II [16]. These actuator units are based on torque-controlled electric motors with integrated joint torque sensors. The robot successfully demonstrated walking on flat ground and stairs, as well as balancing and posture control. No highly-dynamic gaits like running have been demonstrated so far. Another electrically actuated robot with purely impedance- controlled legs is the MIT cheetah robot. Seok et al. [34, 35] presented a quadruped robot with joint torque control, implemented with electric motors with low gear ratio (5.8:1) and current control. No springs or torque sensing elements are needed in this approach (except an elastic spine for energy storage). The robot—supported by a boom—successfully demonstrated a running gait on a treadmill and showed reliable impedance control on joint level. A similar approach was taken by Buchli et al. [7] with LittleDog that had joint level torque control based on electric motor current control. The authors showed how a feedforward torque term obtained by inverse dynamics can reduce the position gains and allow for a successful disturbance rejection of unperceived obstacles. The high gear ratio, low control bandwidth and non-robust gears, however, made it very difficult to implement well controlled dynamic gaits.

There are also a few examples of hydraulically actuated robots with only active impedance. The Sarcos humanoid robots at ATR [21], CMU [39] and more recently at USC [15] have torque controlled joints based on torque sensors. The three research groups have shown balancing and simple stepping experiments on their robots, but none of them has shown any more dynamic gaits. HyQ is a hydraulically-actuated quadruped robot developed at the Istituto Italiano di Tecnologia [32, 33] with joint torque control based on torque sensors [4, 11]. Our robot has successfully demonstrated various dynamic gaits ranging from fast walking (2 m/s), jumping, rearing to balancing over rough and instable terrain [2]. Recent experimental studies [5] on a single leg of HyQ compared active versus passive impedance and showed that high-performance impedance controllers can satisfactorily emulate passive elements such as spring-dampers. In Sect. 5 of this paper we will show how active impedance can enable highly-dynamic and versatile locomotion.

2.2 *Robots Running with a Flying Trot*

Next, we will discuss robots that have successfully demonstrated a flying (or running) trot. Note that we include robots with active and passive impedance, SEA, etc. in this overview. Raibert's quadruped robot of the CMU and later MIT leg lab was the first quadruped robot to demonstrate a flying trot [29]. Its prismatic legs had pneumatic springs in their structure that allowed the robot to run in resonance. BigDog is a hydraulically-actuated quadruped robot [30]. In one of the online videos, this robot demonstrated a flying trot. To date, no experimental results have been published. BigDog has torque-controlled joints and springs in the last segment of its legs. We believe that a combination of active and passive impedance is used in BigDog. StarlETH is a quadruped robot developed at the ETH Zurich with relatively stiff springs in series with its actuators (SEA) making it a fully torque-controlled robot [20]. This platform has recently shown trotting with short flight phases [13]. The Cheetah-cub is a 1.1 kg electric quadruped robot that recently demonstrated a flying trot [37]. Its legs are designed around a spring loaded pantograph mechanism.

Note that all of the above-mentioned robots have passively compliant elements (mostly springs) in their legs.

A few other robots have shown running gaits while some of their degrees of freedom are restricted by a boom or other guiding mechanism (e.g. the biped MABLE [38] (using passive compliance with active force control), KOLT [10] (springs in legs), MIT Cheetah (see Sect. 2.1), Boston Dynamics' Cheetah (no information available)), thus not fully and convincingly demonstrating the versatility required for a useful service robot.

3 Active Impedance

With *active impedance* we mean that the mechanical impedance is (actively) controlled and adjustable in software. Note that in our case we control both stiffness and damping, but did not implement inertia-shaping.

To implement active impedance on HyQ, we use a cascaded control architecture as depicted in Fig. 1. In this control scheme, an outer impedance control loop feeds back the joint angular positions and produces a torque command as output. Then, this torque command becomes the input reference for an inner torque control loop. The high performance of the inner torque controller, obtained through low-level model-

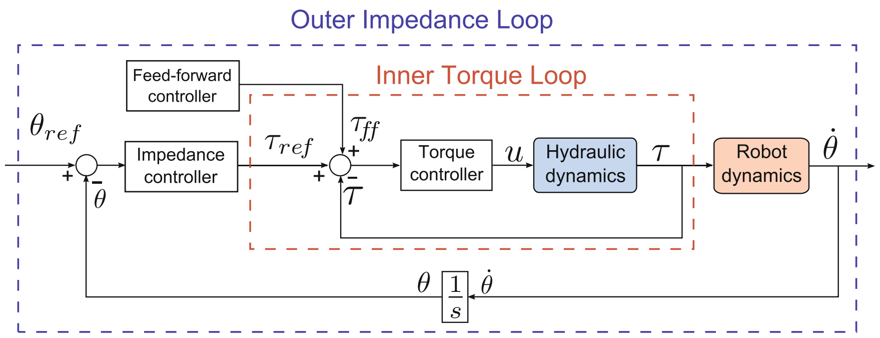


Fig. 1 Block diagram of the HyQ cascade impedance control architecture. It includes an outer impedance loop and an inner torque loop. The outer loop consists of a feedback controller and can include also a feed-forward controller such as rigid body inverse dynamics controller. The inner torque loop uses a feedback linearization approach for an increased tracking performance

based techniques [4, 11], was essential to successfully achieve adjustable impedance through software, without the presence of real springs.

The outer impedance loop defines the impedance characteristics of the robot, either set in joint or task space. The joint stiffness and damping can be implemented through a simple proportional derivative (PD) joint position controller. In this case, due to the presence of the inner torque loop, the proportional gain of the position control acquires units of Nm/rad, which corresponds to a rotational spring, and the derivative gain acquires the unit Nms/rad, which corresponds to a rotational damper. Therefore, by setting the proportional and derivative position gains it is possible to define the stiffness and damping of the robot joints. This joint-space impedance scheme is used for the flying trot experiments described in Sect. 5.2.

On the other hand, sometimes it might be more convenient to set the impedance at the end-effector instead of at the joints. A very intuitive way of defining a task-space stiffness and damping is through the implementation of virtual components [28]. As for the PD position controller mentioned above, these virtual components are also implemented in the impedance loop shown in Fig. 1. In HyQ, we designed a virtual spring-damper between the hip and the foot, as depicted in Fig. 6 on the left. The desired force f created by these virtual components can be linear or nonlinear with respect to the stiffness, damping, and virtual prismatic leg length [4]. Once the end-effector force f is calculated, it is then mapped into joint-space through the Jacobian transpose of the kinematic transform of the virtual model coordinate system to the joint coordinate system. The use of the virtual prismatic leg is also a simple way of actively implementing the well-known spring loaded inverted pendulum (SLIP) model [3], which is a useful abstraction that describes the spring-like behaviour found in human and animal running and walking. This task-space impedance controller is employed in this paper in Sect. 5.3, where the stiffness of the linear spring is changed on the fly to create a resonant hopping with HyQ.

In addition, the inner torque controller permits a straightforward implementation of high-level model-based control techniques, such as rigid body inverse dynamics, and gravity compensation. The output torques from the above mentioned techniques can be easily added as a feed-forward torque to the torque reference command from the outer loop, as shown in Fig. 1. Some of these model-based techniques provide very convenient capabilities for performing robust locomotion in unstructured and partially unknown environments [7]. Essentially, such control methods permit lowering the position gains (hence the system's stiffness) without giving up on position tracking performance.

4 Flying Trot Motion Generation and Control

A trot is a gait in which diagonal leg pairs move simultaneously, alternating with the other pair of legs. A flying trot (or running trot) is a special case characterized by a ballistic body motion, i.e., by a period in which there are no legs in contact with the ground. The body flight phase depends on the ratio between the time that a leg stays in contact with the ground (the *stance phase*) and the time that a leg takes to swing to the next foothold (the *swing phase*). This ratio is called *Duty Factor*, hereafter defined as D_f, and varies between 0 and 1. During trotting, if all the legs have a duty factor of less than 0.5 (i.e. swing phase longer than stance phase) then the body undergoes a flight phase for a certain time fraction of the gait cycle.

A comprehensive locomotion control framework is required to make a robot perform a stable flying trot. This control framework needs to integrate appropriate trajectory generation and body motion control in a closed loop fashion. Our recently presented Reactive Control Framework (RCF) [2] implements these aspects and we adapted it to achieve a flying trot with HyQ. The RCF integrates the basic components for robot motion generation and robot motion control. No information about the environment, such as terrain surface level or obstacles, is required to achieve a basic robust (reactive) locomotion behavior.

Next, we will highlight some of the important features of the RCF in relation to the generation of a flying trot: the generated profile for the feet trajectories; the trajectory generator parameters; and how we choose such parameters to achieve a flying trot.

The generation of the reference trajectories for the feet is loosely inspired by the Central Pattern Generators (CPGs) of animals [22], with the advantage of having intuitive parameters such as step length and step height. Ellipse–shaped trajectories (called primitives) are generated by a network of four non-linear oscillators, whose state represents the Cartesian coordinates of each foot [1], as depicted in Fig. 2 on the left. The oscillator parameters that define the aspect ratio of the ellipse are directly related to the step length L_s and the step height H_s. Each oscillator has an angular frequency w_s, associated to the corresponding leg step frequency f_s; w_s might be different for the stance and swing phases, to achieve a duty factor different from 0.5. Non-linear filters are coupled to the output of the network of oscillators to reshape the

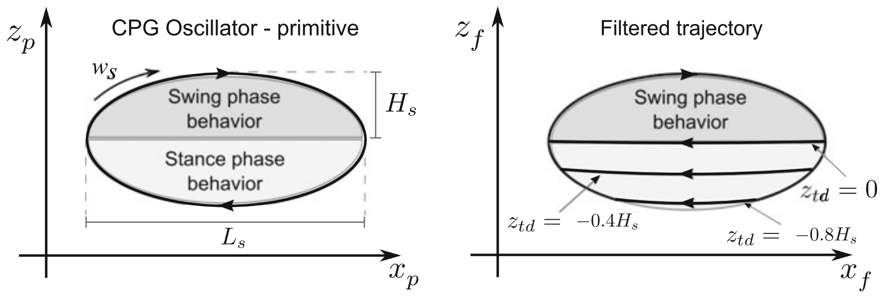


Fig. 2 The foot trajectory generated by the CPG oscillator (on the *left*) and the trajectory modulated by the non-linear filter (on the *right*). z_p and x_p are the reference coordinates of the primitive's trajectory, while z_f and x_f are the filtered references sent to the joint controller. z_{td} is the filter parameter which determines where the original elliptic trajectory has to be interrupted. (Figure modified from Barasuol et al. [2])

elliptical trajectories to semi-elliptical ones, to make the robot capable of adapting to the actual terrain profile. The non-linear filters reshape the primitive's trajectories according to an estimation of the foot position at touchdown; this information is either predefined, when the surface is well known, or computed from sensory information (for example using force sensors). The shape of the adapted trajectories are illustrated in Fig. 2 on the right.

The *step depth* parameter z_{td} affects the reshaping of the trajectory by determining at which height the ellipse has to be interrupted, as depicted in Fig. 2 on the right. The desired robot forward velocity V_f determines the relative velocity of the foot with respect to the robot trunk, which is imposed during the rectified part of the semi-ellipse (i.e. during the stance phase). If a terrain map is available the swing-to-stance transition can be planned in advance, reducing the impact forces. On the other hand, the feet trajectories can be dynamically adjusted even if the robot is walking *blindly*, e.g. by using feedback from the foot or joint force sensing, see [2]. This feature makes locomotion more robust also with respect to poor state estimation.

In this paper we show experiments performed on flat ground. We consider the flat ground as a well-known surface and, therefore, we assume $z_{td} = 0$ for all the legs. With $z_{td} = 0$ the shape of the primitives becomes a half-ellipse.

During a flying trot the most important parameters are the step length L_s, the duty factor D_f, the desired forward velocity V_f and the step frequency f_s. In the RCF approach all these parameters can be independently modulated. To achieve a stable *spring-mass* bouncing motion of the robot's centre of mass (COM), the robot's motion during the stance period needs to match the system's resonant frequency (defined by the robot's mass and leg stiffness). Selecting a proper duty factor and step frequency allows us to obtain a stance phase that matches the natural resonance period. Since D_f and f_s are then defined, choosing a desired forward velocity V_f consequently determines the value of the step length L_s.

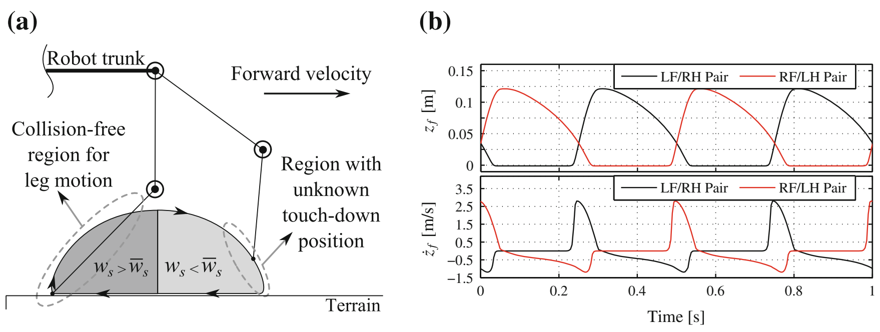


Fig. 3 Modulation of the angular frequency w_s. **a** In the collision-free region the angular frequency w_s is greater than the average angular frequency $\bar{w}_s$ of the swing phase. In the unknown touch-down region w_s is smaller than $\bar{w}_s$. **b** The plot shows the foot's relative position z_f (step height) and the corresponding velocity $\dot{z}_f$ references for each pair of diagonal legs (Left-Front/Right-Hind and Right-Front/Left-Hind legs). The swing period in the collision-free region is chosen to be half of the swing period in the unknown touch-down region. The duty factor is 0.45, the desired forward velocity is 1 m/s, the step frequency is 2 Hz and the step height is 0.12 m

In our flying trot we explore the independent parameter modulation capability of the RCF approach to generate a variable swing velocity of the leg. The idea is to move the leg faster in regions where there is a low risk of impact with obstacles, while slowing it down in proximity of the expected touchdown regions, to reduce the impact forces. We obtain this leg behavior by modulating the angular frequency of the primitives according to the collision-free region and the unknown touch-down region, without affecting the total swing period. See Fig. 3a.

Figure 3b shows an example of Cartesian references for a flying trot run at 1 m/s when the swing period in the collision-free region is chosen to be half of the swing period in the unknown touch-down region.

5 Experimental Results

We performed a series of experiments with our quadruped robot HyQ that uses only active (and no passive) impedance. After a description of the platform, we will present the results of a successful flying trot experiment and a resonant hopping. Both examples illustrate the advantages and potentials of active impedance for legged robots.

5.1 *Experimental Platform HyQ*

The platform used for these experiments is HyQ, a quadruped robot with hydraulically and electrically actuated joints [32, 33]. The machine weighs 80 kg, is roughly

1 m long and has a leg length of 0.78 m with fully-extended legs. All of its 12 degrees of freedom (DOF) are torque-controlled joints: The hip abduction/adduction joints are driven by DC brushless motors with strain-gauge based torque sensors for torque control [11]. All 8 joints in the sagittal plane (hip and knee flexion/extension) are actuated by hydraulic cylinders connected to load cells for force measurement. High-performance servovalves enable joint-level torque control with excellent tracking [4] that led to the implementation of active impedance as described in Sect. 3. Note, that besides a thin rubber layer at the feet, there are no passive stiffness/damping elements (e.g. springs) present anywhere in the robot's leg structure.

Since 2011 HyQ has demonstrated a wide range of static and dynamic motions such as a crawl gait, stair climbing, walking trot over flat, inclined and rough terrain (indoors and outdoors), squat jumps, rearing, balancing under disturbances and step reflexes.

5.2 Flying Trot Experiment

We conducted several experiments of a flying trot with HyQ based on the approach presented in Sect. 4. Figure 4 shows a picture sequence of one of these experiments to illustrate the flight phases (right hand side frames) between the stance phases of the two diagonal leg pairs. A link to a video of this experiment can be found at [25].

Figure 5 shows the knee joint torque plots of the four legs and the vertical ground reaction forces. The duty factor during this experiment was set to 0.45, the step height 0.12 m, forward velocity 1.3 m/s, step length 0.28 m and the joint-level active stiffness

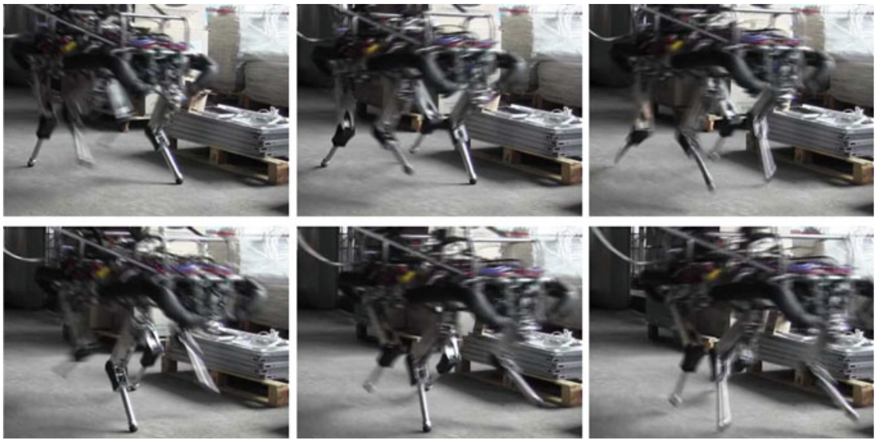


Fig. 4 Picture sequence of the flying trot experiment with the HyQ robot, which shows the flight phase achieved by setting the duty factor at 0.45. The time between each frame is 80 ms, and the whole sequence represents 400 ms of the actual experiment

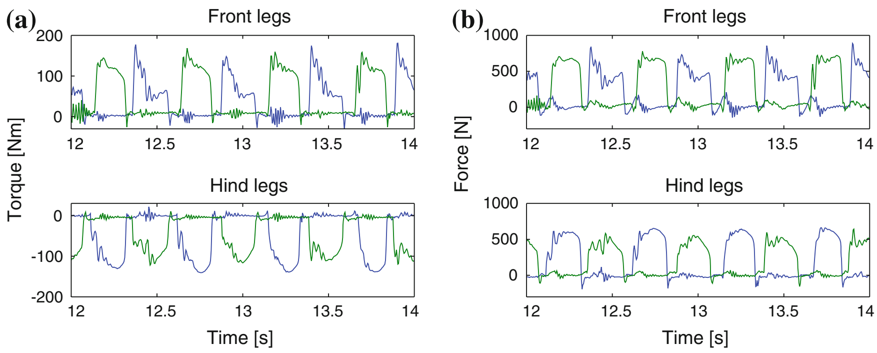


Fig. 5 Force profiles during the flying trot experiment; the *blue lines* refer to the left legs, the *green* ones to the right legs. **a** This plot shows the torques at the four knee joints of the robot; the short intervals during which all the torques are close to zero are due to the flight phase. **b** This plot illustrates the ground reaction forces during the same time interval, estimated from the torques at the knees and hips with the transpose of the Jacobian

300 Nm/rad for the hip and knee flexion/extension joints. Note that the joint torques of all four legs stay inside the maximum torque limits[3] of 181 Nm demonstrating that active compliance can successfully absorb the high impacts during the running and cope with these collisions. The plots also show that the joint torques and ground reaction forces go to zero between the stance phases of the diagonal leg pairs. This illustrates that the robot was indeed in flight phases. To the best of our knowledge no other robot has successfully shown a robust flying trot with active impedance only, i.e. without passive elements such as springs in its legs.

5.3 Resonant Hopping Experiment

In this section we show HyQ's ability of changing the virtual spring stiffness on the fly to achieve a resonant hopping motion. For doing so, we implemented a virtual linear spring-damper for all four legs of HyQ as shown in Fig. 6 on the left. The length of the virtual linear springs ($l = 0.58$ m) is varied sinusoidally ($\delta l = 0.05$ m) at a constant frequency of 1.6 Hz. During the experiment, the stiffness of the virtual springs is linearly changed from $K = 2000$ to $K = 5000$ N/m.

As shown in Fig. 6, after 1 s the spring stiffness starts to increase and, consequently, the amplitude of the ground reaction force oscillations grows due to resonant effects. We show the ground reaction force for the left front (LF) leg in the first plot. When the stiffness and thus spring-mass system resonates with the frequency of the sinusoidal spring length excitation, the robot starts to hop and the ground reaction forces go to

[3]Note that we recently increased the hydraulic system pressure of the HyQ robot to 20 MPa, increasing the maximum torque of the hip and knee flexion/extension joints to 181 Nm.

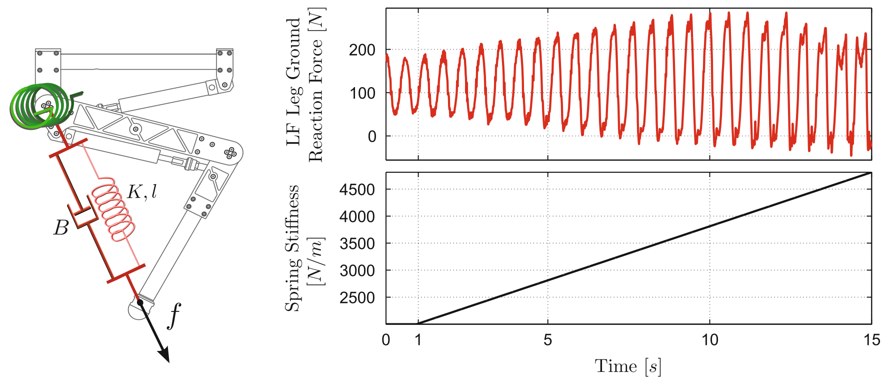


Fig. 6 Experimental results of resonant hopping: *left:* HyQ virtual elements: a spring-damper connects the hip to the foot (*red elements*), creating a *prismatic virtual leg*. In the hip joint, a simple joint-space position PD control can be seen as a rotational spring-damper (*green element*). *Right:* We implemented a hopping motion by exciting the HyQ robot in a resonant way by varying the virtual legs stiffness. The *top* plot shows the ground reaction force for the left front (LF) leg, which reaches zero after around 9 s demonstrating a presence of a flight phase. The *bottom* plot presents the linear change in stiffness applied to the legs

zero during flight phase (all four legs in the air). The resonance peak occurs at about 10 s, when the stiffness is around 3800 N/m.

This example shows how active impedance allows to adjust the dynamics of the system, thus creating a big potential for new control methods for legged robots.

6 Discussion

An important contribution of this work is to discuss the initial question *Is Active Impedance the Key to a Breakthrough for Legged Robots*? To this end, this section will first provide possible reasons why legged robots are still far from a breakthrough. We will then discuss why springs are currently not ideal to use, and mention the pro and cons of active impedance. Finally, we will propose important future topics of research that will help legged robots become a reality in every-day life.

As mentioned in the introduction, despite decades of research on legged locomotion, today's robots are still far from being able to move in human environments. Two of the main requirements for such robots are (1) the ability to cope with collisions and non-smooth interactions, since they cannot be avoided in such environments; and (2) the versatility of such machines to execute a wide range of tasks to become truly useful assistants. Very few examples of robot designs and their associated control framework meet these two requirements.

Springs are often used to meet the first requirement. However, springs are not an ideal solution to meet the second requirement for the following reasons. A truly versatile robot should be able to execute tasks ranging from a precise and careful manipulation of a delicate object, to locomotion in environments with unperceived obstacles where a soft interaction but also fast reflex motions are required. While some tasks require very precisely controlled joints, others need compliant behavior, yet others require very fast joint motions as reaction to an external perturbation, e.g. when being pushed or for safety stops. Precise motions at any speed require either a very good model[4] of the robot and possibly the environment or high gain (i.e. 'stiff') control.[5] In addition, if fast motions are required, as a reaction to an unforeseen event (side step to keep balance, step reflex or stopping a robot arm in front of a person) a very high actuation bandwidth is required. Compliant behavior as reaction to an unforeseen perturbation requires low output impedance and is in contradiction to a quick controlled movement without using a model. It fundamentally limits the ability of a quick stop or a sudden reactive movement. Therefore, a compliant robot (or human) needs to have the required bandwidth and high gain control available to be robust in such situations (e.g. a safety stop of a human arm requires immediate stiffening up). In case of a SEA the spring stiffness fundamentally limits the control bandwidth and a trade-off has been fixed at design time. As mentioned in the introduction, VSA might be a possible solution to this problem, however the technology has (still) several limitations. For a VSA, the ability of a quick stop is fundamentally limited by the (usually slow) adaptation of the stiffness. In case of an active impedance system, the only limitations stem from sensing and actuation delays (actuator physics, data acquisition, data processing), which are, to a large extent, design parameters.

To sum up, a versatile robot needs to be able to control its joint stiffness in a wide range. Springs in the structure of a robot including the stiff springs of SEA reduce the maximum joint stiffness and control bandwidth; and thus the robot's versatility. We argue that legged robots with active impedance, while certainly not the only solution, are a promising solution that meet both of the above mentioned requirements. Importantly, they are implementable with *today's* available technology thus putting versatile service robots within immediate reach.

Active impedance has several advantages when compared to passive springs and dampers. With today's advances in actuator, control and computer technology a wide range of stable stiffness and damping values can be emulated [5], which leads to more versatile robots. These values can be adjusted in real-time to swiftly adapt to changing conditions in the environment or task. Furthermore, robots with active impedance can take advantage of any programmable type of impedance (e.g. exponential springs, nonlinear dampers, muscle-model-based springs, etc.) [4]. A potential drawback of active compliance is low energy efficiency, as no energy can be stored due to a lack of physically compliant elements. Despite this disadvantage we do not consider it as a

[4]Note that the fact that models are required for good performance does not address the question where the model comes from. For robots it can sometimes be derived from CAD data, sometimes must be estimated/learned. For humans models are typically acquired by learning.

[5]It is worthwhile discussing these issues in the control theoretic notions of nominal behavior and disturbance reaction.

major problem for the following reasons: On one hand, new methods of high-density energy storage are currently investigated in various research fields. New compact energy sources will eventually be able to power legged robots for entire days [6]. On the other hand, other ways of energy recovery such as energy regenerative electronics for electric motors have recently been proposed for joints with active impedance [34]. Furthermore, passively compliant elements are only really able to increase energy efficiency of a robot during repetitive motions, such as walking, running, scrubbing etc. if the motion frequency is around the resonant frequency of the system. For an in-depth discussion of the *pro & cons* of active versus passive compliance we refer the interested reader to [5].

For humans we see this trade-off in the example of Oscar Pistorious, a below-knee amputee who has won several medals in sprint running. He uses two carbon-fibre curve-shaped springs as foot prosthesis that allow him to run fast and efficiently. After the races and trainings however he wears normal, stiff prosthesis. It is therefore a good example of the limited versatility imposed by springs.

Now that we understand the limitations introduced by springs we can rethink and adjust future research agendas to focus on the important topics that will lead to a faster breakthrough of legged robots into everyday life. First of all, torque-controlled robots open up a wide range of control methods besides active impedance, e.g. model-based control of rigid body dynamics (gravity compensation, inverse dynamics, etc.) and control of contact forces. These are all methods that will lead to improved manipulation and locomotion skills in human environments. Additionally, research is required into optimal selection of stiffness trajectories for a large range of tasks. Investigations into how to build more compact and less complex VSA with fast stiffness adjustment are important because they might eventually be useful to save energy during repetitive motions. Questions regarding the safety and reliability of active impedance systems were not discussed in this work due to lack of space, but they are important topics that need to be investigated. Last but not least more research into energy efficient active impedance systems is required.

7 Conclusions

We have shown, to the best of our knowledge, for the first time how a legged robot with active impedance only (i.e. without springs) can execute highly dynamic tasks that involve large and impulsive impact forces, such as running and hopping. Our experiments presented here and elsewhere [5] show that it is possible to achieve the same behavior with a fully actively controlled system as with passive systems. Active impedance offers the additional advantage of versatility and flexibility, allowing to specify the most suitable dynamic behavior on the fly. The data shown indicates that the argument of active systems being too slow to control does not hold for the dynamic range that is used for highly dynamic locomotion and interaction tasks on time-, force- and length-scales typical for humans. We consider this approach fundamental to the breakthrough of versatile robotic assistants with arms and legs and we have demonstrated that the required control performance is achievable.

Acknowledgments This research has been funded by the Fondazione Istituto Italiano di Tecnologia. The authors would like to thank CAPES for the scholarship granted to V. Barasuol (Grant Procs. 6463-11-8). T. Boaventura is partially funded through the EU Project BALANCE (Grant 601003 of the EU FP7 program). J. Buchli is supported by a Swiss National Science Foundation professorship. The authors would like to thank also the other members of the Dynamic Legged Systems Lab that contributed to the success of this project: M. Focchi, I. Havoutis, S. Bazeille, J. Goldsmith, H. Khan, B.Rehman, and our team of technicians.

References

1. Barasuol, V., De Negri, V.J., De Pieri, E.R.: WCPG: a central pattern generator for legged robots based on workspaceintentions. In: Proceedings of the ASME Dynamic System and Control Conference (DSCC), pages 111–114 (2011)
2. Barasuol, V., Buchli, J., Semini, C., Frigerio, M., De Pieri, E.R., Caldwell, D.G.: A reactive controller framework for quadrupedal locomotion on challenging terrain. In: IEEE International Conference on Robotics and Automation (ICRA) (2013)
3. Blickhan, R.: The spring-mass model for running and hopping. Biomechanics **22**, 1217–1227 (1989)
4. Boaventura, T., Semini, C., Buchli, J., Frigerio, M., Focchi, M., Caldwell. D.G.: Dynamic torque control of a hydraulic quadruped robot. In: IEEE International Conference in Robotics and Automation (ICRA), pp. 1889–1894 (2012)
5. Boaventura, T., Medrano-Cerda, G.A., Semini, C., Buchli, J., Caldwell, D.G.: Stability and performance of the compliance controller of the quadruped robot HYD. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)
6. Bruce, P.G., Freunberger, S.A., Hardwick, L.J., Tarascon, J.-M.: Li-O2 and Li-S batteries with high energy storage. Nat. Mater. **11**, 19–29 (2012)
7. Buchli, J., Kalakrishnan, M., Mistry, M., Pastor, P., Schaal, S.: Compliant quadruped locomotion over rough terrain. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 814–820 (2009)
8. Buehler, M., Battaglia, R., Cocosco, R., Hawker, G., Sarkis, J., Yamazaki, K.: SCOUT: a simple quadruped that walks, climbs, and runs. Int. Conf. Robot. Autom. (ICRA) **2**, 1707–1712 (1998)
9. Burdet, E., Osu, R., Franklin, D., Milner, T., Kawato, M.: The central nervous system stabilizes unstable dynamics by learning optimal impedance. Nature **414**(6862), 446–449 (2001)
10. Estremera, J., Waldron, K.J.: Thrust control, stabilization and energetics of a quadruped running robot. Int. J. Robot. Res. **27**, 1135–1151 (2008)
11. Focchi, M., Boaventura, T., Semini, C., Frigerio, M., Buchli, J., Caldwell, D.G.: Torque-control based compliant actuation of a quadruped robot. In: Proceedings of the 12th IEEE International Workshop on Advanced Motion Control (AMC) (2012)
12. Franklin, D., Burdet, E., Osu, R., Kawato, M., Milner, T.: Functional significance of stiffness in adaptation of multijoint arm movements to stable and unstable dynamics. Exp. Brain Res. **151**, 145–157 (2003)
13. Gehring, C., Coros, S., Hutter, M., Blösch, M., Höpflinger, M., Siegwart, R.: Control of dynamic gaits for a quadrupedal robot. In: IEEE International Conference on Robotics and Automation (ICRA) (2013)
14. Geyer, H., Herr, H.: A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities. IEEE Trans. Neural Syst. Rehabil. Eng. **18**(3), 263–273 (2010)

15. Herzog, A., Righetti, L., Grimminger, F., Pastor, P., Schaal, S.: Momentum-based balance control for torque-controlled humanoids. In: arXiv preprint arXiv:1305.2042 (2013)
16. Hirzinger, G., Albu-Schäffer, A., Hahnle, M., Schaefer, I., Sporer, N.: On a new generation of torque controlled light-weight robots. In: IEEE International Conference on Robotics and Automation (ICRA), vol. 4, pp. 3356–3363 (2001)
17. Hogan, N.: Adaptive control of mechanical impedance by coactivation of antagonist muscles. IEEE Trans. Autom. Control **29**, 681–690 (1984)
18. Hogan, N.: Impedance control: An approach to manipulation: Part I - Theory. ASME J. Dyn. Syst. Meas. Control **107**, 1–7 (1985)
19. Hogan, N.: Impedance control: An approach to manipulation: Part II - Implementation. ASME J. Dyn. Syst. Meas. Control **107**, 8–16 (1985)
20. Hutter, M., Gehring, C., Blösch, M., Höpflinger, M., Remy, C.D., Siegwart, R.: Starleth: A compliant quadrupedal robot for fast, efficient, and versatile locomotion. In: International Conference on Climbing and Walking Robots (CLAWAR) (2012)
21. Hyon, S., Hale, J., Cheng, G.: Full-body compliant human-humanoid interaction: Balancing in the presence of unknown external forces. IEEE Trans. Robot. **23**(5), 884–898 (2007)
22. Ijspeert, A.J.: Central pattern generators for locomotion control in animals and robots: a review. Neural Netw. **21**(4), 642–653 (2008)
23. Kandel, E., Schwartz, J., Jessell, T.: Principles of Neural Science, 4th edn, McGraw-Hill Medical, (2000)
24. Khatib, O.: A unified approach for motion and force control of robot manipulators: The operational space formulation. IEEE J. Robot. Autom. **3**(1), 43–53 (1987)
25. Online Video: HyQ Robot: Flying Trot with Active Compliance: http://www.iit.it/hyq or http://www.youtube.com/watch?v=27lxHMp9LIA
26. Ott, C., Eiberger, O., Englsberger, J., Roa, M.A., Albu-Schäffer, A.: Hardware and control concept for an experimental bipedal robot with joint torque sensors. J. Robot. Soc. Jpn. **30**(4), 378–382 (2012)
27. Pratt, G., Williamson, M.: Series elastic actuators. In: IEEE International Conference on Intelligent Robots and Systems (IROS) (1995)
28. Pratt, J., Chew, C., Torres, A., Dilworth, P., Pratt, G.: Virtual model control: An intuitive approach for bipedal locomotion. Int. J. Robot. Res **20**(2), 129–143 (2001)
29. Raibert, M.H.: Legged Robots That Balance. The MIT Press, Cambridge (1986)
30. Raibert, M., Blankespoor, K., Nelson, G., Playter, R., the BigDog Team,: Bigdog, the rough-terrain quadruped robot. In: Proceedings of the 17th World Congress The International Federation of Automatic Control (IFAC) (2008)
31. Selen, L., Franklin, D., Wolpert, D.: Impedance control reduces instability that arises from motor noise. J Neurosci **29**(40), 12606–16 (2009)
32. Semini, C.: HyQ—Design and Development of a Hydraulically Actuated Quadruped Robot. Ph.D thesis, Istituto Italiano di Tecnologia (IIT) (2010)
33. Semini, C., Tsagarakis, N.G., Guglielmino, E., Focchi, M., Cannella, F., Caldwell, D.G.: Design of HyQ—a hydraulically and electrically actuated quadruped robot. J. Syst. Control Eng. **225**(6), 831–849 (2011)
34. Seok, S., Wang, A., Otten, D., Kim, S.: Actuator design for high force proprioceptive control in fast legged locomotion. In: IEEE/RSJ Intelligent Robots and Systems (IROS), pp. 1970–1975 (2012)
35. Seok, S., Wang, A., Chuah, M.Y.M., Otten, D., Lang, J., Kim, S.: Design principles for highly efficient quadrupeds and implementation on the mit cheetah robot. In: IEEE International Conference on Robotics and Automation (ICRA) (2013)
36. Shadmer, R., Arbib, M.: A mathematical analysis of the force-stiffness characteristics of muscles in control of a single joint system. Biol. Cybern. **66**, 463–477 (1992)
37. Spröwitz, A., Tuleu, A., Vespignani, M., Ajallooeian, M., Badri, E., Ijspeert, A.: Towards dynamic trot gait locomotion-design, control and experiments with Cheetah-cub, a compliant quadruped robot. Int. J. Robot. Res **32**(8), 933–951 (2013)

38. Sreenath, K., Park, H.W., Grizzle, J.W.: Design and experimental implementation of a compliant hybrid zero dynamics controller with active force control for running on mabel. In: IEEE International Conference in Robotics and Automation (ICRA) (2012)
39. Stephens, B., Atkeson, C.: Modeling and control of periodic humanoid balance using the linear biped model. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids) (2010)
40. Tee, K., Franklin, D., Kawato, M., Milner, T., Burdet, E.: Concurrent adaptation of force and impedance in the redundant muscle system. Biol. Cybern. **120**(1), 31–44 (2009)
41. Tsagarakis, N., Sardellitti, I., Caldwell, D.: A new variable stiffness actuator (compact-vsa): Design and modelling. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 378–383 (2011)
42. Vanderborght, B. et al.: Variable impedance actuators: a review. Robotics and Autonomous Systems (2013)
43. Semini, C., Barasuol, V., Boaventura, T., Frigerio, M., Focchi, M., Caldwell, D.G., Buchli, J.: Towards versatile legged robots through active impedance control. Int. J. Robot. Res **34**(7), 1003–1020 (2015)

Optimal Control of Nonlinear Systems with Temporal Logic Specifications

Eric M. Wolff and Richard M. Murray

Abstract We present a mathematical programming-based method for optimal control of nonlinear systems subject to temporal logic task specifications. We specify tasks using a fragment of linear temporal logic (LTL) that allows both finite- and infinite-horizon properties to be specified, including tasks such as surveillance, periodic motion, repeated assembly, and environmental monitoring. Our method directly encodes an LTL formula as mixed-integer linear constraints on the system variables, avoiding the computationally expensive process of creating a finite abstraction. Our approach is efficient; for common tasks our formulation uses significantly fewer binary variables than related approaches and gives the tightest possible convex relaxation. We apply our method on piecewise affine systems and certain classes of differentially flat systems. In numerical experiments, we solve temporal logic motion planning tasks for high-dimensional (10+ continuous state) systems.

1 Introduction

In safety-critical robotics applications involving autonomous ground and air vehicles, it is desirable to unambiguously specify the desired system behavior and automatically synthesize a controller that provably implements this behavior. Additionally, autonomous systems often have high-dimensional, nonlinear dynamics and require high-performance (not just feasible) controllers.

Linear temporal logic (LTL) is an expressive task-specification language for specifying a variety of tasks such as responding to the environment, visiting goals, periodically monitoring areas, staying safe, and remaining stable. These properties generalize classical point-to-point motion planning. Also, the widespread use of LTL

E.M. Wolff (✉) · R.M. Murray
California Institute of Technology, Pasadena, CA, USA
e-mail: ewolff@caltech.edu

R.M. Murray
e-mail: murray@cds.caltech.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_2

in software verification [2] makes it appealing as a common language for reasoning about the software and dynamics of autonomous systems.

Standard methods for motion planning with LTL task specifications first create a finite abstraction of the original dynamical system. This abstraction can informally be viewed as a labeled graph that represents possible behaviors of the system. Approximate finite abstractions can be computed using either sampling-based methods (e.g., RRTs) [6, 14, 17] or reachability-based approaches [1, 3, 10, 16, 29].

Given a finite abstraction of a dynamical system and an LTL specification, controllers can be automatically constructed using an automata-based approach [2, 6, 9, 14, 16]. This approach first transforms the LTL formula into an equivalent Büchi automaton whose size may be exponential in the length of the formula [2]. A product automaton is created from the finite abstraction and the Büchi automaton, and then a controller is found by graph search in the product automaton.

The main drawback of this approach is that it is expensive to compute a finite abstraction. The product automaton might also be quite large due to the size of the abstraction and the Büchi automaton. Finally, although optimal controllers can be computed for the discrete abstraction [21, 27], optimality is only with respect to the abstraction's level of refinement or asymptotic [14].

Instead of the automata-based approach, we directly encode a large class of temporal logic formulas as mixed-integer linear constraints on the original dynamical system. These constraints enforce that an infinite sequence of system states satisfies a task specification. A key component of our formulation is enforcing that the system is in a (non-convex) region at a given time. We introduce an alternative formulation for this that gives a tighter convex relaxation than the commonly used big-M approach. Our approach applies to any deterministic system model that is amenable to finite-dimensional optimization, as the temporal logic constraints are independent of any particular system dynamics or cost functions. We specifically investigate Mixed Logical Dynamic (MLD) systems [4] and certain differentially flat systems [19], whose dynamics can be encoded with mixed-integer linear constraints. MLD systems include constrained linear systems, linear hybrid automata, and piecewise affine systems. Differentially flat systems include quadrotors and car-like vehicles.

It is well-known that mixed-integer linear programming can be used for reasoning about propositional logic [7, 11], generating state-constrained trajectories [8, 20, 24], and modeling vehicle routing problems [13, 22]. The work most similar to ours is Karaman et al. [15], who consider controller synthesis for MLD systems subject to finite-horizon LTL specifications. However, finite-horizon properties are too restrictive to model a large class of interesting robotics problems, including persistent surveillance, repeated assembly, periodic motion, and environmental monitoring. Our work specifically addresses these types of periodic tasks with a novel mixed-integer formulation.

Our main contributions are (1) a novel method for encoding both finite- and infinite-horizon temporal logic properties as mixed-integer linear constraints on a system and (2) an improved encoding that has a tighter convex relaxation and uses significantly fewer binary variables for common tasks than related work [15]. The fragment of temporal logic that we consider allows one to specify properties such as safety, stability, liveness, guarantee, and response. We demonstrate how this mixed-integer programming formulation can be used with off-the-shelf optimization solvers (e.g. CPLEX [23]) to compute both feasible and optimal controllers for high-dimensional systems with temporal logic specifications.

2 Preliminaries

An *atomic proposition* is a statement that is *True* or *False*. A *propositional formula* is composed of only atomic propositions and propositional connectives, i.e., $\wedge$ (and), $\vee$ (or), and $\neg$ (not). Let $\mathcal{T} = \{0, 1, 2, \ldots, T\} \subset \mathbb{N}$ denote a bounded set of discrete time instances and $\mathcal{T}^{\infty} = \{0, 1, 2, \ldots\}$ denote an unbounded set of discrete time instances.

2.1 System Model

We consider discrete-time nonlinear systems of the form

$$x(t+1) = f(x(t), u(t)), \tag{1}$$

where $t \in \mathcal{T}^{\infty}$, $x \in \mathcal{X} \subseteq \mathbb{R}^{n_c} \times \{0,1\}^{n_l}$ are the continuous and binary states, $u \in \mathcal{U} \subseteq \mathbb{R}^{m_c} \times \{0,1\}^{m_l}$ are the inputs, and $x(0) = x_0 \in \mathcal{X}$ is the initial state. We assume that the system is deterministic, i.e., an initial state x_0 and a control input sequence $\mathbf{u} = u_0 u_1 u_2 \ldots$ produces a unique trajectory (or run) $\mathbf{x} = \mathbf{x}(x_0, \mathbf{u}) = x_0 x_1 x_2 \ldots$.

Let AP be a finite set of atomic propositions. The (time-dependent) *labeling function* $L_t : \mathcal{X} \rightarrow 2^{AP}$ maps the continuous part of each state to the set of atomic propositions that are *True* at time t. Each atomic proposition $\psi \in AP$ is represented by a union of polyhedrons. The finite index set I_t^{ψ} lists the polyhedrons where ψ holds at time t. The ith polyhedron is $\{x \in \mathcal{X} \mid H_t^{\psi_i} x \leq K_t^{\psi_i}\}$, where $i \in I_t^{\psi}$. Thus, the set of states where atomic proposition ψ holds at time t is given by $[\![\psi]\!](t) := \{x \in \mathcal{X} \mid H_t^{\psi_i} x \leq K_t^{\psi_i}$ for some $i \in I_t^{\psi}\}$. This (potentially) time-varying set is the finite union of polyhedrons (finite conjunctions of halfspaces).

2.2 A Fragment of Temporal Logic

We do not attempt to reason about all possible temporal logic formulas (see [2]); instead, we develop a useful library of temporal operators for robotic tasks. This fragment of temporal logic can concisely and unambiguously specify a wide range of tasks such as safe navigation, surveillance, persistent coverage, response to the environment, and visiting goals. In the following definitions, ψ, ϕ, and ψ_j (for a finite number of indices j) are propositional formulas. To simplify the presentation, we split these into three groups: core Φ_{core}, response Φ_{resp}, and fairness Φ_{fair}. We first define the syntax of the temporal operators and then their semantics.

Syntax

The core operators, $\Phi_{\text{core}} := \{\varphi_{\text{safe}}, \varphi_{\text{goal}}, \varphi_{\text{per}}, \varphi_{\text{live}}, \varphi_{\text{until}}\}$, specify fundamental properties such as safety, guarantee, persistence, liveness (recurrence), and until. These operators are,

$$\varphi_{\text{safe}} := \Box\psi, \quad \varphi_{\text{goal}} := \Diamond\psi, \quad \varphi_{\text{per}} := \Diamond\Box\psi, \quad \varphi_{\text{live}} := \Box\Diamond\psi, \quad \varphi_{\text{until}} := \psi \,\mathcal{U}\, \phi,$$

where φ_{safe} specifies safety, i.e., a property should invariantly hold, φ_{goal} specifies goal visitation, i.e., a property should eventually hold, φ_{per} specifies persistence, i.e., a property should eventually hold invariantly, and φ_{live} specifies liveness (recurrence), i.e., a property should hold repeatedly, as in surveillance, and φ_{until} specifies until, i.e., a property ψ should hold until another property ϕ holds.

The response operators, $\Phi_{\text{resp}} := \{\varphi^1_{\text{resp}}, \varphi^2_{\text{resp}}, \varphi^3_{\text{resp}}, \varphi^4_{\text{resp}}\}$, specify how the system responds to the environment. These operators are,

$$\varphi^1_{\text{resp}} := \Box(\psi \implies \bigcirc\phi), \qquad \varphi^2_{\text{resp}} := \Box(\psi \implies \Diamond\phi),$$
$$\varphi^3_{\text{resp}} := \Diamond\Box(\psi \implies \bigcirc\phi), \qquad \varphi^4_{\text{resp}} := \Diamond\Box(\psi \implies \Diamond\phi),$$

where φ^1_{resp} specifies next-step response to the environment, φ^2_{resp} specifies eventual response to the environment, φ^3_{resp} specifies steady-state next-step response to the environment, and φ^4_{resp} specifies steady-state eventual response to the environment.

Finally, the fairness operators, $\Phi_{\text{fair}} := \{\varphi^1_{\text{fair}}, \varphi^2_{\text{fair}}, \varphi^3_{\text{fair}}\}$, allow one to specify conditional tasks. These operators are,

$$\varphi^1_{\text{fair}} := \Diamond\psi \implies \bigwedge_{j=1}^{m} \Diamond\phi_j, \qquad \varphi^2_{\text{fair}} := \Diamond\psi \implies \bigwedge_{j=1}^{m} \Box\Diamond\phi_j,$$
$$\varphi^3_{\text{fair}} := \Box\Diamond\psi \implies \bigwedge_{j=1}^{m} \Box\Diamond\phi_j,$$

where φ^1_{fair} specifies conditional goal visitation, and φ^2_{fair} and φ^3_{fair} specify conditional repeated goal visitation.

The fragment of LTL that we consider is built from the temporal operators defined above as follows,

$$\varphi :: = \varphi_{\text{core}} \mid \varphi_{\text{resp}} \mid \varphi_{\text{fair}} \mid \varphi_1 \wedge \varphi_2, \tag{2}$$

where $\varphi_{\text{core}} \in \Phi_{\text{core}}$, $\varphi_{\text{resp}} \in \Phi_{\text{resp}}$, and $\varphi_{\text{fair}} \in \Phi_{\text{fair}}$.

This LTL fragment specifies many properties relevant to robotics, especially for surveillance tasks for which no mathematical programming-based approaches currently exist. However, it does not include nested properties [2]. Determining all temporal properties that can be expressed in this framework is future work.

Remark 1 To include disjunctions (e.g., $\varphi_1 \vee \varphi_2$), one can rewrite a formula in disjunctive normal form, where each clause is of the form (2). In what follows, each clause can then be considered separately, as the system (1) is deterministic.

Semantics

We use set operations between a trajectory (run) $\mathbf{x} = \mathbf{x}(x_0, \mathbf{u})$ and subsets of $\mathcal{X}$ where particular propositional formulas hold to define satisfaction of a temporal logic formula [2]. We denote the set of states where propositional formula ψ holds by $[\![\psi]\!]$. A run $\mathbf{x}$ *satisfies* the temporal logic formula φ, denoted by $\mathbf{x} \models \varphi$, if and only if certain set operations hold. Given propositional formulas ψ and ϕ, we relate satisfaction of (a partial list of) formulas of the form (2) with set operations as follows:

- $\mathbf{x} \models \Box\psi$ iff $x_i \in [\![\psi]\!]$ for all i,
- $\mathbf{x} \models \Diamond\Box\psi$ iff there exists an index j such that $x_i \in [\![\psi]\!]$ for all $i \geq j$,
- $\mathbf{x} \models \Diamond\psi$ iff $x_i \in [\![\psi]\!]$ for some i,
- $\mathbf{x} \models \Box\Diamond\psi$ iff $x_i \in [\![\psi]\!]$ for infinitely many i,
- $\mathbf{x} \models \psi\ \mathcal{U}\ \phi$ iff there exists an index j such that $x_j \in [\![\phi]\!]$ and $x_i \in [\![\psi]\!]$ for all $i < j$,
- $\mathbf{x} \models \Box(\psi \implies \bigcirc\phi)$ iff $x_i \notin [\![\psi]\!]$ or $x_{i+1} \in [\![\phi]\!]$ for all i,
- $\mathbf{x} \models \Box(\psi \implies \Diamond\phi)$ iff $x_i \notin [\![\psi]\!]$ or $x_k \in [\![\phi]\!]$ for some $k \geq i$ for all i,
- $\mathbf{x} \models \Diamond\Box(\psi \implies \bigcirc\phi)$ iff there exists an index j such that $x_i \notin [\![\psi]\!]$ or $x_{i+1} \in [\![\phi]\!]$ for all $i \geq j$,
- $\mathbf{x} \models \Diamond\Box(\psi \implies \Diamond\phi)$ iff there exists an index j such that $x_i \notin [\![\psi]\!]$ or $x_k \in [\![\phi]\!]$ for some $k \geq i$ for all $i \geq j$.

A run $\mathbf{x}$ *satisfies* a conjunction of temporal logic formulas $\varphi = \bigwedge_{i=1}^{m} \varphi_i$ if and only if the set operations for each temporal logic formula φ_i holds. The LTL formula φ is *satisfiable* by a system at state $x_0 \in \mathcal{X}$ if and only if there exists a control input sequence $\mathbf{u}$ such that $\mathbf{x}(x_0, \mathbf{u}) \models \varphi$.

3 Problem Statement

In this section, we formally state both a feasibility and an optimization problem and give an overview of our solution approach. Let φ be an LTL formula of the form (2) defined over AP.

Problem 1 Given a system of the form (1), with initial condition x_0, and an LTL formula φ of the form (2), determine whether or not there exists a control input sequence $\mathbf{u}$ such that $\mathbf{x}(x_0, \mathbf{u}) \models \varphi$.

We now introduce a cost function to distinguish among all trajectories that satisfy Problem 1. Since LTL formulas are defined over infinite state sequences, we define a cost function over infinite state sequences. We use a maximum cost function to simplify the presentation; it can easily be extended to discounted, limit-maximum, and average cost functions (see [26]). Let the cost $c : \mathcal{X} \times \mathcal{U} \rightarrow \mathbb{R}$ be bounded.

Definition 1 Let $\mathbf{x}$ be a trajectory and $\mathbf{u}$ be the corresponding control input sequence. The *maximum cost* of trajectory $\mathbf{x}$ is

$$J(\mathbf{x}, \mathbf{u}) := \sup_{t \in \mathcal{T}^{\infty}} c(x_t, u_t), \tag{3}$$

where J maps trajectories and control inputs to $\mathbb{R} \cup \infty$.

Problem 2 Given a system of the form (1), with initial condition x_0, and an LTL formula φ of the form (2), compute a control input sequence $\mathbf{u}$ such that $\mathbf{x}(x_0, \mathbf{u}) \models \varphi$ and $J(\mathbf{x}(x_0, \mathbf{u}), \mathbf{u})$ is minimized.

We now give a brief overview of our solution approach. We parameterize the system trajectory (control input) as a periodic prefix-suffix structure. Every LTL operator of the form (2) is encoded as mixed-integer linear constraints on this finite parameterization. These temporal logic constraints (see Sect. 5) are then combined with dynamic constraints (see Sect. 6) as constraints on a combined mixed-integer optimization problem with an appropriate cost function. For MLD systems and certain differentially flat systems (see Sect. 6) with linear costs, Problems 1 and 2 can thus be solved using a mixed-integer linear program (MILP) solver. While even checking feasibility of a MILP is NP-hard, modern solvers using branch and bound methods routinely solve large problems [23]. We show promising results (see Fig. 1) on high-dimensional (10+ continuous state) systems in Sect. 7.

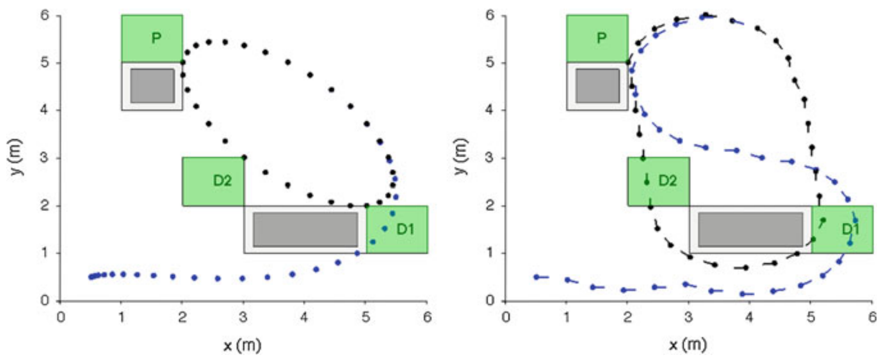


Fig. 1 Illustration of a problem instance. The task is to repeatedly visit regions P, $D1$, and $D2$, where *dark* regions are obstacles that must be avoided. Representative trajectories for a quadrotor (*left*) and nonlinear car (*right*) are shown with the prefix (*blue*) and suffix (*black*)

Remark 2 We only consider open-loop trajectory generation, which is already a challenging problem due to the nonlinear dynamics and LTL specifications. Disturbances can be dealt with by wrapping a feedback controller around the trajectory. Incorporating disturbances during trajectory generation is the subject of future work.

4 A Periodic Trajectory Parameterization

We parameterize the system trajectory by a periodic prefix-suffix form that is commonly used in model checking for finite systems. In this structure, the prefix is a finite trajectory and the suffix is a finite trajectory that is repeated infinitely often. This gives a sufficient condition that is amenable to computation, although it may miss valid non-periodic trajectories.

A *walk* is a finite sequence of states $\mathbf{x} = x_0 x_1 x_2 \dots x_N$ that satisfy the constraints in (1). A *cycle* is a walk $\mathbf{x} = x_0 x_1 x_2 \dots x_N$ where $f(x_N, u) = x_0$ for some $u \in \mathcal{U}$. A trajectory $\mathbf{x}$ induces a corresponding *word* (i.e., sequence of labels) $L(\mathbf{x}) = L_0(x_0) L_1(x_1) L_2(x_2) \dots$ through the labeling function. A word is similarly defined for a walk or cycle. We now define a trajectory in *prefix-suffix* form.

Definition 2 Let $\mathbf{x}_{\text{pre}}$ be a finite walk and $\mathbf{x}_{\text{suf}}$ be a finite cycle. A trajectory $\mathbf{x}$ is in *prefix-suffix* form if it is of the form $\mathbf{x} = \mathbf{x}_{\text{pre}}(\mathbf{x}_{\text{suf}})^{\omega}$, where ω denotes infinite repetition.

We will require that the (time-varying) labeling function L_t is eventually periodic.

Assumption 1 There exists a finite $t' \in \mathcal{T}^{\infty}$ and a $\Omega \in \mathbb{N}$ such that $L_t = L_{t+\Omega}$ for all $t \geq t' \in \mathcal{T}^{\infty}$. We further assume that Ω is minimal among all possible values.

In the sequel, we will only consider trajectories $\mathbf{x} = \mathbf{x}_{\text{pre}}(\mathbf{x}_{\text{suf}})^{\omega}$ in prefix-suffix form. While both $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ are finite, the constraint that $\mathbf{x}_{\text{suf}}$ is a cycle allows us to repeat that sequence of states forever. Repeating the same sequence of states is a sufficient condition that the word $L(\mathbf{x}_{\text{suf}})$ (i.e., the sequence of atomic propositions) is also repeated (using Assumption 1). However, only the word matters for the feasibility of an LTL formula, not the exact sequence of states. In fact, there may exist other trajectories that produce the same word $L(x)$, but are not eventually periodic. Our approach cannot find such trajectories, although we have not noticed this limitation in our experiments. This differs from the case of finite discrete systems, where a prefix-suffix form is sufficient to find a feasible solution if one exists [2].

In the next section, we will encode the temporal operators as mixed-integer constraints on $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$. Let $\mathbf{x}_{\text{cat}} := \mathbf{x}_{\text{pre}}\mathbf{x}_{\text{suf}}$ denote the concatenation of $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$, and assign time indices to $\mathbf{x}_{\text{cat}}$ as $\mathcal{T}_{\text{cat}} := \{0, 1, \dots, T_s, \dots, T\}$. Let $\mathcal{T}_{\text{pre}} := \{0, 1, \dots, T_s - 1\}$ and $\mathcal{T}_{\text{suf}} := \{T_s, \dots, T\}$, where T_s is the first time instance on the suffix. The infinite repetition of $\mathbf{x}_{\text{suf}}$ is enforced by the constraint $\mathbf{x}_{\text{cat}}(T_s) = f(\mathbf{x}_{\text{cat}}(T), u)$ for some $u \in \mathcal{U}$. By Assumption 1, it is sufficient that T_s is greater than t' and that the length of $\mathcal{T}_{\text{suf}}$ is an integer multiple of Ω. We often identify $\mathbf{x}_{\text{pre}}(0)\cdots\mathbf{x}_{\text{pre}}(T_{\text{pre}})$ with $\mathbf{x}_{\text{cat}}(0)\cdots\mathbf{x}_{\text{cat}}(T_s-1)$ and $\mathbf{x}_{\text{suf}}(0)\cdots\mathbf{x}_{\text{suf}}(T_{\text{suf}})$ with $\mathbf{x}_{\text{cat}}(T_s)\cdots\mathbf{x}_{\text{cat}}(T)$ in the obvious manner.

5 A Mixed-Integer Linear Formulation of LTL Constraints

In this section, we develop a mixed-integer programming formulation for a given prefix-suffix trajectory parameterization, $\mathbf{x}_{\text{cat}} = \mathbf{x}_{\text{pre}}\mathbf{x}_{\text{suf}}$. The corresponding system trajectory is $\mathbf{x} = \mathbf{x}_{\text{pre}}(\mathbf{x}_{\text{suf}})^{\omega}$. Since the system is deterministic, this defines a corresponding control input sequence. The split between $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ can either be specified *a priori* or left as a variable (see [26] for details). We mix notation in the following and refer to x and $\mathcal{T}$ instead of $\mathbf{x}_{\text{cat}}$ and $\mathcal{T}_{\text{cat}}$ when clear from context.

5.1 Relating the Dynamics and Propositions

We now relate the state of a system to the set of atomic propositions that are *True* at each time instance. We assume that each propositional formula ψ is described at time t by the union of a finite number of polytopes, indexed by the finite index set I_t^{ψ}. Let $[\![\psi]\!](t) := \{x \in \mathcal{X} \mid H_t^{\psi_i}x \le K_t^{\psi_i} \text{ for some } i \in I_t^{\psi}\}$ represent the set of states that satisfy propositional formula ψ at time t. We assume that these have been constructed as necessary from the system's original atomic propositions. We note that a proposition preserving partition [1] is not necessary or even desired.

For each propositional formula ψ, introduce binary variables $z_t^{\psi_i} \in \{0, 1\}$ for all $i \in I_t^{\psi}$ and for all $t \in \mathcal{T}$. Let x_t be the state of the system at time t and M be a vector of sufficiently large constants. The *big-M* formulation

$$\begin{aligned} &H_t^{\psi_i}x_t \le K_t^{\psi_i} + M(1 - z_t^{\psi_i}), \quad \forall i \in I_t^{\psi} \\ &\sum_{i \in I_t^{\psi}} z_t^{\psi_i} = 1 \end{aligned} \tag{4}$$

enforces the constraint that $x_t \in [\![\psi]\!](t)$ at time t. Define $P_t^{\psi} := \sum_{i\in I_t^{\psi}} z_t^{\psi_i}$. If $P_t^{\psi} = 1$, then $x_t \in [\![\psi]\!](t)$. If $P_t^{\psi} = 0$, then nothing can be inferred.

The big-M formulation may give poor continuous relaxations of the binary variables, i.e., $z_t^{\psi_i} \in [0, 1]$, which may lead to poor performance during optimization [23]. Such relaxations are frequently used during the solution of mixed-integer linear programs [23]. Thus, we introduce an alternate representation whose continuous relaxation is the convex hull of the original set $[\![\psi]\!](t)$. This formulation is well-known in the optimization community [12], but does not appear in the trajectory generation literature ([8, 20, 24] and references therein). As such, this formulation may be of independent interest for trajectory planning with obstacles.

The *convex hull* formulation

$$\begin{aligned}
&H_t^{\psi_i} x_t^i \le K_t^{\psi_i} z_t^{\psi_i}, \quad \forall i \in I_t^{\psi} \\
&\sum_{i \in I_t^{\psi}} z_t^{\psi_i} = 1, \\
&\sum_{i \in I_t^{\psi}} x_t^i = x_t
\end{aligned} \tag{5}$$

represents the same set as the big-M formulation (4). While the convex hull formulation introduces more continuous variables, it gives the tightest linear relaxation of the disjunction of the polytopes and reduces the need to select the M parameters [12]. Note that we will only use the convex hull formulation (5) for safety and persistence formulas (i.e., φ_{safe} and φ_{per}) in Sect. 5.2, as $P_t^{\psi} = 0$ enforces $x = 0$.

Regardless if one uses the big-M or convex hull formulation, only one binary variable is needed for each polyhedron (i.e., finite conjunction of halfspaces). This compares favorably with the approach in [15], where a binary variable is introduced for each halfspace. Additionally, the auxiliary continuous variables and mixed-integer constraints previously used are not needed because we use implication. For simple tasks such as $\varphi = \Diamond\psi$, our method can use significantly fewer binary variables than previously needed, depending on the number of halfspaces and polytopes needed to describe $[\![\psi]\!]$.

For every temporal operator described in the following section, the constraints in (4) or (5) should be understood to be implicitly applied to the corresponding propositional formulas so that $P_t^{\psi} = 1$ implies that the system satisfies ψ at time t. Also, note that we use different binary variables for each formula—even when representing the same set.

5.2 The Mixed-Integer Linear Constraints

In this section, the trajectory parameterization $\mathbf{x}$ has been *a priori* split into a prefix $\mathbf{x}_{\text{pre}}$ and a suffix $\mathbf{x}_{\text{suf}}$. This assumption can be relaxed, so that the size of $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ are optimization variables (see [26] for details). We further assume that $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ satisfy Assumption 1.

In the following, the correctness of the constraints applied to $\mathbf{x}_{\text{pre}}$ and $\mathbf{x}_{\text{suf}}$ comes directly from the temporal logic semantics given in Sect. 2.2 and the form of the trajectory $\mathbf{x} = \mathbf{x}_{\text{pre}}(\mathbf{x}_{\text{suf}})^{\omega}$. The most important factors are whether a property can be verified over finite- or infinite-horizons. All infinite-horizon (liveness) properties must be satisfied on the suffix $\mathbf{x}_{\text{suf}}$.

We begin with the fundamental temporal operators Φ_{core}. Safety and persistence require a mixed-integer linear constraint for each time step, while guarantee and liveness only require a single mixed-integer linear constraint.

Safety, $\varphi_{\text{safe}} = \square\psi$, is satisfied by the constraints

$$\begin{aligned} P_t^{\psi} &= 1, \quad \forall t \in \mathcal{T}_{\text{pre}}, \\ P_t^{\psi} &= 1, \quad \forall t \in \mathcal{T}_{\text{suf}}, \end{aligned}$$

which ensure that the system is always in a $[\![\psi]\!]$ region. Similarly, persistence, $\varphi_{\text{per}} = \lozenge\square\psi$, is enforced by

$$P_t^{\psi} = 1, \quad \forall t \in \mathcal{T}_{\text{suf}},$$

which ensures the system eventually remains in a $[\![\psi]\!]$ region.

Guarantee, $\varphi_{\text{goal}} = \lozenge\psi$, is satisfied by the constraints

$$\sum_{t\in\mathcal{T}_{\text{pre}}} P_t^{\psi} + \sum_{t\in\mathcal{T}_{\text{suf}}} P_t^{\psi} = 1,$$

which ensures the system eventually visits a $[\![\psi]\!]$ region. Similarly, liveness $\varphi_{\text{live}} = \square\lozenge\psi$ is enforced by

$$\sum_{t\in\mathcal{T}_{\text{suf}}} P_t^{\psi} = 1,$$

which ensures the system repeatedly visits a $[\![\psi]\!]$ region.

Until, $\varphi_{\text{until}} = \psi \,\mathcal{U}\, \phi$, is enforced by

$$\begin{aligned} P_0^{\phi} &= s_0, \\ P_t^{\phi} &= s_t - s_{t-1}, \quad t = 1, \ldots, T \\ P_t^{\psi} &= 1 - s_t, \quad \forall t \in \mathcal{T}, \end{aligned}$$

where we use auxiliary binary variables $s_t \in \{0, 1\}$ for all $t \in \mathcal{T}$ such that $s_t \leq s_{t+1}$ for $t = 0, \ldots, T-1$ and $s_T = 1$.

Now consider the response temporal operators Φ_{resp}. For these formulas, the definition of implication is used to convert each inner formula into a disjunction between a property that holds at a state and a property that holds at some point in the future. The response formulas require a mixed-integer linear constraint for each time step.

For next-step response, $\varphi_{\text{resp}}^1 = \square(\psi \implies \bigcirc\phi) = \square(\neg\psi \vee \bigcirc\phi)$, the additional constraints are

$$\begin{aligned} P_t^{\neg\psi} + P_{t+1}^{\phi} &= 1, \quad t = 0, \dots, T_s, \dots, T-1, \\ P_T^{\neg\psi} + P_{T_s}^{\phi} &= 1, \end{aligned}$$

Similarly, steady-state next-step response, $\varphi_{\text{resp}}^3 = \Diamond\Box(\psi \implies \bigcirc\phi) = \Diamond\Box(\neg\psi \vee \bigcirc\phi)$, is satisfied by

$$\begin{aligned} P_t^{\neg\psi} + P_{t+1}^{\phi} &= 1, \quad t = T_s, \dots, T-1, \\ P_T^{\neg\psi} + P_{T_s}^{\phi} &= 1, \end{aligned}$$

Eventual response, $\varphi_{\text{resp}}^2 = \Box(\psi \implies \Diamond\phi) = \Box(\neg\psi \vee \Diamond\phi)$, requires the following constraints

$$\begin{aligned} P_t^{\neg\psi} + \sum_{\tau=t}^{T} P_\tau^{\phi} &= 1, \quad \forall t \in \mathcal{T}_{\text{pre}}, \\ P_t^{\neg\psi} + \sum_{t \in \mathcal{T}_{\text{suf}}} P_t^{\phi} &= 1, \quad \forall t \in \mathcal{T}_{\text{suf}}. \end{aligned}$$

Similarly, for steady-state eventual response, $\varphi_{\text{resp}}^4 = \Diamond\Box(\psi \implies \Diamond\phi) = \Diamond\Box(\neg\psi \vee \Diamond\phi)$, the additional constraints are

$$P_t^{\neg\psi} + \sum_{t \in \mathcal{T}_{\text{suf}}} P_t^{\phi} = 1, \quad \forall t \in \mathcal{T}_{\text{suf}}.$$

Now consider the fairness temporal operators Φ_{fair}. In the following, the definition of implication is used to rewrite the inner formula as disjunction between a single safety (persistence) property and a conjunction of guarantee (liveness) properties. These formulas require a mixed-integer linear constraint for each conjunction in the response and each time step.

Conditional goal visitation, $\varphi_{\text{fair}}^1 = \Diamond\psi \implies \bigwedge_{j=1}^m \Diamond\phi_j = \Box\neg\psi \vee \bigwedge_{j=1}^m \Diamond\phi_j$, is specified by

$$P_t^{\neg\psi} + \sum_{t \in \mathcal{T}} P_t^{\phi_j} = 1, \quad \forall j = 1, \dots, m, \forall t \in \mathcal{T}.$$

Conditional repeated goal visitation, $\varphi_{\text{fair}}^2 = \Diamond\psi \implies \bigwedge_{j=1}^m \Box\Diamond\phi_j = \Box\neg\psi \vee \bigwedge_{j=1}^m \Box\Diamond\phi_j$, is enforced as

$$P_t^{\neg\psi} + \sum_{t \in \mathcal{T}_{\text{suf}}} P_t^{\phi_j} = 1, \quad \forall j = 1, \dots, m, \forall t \in \mathcal{T}.$$

Similarly, $\varphi_{\text{fair}}^3 = \Box\Diamond\psi \implies \bigwedge_{j=1}^m \Box\Diamond\phi_j = \Diamond\Box\neg\psi \vee \bigwedge_{j=1}^m \Box\Diamond\phi_j$, is represented by

$$P_t^{\neg\psi} + \sum_{t \in \mathcal{T}_{\mathrm{suf}}} P_t^{\phi_j} = 1, \quad \forall j = 1, \ldots, m, \; \forall t \in \mathcal{T}_{\mathrm{suf}}.$$

We have encoded the temporal logic specifications on the system variables using mixed-integer linear constraints. Note that the equality constraints on the binary variables dramatically reduce search space. In Sect. 6 we discuss adding dynamics to further constrain the possible behaviors of the system.

6 System Dynamics

The mixed-integer constraints in Sect. 5 are over a sequence of states, and thus are independent of the specific system dynamics. Dynamic constraints on the sequence of states can also be enforced by standard transcription methods [5]. However, the resulting optimization problem may then be a mixed-integer *nonlinear* program due to the dynamics. We highlight two useful classes of nonlinear systems where the dynamics can be encoded using mixed-integer *linear* constraints.

6.1 Mixed Logical Dynamical Systems

Mixed Logical Dynamical (MLD) systems have both continuous and discrete-valued states and allow one to model nonlinearities, logic, and constraints [4]. These systems include constrained linear systems, linear hybrid automata, and piecewise affine systems. An MLD system is of the form

$$\begin{aligned} & x(t+1) = Ax(t) + B_1 u(t) + B_2 \delta(t) + B_3 z(t) \\ \text{subject to} \quad & E_2 \delta(t) + E_3 z(t) \leq E_1 u(t) + E_4 x(t) + E_5, \end{aligned}$$

where $t \in \mathcal{T}^{\infty}$, $x \in \mathcal{X} \subseteq \mathbb{R}^{n_c} \times \{0, 1\}^{n_l}$ are the continuous and binary states, $u \in \mathcal{U} \subseteq \mathbb{R}^{m_c} \times \{0, 1\}^{m_l}$ are the inputs, and $\delta \in \{0, 1\}^{r_l}$ and $z \in \mathbb{R}^{r_l}$ are auxiliary binary and continuous variables, respectively. The terms A, B_1, B_2, B_3, E_1, E_2, E_3, E_4, and E_5 are system matrices of appropriate dimension. We assume that the system is deterministic and well-posed (see Definition 1 in [4]).

6.2 Differentially Flat Systems

A system is *differentially flat* if there exists a set of outputs such that all states and control inputs can be determined from these outputs without integration. If a system has states $x \in \mathbb{R}^n$ and control inputs $u \in \mathbb{R}^m$, then it is flat if we can find outputs $y \in \mathbb{R}^m$ of the form $y = y(x, u, \dot{u}, \ldots, u^{(p)})$ such that $x = x(y, \dot{y}, \ldots, y^{(q)})$ and $u = u(y, \dot{y}, \ldots, y^{(q)})$. Thus, we can plan trajectories in output space and then map these to control inputs [17].

Differentially flat systems may be encoded using mixed integer linear constraints in certain cases, e.g., the flat output is constrained by mixed integer linear constraints. This holds for relevant classes of robotic systems, including quadrotors and car-like robots. However, control input constraints are typically non-convex in the flat output. Common approaches to satisfy control constraints are to plan a sufficiently smooth trajectory or slow down along a trajectory [19].

7 Examples

We demonstrate our techniques on a variety of motion planning problems. The first example is a chain of integrators parameterized by dimension. Our second example is a quadrotor model from [25]. Our final example is a nonlinear car-like vehicle with drift. All computations were done on a laptop with a 2.4 GHz dual-core processor and 4 GB of memory using CPLEX [23] through Yalmip [18].

The environment and task is motivated by a pickup and delivery scenario. All properties should be understood to be with respect to regions in the plane (see Fig. 1). Let P be a region where supplies can be picked up and $D1$ and $D2$ be regions where supplies must be delivered. The robot must remain in the safe region S (in white). Formally, the task specification is $\varphi = \Box S \wedge \Box\Diamond P \wedge \Box\Diamond D1 \wedge \Box\Diamond D2$. Additionally, we minimize the maximum cost function (3) where $c(x_t, u_t) = |u_t|$ penalizes the control input.

In the remainder of this section, we consider this temporal logic motion planning problem for different system models. We use the simultaneous (sim.) approach described in Sect. 5.2, and also a sequential (seq.) approach from [26] that first computes the suffix and then the prefix. A trajectory of length 60 (split evenly between the prefix and suffix) is used in all cases, and all results are averaged over 20 randomly generated environments. The simultaneous approach uses between 300 and 469 binary variables with a mean of 394. Finally, all continuous-time models are discretized using a first-order hold and time-step of 0.5 s.

7.1 *Chain of Integrators*

The first system is a chain of orthogonal integrators in the x and y directions. The kth derivative of the x and y positions are controlled, i.e., $x^{(k)} = u_x$ and $y^{(k)} = u_y$, subject to the constraints $|u_x| \leq 0.5$ and $|u_y| \leq 0.5$. The general state constraints are $|x^{(i)}| \leq 1$ and $|y^{(i)}| \leq 1$ for $i = 1, \ldots, k-1$. Results are given in Tables 1 and 2 under "chain-2", "chain-6", and "chain-10", where "chain-k" indicates that the kth derivative in both the x and y positions is controlled.

Table 1 Time until a feasible solution was found (mean ± standard error) and number of problems (out of 20) solved in 45 s using the big-M formulation (4) with M = 10

Model	Dimensional	Feasible solution (s)		Number of problems solved	
		Simultaneous	Sequential	Simultaneous	Sequential
Chain-2	4	1.10 ± 0.09	0.64 ± 0.06	20	20
Chain-6	12	$4.70 \pm .48$	2.23 ± 0.15	20	20
Chain-10	20	9.38 ± 1.6	3.74 ± 0.29	20	19
Quadrotor	10	4.20 ± 0.66	1.80 ± 0.15	20	20
Quadrotor-flat	10	2.26 ± 0.36	1.99 ± 1.0	20	20
Car-3	3	43.9 ± 0.77	10.7 ± 2.0	4	20
Car-4	3	42.4 ± 1.7	18.7 ± 3.1	2	18
Car-flat	3	15.8 ± 3.8	14.0 ± 4.4	12	14

Table 2 Time until a feasible solution was found (mean ± standard error) and number of problems (out of 20) solved in 45 s using the convex hull formulation (5)

Model	Dimensional	Feasible solution (s)		Number of problems solved	
		Simultaneous	Sequential	Simultaneous	Sequential
Chain-2	4	1.94 ± 0.23	0.94 ± 0.11	20	20
Chain-6	12	12.4 ± 2.7	2.89 ± 0.32	20	20
Chain-10	20	16.9 ± 3.0	7.28 ± 1.2	17	15
Quadrotor	10	18.9 ± 3.8	2.80 ± 0.35	16	20
Car-3	3	37.3 ± 3.1	13.3 ± 1.6	8	20

7.2 *Quadrotor*

We now consider the quadrotor model used in [25] for point-to-point motion planning, to which we refer the reader for a complete description of the model. The state $x = (p, v, r, w)$ is 10-dimensional, consisting of position $p \in \mathbb{R}^3$, velocity $v \in \mathbb{R}^3$, orientation $r \in \mathbb{R}^2$, and angular velocity $w \in \mathbb{R}^2$. This model is the linearization of a nonlinear model about hover with the yaw constrained to be zero. The control input $u \in \mathbb{R}^3$ is the total, roll, and pitch thrust. Results are given in Tables 1 and 2 under "quadrotor", and a sample trajectory is shown in Fig. 1.

Also, we use the fact that the quadrotor is differentially flat [19] to generate trajectories for the nonlinear model (with fixed yaw). We parameterize the flat output $p \in \mathbb{R}^3$ with eight piecewise polynomials of degree three, and then optimize over their coefficients to compute a smooth trajectory. Afterwards, we check that the trajectory does not violate the control input constraints. Results are given in Table 1 under "quadrotor-flat".

7.3 Nonlinear Car

Consider a nonlinear car-like vehicle with state $x = (p_x, p_y, \theta)$ and dynamics $\dot{x} = (v\cos(\theta), v\sin(\theta), u)$. The variables p_x, p_y are position (m) and θ is orientation (rad). The vehicle's speed v is fixed at 0.8 (m/s) and its control input is constrained as $|u| \leq 2.5$. We form a hybrid MLD model by linearizing the system about different orientations $\hat{\theta}_i$ for $i = 1, \ldots, k$. The dynamics are governed by the closest linearization to the current θ. Results with $k = 3$ and $k = 4$ are given in Table 1 under "car-3" and "car-4", respectively. A sample trajectory of "car-4" is show in Fig. 1.

Additionally, we use the flat output $(x, y) \in \mathbb{R}^2$ to generate trajectories for the nonlinear car-like model in a similar manner as for the quadrotor model. Results are given in Table 1 under "car-flat".

7.4 Discussion and Comparison

We first compare our approach to reachability-based algorithms that compute a finite abstraction [16, 28]. We used the method in [28] to compute a discrete abstraction for a two dimensional system in 22 s, and [16] reports abstracting a four dimensional system in just over a minute. This contrasts with our mixed-integer approach that can routinely find solutions to such problems in seconds, although we do not compute a feedback controller. Our results appear particularly promising for situations where the environment is dynamically changing and a finite abstraction must be repeatedly computed.

We also compare to the finite-horizon mixed-integer formulation given in [15]. Consider the task $\varphi = \Diamond\psi$, where $[\![\psi]\!]$ is a convex polytope defined by m halfspaces. Our method uses one binary variable at each time step, while their approach uses m. Additionally, while we encode eventually ($\Diamond$) using a single constraint, their approach uses a number of constraints quadratic in the trajectory length.

In most of our examples, we are able to quickly compute a feasible trajectory that satisfies a temporal logic formula by solving a mixed-integer linear program. This is aided by the sequential approach, which separates the problem into computing a suffix and then a prefix [26]. It typically takes a long time to compute a trajectory that is provably globally optimal, although this does happen in finite time.

Finally, the convex hull formulation performed poorly in our examples. There is an empirical tradeoff between having tighter continuous relaxations and the number of continuous variables in the formulation. We hypothesize that the convex hull formulation will be most useful in cases when (1) the number of binary variables is large, or (2) the cost function is minimized near the boundary of the region.

8 Conclusion

We presented a novel mixed-integer programming-based method for control of nonlinear systems with a useful fragment of LTL that allows both finite- and infinite-horizon properties to be specified. Our method is efficient in the number of binary variables used to model the an LTL formula. Additionally, we showed the computational effectiveness of our approach on temporal logic motion planning examples.

Future work will consider reactive environments by including both continuous and discrete disturbances using a receding horizon control approach. Additionally, we will expand the space of tasks that can be specified by including additional temporal operators and timing constraints.

Acknowledgments The authors thank Matanya Horowitz, Scott Livingston, and Ufuk Topcu for helpful feedback. This work was supported by a NDSEG Fellowship and the Boeing Corporation.

References

1. Alur, R., Henzinger, T.A., Lafferriere, G., Pappas, G.J.: Discrete abstractions of hybrid systems. Proc. IEEE **88**(7), 971–984 (2000)
2. Baier, C., Katoen, J.P.: Principles of Model Checking. MIT Press, Cambridge (2008)
3. Belta, C., Habets, L.C.G.J.M.: Controlling of a class of nonlinear systems on rectangles. IEEE Trans. Autom. Control **51**, 1749–1759 (2006)
4. Bemporad, A., Morari, M.: Control of systems integrating logic, dynamics, and constraints. Automatica **35**, 407–427 (1999)
5. Betts, J.T.: Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, 2nd edition. SIAM (2000)
6. Bhatia, A., Maly, M.R., Kavraki, L.E., Vardi, M.Y.: Motion planning with complex goals. IEEE Robot. Autom. Mag. **18**, 55–64 (2011)
7. Blair, C.E., Jeroslow, R.G., Lowe, J.K.: Some results and experiments in programming techniques for propositional logic. Comput. Oper. Res. **13**, 633–645 (1986)
8. Earl, M.G., D'Andrea, R.: Iterative MILP methods for vehicle-control problems. IEEE Trans. Robot. **21**, 1158–1167 (2005)
9. Fainekos, G.E., Girard, A., Kress-Gazit, H., Pappas, G.J.: Temporal logic motion planning for dynamic robots. Automatica **45**, 343–352 (2009)
10. Habets, L., Collins, P.J., van Schuppen, J.H.: Reachability and control synthesis for piecewise-affine hybrid systems on simplices. IEEE Trans. Autom. Control **51**, 938–948 (2006)
11. Hooker, J.N., Fedjki, C.: Branch-and-cut solution of inference problems in propositional logic. Ann. Math. Artif. Intell. **1**, 123–139 (1990)
12. Jeroslow, R.G.: Representability in mixed integer programming, I: Characterization results. Disc. Appl. Math. **17**, 223–243 (1987)
13. Karaman, S., Frazzoli, E.: Linear temporal logic vehicle routing with applications to multi-UAV mission planning. Int. J. Robust Nonlinear Control **21**, 1372–1395 (2011)
14. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning with deterministic μ-calculus specifications. In: Proceedings of the American Control Conference (2012)
15. Karaman, S., Sanfelice, R.G., Frazzoli, E.: Optimal control of mixed logical dynamical systems with linear temporal logic specifications. In: Proceedings of the IEEE Conference on Decision and Control, pp. 2117–2122 (2008)
16. Kloetzer, M., Belta, C.: A fully automated framework for control of linear systems from temporal logic specifications. IEEE Trans. Autom. Control **53**(1), 287–297 (2008)

17. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
18. Löfberg, J.: YALMIP : A toolbox for modeling and optimization in MATLAB. In: Proceedings of the CACSD Conference. Taipei, Taiwan (2004). Software available at http://control.ee.ethz.ch/~joloef/yalmip.php
19. Mellinger, D., Kushleyev, A., Kumar, V.: Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In: Proceedings of the International Conference on Robotics and Automation (2012)
20. Richards, A., How, J.P.: Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In: American Control Conference (2002)
21. Smith, S.L., Tumova, J., Belta, C., Rus, D.: Optimal path planning for surveillance with temporal-logic constraints. Int. J. Robot. Res. **30**, 1695–1708 (2011)
22. Toth, P., Vigo, D. (eds.): The Vehicle Routing Problem. Philadelphia, PA: SIAM (2001)
23. User's Manual for CPLEX V12.2. IBM (2010)
24. Vitus, M.P., Pradeep, V., Hoffmann, J., Waslander, S.L., Tomlin, C.J.: Tunnel-MILP: path planning with sequential convex polytopes. In: Proceedings of the AIAA Guidance, Navigation, and Control Conference (2008)
25. Webb, D.J., van den Berg, J.: Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics. In: Proceedings of the IEEE International Conference on Robotics and Automation (2013)
26. Wolff, E.M., Murray, R.M.: Optimal control of mixed logical dynamical systems with long-term temporal logic specifications. Technical report, California Institute of Technology (2013). http://resolver.caltech.edu/CaltechCDSTR:2013.001
27. Wolff, E.M., Topcu, U., Murray, R.M.: Optimal control with weighted average costs and temporal logic specifications. In: Proceedings of the Robotics: Science and Systems (2012)
28. Wongpiromsarn, T., Topcu, U., Ozay, N., Xu, H., Murray, R.M.: TuLiP: A software toolbox for receding horizon temporal logic planning. In: Proceedings of the International Conference on Hybrid Systems: Computation and Control (2011). http://tulip-control.sf.net
29. Wongpiromsarn, T., Topcu, U., Murray, R.M.: Receding horizon temporal logic planning. IEEE Trans. Autom. Control (2012)

Extended LQR: Locally-Optimal Feedback Control for Systems with Non-Linear Dynamics and Non-Quadratic Cost

Jur van den Berg

Abstract We present Extended LQR, a novel approach for locally-optimal control for robots with non-linear dynamics and non-quadratic cost functions. Our formulation is conceptually different from existing approaches, and is based on the novel concept of *LQR-smoothing*, which is an LQR-analogue of Kalman smoothing. Our approach iteratively performs both a backward Extended LQR pass, which computes approximate *cost-to-go* functions, and a forward Extended LQR pass, which computes approximate *cost-to-come* functions. The states at which the *sum* of these functions is minimal provide an approximately optimal sequence of states for the control problem, and we use these points to linearize the dynamics and quadratize the cost functions in the subsequent iteration. Our results indicate that Extended LQR converges quickly and reliably to a locally-optimal solution of the non-linear, non-quadratic optimal control problem. In addition, we show that our approach is easily extended to include *temporal optimization*, in which the duration of a trajectory is optimized as part of the control problem. We demonstrate the potential of our approach on two illustrative non-linear control problems involving simulated and physical differential-drive robots and simulated quadrotor helicopters.

1 Introduction

Optimal control is an important problem in robotics. The problem is generally defined in terms of a description of the robot's dynamics and a control objective in the form of a cost function that is to be minimized, and the goal is to compute an optimal feedback control policy that tells the robot what control to apply given the state it is in. The linear-quadratic regulator (LQR) [4] provides a closed-form optimal solution in case the dynamics of the robot are linear and the cost function is quadratic. The linear-quadratic control problem is known to be closely related to the linear-Gaussian state estimation problem [20], for which the Kalman filter provides the optimal closed-form solution. Since the *Extended* Kalman filter provides a natural extension to

J. van den Berg (✉)
School of Computing, University of Utah, Salt Lake, UT, USA
e-mail: berg@cs.utah.edu

state estimation of non-linear systems [1], a natural question is whether LQR can be extended in a similar fashion to systems with general non-linear dynamics and non-quadratic cost functions. The main challenge here is that the LQR controller is derived using a recursion backward in time starting at the final time, and that the future states and control inputs of the robot are unknown at the time of designing the controller. So, choosing suitable points to linearize the dynamics and quadratize the cost functions about is non-trivial.

To address this, we propose *Extended LQR*, a novel approach to the non-linear, non-quadratic control problem that is based on the novel concept of *LQR-smoothing*. The LQR-smoother consists, analogous to the Kalman smoother [14], of a standard backward LQR pass that computes *cost-to-go* functions, and a *forward* LQR pass that computes *cost-to-come* functions. The sum of these functions give *total-cost* functions, and the states at which the total-cost functions are minimal provide an *optimal sequence of states* for the linear-quadratic control problem. To extend this to systems with non-linear dynamics and non-quadratic cost, Extended LQR iteratively performs a backward and a forward pass (analogous to the extended Kalman smoother [2]) to progressively obtain a better idea of the robot's future trajectory. The states at which the approximate total-cost functions are minimal are used to linearize the dynamics and quadratize the cost functions in each iteration, while the control policies computed in each pass provide control inputs to linearize and quadratize about. We will show that this procedure converges quickly and reliably to a locally-optimal solution to the non-linear, non-quadratic control problem.

There is a large body of literature on the non-linear, non-quadratic control problem, and many approaches based on linear-quadratic approximations have previously been proposed (as we discuss in detail in Sect. 2). Even though we will show that Extended LQR improves upon existing methods such as Iterative LQR (iLQR) [12], the main purpose of this paper is to introduce a conceptually novel approach to non-linear, non-quadratic control. Our formulation remains strictly within the LQR framework and does not use the duality between control and estimation [20] (in contrast to e.g. Approximate Inference Control (AICO) [22]), resulting in a conceptually intuitive approach that is easy to implement. Also, we will show that Extended LQR can naturally be applied to *temporal optimization* problems, in which the duration of the trajectory is to be optimized as part of the optimal control problem [15]. We have made source code of our approach publicly available at http://arl.cs.utah.edu/research/extendedlqr/.

We experimented with our approach on two illustrative non-linear control problems with and without temporal optimization, and compared performance to iLQR. Experiments involve both a physical and simulated differential-drive robot in a 2-D environment with obstacles (see Fig. 1), and a simulated quadrotor helicopter with a 12-D state space in 3-D environments with obstacles. Our results indicate that Extended LQR converges more quickly and reliably than iLQR even without providing it with an initial trajectory or implementing special convergence measures.

The remainder of this paper is organized as follows. We discuss related work in Sect. 2. In Sect. 3, we formally define the problem we address in this paper. In Sect. 4 we review the LQR controller and introduce the novel concept of linear-quadratic

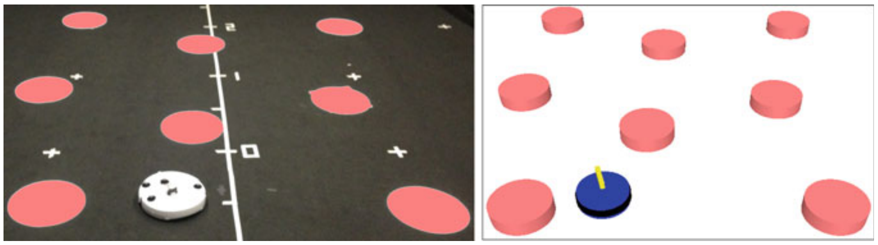


Fig. 1 A physical and simulated iRobot create navigating an environment with obstacles using extended LQR. See http://arl.cs.utah.edu/research/extendedlqr/ for videos of our experiments

smoothing. We use this in Sect. 5 to develop our Extended LQR approach, and show how it can be applied to temporal optimization problems in Sect. 6. We present experimental results in Sect. 7 and conclude in Sect. 8.

2 Related Work

Our approach is conceptually different from existing approaches to approximate optimal control such as Iterative LQR (iLQR) [12] and Differential Dynamic Programming (DDP) [9]. These approaches linearize the dynamics and quadratize the cost functions about a given (dynamically feasible) nominal trajectory and use LQR to compute a control policy. This control policy is then executed to compute a new nominal trajectory, and the procedure is repeated until convergence. These approaches require special measures such as line search [26] to ensure convergence, as the control policies may drive a new trajectory too "far away" from where the LQ-approximation is valid. Our approach, in contrast, relinearizes/requadratizes in both the backward pass and the forward pass, about a sequence of states and control inputs that is dynamically feasible only upon convergence. We will show that as a result, Extended LQR converges reliably without special convergence measures.

Sequential Quadratic Programming (SQP) methods [3, 13] also iteratively relinearize the dynamics and requadratize the cost functions, with the distinction that it formulates the problem in each iteration as a convex optimization problem that allows for the inclusion of convex constraints on the state and the control input. SQP approaches, however, typically do not compute a feedback control policy, but instead give an optimal open-loop sequence of control inputs for the control problem. The approaches of [17, 27] are closely related, and focus on *trajectory optimization* among obstacles for the specific class of robots with holonomic dynamics. Extended LQR can be used for trajectory optimization as well. In this case, like the mentioned approaches, the initial trajectory need not be dynamically feasible.

Approximate Inference Control (AICO) [22] uses the duality between control and estimation [20], and formulates the optimal control problem in terms of Kullback-Leibler divergence minimization [16]. Even though the derivation of Extended LQR differs considerably from that of AICO, ultimately the qualitative differences are subtle. A key technical difference is that AICO focuses on computing an optimal sequence of states and does not compute control policies during iterations. This limits AICO to cost functions that are explicitly quadratic in the control input, and it requires "local" iterations for each stage along the trajectory in addition to global forward and backward passes. Our approach computes control policies in each pass, which are used to select control inputs to linearize/quadratize about in the subsequent pass. Extended LQR is therefore applicable to general non-quadratic cost functions and does not use local iterations.

Extended LQR assumes deterministic dynamics, implicitly relying on the fact that the optimal LQR solution is independent from the process noise variance. Other approaches more directly account for stochastic dynamics: AICO lets the process noise variance interact with the cost matrices in the Ricatti equations, resulting in a form of risk-sensitive control [16, 25], and iLQR and DDP have been extended to explicitly take into account state and control input-dependent process noise [19, 21].

The problem of temporal optimization has previously been addressed in [15], which extends AICO with an EM-approach that alternatingly optimizes the trajectory and its duration. We present a more direct approach that includes the time-step in the state, and penalizes for the duration of the trajectory in the cost function. Extended LQR can then be applied as is to the augmented control problem.

One of the concepts underpinning our approach is *Forward LQR*. While forward Ricatti recursion has been explored in optimal control [6, 24], these works do not use it to compute *cost-to-come* functions as part of an *LQR-smoothing* framework. Our work also has similarities in spirit to [7], which employs both a forward and a backward Dijkstra's algorithm in discrete gridworlds.

3 Problem Definition

Let $\mathbb{X} \subset \mathbb{R}^n$ and $\mathbb{U} \subset \mathbb{R}^m$ be the state space and the control input space, respectively, of the robot, and let its deterministic discrete-time dynamics be given by:

$$\mathbf{x}_{t+1} = \mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_t), \tag{1}$$

with $\mathbf{g}_t \in \mathbb{X} \times \mathbb{U} \to \mathbb{X}$, where $\mathbf{x}_t \in \mathbb{X}$ and $\mathbf{u}_t \in \mathbb{U}$ denote the state and the control input of the robot at stage t. Let the control objective be defined as minimizing a cost function:

$$c_\ell(\mathbf{x}_\ell) + \textstyle\sum_{t=0}^{\ell-1} c_t(\mathbf{x}_t, \mathbf{u}_t), \tag{2}$$

for given horizon ℓ and local cost functions $c_\ell \in \mathbb{X} \to \mathbb{R}$ and $c_t \in \mathbb{X} \times \mathbb{U} \to \mathbb{R}$. Now, the optimal control problem is defined as finding a *control policy* $\pi_t \in \mathbb{X} \to \mathbb{U}$ for all $0 \leq t < \ell$, such that selecting controls $\mathbf{u}_t = \pi_t(\mathbf{x}_t)$ minimizes Eq. (2).

A general solution approach computes the *cost-to-go* functions $s_t \in \mathbb{X} \to \mathbb{R}$ and the optimal control policies π_t using a backward recursion procedure:

$$s_\ell(\mathbf{x}_\ell) = c_\ell(\mathbf{x}_\ell), \qquad s_t(\mathbf{x}_t) = \min_{\mathbf{u}_t}(c_t(\mathbf{x}_t, \mathbf{u}_t) + s_{t+1}(\mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_t))), \tag{3}$$

$$\pi_t(\mathbf{x}_t) = \operatorname{argmin}_{\mathbf{u}_t}(c_t(\mathbf{x}_t, \mathbf{u}_t) + s_{t+1}(\mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_t))). \tag{4}$$

The cost-to-go function $s_t(\mathbf{x}_t)$ gives the total future cost that will be accrued between stage t and stage ℓ by a minimal-cost sequence of states and control inputs that starts in $\mathbf{x}_t$ at stage t. In general, there is no explicit parametric expression for the cost-to-go functions s_t, except in the case where the dynamics are linear and the local cost functions are quadratic (as we will review in Sect. 4.1). The objective of this paper is to extend this approach to create (locally) optimal solutions to the general control problem with non-linear dynamics and non-quadratic cost.

Even though we address the discrete-time control problem, we assume the *continuous-time* dynamics are given:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)). \tag{5}$$

A continuous-time formulation is typically the most natural way to describe the dynamics of (non-linear) robotic systems, and it allows us to evaluate the discrete-time dynamics $\mathbf{g}_t$ (Eq. (1)), as well as its *inverse*, for any given time-step using e.g. Runge-Kutta integration. The *inverse* discrete-time dynamics are denoted by:

$$\mathbf{x}_t = \bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t), \tag{6}$$

where $\bar{\mathbf{g}}_t$ is defined such that $\mathbf{g}_t(\bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t), \mathbf{u}_t) = \mathbf{x}_{t+1}$ and, equivalently, $\bar{\mathbf{g}}_t(\mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_t), \mathbf{u}_t) = \mathbf{x}_t$. It is obtained by integrating Eq. (5) backward in time. We further assume that the local cost functions have positive-(semi)definite Hessians:

$$\frac{\partial^2 c_\ell}{\partial \mathbf{x}_\ell \partial \mathbf{x}_\ell} > 0, \qquad \frac{\partial^2 c_t}{\partial \mathbf{u}_t \partial \mathbf{u}_t} > 0, \qquad \frac{\partial^2 c_t}{\partial \left[\begin{smallmatrix}\mathbf{x}_t \\ \mathbf{u}_t\end{smallmatrix}\right] \partial \left[\begin{smallmatrix}\mathbf{x}_t \\ \mathbf{u}_t\end{smallmatrix}\right]} \geq 0. \tag{7}$$

4 Linear-Quadratic Control and Smoothing

We begin this section by reviewing LQR, which computes *cost-to-go* functions and provides a closed-form optimal solution to the linear-quadratic control problem. We then show how *cost-to-come* functions can be computed by a procedure similar to LQR, but one that runs *forward* in time, and introduce the concept of *linear-quadratic smoothing*, in which the cost-to-go and the cost-to-come functions are combined to create an LQR analogue of the Kalman smoother [14] that provides the optimal sequence of states for the linear-quadratic control problem. These concepts are at the foundation of our Extended LQR approach for non-linear, non-quadratic control, which we discuss in Sect. 5.

4.1 LQR Control

The linear-quadratic control problem is a special case of the general problem defined above for which the *LQR controller* provides a closed-form optimal solution [4]. In this case, the dynamics are linear and given by:

$$\mathbf{x}_{t+1} = \mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_t) = A_t\mathbf{x}_t + B_t\mathbf{u}_t + \mathbf{c}_t, \tag{8}$$

with $A_t \in \mathbb{R}^{n\times n}$, $B_t \in \mathbb{R}^{n\times m}$, and $\mathbf{c}_t \in \mathbb{R}^n$ given for all t. The local cost functions are quadratic, and given by:

$$c_\ell(\mathbf{x}_\ell) = \frac{1}{2}\mathbf{x}_\ell^T Q_\ell \mathbf{x}_\ell + \mathbf{x}_\ell^T \mathbf{q}_\ell, \quad c_t(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{2}\begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T \begin{bmatrix}Q_t & P_t^T\\ P_t & R_t\end{bmatrix}\begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix} + \begin{bmatrix}\mathbf{x}_t\\ \mathbf{u}_t\end{bmatrix}^T\begin{bmatrix}\mathbf{q}_t\\ \mathbf{r}_t\end{bmatrix}, \tag{9}$$

where $Q_t \in \mathbb{R}^{n\times n}$, $R_t \in \mathbb{R}^{m\times m}$, $P_t \in \mathbb{R}^{m\times n}$, $\mathbf{q}_t \in \mathbb{R}^n$, and $\mathbf{r}_t \in \mathbb{R}^m$ are given for all t. Matrices $Q_\ell > 0$ and $R_t > 0$ are positive-definite, and $\left[\begin{smallmatrix}Q_t & P_t^T\\ P_t & R_t\end{smallmatrix}\right] \geq 0$ is positive-semidefinite, in accordance with Eq. (7).

For this control problem, the cost-to-go functions s_t have the following explicit quadratic formulation:

$$s_t(\mathbf{x}_t) = \tfrac{1}{2}\mathbf{x}_t^T S_t \mathbf{x}_t + \mathbf{x}_t^T \mathbf{s}_t + k, \tag{10}$$

where k is a constant, and $S_t \in \mathbb{R}^{n\times n} > 0$ and $\mathbf{s}_t \in \mathbb{R}^n$ are computed using backward recursion. For final stage ℓ, we have $S_\ell = Q_\ell$ and $\mathbf{s}_\ell = \mathbf{q}_\ell$, and for stage $\ell > t \geq 0$:

$$S_t = D_t - C_t^T E_t^{-1} C_t, \qquad \mathbf{s}_t = \mathbf{d}_t - C_t^T E_t^{-1}\mathbf{e}_t, \tag{11}$$

where:

$$C_t = P_t + B_t^T S_{t+1} A_t, \quad D_t = Q_t + A_t^T S_{t+1} A_t, \quad E_t = R_t + B_t^T S_{t+1} B_t,$$
$$\mathbf{d}_t = \mathbf{q}_t + A_t^T \mathbf{s}_{t+1} + A_t^T S_{t+1}\mathbf{c}_t, \quad \mathbf{e}_t = \mathbf{r}_t + B_t^T \mathbf{s}_{t+1} + B_t^T S_{t+1}\mathbf{c}_t.$$

The optimal feedback control policies π_t have an explicit linear formulation:

$$\pi_t(\mathbf{x}_t) = L_t\mathbf{x}_t + \mathbf{l}_t, \qquad L_t = -E_t^{-1}C_t, \qquad \mathbf{l}_t = -E_t^{-1}\mathbf{e}_t, \tag{12}$$

4.2 Cost-to-Come Functions and Forward LQR

As mentioned above, the cost-to-go functions $s_t(\mathbf{x}_t)$ give the total future cost that will be accrued between stage t and stage ℓ (*including* the cost incurred at stage t) by a minimal-cost sequence of states and control inputs that starts in $\mathbf{x}_t$ at stage t. We use the similar concept of *cost-to-come* functions [11], denoted $\bar{s}_t(\mathbf{x}_t)$, which give the total *past* cost that was accrued between stage 0 and stage t (*excluding* the cost incurred at stage t) by a minimal-cost sequence of states and controls that arrives in $\mathbf{x}_t$ at stage t. Given the *inverse* dynamics (see Eq. (6)), the cost-to-come functions are generally defined by the following *forward* recursion procedure:

$$\bar{s}_0(\mathbf{x}_0) = 0, \quad \bar{s}_{t+1}(\mathbf{x}_{t+1}) = \min_{\mathbf{u}_t}(c_t(\bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t), \mathbf{u}_t) + \bar{s}_t(\bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t))), \tag{13}$$

$$\bar{\pi}_t(\mathbf{x}_{t+1}) = \operatorname{argmin}_{\mathbf{u}_t}(c_t(\bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t), \mathbf{u}_t) + \bar{s}_t(\bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t))), \tag{14}$$

Here, $\bar{\pi}_t$ is an *inverse* control policy that given a state $\mathbf{x}_{t+1}$ at stage $t+1$ computes the control input $\mathbf{u}_t = \bar{\pi}_t(\mathbf{x}_{t+1})$ that was applied at stage t in order to arrive at $\mathbf{x}_{t+1}$ with minimal cost-to-come.

If the dynamics are linear and given by Eq. (8), and the local cost functions are quadratic and given by Eq. (9), the cost-to-come functions $\bar{s}_t$ have an explicit quadratic formulation, similar to the cost-to-go functions in LQR:

$$\bar{s}_t(\mathbf{x}_t) = \tfrac{1}{2}\mathbf{x}_t^T \bar{S}_t \mathbf{x}_t + \mathbf{x}_t^T \bar{\mathbf{s}}_t + \bar{k}, \tag{15}$$

where $\bar{S}_t \in \mathbb{R}^{n \times n} \geq 0$ and $\bar{\mathbf{s}}_t \in \mathbb{R}^n$. The recursive update equations for $\bar{S}_t$ and $\bar{\mathbf{s}}_t$ run forward in time, and can be derived in a similar fashion as those of standard LQR, given that the linear dynamics are expressed in their inverse form:

$$\mathbf{x}_t = \bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t) = \bar{A}_t \mathbf{x}_{t+1} + \bar{B}_t \mathbf{u}_t + \bar{\mathbf{c}}_t, \tag{16}$$

with $\bar{A}_t = A_t^{-1}$, $\bar{B}_t = -A_t^{-1}B_t$, and $\bar{\mathbf{c}}_t = -A_t^{-1}\mathbf{c}_t$ (as follows from solving Eq. (8) for $\mathbf{x}_t$). Then, we initially have $\bar{S}_0 = 0$ and $\bar{\mathbf{s}}_0 = \mathbf{0}$, and for stage $0 \leq t < \ell$:

$$\bar{S}_{t+1} = \bar{D}_t - \bar{C}_t^T \bar{E}_t^{-1} \bar{C}_t, \qquad \bar{\mathbf{s}}_{t+1} = \bar{\mathbf{d}}_t - \bar{C}_t^T \bar{E}_t^{-1} \bar{\mathbf{e}}_t, \tag{17}$$

where

$$\bar{C}_t = \bar{B}_t^T(\bar{S}_t + Q_t)\bar{A}_t + P_t\bar{A}_t, \;\; \bar{D}_t = \bar{A}_t^T(\bar{S}_t + Q_t)\bar{A}_t, \;\; \bar{E}_t = \bar{B}_t^T(\bar{S}_t + Q_t)\bar{B}_t + R_t + P_t\bar{B}_t + \bar{B}_t^T P_t^T,$$
$$\bar{\mathbf{d}}_t = \bar{A}_t^T(\bar{\mathbf{s}}_t + \mathbf{q}_t) + \bar{A}_t^T(\bar{S}_t + Q_t)\bar{\mathbf{c}}_t, \quad \bar{\mathbf{e}}_t = \mathbf{r}_t + P_t\bar{\mathbf{c}}_t + \bar{B}_t^T(\bar{\mathbf{s}}_t + \mathbf{q}_t) + \bar{B}_t^T(\bar{S}_t + Q_t)\bar{\mathbf{c}}_t.$$

The inverse control policies $\bar{\pi}_t$ have, similar to the control policies in LQR, an explicit linear formulation:

$$\bar{\pi}_t(\mathbf{x}_{t+1}) = \bar{L}_t \mathbf{x}_{t+1} + \bar{\mathbf{l}}_t, \qquad \bar{L}_t = -\bar{E}_t^{-1}\bar{C}_t, \qquad \bar{\mathbf{l}}_t = -\bar{E}_t^{-1}\bar{\mathbf{e}}_t. \tag{18}$$

In the remainder of this paper, we refer to the above procedure as *Forward LQR*.

4.3 Linear-Quadratic Smoothing

Performing both the standard LQR procedure (following Sect. 4.1) and the forward LQR procedure (following Sect. 4.2) for a given linear-quadratic control problem gives both the cost-to-go functions s_t and cost-to-come functions $\bar{s}_t$. The sum of the cost-to-go $s_t(\mathbf{x}_t)$ and the cost-to-come $\bar{s}_t(\mathbf{x}_t)$ gives the total (past and future) cost $\hat{s}_t(\mathbf{x}_t)$ accrued between stage 0 and stage ℓ by a minimal-cost sequence of states and controls that visits $\mathbf{x}_t$ at stage t:

$$\hat{s}_t(\mathbf{x}_t) = s_t(\mathbf{x}_t) + \bar{s}_t(\mathbf{x}_t) = \tfrac{1}{2}\mathbf{x}_t^T(S_t + \bar{S}_t)\mathbf{x}_t + \mathbf{x}_t^T(\mathbf{s}_t + \bar{\mathbf{s}}_t) + \hat{k}. \tag{19}$$

Let $\hat{\mathbf{x}}_t$ denote the state at stage t for which the total-cost function $\hat{s}_t$ is minimal:

$$\hat{\mathbf{x}}_t = \text{argmin}_{\mathbf{x}_t}\hat{s}_t(\mathbf{x}_t) = -(S_t + \bar{S}_t)^{-1}(\mathbf{s}_t + \bar{\mathbf{s}}_t). \tag{20}$$

(Note that this inverse exists since $S_t > 0$.) Then, the sequence of states $\{\hat{\mathbf{x}}_0, \dots, \hat{\mathbf{x}}_\ell\}$ is the minimum-cost sequence of states for the given linear-quadratic control problem. The associated controls are given by the (inverse) control policies:

$$\hat{\mathbf{x}}_{t+1} = \mathbf{g}_t(\hat{\mathbf{x}}_t, \pi_t(\hat{\mathbf{x}}_t)), \qquad \hat{\mathbf{x}}_t = \bar{\mathbf{g}}_t(\hat{\mathbf{x}}_{t+1}, \bar{\pi}_t(\hat{\mathbf{x}}_{t+1})). \tag{21}$$

Note also that $\pi_t(\hat{\mathbf{x}}_t) = \bar{\pi}_t(\hat{\mathbf{x}}_{t+1})$.

The above can be seen as an LQR-analogue of the *Kalman smoother* [14]. The Kalman smoother performs both a forward and a backward Kalman filter to compute the posterior distributions of the state given all (past and future) observations. The mean of these distributions gives the maximum-likelihood (and maximum-a-posteriori) sequence of states. In fact, it can be shown that the above procedure is the exact dual of the Kalman smoother. We use the insights of Eqs. (20) and (21) below to develop Extended LQR.

5 Extended LQR

In this section we extend LQR and Forward LQR to the case of general non-linear dynamics and non-quadratic local cost functions, in a similar way as how the extended Kalman filter extends the Kalman filter to non-linear systems. In our (forward) Extended LQR approach, we use the same equations as in standard LQR and forward LQR to update the cost-to-go and cost-to-come functions, respectively. The challenge is how to linearize the dynamics and quadratize the local cost functions in each cycle of the recursion, since there are a-priori no obvious candidates for the state and control input to linearize/quadratize about.

The idea of our approach to tackle this problem is to iteratively perform backward and forward Extended LQR passes to compute increasingly better approximations of the total-cost functions $\hat{s}_t$, analogous to the (iterated) extended Kalman smoother [2]. In each pass, we use the states at which the current approximations of the total-cost functions are minimal to linearize and quadratize, and use the control policies from the preceding pass to select control inputs to linearize/quadratize about.

The iteration continues until convergence, i.e. when the minimum-total-cost states no longer change. These states then provide a locally-optimal sequence of states for the non-linear, non-quadratic control problem, and the control policies computed by the last backward pass provide an approximately optimal feedback control policy. The iteration starts with a backward pass, which we discuss first. We then discuss the forward pass, and discuss convergence properties.

5.1 Backward Extended LQR

We assume during the backward pass that the cost-to-come functions, as defined by $\bar{S}_t$ and $\bar{\mathbf{s}}_t$, as well as the inverse control policies $\bar{\pi}_t$ are available for all t from the preceding forward pass. Also, an initial quadratization point $\hat{\mathbf{x}}_\ell$ for stage ℓ is available from the forward pass. If this is the first backward pass (and no forward pass preceded it), one can assume $\bar{S}_t = 0, \bar{\mathbf{s}}_t = \mathbf{0}, \bar{\pi}_t(\mathbf{x}_{t+1}) = \mathbf{0}$, and $\hat{\mathbf{x}}_\ell = \mathbf{0}$, or alternatively set these values in accordance with any available prior information (such as a given initial trajectory).

The backward pass closely follows the standard LQR approach of Sect. 4.1. We initialize the backward pass by setting $S_\ell = Q_\ell$ and $\mathbf{s}_\ell = \mathbf{q}_\ell$ as in standard LQR, where matrix Q_ℓ and vector $\mathbf{q}_\ell$ are obtained by quadratizing the final cost function $c_\ell(\mathbf{x}_\ell)$ about the given point $\hat{\mathbf{x}}_\ell$, which puts it in the form of Eq. (9), with:

$$Q_\ell = \frac{\partial^2 c_\ell}{\partial \mathbf{x}_\ell \partial \mathbf{x}_\ell}(\hat{\mathbf{x}}_\ell), \qquad \mathbf{q}_\ell = \frac{\partial c_\ell}{\partial \mathbf{x}_\ell}(\hat{\mathbf{x}}_\ell) - Q_\ell \hat{\mathbf{x}}_\ell. \tag{22}$$

We then proceed by performing recursive updates of the cost-to-go function for $\ell > t \geq 0$. In each step of the recursion, we compute S_t and $\mathbf{s}_t$ given S_{t+1} and $\mathbf{s}_{t+1}$, starting with $t = \ell - 1$. For this, we use Eq. (11) as in standard LQR, where matrices A_t, B_t, and $\mathbf{c}_t$ are obtained by linearizing the dynamics $\mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_t)$, and matrices P_t, Q_t, R_t, $\mathbf{q}_t$, and $\mathbf{r}_t$ are obtained by quadratizing the local cost function $c_t(\mathbf{x}_t, \mathbf{u}_t)$.

The challenge is to choose a state $\hat{\mathbf{x}}_t$ and control input $\hat{\mathbf{u}}_t$ about which to linearize the dynamics and quadratize the local cost function. In principle, we would like to set $\hat{\mathbf{x}}_t$ to the minimum-total-cost state at stage t, as defined by Eq. (20). However, we do not yet have an estimate of the total-cost function at stage t, since S_t and $\mathbf{s}_t$ are not yet computed (at least not in the current backward pass). We do have a current-best estimate of the state with minimal total-cost at stage $t+1$, though:

$$\hat{\mathbf{x}}_{t+1} = -(S_{t+1} + \bar{S}_{t+1})^{-1}(\mathbf{s}_{t+1} + \bar{\mathbf{s}}_{t+1}). \tag{23}$$

We can then set the state $\hat{\mathbf{x}}_t$ and the control input $\hat{\mathbf{u}}_t$ about which to linearize and quadratize in accordance with Eq. (21), using the inverse dynamics and the inverse control policy $\bar{\pi}_t$ available from the preceding forward pass:

$$\hat{\mathbf{u}}_t = \bar{\pi}_t(\hat{\mathbf{x}}_{t+1}), \qquad \hat{\mathbf{x}}_t = \bar{\mathbf{g}}_t(\hat{\mathbf{x}}_{t+1}, \hat{\mathbf{u}}_t). \tag{24}$$

Linearizing the (forward) dynamics $\mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_t)$ about $\hat{\mathbf{x}}_t$ and $\hat{\mathbf{u}}_t$ then puts it in the form of Eq. (8), with:

$$A_t = \frac{\partial \mathbf{g}_t}{\partial \mathbf{x}_t}(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t), \qquad B_t = \frac{\partial \mathbf{g}_t}{\partial \mathbf{u}_t}(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t), \qquad \mathbf{c}_t = \hat{\mathbf{x}}_{t+1} - A_t\hat{\mathbf{x}}_t - B_t\hat{\mathbf{u}}_t, \tag{25}$$

and quadratizing the local cost function $c_t(\mathbf{x}_t, \mathbf{u}_t)$ about $\hat{\mathbf{x}}_t$ and $\hat{\mathbf{u}}_t$ puts it in the form of Eq. (9), with:

$$\begin{bmatrix} Q_t & P_t^T \\ P_t & R_t \end{bmatrix} = \frac{\partial^2 c_t}{\partial\left[\begin{smallmatrix}\mathbf{x}_t \\ \mathbf{u}_t\end{smallmatrix}\right]\partial\left[\begin{smallmatrix}\mathbf{x}_t \\ \mathbf{u}_t\end{smallmatrix}\right]}(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t), \qquad \begin{bmatrix} \mathbf{q}_t \\ \mathbf{r}_t \end{bmatrix} = \frac{\partial c_t}{\partial\left[\begin{smallmatrix}\mathbf{x}_t \\ \mathbf{u}_t\end{smallmatrix}\right]}(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t) - \begin{bmatrix} Q_t & P_t^T \\ P_t & R_t \end{bmatrix}\begin{bmatrix} \hat{\mathbf{x}}_t \\ \hat{\mathbf{u}}_t \end{bmatrix}. \tag{26}$$

Given these matrices and vectors we can compute S_t and $\mathbf{s}_t$ using Eq. (11), and the control policy π_t using Eq. (12). This recursion is repeated until $t = 0$.

5.2 Forward Extended LQR

We assume during the forward pass that the cost-to-go functions, as defined by S_t and $\mathbf{s}_t$, as well as the control policies π_t are available for all t from the preceding backward pass. The forward pass closely follows the Forward LQR approach of Sect. 4.2. We initialize the forward pass by setting $\bar{S}_0 = 0$ and $\bar{\mathbf{s}}_0 = \mathbf{0}$, and then proceed by performing recursive updates of the cost-to-come functions for $0 \leq t < \ell$.

In each step of the recursion, we compute $\bar{S}_{t+1}$ and $\bar{\mathbf{s}}_{t+1}$ given $\bar{S}_t$ and $\bar{\mathbf{s}}_t$, starting with $t = 0$. For this, we use Eq. (17), where matrices and vector $\bar{A}_t$, $\bar{B}_t$, and $\bar{\mathbf{c}}_t$ are obtained by linearizing the inverse dynamics $\bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t)$, and matrices and vectors P_t, Q_t, R_t, $\mathbf{q}_t$, and $\mathbf{r}_t$ are obtained by quadratizing the local cost function $c_t(\mathbf{x}_t, \mathbf{u}_t)$.

We select linearization and quadratization points in a similar manner as in the backward pass. We do not yet have an estimate of the minimum-total-cost state at stage $t + 1$ as defined by Eq. (20) to linearize the inverse dynamics about, since $\bar{S}_{t+1}$ and $\bar{\mathbf{s}}_{t+1}$ are not yet computed, but we do have a current-best estimate of the state with minimal total-cost at stage t:

$$\hat{\mathbf{x}}_t = -(S_t + \bar{S}_t)^{-1}(\mathbf{s}_t + \bar{\mathbf{s}}_t). \tag{27}$$

Using the forward dynamics and the control policy π_t available from the backward pass, we then set $\hat{\mathbf{x}}_{t+1}$ and $\hat{\mathbf{u}}_t$ in accordance with Eq. (21):

$$\hat{\mathbf{u}}_t = \pi_t(\hat{\mathbf{x}}_t), \qquad \hat{\mathbf{x}}_{t+1} = \mathbf{g}_t(\hat{\mathbf{x}}_t, \hat{\mathbf{u}}_t). \tag{28}$$

Linearizing the inverse dynamics $\bar{\mathbf{g}}_t(\mathbf{x}_{t+1}, \mathbf{u}_t)$ about $\hat{\mathbf{x}}_{t+1}$ and $\hat{\mathbf{u}}_t$ and quadratizing the local cost function about $\hat{\mathbf{x}}_t$ and $\hat{\mathbf{u}}_t$ then gives:

$$\bar{A}_t = \frac{\partial \bar{\mathbf{g}}_t}{\partial \mathbf{x}_{t+1}}(\hat{\mathbf{x}}_{t+1}, \hat{\mathbf{u}}_t), \quad \bar{B}_t = \frac{\partial \bar{\mathbf{g}}_t}{\partial \mathbf{u}_t}(\hat{\mathbf{x}}_{t+1}, \hat{\mathbf{u}}_t), \quad \bar{\mathbf{c}}_t = \hat{\mathbf{x}}_t - \bar{A}_t\hat{\mathbf{x}}_{t+1} - \bar{B}_t\hat{\mathbf{u}}_t, \tag{29}$$

and P_t, Q_t, R_t, $\mathbf{q}_t$, and $\mathbf{r}_t$ as in Eq. (26). We can then compute $\bar{S}_{t+1}$ and $\bar{\mathbf{s}}_{t+1}$ using Eq. (17), and the inverse control policy $\bar{\pi}_t$ using Eq. (18). This recursion is repeated until $t = \ell - 1$. Finally, we set an initial quadratization point for stage ℓ for the subsequent backward pass:

$$\hat{\mathbf{x}}_\ell = -(S_\ell + \bar{S}_\ell)^{-1}(\mathbf{s}_\ell + \bar{\mathbf{s}}_\ell). \tag{30}$$

5.3 Convergence Properties

Upon convergence, the properties of Eq. (21) hold, and the minimum-total-cost states $\hat{\mathbf{x}}_t$ form an exact *locally-optimal* sequence of states for the non-linear, non-quadratic control problem. The (linear) control policies π_t from the last backward pass then provide a first-order Taylor approximation about the minimum-total-cost states $\hat{\mathbf{x}}_t$ of the true locally-optimal control policy, and the computed (quadratic) cost-to-go functions provide a second-order Taylor approximation of the true locally-optimal cost-to-go functions. Using the analogy with the iterated Kalman smoother [2], our approach can be shown to perform Gauss-Newton updates towards a local optimum and thus should exhibit a rate of convergence approaching second-order [5]. The running-time per iteration is $O(\ell n^3)$.

Before convergence is achieved, the sequence of minimum-total-cost states is not necessarily consistent with the non-linear dynamics. Also, as the minimal-total-cost states are in a way an "average" between the minimal-cost-to-go states and the minimal-cost-to-come states, of which only one is updated in each pass, the minimal-total-cost states smoothly evolve. This is why our approach can converge reliably without implementing convergence measures such as line search.

6 Temporal Optimization

In many cases it is desirable to optimize the duration of the trajectory as part of the optimal control problem [15]. Such an objective naturally fits within our framework, by keeping the number of steps ℓ constant, but making the *time-step* τ part of the state, and penalizing linearly for the duration of the trajectory in the cost function.

Since the time-step must be strictly positive, we make its logarithm $\lambda = \log \tau$ part of the state. The augmented state $\tilde{\mathbf{x}}$ and the augmented (continuous-time) dynamics $\dot{\tilde{\mathbf{x}}}(t) = \tilde{\mathbf{f}}(t, \tilde{\mathbf{x}}(t), \mathbf{u}(t))$ are then defined as:

$$\tilde{\mathbf{x}} = [\mathbf{x}^T, \lambda]^T, \qquad \dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)), \qquad \dot{\lambda}(t) = 0, \tag{31}$$

where we use a time-step of $\tau = \exp \lambda_t$ when evaluating the discrete-time dynamics $\mathbf{g}_t(\tilde{\mathbf{x}}_t, \mathbf{u}_t)$ (or its inverse). In addition, we create augmented cost functions $\tilde{c}_t$ that add a term $\exp \lambda_t$ to the local cost functions c_t of each stage $0 < t \leq \ell$ (excluding 0), such that in total the duration of the trajectory is linearly penalized:

$$\tilde{c}_\ell(\tilde{\mathbf{x}}_\ell) = \exp \lambda_\ell + c_\ell(\mathbf{x}_\ell), \quad \tilde{c}_t(\tilde{\mathbf{x}}_t, \mathbf{u}_t) = \begin{cases} c_t(\mathbf{x}_t, \mathbf{u}_t) & \text{if } t = 0, \\ \exp \lambda_t + c_t(\mathbf{x}_t, \mathbf{u}_t) & \text{if } 0 < t < \ell. \end{cases} \tag{32}$$

Since the second derivative of $\exp \lambda_t$ with respect to λ_t is positive, the augmented cost functions obey the requirements of Eq. (7).

With these definitions of augmented state, dynamics, and cost functions, we can use Extended LQR as is to solve control problems with temporal optimization. Upon convergence, the value of λ_t in the minimum-total-cost states $\hat{\tilde{\mathbf{x}}}_t$ is the same for all t, as follows from Eq. (21) and the fact that $\dot{\lambda}(t) = 0$ in the augmented dynamics.

7 Experiments

We experimented with Extended LQR on two systems; an iRobot Create differential-drive robot, which we use mainly for illustrative purposes to gain insight into the working of our approach and temporal optimization, and a simulated quadrotor helicopter, on which we perform extensive quantitative analysis and performance comparison with Iterative LQR (iLQR).

7.1 IRobot Create Differential-Drive Robot

Our first experiment involves a simulated and physical iRobot Create differential-drive robot. Its state $\mathbf{x} = [p_x, p_y, \theta]^T$ is described by its two-dimensional position

(p_x, p_y) (m) and orientation θ (rad), and its control input $\mathbf{u} = [v_\ell, v_r]^T$ consists of the speeds (m/s) of the left and right wheel, respectively. The dynamics $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ of the robot are non-linear and given by:

$$\dot{p}_x = \tfrac{1}{2}(v_\ell + v_r)\cos\theta, \qquad \dot{p}_y = \tfrac{1}{2}(v_\ell + v_r)\sin\theta, \qquad \dot{\theta} = (v_r - v_\ell)/w,$$

where $w = 0.258$ m is the distance between the wheels of the iRobot Create.

We use the following cost functions in our experiments:

$$\begin{aligned} c_\ell(\mathbf{x}) &= \tfrac{1}{2}(\mathbf{x} - \mathbf{x}^\star_\ell)^T Q(\mathbf{x} - \mathbf{x}^\star_\ell), \\ c_0(\mathbf{x}, \mathbf{u}) &= \tfrac{1}{2}(\mathbf{x} - \mathbf{x}^\star_0)^T Q(\mathbf{x} - \mathbf{x}^\star_0) + \tfrac{1}{2}(\mathbf{u} - \mathbf{u}^\star)^T R(\mathbf{u} - \mathbf{u}^\star), \\ c_t(\mathbf{x}, \mathbf{u}) &= \tfrac{1}{2}(\mathbf{u} - \mathbf{u}^\star)^T R(\mathbf{u} - \mathbf{u}^\star) + q \textstyle\sum_i \exp(-d_i(\mathbf{x})), \end{aligned}$$

for $0 < t < \ell$, where $\mathbf{x}^\star_\ell$ is the target state, $\mathbf{x}^\star_0$ the initial state, and $\mathbf{u}^\star$ the nominal control input, which we set to $(0.25, 0.25)$ m/s for the iRobot Create (which has maximum wheel speeds of $v_\ell, v_r \in [-0.5, 0.5]$ m/s). Matrices Q and R and scalar q are positive weight factors. The function $d_i(\mathbf{x})$ gives the (signed) distance between the robot configured at $\mathbf{x}$ and the i'th obstacle in the environment. The term $q \sum_i \exp(-d_i(\mathbf{x}))$ makes this local cost function non-quadratic in the state $\mathbf{x}$. Since its Hessian is not always positive-semidefinite, counter to the requirement of Eq. (7), it is *regularized* when quadratizing the cost function. That is, its eigendecomposition is computed and its negative eigenvalues are set to zero [8].

We experimented in the environment of Fig. 2, which measures 4 m by 6 m. The obstacles each have a radius of 0.2 m, and the iRobot Create has a physical radius of 0.17 m. The initial state was set to $\mathbf{x}^\star_0 = (0, -2.5, \pi)$ and the target state to $\mathbf{x}^\star_\ell = (0, 2.5, \pi)$. We ran our approach for various fixed time-steps τ and temporal optimization, with a fixed number of steps $\ell = 150$. Extended LQR was not seeded

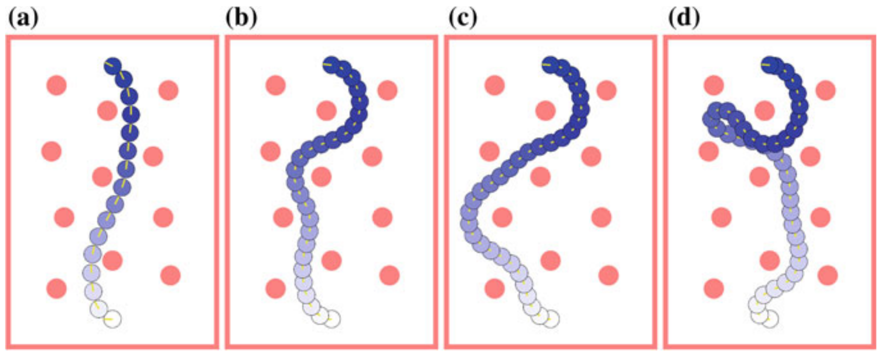


Fig. 2 Trajectories resulting from the differential-drive robot experiments. The robot is shown every second. See http://arl.cs.utah.edu/research/extendedlqr/ for videos of our experiments. **a** $\tau = \frac{1}{10}$ s. **b** $\tau = \frac{1}{6}$ s, $\tau \to 0.168$ s. **c** $\tau = \frac{1}{5}$ s. **d** $\tau = \frac{1}{4}$ s

Table 1 **a** Results of the simulations of Sect. 7.1 with a differential-drive robot. **b** Results of the simulations of Sect. 7.2 with a quadrotor helicopter, averaged over 100 queries

a Differential-Drive Robot ($n = 3$, $\ell = 150$)				**b** Quadrotor Helicopter ($n = 12$, $\ell = 150$)				
Extended LQR				Extended LQR			Iterative LQR	
time-step (s)	#iters	time (s)	speed (m/s)	time-step (s)	#iters	time (s)	#iters	time (s)
$\tau = 1/10$	8	0.014	0.359	$\tau = 1/40$	9.39	0.33	27.7	0.48
$\tau = 1/6$	7	0.012	0.251	$\tau = 1/30$	12.9	0.46	37.9	0.66
$\tau = 1/5$	8	0.014	0.238	$\tau = 1/20$	17.61	0.63	50.08	0.86
$\tau = 1/4$	14	0.027	0.242	$\tau = 1/10$	24.22	0.87	92.66	1.59
$\tau \to 0.168$	7	0.016	0.250	$\tau \to 0.070$	20.85	0.81	N/A	

with an initial trajectory, and we let the algorithm run until the relative improvement dropped below 10^{-4}. Figure 2 shows the resulting trajectories. Table 1a gives quantitative results, where the second column gives the number of iterations until convergence, the third column the computation time required (for a C++ implementation on an Intel i5 1.60 GHz with 4 GB RAM), and the fourth column gives the average speed of the robot along the trajectory.

To appreciate these results, it is important to note that the nominal control input $\mathbf{u}^\star$ was set to 0.25 m/s. Hence, the robot is penalized for driving either faster or slower. For a time-step of $\tau = \frac{1}{10}$ s, the robot only has $\ell\tau = 15$ s to reach the goal; it therefore chooses a short trajectory that comes close to the obstacles with a speed far above the nominal. For $\tau = \frac{1}{6}$ s, the robot has 25 s to reach the goal, and we see a speed close to the nominal and a trajectory that takes a safer margin with respect to the obstacles. For $\tau = \frac{1}{4}$ s, on the other hand, the robot has as much as 37.5 s to reach the goal, while it still wants to keep driving at a speed of 0.25 m/s. The result is that the robot takes a long detour, while maintaining a speed slightly below the nominal.

The above results clearly show why temporal optimization can be useful. The results of temporal optimization are shown the bottom row of Table 1a. In this case, the resulting trajectory is visually indistinguishable from the trajectory resulting from $\tau = 1/6$ s (Fig. 2b) and the resulting time-step (0.168 s) is only slightly longer to allow for an average speed of exactly the nominal. The results suggest that including temporal optimization does not significantly affect the convergence rate of Extended LQR. The computation time slightly increases, which reflects the higher dimension of the state after adding the time-step.

We also successfully executed the control policy resulting from these experiments on a physical iRobot Create (see Fig. 1) in a laboratory with motion capture for state estimation. This shows that the control policy can account for disturbances to which a real robot is inherently subject, despite the fact that Extended LQR does not specifically take into account motion uncertainty.

7.2 Quadrotor Helicopter in 3-D Environment

Our second experiment considers a simulated quadrotor helicopter, modeled after the Ascending Technologies' ResearchPilot. Its state $\mathbf{x} = [\mathbf{p}^T, \mathbf{v}^T, \mathbf{r}^T, \mathbf{w}^T]^T$ is 12-dimensional, and consists of its position $\mathbf{p}$ (m), velocity $\mathbf{v}$ (m/s), orientation $\mathbf{r}$ (rotation about axis $\mathbf{r}$ by angle $\|\mathbf{r}\|$ (rad)), and angular velocity $\mathbf{w}$ (rad/s). Its control input $\mathbf{u} = [u_1, u_2, u_3, u_4]^T$ (N) consists of the forces exerted by each of the four rotors. The dynamics are non-linear, and given by:

$$\begin{aligned}
\dot{\mathbf{p}} &= \mathbf{v},\\
\dot{\mathbf{v}} &= -g\mathbf{e}_3 + ((u_1 + u_2 + u_3 + u_4)\exp([\mathbf{r}])\mathbf{e}_3 - k_v\mathbf{v})/m,\\
\dot{\mathbf{r}} &= \mathbf{w} + \tfrac{1}{2}[\mathbf{r}]\mathbf{w} + (1 - \tfrac{1}{2}\|\mathbf{r}\|/\tan(\tfrac{1}{2}\|\mathbf{r}\|))[\mathbf{r}]^2\mathbf{w}/\|\mathbf{r}\|^2,\\
\dot{\mathbf{w}} &= J^{-1}(\rho(u_2 - u_4)\mathbf{e}_1 + \rho(u_3 - u_1)\mathbf{e}_2 + k_m(u_1 - u_2 + u_3 - u_4)\mathbf{e}_3 - [\mathbf{w}]J\mathbf{w}),
\end{aligned}$$

where $\mathbf{e}_i$ are the standard basis vectors, $g = 9.8\,\mathrm{m/s}^2$ is the gravity, $k_v = 0.15$ is a constant relating the velocity to an opposite force (caused by rotor drag and induced inflow), $m = 0.5\,\mathrm{kg}$ is the mass, $J = 0.05I$ (kg m^2) is the moment of inertia matrix, $\rho = 0.17\,\mathrm{m}$ is the distance between the center of mass and the center of the rotors, and $k_m = 0.025$ is a constant relating the force of a rotor to its torque. The notation $[\mathbf{a}]$ refers to the skew-symmetric cross-product matrix of $\mathbf{a}$. We used the same local cost functions as in Sect. 7.1, where the nominal control input $\mathbf{u}^\star$ was set to $\frac{1}{4}mg$ N for each of the rotors, which is the force required to let the quadrotor hover.

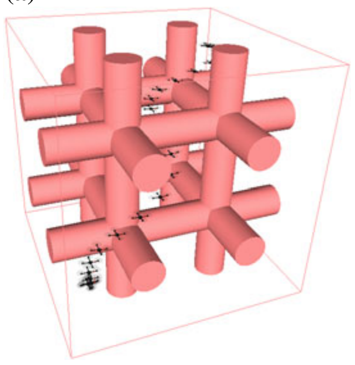
We experimented in the 3-D environment of Fig. 3a measuring 6 m by 6 m by 6 m for varying initial and target states and time-steps, and compared the performance of Extended LQR to iLQR. Neither algorithm was initialized with a given trajectory, but iLQR was implemented with line search. In all simulations we used a fixed number of steps $\ell = 150$, and modeled the geometry of the quadrotor as a sphere with a radius of 0.3 m. The initial state $\mathbf{x}_0^\star$ was set to $(\mathbf{p}, \mathbf{0}, \mathbf{0}, \mathbf{0})$, where initial position $\mathbf{p}$ was randomly sampled from the edges of the environment. The target state $\mathbf{x}_\ell^\star = -\mathbf{x}_0^\star$ was set to the antipodal point in the environment. Due to the multiple homotopy classes in the environment, Extended LQR and iLQR often converge to different local optima. Therefore, we averaged the computation time and the number of iterations for both methods over the same 100 random queries for each value of the time-step, which we let range from $\tau = \frac{1}{40}$ s to $\tau = \frac{1}{10}$ s. Figure 3b graphs the quantitative results, and Table 1b gives actual numbers for part of the experiments.

These results suggest that Extended LQR requires on average about a factor 3 less iterations than iLQR. However, per iteration, Extended LQR requires about twice the computation time of iLQR (35 ms versus 17 ms). This is not surprising, as Extended LQR relinearizes and requadratizes in both the backward and the forward pass, whereas iLQR only does so in its backward pass. Combining these effects, the performance gain of Extended LQR over iLQR is about a factor 1.5. The graph also shows that the number of iterations required increases as the time-step (and hence the duration of the trajectory) increases. This is true for both Extended LQR and iLQR, although our results suggest that this effect is stronger for iLQR. The cause of this is not entirely clear; an intuitive explanation may be that the "well" of the cost-potential is less "deep" for longer trajectories, as there is less hurry to arrive at the goal. For the quadrotor simulations Extended LQR required about 18 times more computation time per iteration than for the differential-drive robot simulations, reflecting the trebling of the dimension n of the state.

We also ran Extended LQR with temporal optimization on the same 100 queries (see the bottom row of Table 1b; an example trajectory is shown in Fig. 3a). In this case the resulting time-step was on average 0.070 s, and it took on average 20.85 iterations until convergence. This is consistent with the number of iterations required for a fixed time-step of 0.07 s (see Fig. 3b), confirming that temporal optimization does not negatively affect the performance of Extended LQR.

Overall, we observed that the local optimum Extended LQR converges to is relatively sensitive to the parameter settings. This is because we did not seed our approach with an initial trajectory, and in the first few iterations the minimum-cost-states "bounce around" relatively unpredictably. In most cases Extended LQR converged quickly, where the relative improvement typically declined by about a constant factor with each iteration. A second-order convergence rate was not observed in our experiments, for neither iLQR nor Extended LQR. This is likely the result of the way the obstacle-cost term is quadratized, in which negative second-order information is essentially "thrown away" in order to ensure positive-semidefiniteness. In all experiments, Extended LQR converged reliably without line search.

(a)



$\tau \to$ 0:080s

(b)

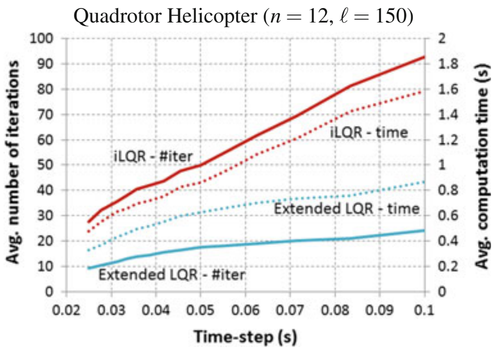


Fig. 3 **a** A trajectory resulting from the quadrotor helicopter experiments of Sect. 7.2 with temporal optimization. The robot is shown every half second. For videos of our experiments, see http://arl.cs.utah.edu/research/extendedlqr/. **b** Chart indicating the relative performance of extended LQR (*blue lines*) and iLQR (*red lines*) for varying values of the time-step in terms of computation time (*dotted lines*) and number of iterations (*solid lines*) averaged over 100 queries

8 Conclusion

We presented Extended LQR, a novel approach to the non-linear, non-quadratic optimal control problem. Experiments showed that our approach converges quickly to a locally-optimal solution, outperforming iLQR, and does not require additional convergence measures for reliability. In addition, this paper introduced the novel concept of *LQR-smoothing*, which is at the foundation of Extended LQR. We also showed that Extended LQR can naturally be applied to temporal optimization problems. We made source code of Extended LQR publicly available for download at http://arl.cs.utah.edu/research/extendedlqr/.

We have presented our approach for deterministic dynamics, implicitly relying on the independence of the optimal LQR solution to the process noise variance. An interesting question for future work is whether our approach can be extended for the risk-sensitive control problem or the stochastic optimal control problem with state and control input-dependent process noise. In [25] and [21] it is shown that LQR can naturally be extended for such settings. Our approach would require a formulation of the *inverse* stochastic dynamics, which seems to be the main challenge.

Potential application domains of Extended LQR include *optimal kinodynamic motion planning*, where Extended LQR could serve as a local planner in RRT* [10] or as part of LQR-trees [18], and *belief space planning*, where Extended LQR could improve upon the iLQR-based approach of [23].

References

1. Bar-Shalom, Y., Li, R., Kirubarajan, T.: Estimation with Applications to Tracking and Navigation, Wiley-Interscience, New Jersey (2004)
2. Bell, B.: The iterated Kalman smoother as a Gauss-Newton method. SIAM J. Optim. **4**(3), 626–636 (1994)
3. Betts, J.: Practical methods for optimal control and estimation using nonlinear programming, vol. 19, SIAM (2009)
4. Bertsekas, D.: Dynamic Programming and Optimal Control. Athena Scientific, Belmont (2001)
5. A. Björck. Numerical Methods for Least Squares Problems. SIAM, Philadelphia (1996)
6. Chen, M.S., Kao, C.Y.: Control of linear time-varying systems using forward Riccati equation. J. Dyn. Syst. Meas. Control **119**(3), 536540 (1997)
7. Fujita, Y., Nakamura, Y., Shiller, Z.: Dual Dijkstra search for paths with different topologies. In: Proceedings of the IEEE International Conference on Robotics and Automation (2003)
8. Higham, N.: Computing a nearest symmetric positive semidefinite matrix. Linear Algebra Appl. **103**, 103–118 (1988)
9. Jacobsen, D., Mayne, D.: Differential Dynamic Programming. Elsevier, New York (1970)
10. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. Int. J. Robot. Res. **30**(7), 846–894 (2011)
11. Lavalle, S.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
12. Li, W., Todorov, E.: Iterative linear-quadratic regulator design for nonlinear biological movement systems. In: Proceedings of the International Conference on Informatics in Control, Automation and Robotics (2004)
13. Nocedal, J., Wright, S.: Numerical Optimization. Springer Science+ Business Media, Germany (2006)

14. Rauch, H., Tung, F., Striebel, C.: Maximum likelihood estimates of linear dynamic systems. AIAA J. **3**(8), 1445–1450 (1965)
15. Rawlik, K., Toussaint, M., Vijayakumar, S.: An approximate inference approach to temporal optimization in optimal control. In: Proceedings of the Advances in Neural Information Processing Systems, pp. 2011–2019 (2010)
16. Rawlik, K., Toussaint, M., Vijayakumar, S.: On stochastic optimal control and reinforcement learning by approximate inference. In: Proceedings of the Robotics Science and Systems Conference (R:SS 2012), Sydney, Australia (2012)
17. Schulman, J., Ho, J., Lee, A., Awwal, I., Bradlow, H., Abbeel, P.: Finding locally optimal, collision-free trajectories with sequential convex optimization. In: Robotics: Science and Systems (2013)
18. Tedrake, R., Manchester, I., Tobenkin, M., Roberts, J.: LQR-trees: Feedback motion planning via sums-of-squares verification. Int. J. Robot. Res. **29**(8), 1038–1052 (2010)
19. Theodorou, E., Tassa, Y., Todorov, E.: Stochastic differential dynamic programming. Proceedings of the American Control Conference (2010)
20. Todorov, E.: General duality between optimal control and estimation. In: Proceedings of the IEEE Conference on Decision and Control (2008)
21. Todorov, E., Li, W.: A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In: Proceedings of the American Control Conference (2005)
22. Toussaint, M.: Robot trajectory optimization using approximate inference. In: Proceedings of the International Conference on Machine Learning (2009)
23. van den Berg, J., Patil, S., Alterovitz, R.: Motion planning under uncertainty using iterative local optimization in belief space. Int. J. Robot. Res. **31**(11), 1263–1278 (2012)
24. Weiss, A., Kolmanovsky, I., Bernstein, D.: Forward-integration Riccati-based output-feedback control of linear time-varying systems. In: American Control Conference (2012)
25. Whittle, P.: Risk-sensitive linear/quadratic/Gaussian control. Adv. Appl. Prob. **13**(4), 764–777 (1981)
26. Yakowitz, S.: Algorithms and computational techniques in differential dynamic programming. Control Dyn. Syst. **31**, 75–91 (1989)
27. Zucker, M., Ratliff, N., Dragan, A., Pivtoraiko, M., Klingensmith, M., Dellin, C., Bagnell, J., Srinivasa, S.: CHOMP: Covariant Hamiltonian optimization for motion planning. Int. J. Robot. Res. (2013)

Adaptive Communication in Multi-robot Systems Using Directionality of Signal Strength

Stephanie Gil, Swarun Kumar, Dina Katabi and Daniela Rus

Abstract We consider the problem of satisfying communication demands in a multi-agent system where several robots cooperate on a task and a fixed subset of the agents act as mobile routers. Our goal is to position the team of robotic routers to provide communication coverage to the remaining client robots. We allow for dynamic environments and variable client demands, thus necessitating an adaptive solution. We present an innovative method that calculates a mapping between a robot's current position and the signal strength that it receives along *each spatial direction*, for its wireless links to every other robot. We show that this information can be used to design a simple positional controller that retains a quadratic structure, while capturing the behavior of wireless signals in real-world environments. Notably, our approach does not necessitate stochastic sampling along directions that are counter-productive to the overall coordination goal, nor does it require exact client positions, or a known map of the environment.

1 Introduction

Multi-agent robotic systems perform many complex tasks through coordination, such as cooperative search of an environment, consensus, rendezvous, and formation control [1–3]. As cooperation is at the core of multi-robot tasks, the performance of these systems directly hinges on the robots' ability to communicate reliably. To maintain certain communication guarantees, these systems need a mapping of communication

S. Gil (✉) · S. Kumar · D. Katabi · D. Rus
Massachusetts Institute of Technology, Boston, MA, USA
e-mail: sgil@mit.edu

S. Kumar
e-mail: swarun@cmu.edu

D. Katabi
e-mail: dk@mit.edu

D. Rus
e-mail: rus@mit.edu

quality to robot placement. Producing such a mapping however is quite challenging [4]. Past literature employs two broad strategies to address this challenge: On the one hand, there is the Euclidean disk model which assumes that the signal quality of a link is a function of distance between the communicating vehicles. This model is deterministic and simple, and hence when incorporated in a robotic controller, yields simple positional optimizations for a wide range of collaborative tasks [1–3]. Unfortunately, the Euclidean model is too simplistic and fails to represent wireless signals in realistic environments [4]. On the other hand, there are stochastic sampling methods [4–6] that measure the wireless signal strength in a robot's vicinity to fit parameters for intricate probabilistic communication models. While such methods are not oblivious to wireless channels, they require exploratory sampling [7] along directions that may be counter-productive to the overall coordination goal. Further, they often assume the knowledge of parameters based on the structure and material composition of the environment.

Our objective here is to (i) present a novel method for capturing the spatial variation of wireless signals in the local environment *without* sampling along counter-productive directions, or requiring information about the environment and/or the channel's distributions and (ii) derive a control formulation that maintains the structural (quadratic) simplicity allowed by the Euclidean disk model while accounting for this wireless channel feedback. First, we introduce an innovative approach for mapping communication quality to robot placement. We calculate a mapping between a robot's current position and the signal strength that it receives along *each spatial direction*, for every wireless link with other robots. This is in contrast to existing methods [5, 6], which compute an aggregate signal power at each position but cannot distinguish the amount of signal power received from each spatial direction.

Second, we construct an optimization for positioning a team of robot routers to provide communication coverage to an independent set of client vehicles using the directional information provided by our mapping. We aim for a solution that is adaptive to variable communication quality demands by the clients, as well as changes in the wireless channels due to natural fluctuations or a dynamic environment. Being able to measure the profile of signal strength across spatial directions in real-time yields a much more capable controller. For example, the controller uses the profile to find directions of movement that yields better communication quality. The profile also helps estimate the confidence with which the controller can improve signal power by navigating the robot along any of these directions. The confidence can then be used to control the speed of the robot, thereby improving stability and convergence time. Furthermore, the controller can leverage the entire profile of signal strength across directions, to optimize communication with multiple robots by choosing a direction of movement corresponding to a strong signal that strikes trade-offs between competing demands. Interestingly, we show that such optimizations can be formulated in terms of simple quadratic costs, similar in spirit to the disk model. Further, they can be made independent of environment-dependent parameters, or even client positions.

A key question remains: how do we calculate the signal strength along each spatial direction? The naive approach would use directional antennas, a type of antenna that receives signals only from a cone in space. Unfortunately, directional antennas are bulky and have low spatial resolution [8] (about 60°), making them ill-suited for small agile robots. To address this problem, we employ Synthetic Aperture Radar (SAR), a technique that leverages movement to emulate a high-resolution directional antenna [9]. In order to achieve this, we must derive a method for implementing SAR using off-the-shelf wireless cards, a challenging task since these devices are not intended for this purpose.

We implement our method in a multi-robot testbed that has two robotic routers serving three robotic clients. We conduct our experiments in different indoor environments without providing the robotic controller the environment map or the clients' positions. We observe the following: (1) Our system consistently positions the robotic routers to satisfy the robotic client demands, while adapting to changes in the environment and fluctuations in the wireless channels; (2) Compared to the disk model [1, 2] and the stochastic approach [10, 11] under identical settings, our system converges to accurately satisfy the communication demands, unlike the disk model, while significantly out-performing the stochastic method in terms of empirical convergence rate (see Fig. 8 in Sect. 5.4).

Contributions: The contributions of this paper are three-fold: (1) We present a method to enable a robotic receiver to find the profile of signal strength across spatial directions for each sender of interest. To this end, we perform synthetic aperture radar (SAR) techniques using standard Wi-Fi packets exchanged between two independent nodes; (2) We develop an optimization that leverages this directional signal profile to position robotic routers to satisfy heterogeneous communication demands of a network of robotic clients, while adapting to real-time environmental changes; (3) We implement our design and demonstrate its empirical gains in comparison to both the disk model and the stochastic method.

2 Related Work

Our work is related to past papers on multi-robot coordination to achieve a collaborative task while supporting specific communication demands [4–6, 10, 12]. These papers recognize the importance of measuring the signal strength on real-world wireless links to model communication quality. Such papers typically build analytical models of the signal strength on a wireless link to account for the effects of distance, obstacles, and reflections on the signal. The models are then supplemented with measurement data. While these approaches provide a more realistic integration of robot coordination with communication constraints than the disk model, they often necessitate parameter fitting that are environment-dependent. Further, they require sampling of signal strength along stochastic directions that may be counter-productive to the overall coordination goal. In comparison to these papers, we introduce a system that captures the different *directions* of a signal, as opposed to only its magnitude at a particular position. This allows us to satisfy variable demands from multiple robotic

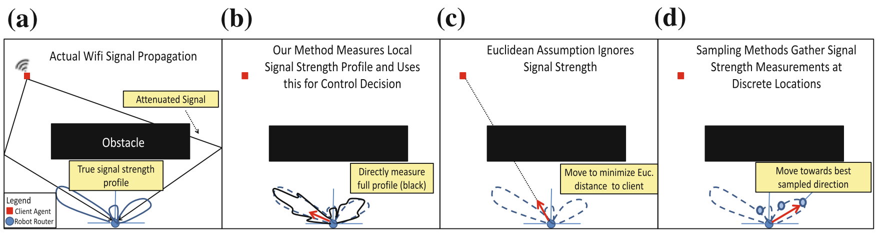

(a)
(b)
(c)
(d)
Actual Wifi Signal Propagation
Attenuated Signal
Obstacle
True signal strength profile
Legend
Client Agent
Robot Router
Our Method Measures Local Signal Strength Profile and Uses this for Control Decision
Directly measure full profile (black)
Euclidean Assumption Ignores Signal Strength
Move to minimize Euc. distance to client
Sampling Methods Gather Signal Strength Measurements at Discrete Locations
Move towards best sampled direction

Fig. 1 Schematic drawings demonstrating the differences between the current method and previous methods

clients in an environment-oblivious fashion, and without sampling the signal along stochastic directions (Fig. 1).

Our work is also related to past work on synthetic aperture radar (SAR). SAR allows us to exploit the natural movements of robots to calculate the signal strength along each spatial direction. Past work on SAR however assumes a single device that transmits a signal and receives its reflections [9, 13–15], and none of this work can use off-the-shelf Wi-Fi cards. In contrast, we present a system that extracts SAR information from standard Wi-Fi packets transmitted between different devices.

3 Problem Statement

We consider a mobile network with two classes of members, n robotic clients (or *clients*) whose positions are not controlled, and a team of k robotic routers whose mobility we control. Our goal is to position the robotic routers to provide adaptive wireless communication coverage to the clients, while allowing variable communication quality demands for all clients, and where exact client positions are unknown. For each client $j \in [n] = \{1, \ldots, n\}$, we define demanded communication quality $q_j > 0$,[1] and achieved communication quality ρ_{ij} to each router i (where $i \in [k]$), both expressed in terms of *Effective Signal to Noise Ratio* (ESNR) that has a direct mapping to rate in Mb/s [16].[2] Additionally, let every client j be given an importance $\alpha_j > 0$. We define the notion of *service discrepancy* for each pair of robots (i, j) to be the difference between the demanded and achieved communication quality scaled by the importance of the client.

$$w_{ij} = \max(\alpha_j(q_j - \rho_{ij})/q_j, 0) \tag{1}$$

[1]Note that all quantities in this section are time-dependent; we omit this dependency for simplicity.

[2]We choose to work with ESNR values rather than rates since the rates supported on a link are discretized (non-continuous).

Physically, this is the fraction of the client's communication demand that remains to be satisfied, scaled by α_j. Denote by $c_i \in \mathbb{R}^d$ the position of the ith robot router and by $p_j \in \mathbb{R}^d$ the position of the jth client[3] and $C_t = \{c_{1,t}, \ldots, c_{k,t}\}$ is the set of all router positions at time t. Given a cost g in terms of signal quality, communication demands, and agent positions, we wish to position each robotic router to minimize the largest discrepancy of service between routers and clients. However, the true form of this function g has an intricate dependence on the position of the client, router, and the environment. Thus an inherent challenge to solving this problem is approximating the influence of spatial positioning on communication quality in a way general to different environments. We have a joint goal to (1) find $f_{ij} : [-\frac{\pi}{2}, \frac{\pi}{2}] \to \mathbb{R}$ (a relation capturing directional information about the signal quality between i and j), and an approximation $\tilde{g}$ of g that is a cost characterizing the anticipated communication quality for the router-client pair (i, j) at a proposed router position c_i, and (2) use this cost to optimize router positions to minimize the service discrepancy to each client. Formally,

Problem 1 Find a mapping

$$f_{ij} : \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \to \mathbb{R} \tag{2}$$

that maps spatial direction to wireless signal strength directly from channel measurements, and a cost

$$\tilde{g}(c_i, C_t, w_{ij}, f_{ij}) > 0 \tag{3}$$

that is independent of the environment and client positions, has a simple quadratic form, and whose minimization over c_i directly relates to increasing signal quality. We aim to find robot router positions, C_{t+1} that minimize the maximum service discrepancy over all clients j by solving the following min–max problem:

$$C_{t+1} = \arg\min_{c_i \in C}\{\max_j \min_i \tilde{g}(c_i, C_t, w_{ij}, f_{ij})\} \tag{4}$$

Intuitively, the solution to this optimization problem favors "fair" solutions where the maximum service discrepancy is minimized over all clients. We dedicate the next sections of this paper to (1) Developing a method that computes f_{ij} as the profile of signal qualities along each direction θ for each link (i, j) found directly from channel measurements; and (2) Developing an optimization framework that utilizes this directional information to handle trade-offs between competing client demands, and position all routers to jointly minimize the maximum service discrepancy over the links in the communication network.

[3]In this paper we mainly consider $d = 2$ although all concepts are extensible to $d = 3$.

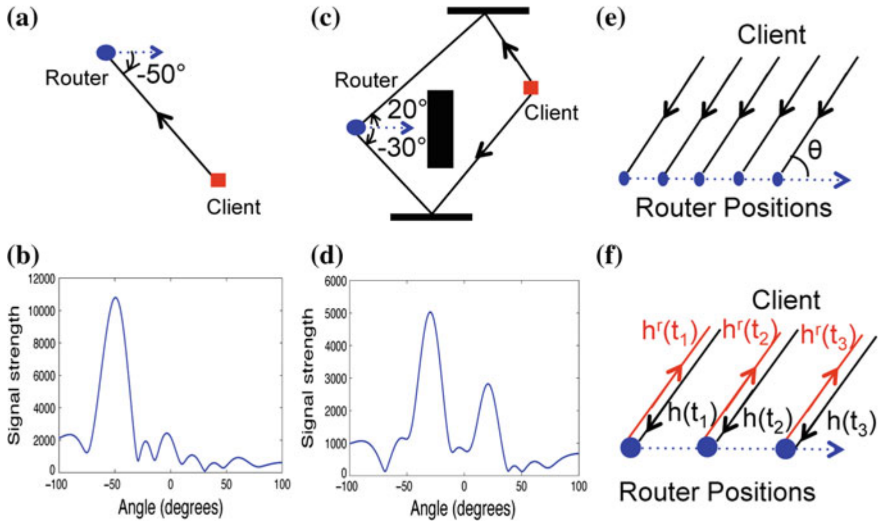


Fig. 2 **a**, **c** LOS and NLOS topologies annotated with signal paths. **b**, **d** $f(\theta)$ of the signal in LOS and NLOS. **e** Shows how θ is defined in SAR. **f** Shows $h(t_i)$, the forward channel from transmitter to receiver and $h^r(t_i)$, the reverse channel from receiver to transmitter at time t_i

4 Approach

4.1 *Computing the Directional Power Profile of a Wireless Link*

In this section, we develop the first component of the solution of Problem 1; namely, we derive a method to calculate $f(\theta)$, the mapping which captures the strength of the signal from a robotic client to its router along each direction θ.[4] Where this mapping can be updated often, roughly once every 6 cm of motion.

Before we explain how we compute $f(\theta)$, we describe this function to help understand the information it captures. Assume we have a robotic client and router, where the router moves along some trajectory. We will define the direction θ relative to the tangent to the router's trajectory at each point. Consider the scenario in Fig. 2a, where the robotic client is in line-of-sight at $-50°$ relative to the robotic router, which is moving along the horizontal axis. In this case, one would expect $f(\theta)$ to have a single dominant peak at $-50°$, as shown in Fig. 2b. Now consider the more complex scenario in Fig. 2c, where the environment has some obstacles and one of these obstacles obstructs the line-of-sight path between the router and its client. In

[4]For simplicity, we denote $f_{ij}(\theta)$ as $f(\theta)$ as we consider only the single link between robotic router i and client j for the rest of this section.

this case, $f(\theta)$ would show two dominant peaks at 20° and −30° that correspond to the two reflected paths from surrounding obstacles, as shown in Fig. 2d.

Advantage over Sampling Methods: One may estimate $f(\theta)$ by sampling the signal power similar to stochastic techniques [5, 10, 11]. In this case, one has to move the router along each direction, compute the power in all these new positions relative to the first, and draw the profile $f(\theta)$. Unfortunately, this approach leads to much wasted exploration. This is because the signal power does not change reliably when the robot moves. For example, if the robot moves for 5 or 10 cm, it is very likely that the resulting change in the signal power is below the variability in noise. Hence, measurements of power over short distances are likely to be marred by noise. To obtain reliable measurements of changes in the signal power, the robot has to move significantly along potentially counter-productive paths.

To address this limitation, our approach relies on the channel phase as opposed to the power. Specifically, at any position the wireless channel can be expressed as a complex number $h(t)$ [17]. The magnitude of this complex channel captures the signal power (more accurately, its square-root). The phase of the channel has traditionally been ignored by robotic systems. However, the phase changes rapidly with motion. For Wi-Fi signals at a frequency of 5 GHz, the phase of the channel rotates by π every 3 cm. This far exceeds any rotation due to noise variability. Thus, by measuring channels as complex numbers and tracking changes in its phase as the robot moves, we reliably estimate signal variation without much exploration. In the next section, we explain how to use a technique called synthetic aperture radar (SAR) to extract the received signal strength along each direction from changes in channel phase. Note that SAR does not need exploring all directions; the robot can move along its path without extra exploration or sampling. SAR uses the resulting variations in channel phase over distances of a few centimeters to find $f(\theta)$.

4.1.1 Synthetic Aperture Radar (SAR)

Synthetic Aperture Radar (SAR) enables a single antenna mounted on a mobile device to estimate the strength of the signal received along every spatial direction. We leverage the natural motion of a robotic router to implement SAR and measure $f(\theta)$ for each of its robotic clients using an omni-directional antenna. To do so, the robotic router measures the channel $h(t)$ from its client as it moves along any straight line. The straight line path over which the router acquires data is on the order of half a wavelength (centimeters); assuming the source is stationary and the router either moves at a known constant velocity or its position is known for the traversal time window, then a sufficient amount of usable channel data can be collected. This means every few centimeters the router can have an updated measurement of $f(\theta)$, for all values of θ.

Specifically, Let $h(t)$ for $t \in \{t_0, \ldots, t_m\}$ be the $m+1$ most recent channel measurements, corresponding to the robot moving a distance $d(t_0) \ldots d(t_m)$. SAR computes the received signal strength across spatial directions $f(\theta)$ as:

$$f(\theta) = \left| \sum_t h(t) e^{-j\frac{2\pi}{\lambda} d(t) cos\theta} \right|^2, \tag{5}$$

where λ is the wavelength of the Wi-Fi signal. We refer the reader to [18] for the analysis of this standard SAR equation. At a high level, the multiplying terms $e^{-j\frac{2\pi}{\lambda} d(t) cos\theta}$ in Eq. (5) project the channels $h(t)$ along the direction of interest θ by compensating for incremental phase rotations introduced by the robot's movement to all paths of the signal arriving along θ.

Note that SAR finds the signal power from every angle θ simply by measuring the channels,[5] without needing prior tuning to any given direction. In fact, moving by around a wavelength (about 6 cm) is sufficient to measure the full profile of $f(\theta)$.

Therefore, SAR is a natural choice for autonomous robotic networks since it exploits the mobility of the robots to compute $f(\theta)$. Further, it only requires the robot to move along a small straight line along any arbitrary direction, and does not require it to explore directions counter-productive to the overall coordination goal. Note that SAR requires only the relative position of the robotic router $d(t)$ and the both the magnitude and phase of the channel $h(t)$. It does not require the topology of the environment nor the exact location of the transmitter.

4.1.2 Challenges in Implementing SAR on Independent Wireless Devices

A key challenge in adapting SAR to multi-robot systems is that all past SAR-based solutions [9, 13, 15] are for radar-like applications, where a single device transmits a radar signal and receives its reflections off an imaged object, e.g., an airplane. However, in our scenario the transmitter and receiver are completely independent wireless devices (i.e., the robotic client and router, respectively). This means that the transmitter robot and the receiver robot have different frequency oscillators. In practice, there is always a small difference between the frequency of two independent oscillators. Unfortunately, even a small offset Δ_f in the frequency of the oscillators introduces a time varying phase to the wireless channel.

For instance, let $h(t_0)$, $h(t_1)$, ..., $h(t_m)$ be the actual wireless channel from the robotic client to the robotic router at times $t_0, t_1, \ldots, t_m$. The channel observed by the router from its client $\hat{h}(t_0)$, $\hat{h}(t_1)$, ..., $\hat{h}(t_m)$ are given by:

$$\hat{h}(t_0) = h(t_0), \quad \hat{h}(t_1) = h(t_1)e^{-2\pi \Delta_f (t_1 - t_0)}, \ldots, \quad \hat{h}(t_m) = h(t_m)e^{-2\pi \Delta_f (t_m - t_0)}. \tag{6}$$

Hence, the phase of the channels are corrupted by time-varying values due to the frequency offset between the transmitter and the receiver. Fortunately, we can correct

[5]Of course, the resolution at which θ is available depends on the number of channel measurements.

for this offset using the well-known concept of channel reciprocity [17]. Specifically, let $h^r(t)$ denote the reverse channel from the robotic router to its client, as shown in Fig. 2f. Reciprocity states that the ratio of the forward and reverse channels stays constant over time, subject to frequency offset, i.e. $h^r(t) = \gamma h(t)$, where γ is constant. Further, the frequency offset in the reverse direction Δ_f^r is negative of the offset in the forward direction, i.e. $\Delta_f^r = -\Delta_f$. Thus, the observed reverse channels $\hat{h}^r(t_0)$, $\hat{h}^r(t_1), \ldots, \hat{h}^r(t_m)$ are given by:

$$\hat{h}^r(t_0) = h^r(t_0),\ \ \hat{h}^r(t_1) = h^r(t_1)e^{2\pi\Delta_f(t_1-t_0)},\ \ \ldots,\ \ \hat{h}^r(t_m) = h^r(t_m)e^{2\pi\Delta_f(t_m-t_0)}. \tag{7}$$

Multiplying Eqs. 6 and 7 above and using $h^r(t) = \gamma h(t)$, we have $\hat{h}(t)\hat{h}^r(t) = h(t)h^r(t) = \gamma h(t)^2 \Rightarrow h(t) = \sqrt{\hat{h}(t)\hat{h}^r(t)/\gamma}$. Hence we re-write Eq. (5) as:

$$f(\theta) = \left|\sum_t \sqrt{\hat{h}(t)\hat{h}^r(t)}e^{-j\frac{2\pi}{\lambda}d(t)cos\theta}\right|^2, \tag{8}$$

where the constant scaling γ is dropped for simplicity. Hence, to measure $f(\theta)$ the router and client simply need to measure their channels at both ends.[6] In the next section, we explain how we leverage $f(\theta)$ on each link to control the position of multiple robotic routers to meet the clients' communication demands.

4.2 *Optimizing Robotic Router Placement Using Channel Feedback*

In this section, we target the problem of placing a team of mobile router vehicles at locations such that they provide wireless coverage to client vehicles, each with different communication demands. Specifically, using as input the channel feedback $f_{ij}(\theta)$ derived in the previous section, we aim to find a function $\tilde{g}$ that can be optimized over router positions such that:

$$C_{t+1} = \arg\min_C\{\max_j \min_{c_i \in C} \tilde{g}(c_i, C_t, w_{ij}, f_{ij})\} \tag{9}$$

Our focus in this section is to find a function $\tilde{g}$ that has three desirable properties: (1) It is quadratic; (2) It allows for trade-offs between clients with competing demands as captured by the service discrepancies w_{ij}; and (3) It is independent of client positions p_j. In the rest of this section, we show how to capitalize the rich spatial information

[6] In practice, the router and client transmit back-to-back packets with a small gap $\delta \approx 200\,\mu$s to obtain $\hat{h}^r(t+\delta)$ and $\hat{h}(t)$, respectively. The router collects these values and approximates $\hat{h}(t)\hat{h}^r(t)$ as $\hat{h}(t)\hat{h}^r(t+\delta)e^{-j2\Delta_f\delta}$. The router computes this 10 times per second (an overhead of just 0.1 %).

provided by $f_{ij}(\theta)$, to derive a cost $\tilde{g}$ possessing the three desired qualities. We can then optimize this cost to complete our objective of robot router placement that best satisfies the communication demands of the clients.

4.2.1 A Generalized Distance Metric for Incorporation of Channel Feedback

Our first goal is to translate signal quality over all directions, $f_{ij}(\theta)$, to a cost $\tilde{g}$ that can be optimized over router positions. We begin with the case where all positions are known and extend to the position independent case in Sect. 4.2.3. Consider a single router-client pair (i, j) located at positions (c_i, p_j). A disk model approach to service this client does not use $f_{ij}(\theta)$ at all. Instead, it relates improving communication quality between the router and client to reducing the Euclidean distance between them, i.e. the cost $\tilde{g} := dist(p_j, c_i)$. The appeal of such a cost is in its simple quadratic form that can be easily optimized. Unfortunately, the cost is oblivious to the actual wireless channel at the client and fails to capture the current service discrepancy which can be large even at small distances (say, due to obstacles).

Our system avoids this pitfall, while retaining simplicity, by incorporating real-time channel feedback into a generalized distance metric. In particular, we do not assume that the shortest distance for enabling better communication between two robots is the straight line path between them, but rather the path along the $\theta_{\max}$, the direction of maximum signal strength from the mapping $f_{ij}(\theta)$. Thus, the client is recommended to move towards $\mathbf{v}_{\theta_{\max}}$, the unit vector along $\theta_{\max}$.

Importantly, the recommended heading direction $\mathbf{v}_{\theta_{\max}}$ may exhibit variation due to noise or multipath affecting the wireless link. To account for these effects, while not over-fitting to noise, we leverage the entire f_{ij} signal profile to design a *confidence* metric σ_{ij} in heading direction. Intuitively, σ_{ij} captures the"variance" of f_{ij} around $\theta_{\max}$.[7] We would like to encode this quantity into our controller such that $\mathbf{v}_{\theta_{\max}}$ directions of high confidence are followed more aggressively (larger displacements along these directions), and the opposite is true of $\mathbf{v}_{\theta_{\max}}$ directions with low confidence. Specifically, σ_{ij} falls under the following categories: (1) $\sigma_{ij} < 1$: Indicates a high confidence in $\mathbf{v}_{\theta_{\max}}$ due to a sharp peak in f_{ij}. The robot is moved at higher speeds; (2) $\sigma_{ij} \approx 1$: Indicates that f_{ij} is noisy, so the robot must move slowly; (3) $\sigma_{ij} > 1$: Indicates that f_{ij} has multiple significant peaks owing to multi-path. We study this case, and particularly the opportunity it presents for making trade-offs between clients, more elaborately in Sect. 4.2.2.

We can use the heading direction and confidence to design a cost function $\tilde{g}$ that accurately captures the cost of communication in the spatial domain. Interestingly, we can express this cost as a generalized distance metric called the *Mahalanobis distance*. The square of the Mahalanobis distance is a cost function (paraboloid) with ellipsoidal level sets (Fig. 3). We design our cost by orienting these level sets so that the direction of steepest descent is along $\mathbf{v}_{\theta_{\max}}$. We then skew the ellipsoidal

[7]Mathematically, $\sigma_{ij} = \sum_\theta [(\theta - \theta_{\max})^2 f_{ij}(\theta)] / \sum_\theta [(\theta - \theta_{\max})^2 \text{mean}\{f_{ij}(\theta)\}]$.

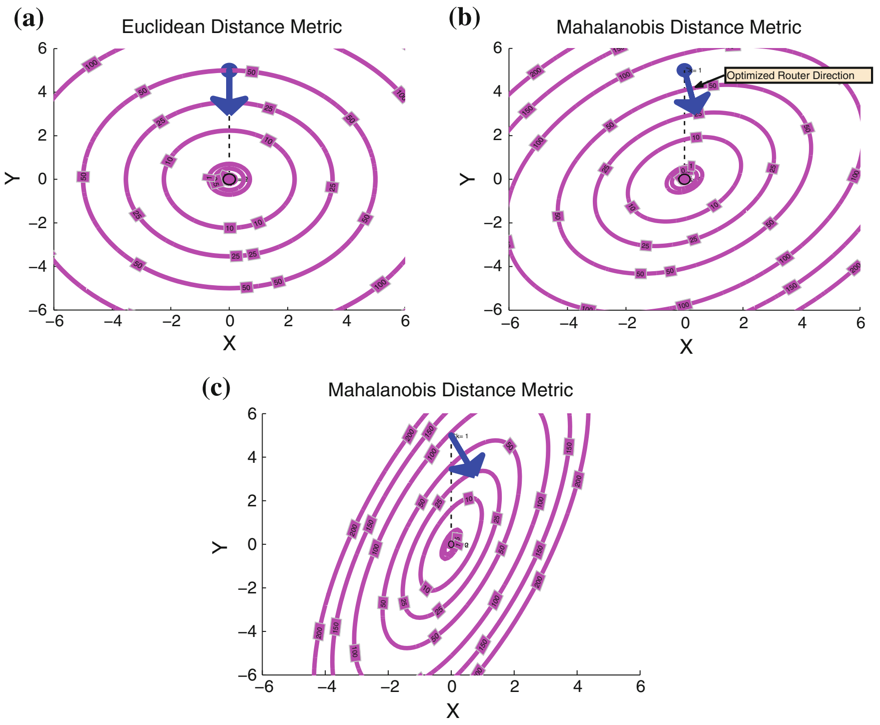


Fig. 3 These plots show the level sets of Euclidean and Mahalanobis distance functions. **a** Euclidean distance. **b** Mahalanobis (Low Conf). **c** Mahalanobis (High Conf)

level sets using the confidence σ_{ij}, so that a higher confidence translates to a steeper descent. Mathematically, the Mahalanobis distance is given by:

Definition 1 (*Mahalanobis Distance*) Given a positive definite matrix $M \in \mathbb{R}^{dxd}$, a vector $x \in \mathbb{R}^d$, and a vector $y \in \mathbb{R}^d$, the Mahalanobis Distance between x and y is:

$$\text{dist}_M(x, y) = \sqrt{(x-y)^T M (x-y)} \tag{10}$$

Euclidean distance is a special case of the Mahalanobis distance (see Fig. 3a) with $M = I$ where I is the identity matrix of appropriate dimension.

Here, $M = Q\Lambda Q^T$ is a positive-definite matrix, where Q consists of orthogonal eigen-vectors and Λ contains the corresponding eigen-values. We simply set one of the eigen-vectors of Q to the heading direction $\mathbf{v}_{\theta_{\max}}$. To skew the ellipsoid, we set the ratio of the eigen values $\{\lambda_1, \lambda_2\}$ in Λ to the confidence σ^2, i.e. $\lambda_2/\lambda_1 = \sigma^2$, where λ_1 is the eigen-value corresponding to $\mathbf{v}_{\theta_{\max}}$. For e.g., In Fig. 3b, where $\sigma \approx 1$ (i.e. poor confidence), the level sets are nearly circular, leading to a shallow descent in cost; while Fig. 3c, where $\sigma < 1$ (i.e. high confidence), the level sets are skewed, leading to a steep descent in cost along $\mathbf{v}_{\theta_{\max}}$. In other words, the cost function has an elegant geometric interpretation, akin to Euclidean distance, but is derived directly from channel measurements. Further, the cost function $\tilde{g} := \text{dist}_{M_{ij}}(p_i, c_j)$ from Eq. (10) is quadratic, a desirable property for optimizations.

4.2.2 Network Trade-Offs

In this section, we show how our optimization framework readily extends to a multi-agent scenario and study the different trade-offs. We show that via the setting of two parameters, both set automatically from wireless channel data, the resulting positional controller can be made to greedily optimize one client vs. strike trade-offs between multiple clients. First, we focus on managing service discrepancies specified by w_{ij}. w_{ij} aims to bias the controller by assigning higher weight to users with larger service discrepancies. To do this, we scale the cost function $\tilde{g} = \text{dist}_{M_{ij}}(p_i, c_j)$ by the square of the discrepancy w_{ij}^2 to optimize:

$$r_M(P, C) = \max_{p_j \in P} \min_{c_i \in C} \{w_{ij}^2 \text{dist}_{M_{ij}}(p_i, c_j)\} \tag{11}$$

Second, we highlight the subtle role played by the confidence σ_{ij} in managing network trade-offs. For instance, consider a scenario with two clients: 1 and 2, where client-1 demands greater communication quality (as specified by w_{ij}'s). Suppose client-1 has a highly confident $\mathbf{v}_{\theta_{\max}}$ as shown in Fig. 4a (i.e. $\sigma_{ij} < 1$). As expected, the robotic router is directed towards client-1 as shown in Fig. 4c. In the more interesting scenario in Fig. 4b, client-1's confidence is poor due to multiple peaks in the signal profile f_{ij} (i.e. $\sigma_{ij} > 1$). Here, the router strikes a trade-off and services client-2 instead, as this may potentially benefit client-1 as well due to the multipath recognized in client-1's $f_{ij}(\theta)$ map. The intuition behind this is simple. Eq. (13) above, scales the ellipsoidal cost function based on the discrepancies w_{ij}'s. However, recall that the ellipsoidal cost function is steep (or shallow) depending on whether the confidence is high (or low) and this is attained by setting the ratio of eigenvalues λ_2/λ_1 of M_{ij}. In extremely low confidence scenarios such as Fig. 4b, the higher value of discrepancy of client-1 is masked by its low value of confidence. This balances the trade-off in favor of client-2, despite a lower discrepancy.

4.2.3 A Position-Independent Solution

A simple relaxation to the cost from the previous section frees the optimization of using client positions, while maintaining its simple structure and desirable properties developed above. Consider a given stepsize $\gamma > 0$. We replace client positions p_j in

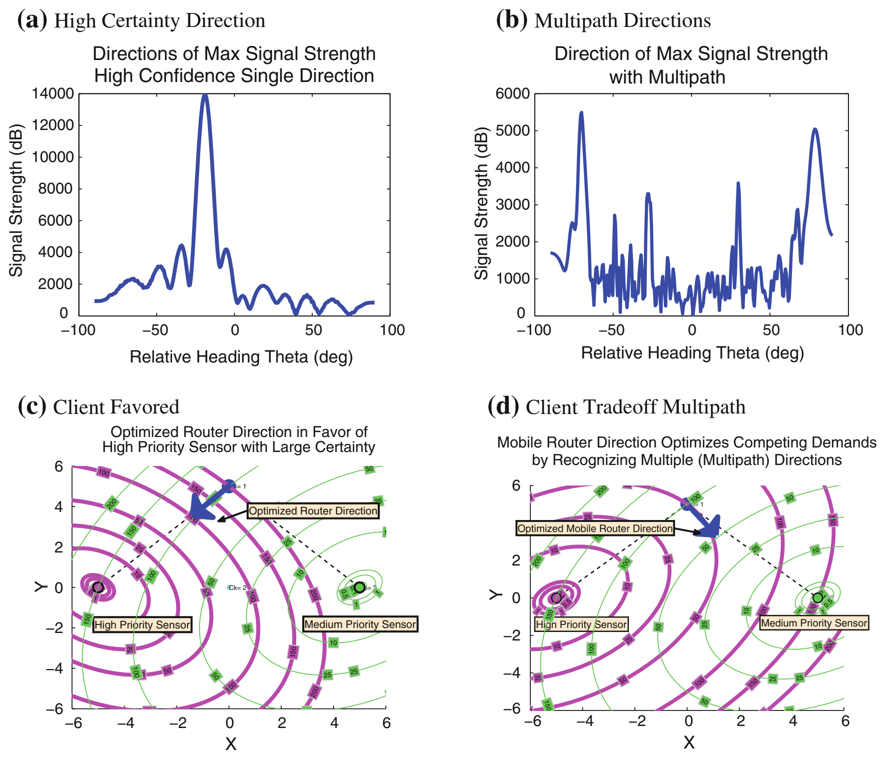


Fig. 4 **Trade-offs between Clients**: **a**, **b** show the $f_{ij}(\theta)$ map for the high demand client; **c**, **d** show the optimized router direction

Eq. (13) with "virtual" positions p'_{ij}:

$$p'_{ij} = c_{i,t} + \gamma w_{ij} \mathbf{v}_{\theta_{\max}} \tag{12}$$

Loosely, a client is no longer directly observed but rather estimated to be along the relative direction $\mathbf{v}_{\theta_{\max}}$ and at a distance of γw_{ij} with respect to the ith router. As before, $\mathbf{v}_{\theta_{\max}}$, is the heading direction associated with the maximum stength signal direction $\theta_{\max}$. As a client's demand is better satisfied by router i, the service discrepancy w_{ij} tends to 0 and the client is perceived as being closer to router i. The intuition here is that routers better equipped to service a particular client as reflected by the w_{ij} term, will view the client as "closer" and those routers with a weaker signal to the same client will view this client as farther away. This results in a natural method of assigning client nodes to routers by effectively sensing over the wireless channels. Our final cost takes the form:

$$r_M(C) = \max_{j \in \{1,\ldots,n\}} \min_{c'_i \in C'} \{\text{dist}^2_{M_{ij}}(c_{i,t} + \gamma w_{ij} \mathbf{v}_{\theta_{\max}}, c'_i)\} \tag{13}$$

By expanding the squared per-link cost $\mathrm{dist}^2_{M_{ij}}(c_i + \gamma w_{ij}\mathbf{v}_{\theta_{\max}}, c'_i)$ from Eq. (13):

$$(c'_i - c_{i,t})^T M_{ij}(c'_i - c_{i,t}) - 2\gamma w_{ij}\lambda_{\theta_{ij}}\mathbf{v}^T_{\theta_{\max}}(c'_i - c_{i,t}) + \gamma^2 w^2_{ij}\lambda_{\theta\, ij} \tag{14}$$

we note that as $w_{ij} \to 0$ the first term in Eq. (14) favors stable solutions where $c'_i = c_{i,t}$, i.e. the router reaches a static solution when all of its assigned clients have zero service discrepancy.

Finally for a set of routers with positions C, $r_M(C)$ reflects the cost of the client with the largest service discrepancy. The positions C that minimize $r_M(C)$ can be found by solving this optimization as a second order cone program as in [19, 20].

As defined in our problem statement, Problem 1, we have found a set of quadratic costs $g(p_j, c_i, C_t, w_{ij}, f_{ij}) = (p'_{ij}(c_{i,t}, w_{ij}, \mathbf{v}_{\theta_{\max}}) - c_i)^T M_{ij}(p'_{ij}(c_{i,t}, w_{ij}, \mathbf{v}_{\theta_{\max}}) - c_i)$ that can be optimized in the desired min–max formulation from (4) in order to find an optimized robotic router placement for our wireless network.

5 Experimental Results

We evaluated our system on a five-node testbed with two routers and three clients. Each node was an ASUS 1015PX netbook equipped with an Intel 5300 Wi-Fi card mounted on an iRobot Create robot. We implemented SAR by modifying the iwlwifi driver on Ubuntu 10.04. We used the 802.11 CSI tool [21] to obtain channel information ($\hat{h}(t)$ in Eq. (8)). The routers communicated with a central laptop emulating the base for control information and human input. We performed our experiments in a room with a Vicon motion capture system to aid robot navigation. Our testbed contains obstacles to simulate both line-of-sight and non-line-of-sight scenarios.

5.1 Computing Direction of Maximum Signal Strength

We first observe how effectively our system computes the direction of maximum signal strength $\theta_{\max}$, on a wireless link. We consider a single client, serviced by a robot router that is: (1) In direct line-of-sight (LOS) as shown in Fig. 5a. (2) In possible non-line-of sight (NLOS) scenarios due to obstacles as shown in Fig. 5b. We drive the robot router in a lawn-mover pattern and get $\theta_{\max}$ at regular intervals.

Results: Figure 5a and b depict the gradient field with the arrows indicating $\theta_{\max}$ in LOS and NLOS, respectively. The gradient field in LOS accurately directs the robot router towards the client regardless of its initial position. In NLOS, the robot is directed away from obstacles so that controller can route around obstacles to improve signal strength. We stress that $\theta_{\max}$ is found locally at the router purely via wireless channels and its own position, *without* prior knowledge of the environment. Further, the plots are not static and naturally *change over time*, especially in dynamic settings. Thus our system obtains instantaneous $\theta_{\max}$ values locally in real-time.

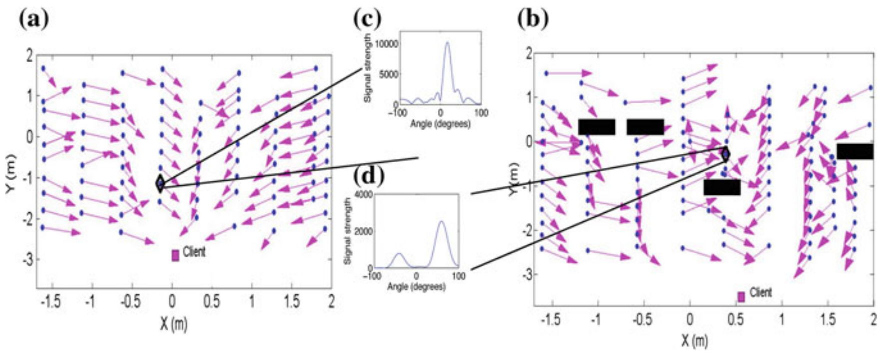


Fig. 5 Gradient field of $\theta_{\max}$ and power profile for **a** Line-of-sight and **b** Non-line-of-sight

Figure 5c and d plot $f_{ij}(\theta)$, the power profile of the signal along different directions, for a candidate location in line-of-sight and non-line-of-sight scenarios, respectively. Clearly, the power profile in line-of-sight is dominated by a single peak at $\theta_{\max}$, directed along the line-of-sight path to the client. In contrast, the power profile in non-line-of-sight close to an obstacle has two significant peaks, each corresponding to reflected paths along walls or other objects in the environment.

5.2 Controlling Router Trajectory to Satisfy Client Demands

We evaluate how a single robotic router finds a trajectory to satisfy the demands of three clients (specified in terms of effective signal-to-noise ratio or ESNR) using $\theta_{\max}$ on each link. We consider the candidate non-line-of-sight setting in Fig. 6a. The router is unaware of exact client positions or the layout of the environment.

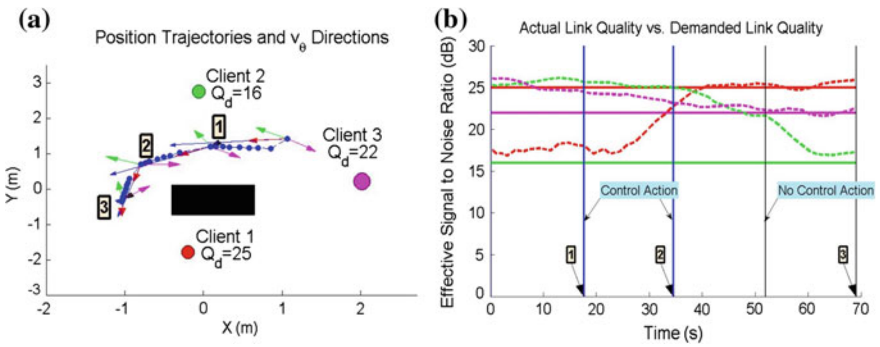


Fig. 6 **a** Depicts testbed with robot router servicing three clients in a candidate non-line-of-sight setting. The *blue line* depicts the trajectory, and *colored arrows* indicate instantaneous $\theta_{\max}$ for the corresponding clients. **b** Plots the ESNR across time (as *dotted lines*) for each client through the experiment. *Solid lines* denote client demands

Results: Figure 6a depicts the trajectory of the robotic router in blue. The colored arrows denote the recommended $\mathbf{v}_{\theta_{\max}}$ directions for each client at every control point. The figure shows how the robot performs non-zero control actions until it eventually satisfies network demands. Figure 6b tracks the ESNR of the clients across time (dotted lines). The plot shows that the ESNR demands of each client (solid lines) are satisfied upon convergence. Note that the whenever the robot decides to follow the $\mathbf{v}_{\theta_{\max}}$ of a client at a control point (vertical line), the client's ESNR increases. This validates our claim that following a heading direction based on $\mathbf{v}_{\theta_{\max}}$ indeed improves the ESNR of the corresponding client.

5.3 Aggregate System Results

We evaluate our full system with two robot routers serving three clients with different ESNR demands. We perform the experiment in line-of-sight (LOS) and non-line-of-sight (NLOS) settings as shown in the inset maps of Fig. 7b and 7d respectively. We repeat the experiment five times in each setting and plot the results.

Results: Figure 7a and b plot the mean and variance of ESNR over time across experiments for each client (dotted colored lines) in LOS and NLOS. Clearly, each client's ESNR demand (solid lines) is satisfied at the converged position across

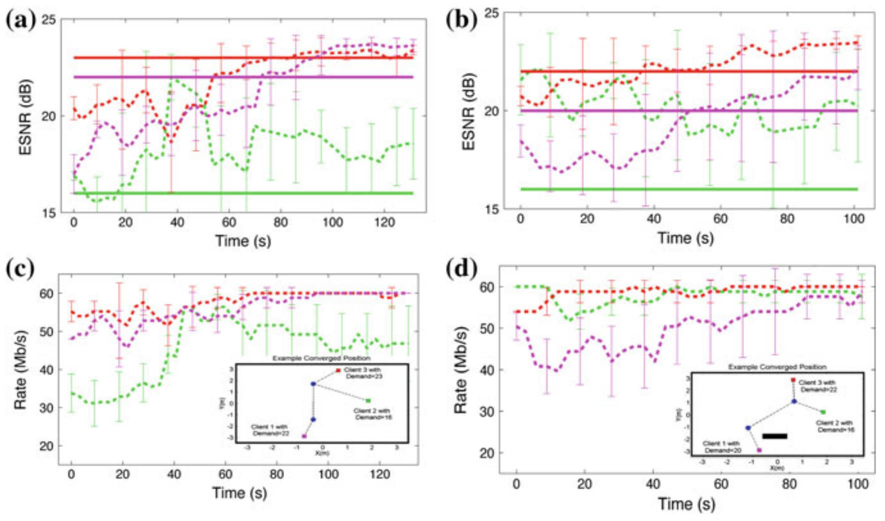


Fig. 7 Agregate results obtained over 5 runs show demands are consistently met even in the presence of obstacles as demonstrated by the candidate converged solutions. **a** ESNR versus time (Line of sight). **b** ESNR versus time (Non line of sight). **c** Rate versus time (Line of sight). **d** Rate versus time (Non line of sight)

experiments. Figure 7c and d plot the corresponding aggregate link rate across time, which follows the same trend as the ESNR [16].[8] The inset plots in Fig. 7c and d depict the final converged position of the routers (blue dots) in LOS and NLOS. The results show that our system consistently satisfies client demands while adapting to real-time changes in wireless channels, even in the presence of obstacles.

5.4 *Comparison with Existing Schemes*

We test our method against two other popular approaches to the communication problem in robotics: (1) Euclidean Disk Model as used in [1, 2], where communication constraints are in terms of Euclidean distance; (2) Stochastic Gradient Approach, where we implement the Simultaneous Perturbation method (SPSA) [11] for estimating the gradient of signal power by sampling the ESNR (which provides greater granularity than RSSI), along randomized directions, similar to the approach utilized by [10]. For the generation of each direction in the SPSA method we use a Bernoulli random variable (as in [11]) and diminishing step sizes satisfying the conditions stated in [11] for convergence. Our largest step size was allowed to be the same maximum vehicle velocity of v_c for all experiments. We consider a robotic router and three clients, each with an ESNR demand of 20 dB. We repeat the experiment five times in the non-line-of-sight environment in Fig. 8b–d. In each instance, we measure $r_{\max}$, the maximum ratio of ESNR demand versus the ESNR achieved among all three clients. In particular, $r_{\max}$ is below one at the converged position (i.e. all client demands are satisfied), and above one otherwise.

Results: Figure 8a plots the aggregate mean and variance of $r_{\max}$ across time, for all the three approaches. Figure 8b–d shows a candidate trajectory adopted by the robotic router for the three schemes. The plots demonstrate while the disk model converges quickly to a solution, ignorance of the wireless channels leads to solutions not meeting client demands; especially in non-line-of-sight settings. In contrast, the stochastic gradient approach (in blue), which sample the instantaneous ESNR, eventually satisfies network demands. However, the convergence is often laborious as the router often traverses counter-productive directions (see Fig. 8c). Indeed such techniques are noisy at low signal power, as even a large change in distance translates to a small change in signal power (a well-studied problem in communications literature [22–24]). Figure 8c shows that this leads to areas at non-line-of-sight or far distances from the client, where the robot easily gets lost.

Our method leverages full information of the channel, including signal power and phase, to find the direction of signal power as opposed to its magnitude. The result is an algorithm that converges to positions that satisfy network demands while not necessitating counter-productive exploration steps of a pure sampling approach.

[8]Note that the data-rate is capped by 60 Mb/s causing the plot to appear flat at times unlike ESNR.

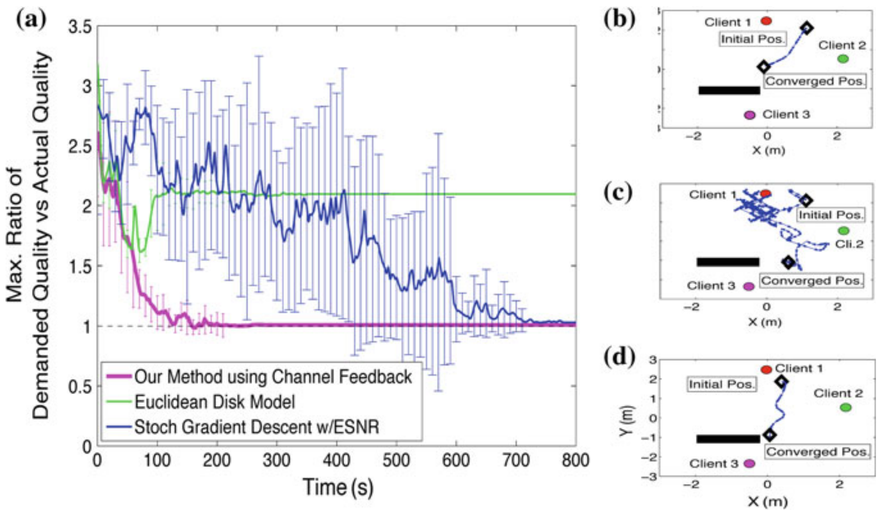


Fig. 8 Plots comparing our method against the Euclidean disk model and a stochastic gradient descent method based on ESNR. Our method both converges to a position that meets communication demands, and converges quickly along an efficient path. **a** Comparison. **b** Euclidean disk model. **c** Stochastic method. **d** Our method

5.5 Robustness to Dynamic Obstacle Positions

We evaluate how our system adapts to changes in the environment without an *a priori* known map. Consider two robotic routers and three clients in an environment with an obstacle located initially as shown in Fig. 9a. We allow the robot routers to navigate to their converged positions. At $t = 120$ s, we move the obstacle to a different location as in Fig. 9c, and let the routers re-converge.

Results: Figure 9b and c depict the converged position of the routers before and after the obstacle was moved. Figure 9d plots the data-rate across time for each client. The plot shows that our system satisfies client demands at the initial position. Further it recovers from the sharp fall in data-rate to one of the clients to successfully re-converge after the obstacle is moved.

5.6 Complex Indoor Environments

We evaluate our system in a large complex indoor environment with concrete walls and columns. We place a robotic router and client and line-of-sight (LOS) and non-line-of-sight (NLOS) as in Fig. 10. We trace the router's gradient field towards the client starting from multiple initial positions.

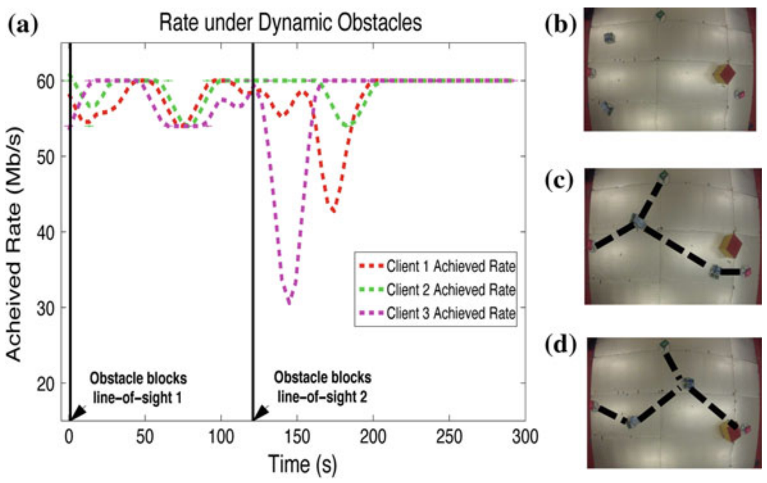


Fig. 9 These plots show the result of disturbing the wireless channels via movement of a line-of-sight obstructing obstacle. Actual testbed snapshots are shown on the right. **b** Initial position. **c** Obstacle pos. 1. **d** Obstacle pos. 2

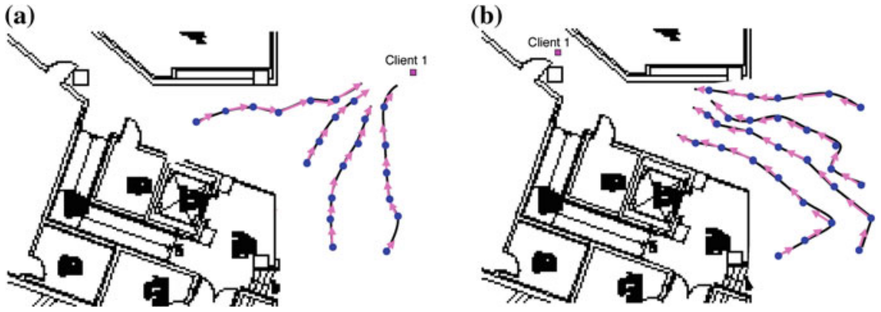


Fig. 10 Trajectories using measured $\mathbf{v}_{\theta_{\max}}$ directions satisfy a client's demand in line-of-sight and non-line-of-sight settings in complex indoor environments. **a** Line of sight. **b** Non line of sight

Results: Figure 10a, b plot of candidate trajectories (from gradient field) in LOS and NLOS across initial locations. The plots show that our system successfully navigates towards the client to satisfy its demands, without knowledge of the environment or client location.

6 Conclusion

In this paper, we present a framework to satisfy real-time variable communication demands for a changing network. We develop a solution enabling a robotic receiver to find the profile of signal strength across spatial directions for each sender of interest.

While our technique retrieves these spatial signal profiles in real time, we note that it faces an important limitation: it assumes access to wireless channels from both the transmitter and the receiver. Developing a system that can work with unmodified transmitters remains an open challenge. Our system integrates the signal profiles with a controller that optimizes communication quality while maintaining quadratic edge costs, and thus has natural extensions to many communication-aware coordination problems such as coverage [1], consensus [3], formation control [2], etc. We believe our system provides the necessary robustness to bring the benefits of these important contributions to practical robotic systems.

Acknowledgments We thank Dan Feldman and Brian Julian for experimental and theoretical contributions to this work. The authors acknowledge MIT Lincoln Laboratory and MAST project under ARL Grant W911NF-08-2-0004 for their support.

References

1. Ganguli, A., S. Susca, A., Martinez, S., Bullo, F., Cortes, J.: On collective motion in sensor networks: sample problems and distributed algorithms. In: CDC-ECC (2005)
2. Jadbabaie, A., Lin, J., Morse, A.S.: Coordination of groups of mobile autonomous agents using nearest neighbor rules. In: IEEE Transactions on Automatic Control (2003)
3. Olfati-Saber, R., Fax, J.A., Murray, R.M.: Consensus and cooperation in networked multi-agent systems. In: Proceedings of the IEEE (2007)
4. MalmirChegini, M., Mostofi, Y.: On the spatial predictability of communication channels. IEEE Trans. Wirel. Commun. **11**(3) (2012)
5. Yan, Y., Mostofi, Y.: Co-optimization of communication and motion planning of a robotic operation under resource constraints and in fading environments. IEEE Trans. Wirel. Commun. **12**(4) (2013)
6. Fink, J., Ribeiro, A., Kumar, V.: Robust control for mobility and wireless communication in cyber-physical systems with application to robot teams. In: Proceedings of the IEEE (2012)
7. Lindh, M., Johansson, K., Bicchi, A.: An experimental study of exploiting multipath fading for robot communications. In: RSS (2007)
8. Aruba Networks. Outdoor antennas and RF coverage strategies http://www.arubanetworks.com/vrd/outdoormimovrd/wwhelp/wwhimpl/common/html/wwhelp.htm#context=OutdoorMIMOVRD&file=chap4.html
9. Fitch, P.J.: Synthetic Aperture Radar. Springer, New York (1988)
10. Le Ny, J., Ribeiro, A., Pappas, G.J.: Adaptive communication-constrained deployment of mobile robotic networks. In: ACC (2012)
11. Spall, J.C.: Adaptive stochastic approximation by the simultaneous perturbation method. In: IEEE Transactions on Automatic Control (2000)
12. Kumar, S., Shi, L., Ahmed, N., Gil, S., Katabi, D., Rus, D.: Carspeak: a content-centric network for autonomous driving. SIGCOMM (2012)
13. Wang, J., Katabi, D.: Dude, where's my card?. RFID positioning that works with multipath and non-line of sight, In: SIGCOMM (2013)
14. Wang, J., Adib, F., Knepper, R., Katabi, D., Daniela, R.: Robot object manipulation using rfids. In: MobiCom, RF-Compass (2013)
15. Adib, F., Katabi, D.: See through walls with wi-fi. In: SIGCOMM (2013)
16. Halperin, D., Hu, W., Sheth, A., Wetherall, D.: Predictable 802.11 packet delivery from wireless channel measurements. In: CCR (2010)

17. Rahul, H., Kumar, S.S., Katabi, D.: Scaling wireless capacity with user demand. In: SIGCOMM, Megamimo (2012)
18. Stoica, P., Moses, R.L.: Spectral Analysis of Signals. Prentice Hall, New Jersey (2005)
19. Gil, S., Feldman, D., Rus, D.: Communication coverage for independently moving robots. In: IROS (2012)
20. Feldman, D., Gil, S., Knepper, R., Julian, B., Rus, D.: K-robots clustering of moving sensors using coresets. In: ICRA 2013 (2013)
21. Halperin, D., Hu, W., Sheth, A., Wetherall, D.: Tool release: Gathering 802.11n traces with channel state information. In: ACM SIGCOMM CCR (2011)
22. Chen, H.-C., Lin, T.-H., Kung, H.T., Lin, C.-K., Gwon, Y.: Determining RF angle of arrival using cots antenna arrays: A field evaluation. In: MILCOM (2012)
23. Xiong, J., Jamieson, K.: Arraytrack: a fine-grained indoor location system. In: NSDI (2013)
24. Joshi, K., Hong, S., Katti, S.: Pinpoint: localizing interfering radios. In: NSDI (2013)

Multi-vehicle Dynamic Pursuit Using Underwater Acoustics

Brooks Reed, Josh Leighton, Milica Stojanovic and Franz Hover

Abstract Marine robots communicating wirelessly is an increasingly attractive means for observing and monitoring the ocean, but acoustic communication remains a major impediment to real-time control. In this paper we address through experiments the capability of acoustics to sustain highly dynamic, multi-agent missions, in particular range-only pursuit in a challenging shallow-water environment. We present in detail results comparing the tracking performance of three different communication configurations, at operating speeds near 1.5 m/s. A "lower bound" case with RF wireless communication, a 4-second cycle and no quantization has a tracking bandwidth of $\approx$0.5 rad/s. When using full-sized modem packets with negligible quantization and a 23-second cycle time, the tracking bandwidth is $\approx$0.065 rad/s. With 13-bit mini-packets, we employ logarithmic quantization to achieve a cycle time of 12 s and a tracking bandwidth of $\approx$0.13 rad/s. These outcomes show definitively that aggressive dynamic control of multi-agent systems underwater is tractable today.

1 Introduction

Marine robots have played an increasing role in ocean operations during recent years, with the proliferation of many commercial platforms and sensors. A major trend is toward tetherless operation, for which each vehicle has to carry its own power

B. Reed (✉)
MIT/WHOI Joint Program in Oceanographic Engineering, Cambridge, USA
e-mail: brooksr8@mit.edu

J. Leighton · F. Hover
Massachusetts Institute of Technology, 77 Massachusetts Ave., Cambridge, MA 02139, USA
e-mail: jleight@mit.edu

F. Hover
e-mail: hover@mit.edu

M. Stojanovic
Northeastern University, 360 Huntington Ave., Boston, MA 02115, USA
e-mail: millitsa@ece.neu.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_5

source and have a means of wireless communication. Over distances beyond about one hundred meters, underwater communication is almost exclusively accomplished through acoustics. Acoustic communications bring many challenges, however, such as packet loss, low data rates, and delays; Heidemann et al. provide a recent review [24]. These undesirable properties of acoustic communication have limited its use in high-performance, real-time tasks. Typical experiments with acoustic modems and vehicles address packet loss rates [7, 9], distributed navigation [1], and command and control of vehicles from ships [30].

If a capability existed, truly dynamic missions of interest would include networked ocean vehicles following a submarine or a marine animal; the latter has been a dream of biologists for decades. Major gaps exist in our understanding of the life cycles of jellyfish [29], sharks [32, 37], lobsters [35], and more. A broader and more challenging problem is monitoring and following a quickly-evolving plume or other oceanic process [8, 19], where distributed measurements must be combined into an estimate, potentially taking into account prior model information [26]. These tasks involve *dynamic feedback control that relies explicitly on acoustic communication*, and fit into the growing field of network-based control, as described in a recent review by Baillieul et al. [2].

In an effort to lay some groundwork for exploiting advanced algorithms in a real-world ocean application, this paper addresses with experiments an approach for joint estimation and pursuit of a moving target using acoustic communications; see Fig. 1. Needless to say, the general pursuit problem has held high interest for

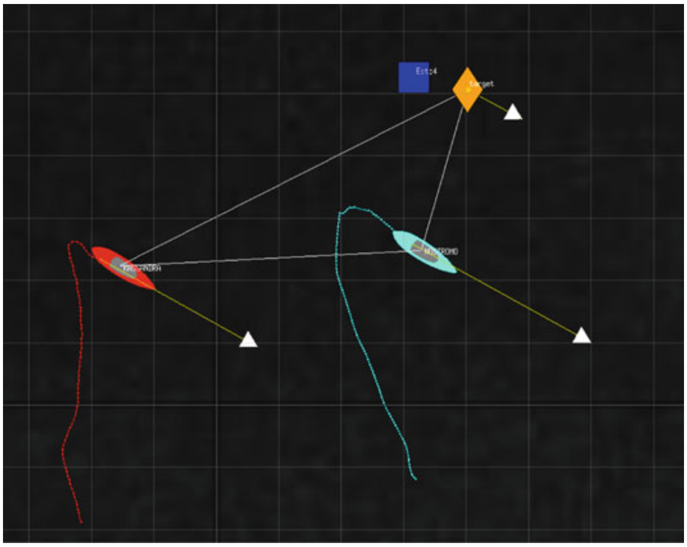


Fig. 1 Screenshot from an active localization and pursuit experiment with acoustic communications. The two vehicles jointly estimate the target location based on range measurements, and move to stay in formation relative to it

decades; it is a canonical mission in space and air, on land, and at sea. Probabilistic pursuit-evasion games have been studied extensively in the robotics literature [36], and pursuer and evader dynamics as well as nonlinear estimation are important factors in these algorithms [27, 38]. The effects of communication constraints however have not received much attention [28]. These are often addressed indirectly via decentralized approaches that require minimal exchange of information between agents [11]; see [17, 22] for ocean-specific implementations.

There have been some recent experimental works that are related to our pursuit scenario. Perhaps most intriguing is tracking a leopard shark in extremely shallow water, using a single autonomous vehicle with a two-element hydrophone array [12]. The system was successful but the shark evidently moved only 200 m or so in 48 min reported. Bean et al. studied range-based leader—follower regulation with Micro-Modem mini-packets and 1 m/s speeds [3], while Brignone et al. looked at a similar problem with DSPComm modems and two vehicles operating at 0.7 and 3 m/s [6]. Both works present data from proof-of-concept field trials with mostly straight trajectories. Soares et al. consider a vehicle following two leaders in a triangle formation, with ranges of about fifteen meters, speeds around 0.5 m/s, and a total loop time of 4 s [33]. In contrast, Cruz et al. consider a complete feedback system—in the sense of two-way communications—for which a stationary controller transmits commands for two mobile followers, who then transmit back their positions [13]. The vehicle speeds are slow, in the neighborhood of 10 cm/s, and the cycle time is around 20 s. Through analysis, Chen and Pompili addressed optimization of acoustic communications in coordinated flight of ocean gliders, where currents are especially important [10].

None of these prior works explicitly deal with designing and improving closed-loop frequency response of an integrated multi-vehicle feedback system. This is exactly our objective here. Our design does not rigorously account for stability margins, the multi-rate nature of acoustic communications, inherent geometric nonlinearities, or the fact that autonomous marine vehicles are not ideal actuators. On the other hand, our approach demonstrates practical closed-loop performance at half the Nyquist rate, with little evidence of stability breakdown.

We detail the experiment setup in the following section with descriptions of the vehicles and communication hardware used, the experimental domain, and the estimation and control strategies and parameters. We then give results from three integrated tests, demonstrating the performance achieved.

2 Experimental Setup

Out experiment in joint localization and pursuit has two mobile agents sharing sensor information and commands through acoustic links. We make scalar range measurements at each agent, and thus tracking is impossible without their coordination. One agent is designated as the leader that coordinates the measurements and the actions of the follower. This arrangement involves lossy channels at both locations in the

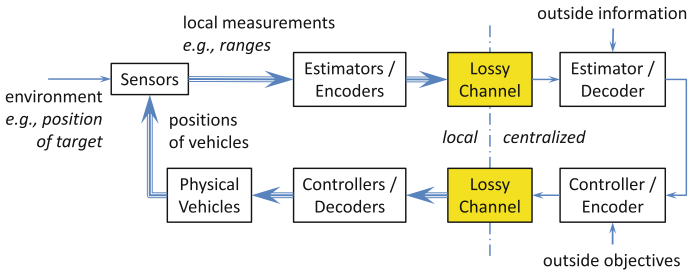


Fig. 2 Block diagram of a generic multi-vehicle feedback system with a centralized estimator and controller, and communication channels at two locations within the loop. Vehicles act as mobile sensors

feedback loop of Fig. 2. In the general case, a centralized architecture such as this allows integration with remote sensing, large-scale computations (such as data assimilation), and human-in-the-loop decision-making. The mobile agents attempt to stay close to the target, and in a formation conducive to good sensor performance.

The next five subsections detail the arrangement and operation of this system.

2.1 Autonomous Surface Vehicles

We use autonomous kayaks as shown in Fig. 3 for our experiments; they are also described in [23]. Each craft is 1.8 m long, weighs about 40 kg, and has a rotating thruster near the bow for propulsion and steering. In these tests, the vehicles operate

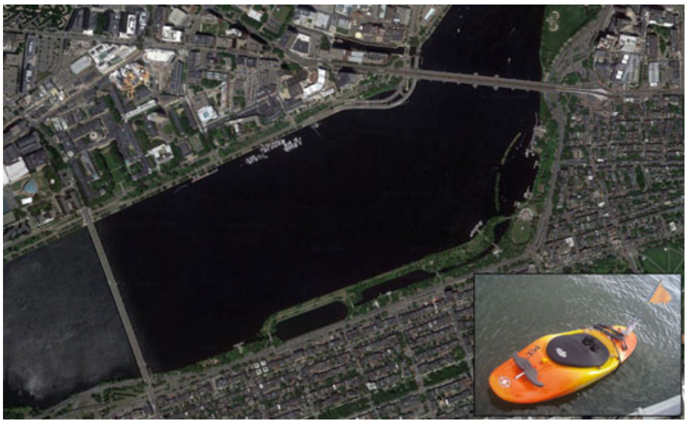


Fig. 3 The Charles River Basin in Cambridge/Boston, MA, and the autonomous kayak Nostromo. Water depth is 2–12 m

at a nominal speed of $V = 1.5\,\text{m/s}$. The relevant navigation sensors available on each vehicle are a tilt-compensated compass and RTK GPS. We use Novotel GPS antennas, uBlox GPS receivers, and the RTKlib software package [34], and have observed position variances on the order of $10^{-4}\,\text{m}^2$. Raw compass measurements are passed through a first-order low-pass filter with time constant 2 s, and the noise variance on this signal is estimated as $10\,\text{deg}^2$.

The vehicles run MOOS-IvP autonomy software [4] integrated with custom control algorithms and modem interfaces. We rely on the the MOOS heading PID controller, which runs at 5 Hz, and the MOOS trackline controller, which runs at 2 Hz. Step response experiments with the kayak under closed-loop heading control indicate a rise time of roughly 4 s, and 30 % overshoot; we also note the kayaks are able to turn 180° in approximately 3 s. The MOOS trackline controller is an inner-outer loop that modulates the desired vehicle heading so as to steer it toward a point on the trackline, some lead distance l_d ahead. When the waypoint is closer than the lead distance, the vehicle simply drives towards the waypoint. For longer distances the result for small errors is a proportional map for desired heading: $\phi^d \simeq e_x/l_d$, where e_x is the cross-track error in meters and ϕ^d is in radians.[1] We set $l_d = 15\,\text{m}$ for these experiments.

2.2 Acoustic Communications

We use the the WHOI Micro-Modem [20], a well-established and commercially available technology for underwater acoustic communication. Modems are towed by the vehicles, suspended at a depth of about 1.5 m; this gives us realistic shallow-water acoustic performance, but with direct access to GPS and RF wireless connectivity at the surface for conducting controlled tests. Along with messaging, we use the modem for one-way travel-time ranging [18]. For messaging, the Micro-Modem has six different packet types with different lengths and data capacities. In this work, we use the FSK mini-packet ("MP"), which is regarded as the most robust of the packet types, but contains only thirteen bits of information. The mini-packets take slightly over 1 s to transmit. We also use the full-sized Rate 0 FSK packets ("FSK0"), which carry thirty-two bytes of information and take approximately 5 s to transmit. We have observed very large increases in packet loss when using small guard times with both packets, and have found communications to be most reliable with 4-second slots for mini-packets and 9.5-second slots for FSK0 packets. All Micro-Modem packets are sent with an acoustic source level of 190 dB rel µPa.

The Charles River Basin has fresh water 2–12 m deep, a complex bathymetry, and some hard surfaces on the boundaries (seawalls and bridges); our working space is about 1500 m long and 500 m wide. Acoustic performance in this environment is

[1]The linear form written is based on approximation of the tangent function. For errors less than 1 m, the MOOS Trackline controller increases the lead distance proportionally, effectively lowering the gain to limit oscillations.

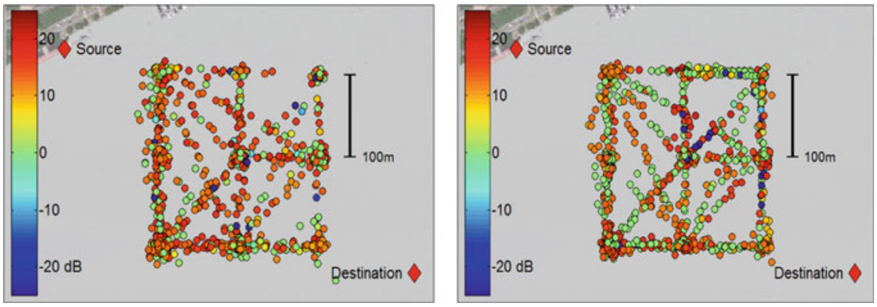


Fig. 4 Micro-Modem performance data in the Charles River Basin, an environment limited by multipath, not power. The *left plot* shows transmissions from the source to a mobile relay, and the *right plot* shows transmissions from the relay to the destination. The SNR value indicates sound pressure level relative to ambient noise

different from an open-water deep ocean scenario, where multipath and reverberation are much lower, but the ranges are higher. Operations in the Basin can have highly variable acoustic performance, as shown in Fig. 4. Our conditions are multipath-limited and travel times are short.

2.3 *Physical Layout*

The two-vehicle pursuit mission encompasses limited communication performance in both the sensing and control channels. In this experiment there is a target to be tracked, “Icarus”, and two cooperating agents “Silvana” and “Nostromo”. We will denote these three nodes with the symbols $\mathcal{I}$, $\mathcal{S}$, and $\mathcal{N}$, respectively. $\mathcal{N}$ can be thought of as a leader, and $\mathcal{S}$ a follower. The sensing objective is a simple one: to maintain $\mathcal{S}$ and $\mathcal{N}$ in a fixed triangular configuration relative to the estimated location of $\mathcal{I}$, so that measurements will be of high fidelity, i.e., in the sense of a good HDOP [5], and in the sense of a short range. Our pursuit arrangement models the general situation where range or other target sensing degrades with distance, but a high level of tracking precision is desired. Maintaining a close pursuit formation keeps ranges close to a nominal value, allowing for more precise quantization.

An “unstable” situation is encountered if the target crosses the baseline (the line in between the two vehicles acting as a moving long baseline network)—the estimate begins to diverge from the target location. Thus, a major disadvantage of a small pursuit formation is that it is easier for the target to cross the baseline, bringing up a key tradeoff between robustness of a larger formation and accuracy of a smaller formation (which requires good closed-loop performance).

2.4 Cycle Description, Timing and Quantization

We detail the stages of the control loop for the MP and FSK0 cases. Within a cycle step, $\mathcal{S}$ and $\mathcal{N}$ each receive a measurement of range to $\mathcal{T}$ via the Micro-Modems in ranging mode. After a guard period, $\mathcal{S}$ transmits its current location and range data to $\mathcal{N}$ through acoustic communication. $\mathcal{N}$ combines this information with its own location and range information to generate an estimated location of $\mathcal{T}$. $\mathcal{N}$ calculates control actions for itself and for $\mathcal{S}$, and transmits the latter back to $\mathcal{S}$. The cycle includes three separate transmissions and there are no acknowledgments. We enforce the fixed time slots with a number of timeouts, as indicated in Fig. 5. We synchronize clocks using the network time protocol; in the absence of clock synchronization, we note that precision clocks are becoming increasingly practical for use on underwater vehicles [18].

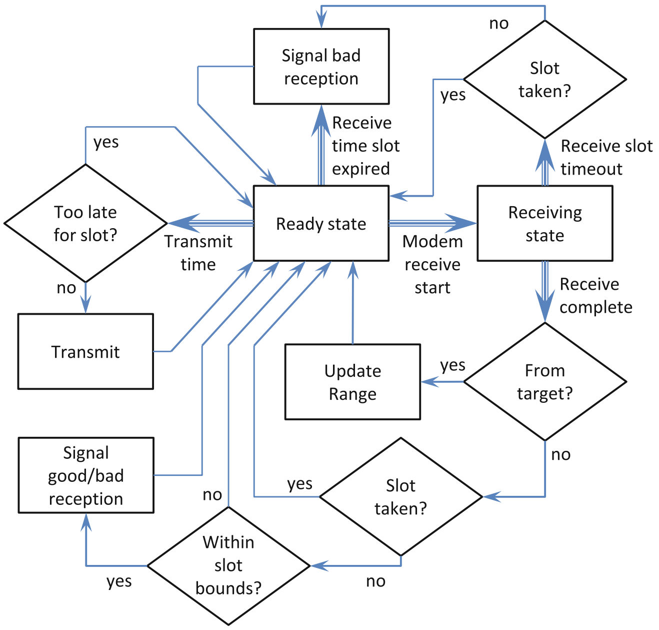


Fig. 5 The internal state machine used on each vehicle to maintain consistent timing with respect to predefined transmission and reception slots. *Thick arrows* distinguish acoustic events that initiate state changes or other actions from normal logic flow. Special operations are indicated to handle detection of erroneous multipath receptions, which frequently occur in this environment. For example, a good reception for a time slot T_i will follow the "Receive complete" path (*bottom*) to a good signal. A trailing multipath reception will return to the receiving state, but the end of time slot T_i will arrive before the end of the packet. In the *top right*, slot T_i is already taken by the good reception, so we return to the ready state with no action taken

For feedback control, there is a problem-dependent tradeoff to be made between time-averaged throughput (usually achieved with long coding blocks) and timeliness of the information (shorter messages). We present data using both 13-bit mini-packets and 32-byte FSK0 packets as an initial study of this tradeoff. The MP scenario minimizes cycle time at the expense of data quantization; we achieved a total cycle time of 12 s in this configuration.[2] With the FSK0 configuration, packets require no quantization for the data types we send, but do require a 9.5 s time slot for each transmission, resulting in a total cycle time of 23 s.[3] The "wifi" scenario involves a 4 s slot for acoustic ranging, as detailed above. However, the inter-vehicle communications are handled instantaneously via wifi, so the estimate is available immediately upon reception of ranges.

For the message from $\mathcal{S}$ to $\mathcal{N}$ in the MP case, we used three bits for the range, and five bits each direction for $\mathcal{S}$'s location in a 32×32 discretized workspace; this workspace had ten-meter resolution. The range data were logarithmically quantized relative to a desired range of 50 m, with seven bin edges located at [19.2 32.5 42.5 50 57.5 67.5 80.8] m, and the three-bit messages decoded as [11.5 26.8 38.2 46.8 53.2 61.8 73.2 88.5] m. This correlates with the density $\rho = 0.75$ [21]. For the message from $\mathcal{N}$ back to $\mathcal{S}$, we used five bits in each of x and y for the desired location in the workspace. This left three bits unused. This quantization makes stark the tradeoff between range and precision. Any range larger than 80.8 m is decoded as the furthest range bin, so when ranges are very large, estimation suffers. Increasing this outer range would come at the expense of resolution of the bins near the 50 m nominal range; it is the control system's job to keep the vehicles in the desired formation so that small bins can be used.

2.5 Settings and User Choices

The tracking system contains a nonlinear sigma-point filter (SPF) [25], well-suited for this type of application.[4] The nonholonomic target $\mathcal{T}$ (a small motorboat) was assumed to be moving at constant 1.55 m/s, with stochastic low-pass, zero-mean turning rate having variance Q. The observation vector contains the two noisy ranges, with variances $R_{\mathcal{S}}$ and $R_{\mathcal{N}}$ for range measurements to Silvana and Nostromo, respectively. The range sensor noise was chosen based on prior characterizations of the WHOI Micro-Modem ranging capability [14, 20] and our own observed LBL performance. The sensor noise for the follower range measurement ($\mathcal{T}$ to $\mathcal{S}$) in the MP experiment was set to a higher value to account for the effects of quantization during

[2]When range measurements do not interfere with modem packets and the cycle consists of just two-way communications (e.g. using GPS and wifi for ranges), we have achieved a 6-second total cycle time with mini-packets in the field.

[3]As we were submitting this paper we became aware of several modifications in the operation of the Micro-Modems that likely will allow for slightly faster cycle times.

[4]Other nonlinear, range-only filters, such as particle filters, could also be used [15].

Table 1 Settings and results for the three configurations

Config	Cycle time (s)	DesRange (m)	$R_{\mathcal{S}}$ (m^2)	$R_{\mathcal{N}}$ (m^2)	Q (rad/s)^2	BW (rad/s)	Atten (dB)
FSK0	23	100	0.25	0.25	0.01	0.065	0
Wifi	4	50	0.25	0.25	0.05	0.5	18
MP	12	50	0.25	9	0.05	0.13	7

DesRange is the length of the legs in the desired sensing formation. The columns with R are the sensor noise variances for the range measurements to each vehicle. Q is the target heading rate variance. BW is the closed-loop tracking bandwidth, and Atten is the tracking error attenuation at 0.065 rad/s. Also see Fig. 9

communication of the measurement from $\mathcal{S}$ to the filter running on $\mathcal{N}$. Settings for the three configurations are given in Table 1.

When a measurement is not available (either due to a missed LBL range, or a dropped measurement packet from $\mathcal{S}$ to $\mathcal{N}$), we take the standard approach of setting the noise of the lost measurement to infinity [31]. In the MP and FSK0 configurations, when a control command from $\mathcal{N}$ to $\mathcal{S}$ is dropped, the previously-received command for $\mathcal{S}$ remains the desired waypoint. This approach ensures safe operation in the case of many missed packets. In the MP case, three bits are left unused in the command packet which could encode contingency plans.

The desired observation triangle has a sixty-degree vertex at $\mathcal{T}$. For the MP and wifi cases, the ranges to each of $\mathcal{S}$ and $\mathcal{N}$ were 50 m; for the FSK0 case the desired ranges were 100 m due to the slower cycle time.[5]

3 Experimental Results

We compare the tracking performance of three different communication configurations: full-sized packets ("FSK0") with negligible quantization and a 23 s cycle, RF wireless communication ("wifi") with a 4 s cycle, and 13-bit mini-packets ("MP") with a 12 s cycle. The "wifi" configuration roughly represents a single vehicle towing a long two-element array, as inter-vehicle communication is lossless and immediate. However, a true single-vehicle with array would be far less maneuverable than vehicles without arrays, and could not pursue the target as closely without risking the target crossing the baseline. For close pursuit with multiple vehicles, we can view the "wifi" case as a lower bound on performance.

The experiments we report were conducted on 8–9 July 2013, both days with light winds.[6] Figures 6, 7, 8 give results from the FSK0, wifi and MP tests, respectively. In each test, $\mathcal{T}$ moved in a largely random trajectory, as shown in the birds-eye

[5]The ranges are set relative to the distance the target can drive in a time step, so that the target is unlikely to cross the baseline before the control system can react.

[6]This data set, along with videos, is publicly available at http://web.mit.edu/hovergroup/resources.html.

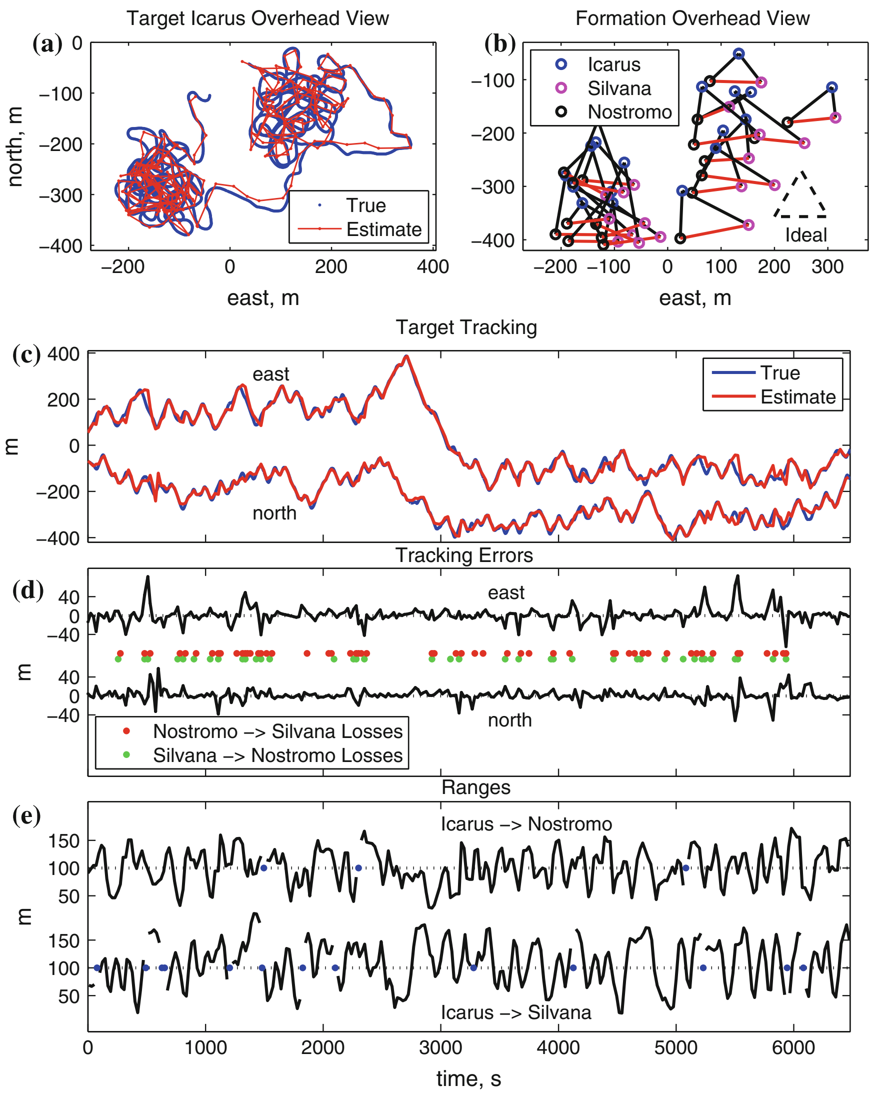


Fig. 6 FSK0 test results (6463 s, 281 cycles). **a** Overview of true and estimated trajectories of the target Icarus. **b** Sensing formation every 15 time steps. **c** Actual (GPS) and estimated trajectory of target Icarus. **d** Estimation error of Icarus' location. The RMS radius of estimation errors was 20.2 m. Data packet losses are also shown; loss rates were: $\mathcal{N} \rightarrow \mathcal{S} = 19.9\,\%$, $\mathcal{S} \rightarrow \mathcal{N} = 14.0\,\%$. **e** Range measurements from Icarus to each kayak, and losses. Range loss rates were: $\mathcal{I} \rightarrow \mathcal{N} = 1.1\,\%$, $\mathcal{I} \rightarrow \mathcal{S} = 4.8\,\%$

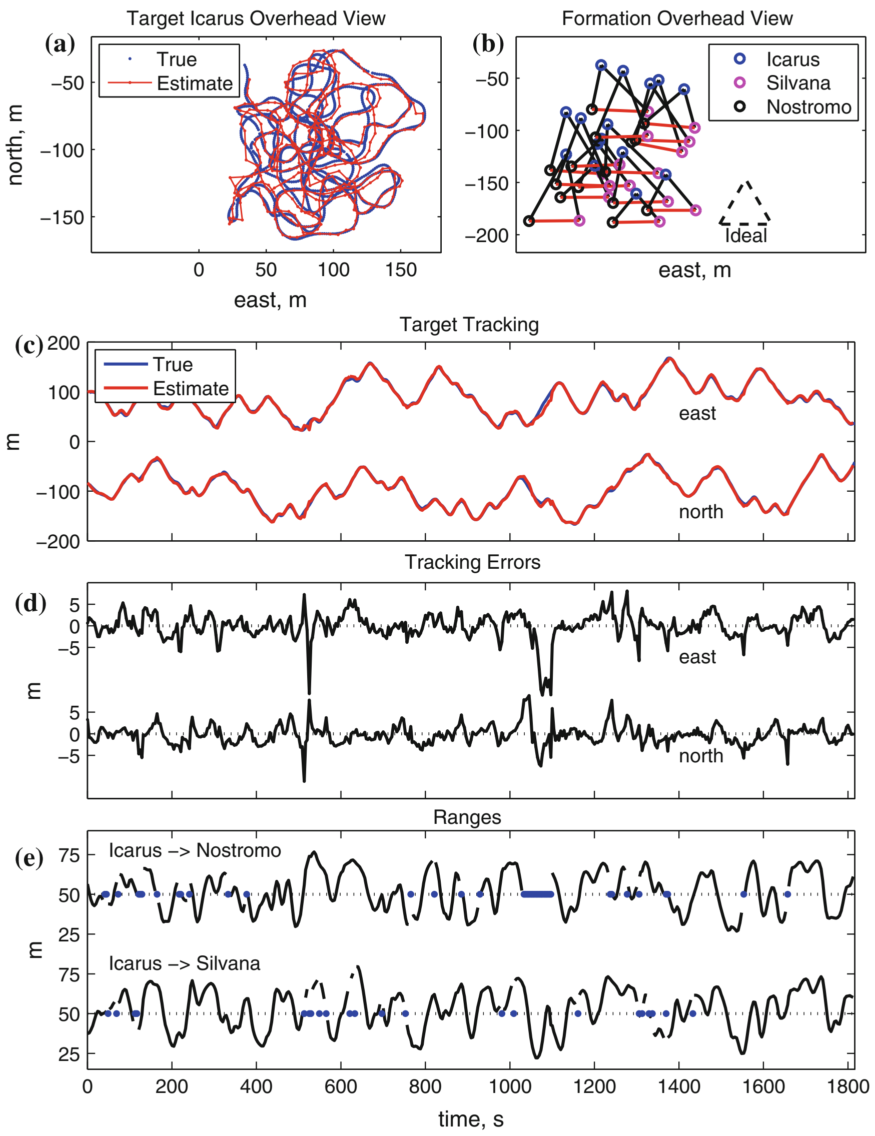


Fig. 7 Wifi test results (1820 s, 455 cycles). **a** Overview of the true and estimated trajectories of the target Icarus. **b** Sensing formation every 30 time steps. **c** Actual (GPS) and estimated trajectory of the target Icarus. **d** Estimation error of Icarus' location. The RMS radius of estimation errors was 3.8 m. **e** Range measurements from Icarus to each vehicle, and losses. Range loss rates were: $\mathcal{I} \rightarrow \mathcal{N} = 9.0\,\%$, $\mathcal{I} \rightarrow \mathcal{S} = 4.8\,\%$

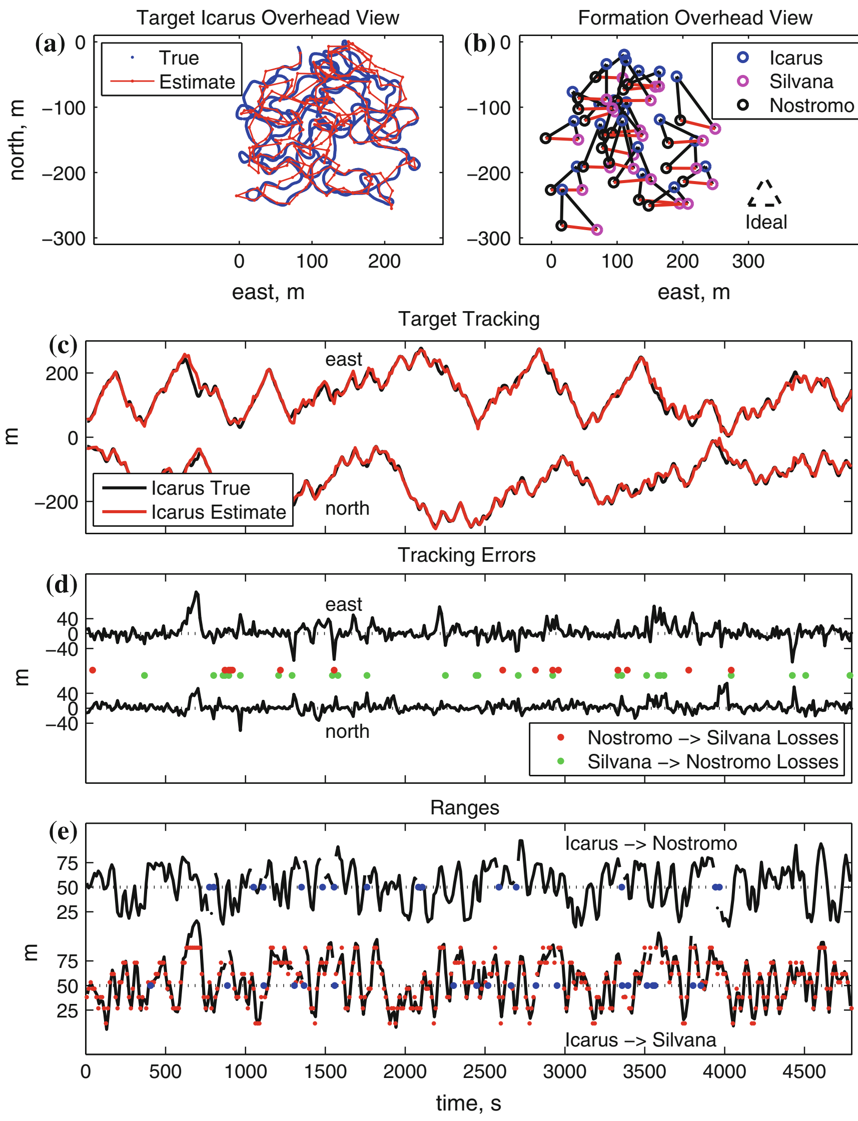


Fig. 8 MP test results (4800 s, 400 cycles). **a** Overview of the true and estimated trajectories of the target Icarus. **b** Sensing formation every 15 time steps. **c** Actual (GPS) and estimated trajectory of the target Icarus. **d** Estimation error of Icarus' location. The RMS radius of estimation errors was 12.7 m. Data packet losses are also shown; the loss rates were: $\mathcal{N} \to \mathcal{S} = 3.8\,\%$, $\mathcal{S} \to \mathcal{N} = 6.5\,\%$. **e** Range measurements from Icarus to each vehicle, and losses. Range loss rates were: $\mathcal{I} \to \mathcal{N} = 3.8\,\%$, $\mathcal{I} \to \mathcal{S} = 4.8\,\%$. Quantized measurements sent from Silvana to Nostromo are shown in *red* on *top* of the true measured ranges

view in the upper left (Subplot **a**) and the time traces in Subplot **c**. The upper right (Subplot **b**) shows the sensing formation every fifteen time steps; we see that while the ideal triangle configuration was rarely achieved in the FSK0 and MP tests, the target did not cross the baseline (the red straight line between the two nodes acting as a moving LBL network), nor did the geometry ever stay poor for a sustained period. The tracking and pursuit system did not lose the target.

The measured ranges are reported in Subplot **e** in each figure, including quantization of raw values sent to $\mathcal{N}$ from $\mathcal{S}$ in the subsequent measurement packet for the MP case. Range losses in all cases are low, as the Micro-Modem ranging ping is fairly robust; see figure captions for loss statistics. Subplot **d** shows the north and east tracking errors over time, along with dropped communication packets for the MP and FSK cases. The packet losses are significantly higher for the FSK0 test. Most of the larger errors occur following packet losses, but some large spikes (such as around 500 s in the mini-packet test) are not near packet losses—errors can also occur due to poor sensing geometry, and in the MP case, quantization.

Recalling our broad objective to achieve dynamic control through mobile acoustic networks, it is revealing to ask what is the effective closed-loop estimation bandwidth achieved. A direct FFT-based empirical transfer function for the estimation error divided by target motion is shown for each test in Fig. 9; spectra have been smoothed with a 5-point centered moving average. The FSK0 test has a break frequency for tracking the motion of $\mathcal{T}$ at approximately 0.065 rad/s, slightly less than half the Nyquist rate for the 23 s cycle. The wifi test has a break frequency of approximately 0.5 rad/s. The MP test has a break frequency of approximately 0.13 rad/s. We can also compare the attenuation of tracking error for each configuration at 0.065 rad/s. FSK0 has zero attenuation, wifi has 18 dB attenuation, and MP has 7 dB.

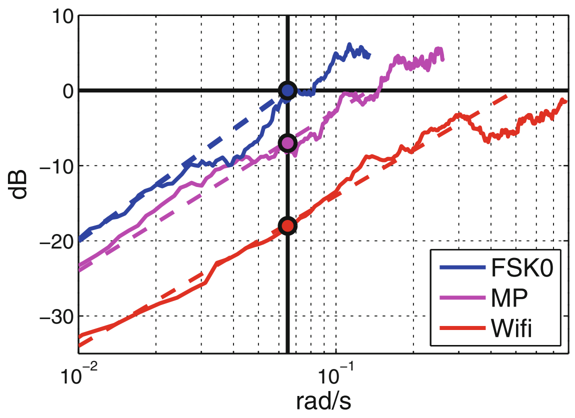


Fig. 9 Empirical FFT-based transfer function for estimator error divided by target motion. The *solid lines* show the mean of the X and Y spectra. The *dashed lines* show an approximate linear fit for low-frequency attenuation. *Dots* show the approximate attenuation at 0.065 rad/s

4 Conclusion

Our experiment has achieved aggressive target pursuit in the underwater environment. As opposed to a traditional control and estimation design scenario, the mission here is accomplished through a highly integrated vehicle system performing full joint estimation and coordination through lossy acoustic communications underwater. The three experimental configurations studied show the effects of cycle time, quantization, and reliability on the frequency response of the system. In particular, the MP and FSK0 experiments demonstrate that for tracking highly dynamic targets it is beneficial to trade-off quantization for low cycle time.

The pursuit mission presented in this paper is one special case of a much larger picture; we believe that undersea communications and coordinated control will soon enable truly distributed and dynamic tracking of moving ocean features, such as eddies, plumes and fronts. Such vehicle systems would be able to observe important chemical, biological, and physical processes over larger physical scales than a single vehicle can cover, and would interface with observation systems on land and in the atmosphere, as well as with humans. Operations like this—“oceanographic pursuit”—are a natural progression of marine technology toward the group autonomy and dynamic behavior that we have seen developed already in the terrestrial environment and in the air [16]. Specification of physical configurations, scheduling, routing, and multi-rate control design will undoubtedly join the mix, making underwater pursuit a rich problem for future work.

Acknowledgments Work is supported by the Office of Naval Research, Grant N00014-09-1-0700, the National Science Foundation, Contract CNS-1212597, and the Finmeccanica Career Development Professorship. We thank Mei Cheung for providing Fig. 4 and help with experimental implementation. We thank Toby Schneider and Mike Benjamin at MIT, and Keenan Ball and Sandipa Singh at WHOI, for their help on technical items. We also acknowledge MIT Sailing Master Fran Charles.

References

1. Bahr, A., Leonard, J., Fallon, M.: Cooperative localization for autonomous underwater vehicles. Int. J. Robot. Res. **28**(6), 714 (2009)
2. Baillieul, J., Antsaklis, P.: Control and communication challenges in networked real-time systems. Proc. IEEE **95**(1), 9–28 (2007)
3. Bean, T., Canning, J., Beidler, G., O’Rourke, M., Edwards, D.: Designing and implementing collaborative behaviors for autonomous underwater vehicles. In: Proceedings of the International Symposium on Unmanned Untethered Submersible Technology (UUST) (2007)
4. Benjamin, M., Leonard, J., Schmidt, H., Newman, P.: An overview of MOOS-IvP and a brief users guide to the IvP helm autonomy software. Massachusetts Institute of Technology, MIT CSAIL, Technical report TR-2009-28-07 (2009)
5. Bingham, B.: Predicting the navigation performance of underwater vehicles. In: Proceedings of the IEEE/RSJ Intelligent Robots and Systems (2009)

6. Brignone, L., Alves, J., Opderbecke, J.: GREX sea trials: first experiences in multiple underwater vehicle coordination based on acoustic communication. In: Proceedings of the MTS/IEEE OCEANS (2009)
7. Caiti, A., Calabro, V., Dini, G., Duca, A., Munafo, A.: AUVs as mobile nodes in acoustic communication networks: field experience at the UAN10 experiments. In: Proceedings of the MTS/IEEE OCEANS (2011)
8. Camilli, R., Reddy, C., Yoerger, D., Van Mooy, B., Jakuba, M., Kinsey, J., McIntyre, C., Sylva, S., Maloney, J.: Tracking hydrocarbon plume transport and biodegradation at Deepwater Horizon. Science **330**(6001), 201 (2010)
9. Canning, J., Anderson, M., Edwards, D., O'Rourke, M., Bean, T., Pentzer, J., Odell, D.: A low bandwidth acoustic communication strategy for supporting collaborative behaviors in a fleet of autonomous underwater vehicles. US Navy J. Underw. Acoust. **59**(3), 285–299 (2009)
10. Chen, B., Pompili, D.: Team formation and steering algorithms for underwater gliders using acoustic communications. Comput. Commun. **35**(9), 1017–1028 (2012)
11. Chung, T., Burdick, J., Murray, R.: A decentralized motion coordination strategy for dynamic target tracking. In: Proceedings of the IEEE International Conference on Robotics and Automation (2006)
12. Clark, C.M., Forney, C., Manii, E., Shinzaki, D., Gage, C., Farris, M., Lowe, C.G., Moline, M.: Tracking and following a tagged leopard shark with an autonomous underwater vehicle. J. Field Robot. **30**(3), 309–322 (2013)
13. Cruz, N.A., Ferreira, B.M., Matos, A.C., Petrioli, C., Petroccia, R., Spaccini, D.: Implementation of an underwater acoustic network using multiple heterogeneous vehicles. In: Proceedings of the MTS/IEEE Oceans (2012)
14. Curcio, J., Leonard, J., Vaganay, J., Patrikalakis, A., Bahr, A., Battle, D., Schmidt, H., Grund, M.: Experiments in moving baseline navigation using autonomous surface craft. In: Proceedings of the MTS/IEEE OCEANS (2005)
15. Dellaert, F., Fox, D., Burgard, W., Thrun, S.: Monte carlo localization for mobile robots. In: Proceedings of the International Conference on Robotics and Automation (ICRA), vol. 2, pp. 1322–1328. IEEE (1999)
16. Dunbabin, M., Marques, L.: Robots for environmental monitoring: significant advancements and applications. IEEE Robot. Autom. Mag. **19**(1), 24–39 (2012)
17. Eickstedt, D., Benjamin, M., Schmidt, H., Leonard, J.: Adaptive tracking of underwater targets with autonomous sensor networks. US Navy J. Underw. Acoust. **56**, 465–495 (2006)
18. Eustice, R.M., Singh, H., Whitcomb, L.L.: Synchronous-clock one-way-travel-time acoustic navigation for underwater vehicles. Special issue on state of the art in maritime autonomous surface and underwater vehicles, J. Field Robot. **28**(1), 121–136 (2011)
19. Farrell, J., Pang, S., Li, W., Arrieta, R.: Chemical plume tracing experimental results with a REMUS AUV. In: Proceedings of the MTS/IEEE OCEANS, vol. 2, pp. 962–968. IEEE (2003)
20. Freitag, L., Grund, M., Singh, S., Partan, J., Koski, P., Ball, K.: The WHOI micro-modem: an acoustic communications and navigation system for multiple platforms. In: Proceedings of the MTS/IEEE OCEANS (2005)
21. Fu, M., Xie, L.: The sector bound approach to quantized feedback control. IEEE Trans. Autom. Control **50**(11), 1698–1711 (2005)
22. Gadre, A., Maczka, D., Spinello, D., McCarter, B., Stilwell, D., Neu, W., Roan, M., Hennage, J.: Cooperative localization of an acoustic source using towed hydrophone arrays. In: Proceedings of the IEEE/OES Autonomous Underwater Vehicles, pp. 1–8 (2008)
23. Gilbertson, E., Reed, B., Leighton, J., Cheung, M., Hover, F.: Experiments in dynamic control of autonomous marine vehicles using acoustic modems. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2013)
24. Heidemann, J., Stojanovic, M., Zorzi, M.: Underwater sensor networks: applications, advances and challenges. Philos. Trans. R. Soc.: Math. Phys. Eng. Sci. **370**(1958), 158–175 (2012)
25. Julier, S., Uhlmann, J.: New extension of the Kalman filter to nonlinear systems. In: AeroSense'97. International Society for Optics and Photonics (1997)

26. Leonard, N., Paley, D., Lekien, F., Sepulchre, R., Fratantoni, D., Davis, R.: Collective motion, sensor networks, and ocean sampling. Proc. IEEE **95**(1), 48–74 (2007)
27. Liao, E., Hollinger, G., Djugash, J., Singh, S.: Preliminary results in tracking mobile targets using range sensors from multiple robots. Distrib. Auton. Robot. Syst. **7**, 125–134 (2006)
28. Mostofi, Y., Chung, T., Murray, R., Burdick, J.: Communication and sensing trade-offs in decentralized mobile sensor networks: a cross-layer design approach. In: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (2005)
29. Rife, J., Rock, S.: Design and validation of a robotic control law for observation of deep-ocean jellyfish. IEEE Trans. Robot. **22**(2), 282–291 (2006)
30. Schneider, T., Schmidt, H.: Unified command and control for heterogeneous marine sensing networks. J. Field Robot. **27**(6), 876–889 (2010)
31. Sinopoli, B., Schenato, L., Franceschetti, M., Poolla, K., Jordan, M., Sastry, S.: Kalman filtering with intermittent observations. IEEE Trans. Autom. Control **49**(9), 1453–1464 (2004)
32. Skomal, G., Benz, G.: Ultrasonic tracking of Greenland sharks, Somniosus microcephalus, under Arctic ice. Mar. Biol. **145**(3), 489–498 (2004)
33. Soares, J., Aguiar, A., Pascoal, A., Martinoli, A.: Joint ASV/AUV range-based formation control: theory and experimental results. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2013)
34. Takasu, T., Yasuda, A.: Development of the low-cost RTK-GPS receiver with an open source program package RTKLIB. In: Proceedings of the International Symposium on GPS/GNSS, International Convention Center Jeju, Korea (2009)
35. Vázquez-Rowe, I., Iribarren, D., Moreira, M.T., Feijoo, G.: Combined application of life cycle assessment and data envelopment analysis as a methodological approach for the assessment of fisheries. Int. J. Life Cycle Assess. **15**(3), 272–283 (2010)
36. Vidal, R., Shakernia, O., Kim, H., Shim, D., Sastry, S.: Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. IEEE Trans. Robot. Autom. **18**(5), 662–669 (2002)
37. Voegeli, F., Smale, M., Webber, D., Andrade, Y., O'Dor, R.: Ultrasonic telemetry, tracking and automated monitoring technology for sharks. Environ. Biol. Fishes **60**(1), 267–282 (2001)
38. Zhou, K., Roumeliotis, S.: Optimal motion strategies for range-only constrained multisensor target tracking. IEEE Trans. Robot. **24**(5), 1168–1185 (2008)

Aggressive Maneuver Regulation of a Quadrotor UAV

Sara Spedicato, Giuseppe Notarstefano, Heinrich H. Bülthoff and Antonio Franchi

Abstract In this paper we design a nonlinear controller for aggressive maneuvering of a quadrotor. We take a maneuver regulation perspective. Differently from the classical trajectory tracking approach, maneuver regulation does not require following a timed reference state, but a geometric "path" with a velocity (and possibly orientation) profile assigned on it. The proposed controller relies on three main ideas. Given a desired maneuver, i.e., a set of state trajectories equivalent under time translations, the system dynamics is decomposed into dynamics longitudinal and transverse to the maneuver. A *space-dependent* version of the transverse dynamics is derived, by using the longitudinal state, i.e., the arc-length of the path, as an independent variable. Then the controller is obtained as a function of the arc-length consisting of two terms: a feedforward term, being the nominal input to apply when on the path at the current arc-length, and a feedback term exponentially stabilizing the state-dependent transverse dynamics. Numerical computations are presented to prove the effectiveness of the proposed strategy. The controller performances are tested in presence of uncertainty of the model parameters and input noise and saturations. The controller is also tested in a realistic simulation environment validated against an experimental test-bed.

S. Spedicato (✉) · G. Notarstefano
Department of Engineering, Università del Salento, Via per Monteroni,
73100 Lecce, Italy
e-mail: sara.spedicato@unisalento.it

G. Notarstefano
e-mail: giuseppe.notarstefano@unisalento.it

H.H. Bülthoff · A. Franchi
Max Planck Institute for Biological Cybernetics, Spemannstraße 38,
72076 Tubingen, Germany
e-mail: hhb@tuebingen.mpg.de

A. Franchi
e-mail: antonio.franchi@tuebingen.mpg.de

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts
in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_6

1 Introduction

In the recent years, the steadily growing number of applications involving Unmanned Aerial Vehicles (UAVs), as quadrotors, has raised attention on the execution of precise aggressive motions. This is, in fact, a fundamental requirement in several (complex) tasks. The classical approaches used in the literature to perform such motions fall into the categories of *trajectory tracking* and *path following* methods. Trajectory tracking techniques aim at limiting the error between the actual system state and the desired state at a specified time. The system state includes position, orientation, and linear and angular velocities. Whenever an exogenous disturb (e.g., wind) forces the robot to momentarily lag behind the "moving reference" of the desired trajectory, then undesired phenomena are likely to arise, such as: (i) huge peaks of acceleration and (ii) a poor geometric tracking of the planned path that may lead to collisions with the surrounding world.

In order to highlight this last sensitive issue, let us consider the example depicted in Fig. 1. A quadrotor has to fly around an obstacle tracking a given planned trajectory, which is specified as a desired state at each time t. At $t = t_1$ strong opposing wind significantly decelerates the actual motion of the quadrotor for a few seconds. When the wind ceases, at $t = t_3$, the quadrotor recovers the full control of its motion and tries to quickly catch up with the moving desired state that is now on the other side of the obstacle, thus dramatically crashing into it.

These drawbacks do not arise in classical path following techniques, whose objective is to have the system position follow a geometric path without a predefined time scheduling. Since a pre-defined timing law on the path is not given, these methods consider a tracking error between the current robot position and the set of positions on the entire geometric path; while orientation, linear and angular velocities are not usually taken into account. Classical path following techniques are able to avoid undesired phenomena as the one described above. However, they can only drive the center of mass of the UAV along the path without ensuring a desired orientation and velocity profile along it.

To overcome the drawbacks that affect both these two techniques, we deal with an extended version of the path following, called *maneuver regulation*, that aims at satisfying additional requirements (such as assigning orientation, linear velocity, and angular velocity on the path).

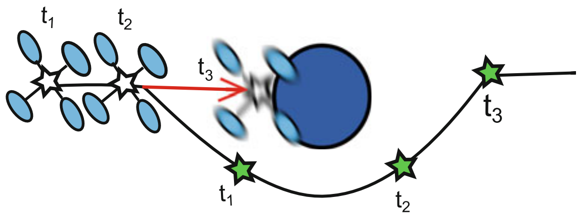


Fig. 1 Example of an undesired phenomenon arising using trajectory tracking

We organize the literature on quadrotor controllers in two parts. First, a vast number of trajectory tracking techniques for quadrotors have been proposed. In [1] a dynamic feedback controller, which renders the system linear and controllable, has been developed. In [2] a full backstepping technique is presented, based on a decomposition of the dynamic model into an underactuated subsystem, and a fully-actuated subsystem. A geometric tracking control is presented in [3]. The nonlinear tracking controller is developed on the special Euclidean group $SE(3)$ and it is shown to have desirable closed loop properties that are almost global. The trajectory tracking controller in [3] has been successfully implemented in [4] to perform fast aerobatic maneuvers without exogenous disturbances and precise measures from an external motion capture system (aerobatic maneuvering is also addressed, for autonomous helicopters, in [5, 6]). A sliding mode controller is proposed in [7] in order to stabilize the quadrotor model as a class of cascaded under-actuated systems. In [8], an underactuated nonlinear $\mathscr{H}_\infty$ controller based on the six degrees of freedom dynamic quadrotor model is designed to control the attitude and altitude in an inner-loop. The outer-loop control is performed using a model-based predictive controller to track the reference trajectory. Finally a trajectory tracker based on a linear quadratic regulator (LQR) is proposed in [9].

Second, only recently, extended path following techniques for quadrotors have been presented. In [10] the problem is solved by means of a backstepping technique. In [11] the problem is addressed as the stabilization of the zero dynamics for a nonlinear control system and solved using input-output feedback linearization on an augmented quadrotor system. The technique is refined in [12] with the objective of executing a more general class of paths "in a more general manner". The definition of the position error as the distance between the actual quadrotor position and the desired path is used in [13], where a "commanded acceleration" is computed using a PD feedback of the position and velocity errors.

The first and main contribution of the paper is the design of a maneuver regulation controller for aggressive maneuvering of a quadrotor on a three-dimensional path with assigned orientation and velocity profiles along it. The control strategy, inspired to the one proposed in [14] for a motorcycle on a bi-dimensional path, is based on the idea of *transverse linearization* of the dynamic system. Given a desired trajectory, the system dynamics is rewritten in terms of a longitudinal and a transverse dynamics. The new system states are the arc-length s and a set of *transverse coordinates* $w(s)$ (defined by means of an appropriate distance between the current state and the desired trajectory states). Differently from the (extended) path following approaches in [10–12], we consider a *space-dependent* version of the transverse dynamics. That is, we derive a differential equation in which the arc-length s is the independent variable, so that the transverse linearization is obtained by linearizing such a space-dependent dynamics. By solving an infinite-horizon linear quadratic regulator optimal control problem, the (space-dependent) transverse dynamics linearization can be exponentially stabilized (under standard controllability assumptions). Thus, the quadrotor maneuver regulation controller is a function of the arc-length s: a feedforward term being the desired input for the given s, plus a feedback $K(s)w(s)$ to stabilize the (space-dependent) transverse dynamics. Notably, although the feedback is linear as a function of the arc-length, it turns to be a nonlinear feedback of the original system state in the time-domain as $s(t) = s(x(t))$.

As second contribution, and preliminary step for experimental tests, we test our maneuver regulation controller on a physical quadrotor simulator, in order to show the performances of the proposed maneuver regulation controller in a realistic simulation scenario. Furthermore, we perform numerical computations under uncertainties on the model parameters and input noise and saturations. The computations show how the controller well behaves in performing a fairly aggressive maneuver as a barrel roll. In fact, we verified that even if the design is executed with some nominal inertial parameters the controller is able to adapt the control effort depending to the actual inertia and to maintain the system stability even if the severe input saturation and noise do not allow a perfect tracking of the desired maneuver.

Compared to the other extended path following approaches designed for a quadrotor, [10–12], our maneuver regulation controller has three key differences. First, the other schemes only ensure the distance from the path and the error on the yaw angle to converge to zero. The resulting roll and pitch angle, although stable, cannot be assigned a priori, so that problems could arise for example in narrow passageways. Second, the scheme does not rely on structural properties of the simplified quadrotor model as, e.g., flatness or non-minimum-phase-ness. Thus, it can be applied also to more complex (possibly non-minimum phase) models. Third, the proposed maneuver regulation scheme can be decoupled into a slow time-scale block and a realtime one. The desired trajectory and the feedback gains can be computed in a slow time-scale. In particular, the desired trajectory can be computed by using trajectory optimization techniques as the ones proposed in [15, 16]. The online computation only requires the calculation of the scheduled arc-length and the application of the feedback gain. Due to this structure, the proposed controller can be seen as a preliminary step toward the development of a receding-horizon scheme involving both trajectory generation and regulation in a coupled scheme.

The paper is organized as follows. In Sect. 2 the quadrotor model and the maneuver regulation problem are introduced. Section 3 addresses the design of the maneuver regulation controller for the quadrotor. Finally, in Sect. 4 numerical computations and physical simulations are provided in order to prove the effectiveness of the LQR based controller under parameters uncertainty and input saturation.

2 Quadrotor Model and the Maneuver Regulation Problem

In this section we present the standard quadrotor model that is instrumental to formally define the maneuver regulation problem and the proposed controller.

In the following we denote vectors using bold small symbols and matrices using capital letters. Given an inertial reference frame (with x–y–z axes oriented in a north-east-down fashion) and a body reference frame attached to the quadrotor center of mass (with x–y–z axes oriented in a forward-right-down fashion), let $\boldsymbol{p} \in \mathbb{R}^3$ be the position vector from the origin of the inertial frame to the origin of the body frame, expressed in the inertial frame. The orientation of the body frame with respect to the inertial frame is denoted by the rotation matrix $R \in SO(3)$, which maps vectors in the body frame into vectors in the inertial frame. Let $\boldsymbol{v} \in \mathbb{R}^3$ and $\boldsymbol{\omega} \in \mathbb{R}^3$ denote respectively the linear and angular velocities expressed in the body frame.

The quadrotor is driven by four forces and torques produced by the four propellers. Each thrust force is directed along the body z-axis but pointing in the negative direction. The thrust forces produce the torques γ_1 and γ_2 around the body x-axis and y-axis respectively. A torque γ_3 around the z-axis is produced by the reaction moments acting on the propellers. The sum of the thrust forces is denoted by f and the torques vector is denoted by $\boldsymbol{\gamma} = (\gamma_1\ \gamma_2\ \gamma_3)^T$.

The standard quadrotor model, see, e.g., [17], is

$$\dot{\boldsymbol{p}} = R\boldsymbol{v} \tag{1}$$

$$m\dot{\boldsymbol{v}} = -m\Omega\boldsymbol{v} + mgR^T\boldsymbol{e}_3 - f\boldsymbol{e}_3 \tag{2}$$

$$\dot{R} = R\Omega \tag{3}$$

$$J\dot{\boldsymbol{\omega}} = -\Omega J\boldsymbol{\omega} + \boldsymbol{\gamma} \tag{4}$$

where m is the quadrotor mass, $J = \mathrm{diag}(j_x, j_y, j_z)$ is the inertia matrix, g is the gravity constant, Ω is the screw-symmetric matrix associated with $\boldsymbol{\omega}$ and $\boldsymbol{e}_3 = (0\ 0\ 1)^T$.

We choose to parameterize the rotation matrix R by roll-pitch-yaw angles.[1] From the inertial reference frame the first rotation is taken around the z-axis by the yaw angle ψ. The coordinate system is then rotated around the new y-axis by the pitch angle θ and finally rotated about the new x-axis by the roll angle φ. The rotation matrix is thus

$$R = \begin{pmatrix} c\psi c\theta & -s\psi c\varphi + c\psi s\theta s\varphi & s\psi s\varphi + c\psi s\theta c\varphi \\ s\psi c\theta & c\psi c\varphi + s\psi s\theta s\varphi & -s\varphi c\psi + s\psi s\theta c\varphi \\ -s\theta & s\varphi c\theta & c\theta c\varphi \end{pmatrix}, \tag{5}$$

where for a generic angle ϕ we define $c\phi := \cos(\phi)$ and $s\phi := \sin(\phi)$.

Let us define $\boldsymbol{p} = (p_1\ p_2\ p_3)^T$, $\boldsymbol{v} = (v_1\ v_2\ v_3)^T$ and $\boldsymbol{\omega} = (p\ q\ r)^T$. Using the roll-pitch-yaw parametrization equations (1–4) are

$$\dot{p}_1 = c\psi c\theta v_1 + (-s\psi c\varphi + c\psi s\theta s\varphi)v_2 + (s\psi s\varphi + c\psi s\theta c\varphi)v_3 \tag{6}$$

$$\dot{p}_2 = s\psi c\theta v_1 + (c\psi c\varphi + s\psi s\theta s\varphi)v_2 + (-s\varphi c\psi + s\psi s\theta c\varphi)v_3 \tag{7}$$

$$\dot{p}_3 = -s\theta v_1 + s\varphi c\theta v_2 + c\theta c\varphi v_3 \tag{8}$$

$$\dot{\varphi} = p + qs\varphi \tan\theta + rc\varphi \tan\theta \tag{9}$$

$$\dot{\theta} = qc\varphi - rs\varphi \tag{10}$$

$$\dot{\psi} = qs\varphi\frac{1}{c\theta} + rc\varphi\frac{1}{c\theta} \tag{11}$$

$$\dot{v}_1 = rv_2 - qv_3 - gs\theta \tag{12}$$

[1]This parametrization of R, largely used in the literature, has a singularity when the pitch angle reaches $\pi/2$. However, the proposed techniques can be developed for any other parametrization. Thus, given the desired maneuver, the best suited parametrization can be used to avoid singularities.

$$\dot{v}_2 = -rv_1 + pv_3 + gs\varphi c\theta \tag{13}$$

$$\dot{v}_3 = qv_1 - pv_2 + gc\theta c\varphi - \frac{f}{m} \tag{14}$$

$$\dot{p} = qr\left(\frac{j_y - j_z}{j_x}\right) + \frac{\gamma_1}{j_x} \tag{15}$$

$$\dot{q} = pr\left(\frac{j_z - j_x}{j_y}\right) + \frac{\gamma_2}{j_y} \tag{16}$$

$$\dot{r} = pq\left(\frac{j_x - j_y}{j_z}\right) + \frac{\gamma_3}{j_z} \tag{17}$$

Equations (6–17) represent a nonlinear, time-invariant control system of the form

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)) \tag{18}$$

$$\boldsymbol{y}(t) = h(\boldsymbol{x}(t)) \tag{19}$$

with state $\boldsymbol{x} = (p_1\ p_2\ p_3\ \varphi\ \theta\ \psi\ v_1\ v_2\ v_3\ p\ q\ r)^T \in \mathbb{R}^{12}$, input $\boldsymbol{u} = (f\ \gamma_1\ \gamma_2\ \gamma_3)^T \in \mathbb{R}^4$ and output $\boldsymbol{y} = (p_1\ p_2\ p_3)^T \in \mathbb{R}^3$. Equation (18) can be also written in scalar form as

$$\dot{x}_i(t) = f_i(\boldsymbol{x}(t), \boldsymbol{u}(t)), \quad \forall i = 1, \ldots, 12,$$

where x_i is the ith component of $\boldsymbol{x}$ and $f_i(\cdot)$ is the scalar ith component of the vector function $f(\cdot)$.

Given the quadrotor model, we can formalize the maneuver regulation task that we want to solve. In the approach we propose in this paper, we can decouple the task into two parts: (i) generation of a desired, or nominal, (state-control) trajectory, and (ii) regulation of the corresponding desired maneuver.

Although in this paper we focus on the regulation sub-task (ii), we briefly describe the whole task. Usually, to accomplish a complex mission, the quadrotor is required to follow a given three-dimensional path and, maybe, satisfy some (soft) constraints on the orientation (e.g., because it has to traverse a narrow passageway).

The trajectory generation sub-task (i) is the following. We suppose that a suitable output curve is assigned by the mission to satisfy some geometric constraints. A reasonable choice of output curve for the quadrotor is $\boldsymbol{y}_\xi(t) = (p_{1\xi}(t)\ p_{2\xi}(t)\ p_{3\xi}(t))^T$, $t \geq 0$. The use of the subscript ξ will be clear in the next lines. We say that an output curve $\boldsymbol{y}_\xi(\cdot)$ is admissible, if there exists a state-control trajectory $\boldsymbol{\xi} = (\boldsymbol{x}_\xi(\cdot), \boldsymbol{u}_\xi(\cdot))$, such that

$$\dot{\boldsymbol{x}}_\xi(t) = f(\boldsymbol{x}_\xi(t), \boldsymbol{u}_\xi(t)), \quad \boldsymbol{y}_\xi(t) = h(\boldsymbol{x}_\xi(t))$$

for all $t \geq 0$, $\|\dot{\boldsymbol{y}}_\xi(\cdot)\|$ is bounded away from zero and $\|\ddot{\boldsymbol{y}}_\xi(\cdot)\|$ is bounded.

State-control trajectories for the standard quadrotor model used in this paper can be generated by exploiting its differential flatness. For more general models or in case state and input constraints need to be explicitly taken into account in the desired trajectory generation, nonlinear optimal control based trajectory-generation techniques, as the ones developed in [15, 16], may be used.

Given a desired (state-control) trajectory $(\boldsymbol{x}_\xi(\cdot), \boldsymbol{u}_\xi(\cdot))$, we define a maneuver $[\boldsymbol{x}_\xi, \boldsymbol{u}_\xi]$ as the set of all the trajectories equivalent under time translation to the trajectory $(\boldsymbol{x}_\xi(\cdot), \boldsymbol{u}_\xi(\cdot))$. Given a trajectory $(\tilde{\boldsymbol{x}}_\xi(\cdot), \tilde{\boldsymbol{u}}_\xi(\cdot))$ of $f(\cdot)$, we say that it is equivalent to $(\boldsymbol{x}_\xi(\cdot), \boldsymbol{u}_\xi(\cdot))$ under time translation if and only if there exists $\Delta \in \mathbb{R}$, such that $\tilde{\boldsymbol{x}}_\xi(t) = \boldsymbol{x}_\xi(t + \Delta)$ and $\tilde{\boldsymbol{u}}_\xi(t) = \boldsymbol{u}_\xi(t + \Delta)$, $\forall t \geq 0$.

We are now ready to formally define the maneuver regulation problem.

Maneuver regulation problem. Let an output $\boldsymbol{y}_\xi(\cdot)$ and an associated maneuver $[\boldsymbol{x}_\xi, \boldsymbol{u}_\xi]$ of the quadrotor be given. Find a feedback control law $u = k(\boldsymbol{x}; [\boldsymbol{x}_\xi, \boldsymbol{u}_\xi])$ that exponentially stabilizes the maneuver $[\boldsymbol{x}_\xi, \boldsymbol{u}_\xi]$, i.e. such that there exist $\Delta, k, \lambda > 0$ such that

$$\lim_{t\to\infty} ||\boldsymbol{x}(t) - \boldsymbol{x}_\xi(t + \Delta)|| \leq k e^{-\lambda t}.$$

Remark 1 From the above definition it is clear why the maneuver regulation task protects from dangerous situations as the one described in Fig. 1. Indeed, for the task to be accomplished the quadrotor is not required to catch up a reference on the desired maneuver. In the specific scenario, when the quadrotor regains control after the disturbance has ceased, it can track a time-translated trajectory belonging to the same maneuver, whose initial condition is close to the current quadrotor state, thus avoiding to fall into the obstacle. □

3 Transverse Linearization Based Maneuver Regulation Controller

In order to exponentially stabilize a desired maneuver, rather than a trajectory, we seek a controller scheduled by points on the desired output, rather than by the time. As a first step, we rewrite the quadrotor dynamics in terms of a longitudinal and a transverse dynamics.

The longitudinal dynamics describes the evolution of the system position along the curve, while the transverse dynamics describes the evolution of a suitable error between the actual state and the desired one at the current longitudinal coordinate.

We start by parametrizing the (admissible) output curve in terms of the arc-length $\sigma_\xi : \mathbb{R}_0^+ \to \mathbb{R}_0^+$, defined as

$$\sigma_\xi(t) = \int_0^t \sqrt{\dot{p}_{1\xi}^2(\tau) + \dot{p}_{2\xi}^2(\tau) + \dot{p}_{3\xi}^2(\tau)}\, d\tau, \quad \forall t \geq 0. \tag{20}$$

Defining the inverse of $\sigma_\xi(\cdot)$ as the function $\bar{t}_\xi : \mathbb{R}_0^+ \to \mathbb{R}_0^+$, the output, parametrized using the arc-length, is $\bar{\boldsymbol{y}}_\xi(\sigma) = \boldsymbol{y}_\xi(\bar{t}_\xi(\sigma))$. We shall denote the σ-parametrized curves with a bar, and the derivatives with respect to the coordinate σ with a prime symbol.

We parameterize the position of the quadrotor center of mass $\boldsymbol{y} = (p_1\ p_2\ p_3)^T$ in a tubular neighborhood of $\bar{\boldsymbol{y}}_\xi(\cdot)$ using a set of coordinates $(s, w_1, w_2) \in \mathbb{R} \times \mathbb{R} \times \mathbb{R}$. In order to do this, we construct a locally invertible function $\phi : \mathbb{R}^3 \to \mathbb{R} \times \mathbb{R} \times \mathbb{R}$ such that

$$(s, w_1, w_2) = \phi(\boldsymbol{y}), \tag{21}$$

with ϕ_i, $i \in \{1, 2, 3\}$, the ith scalar component of the vector function $\phi(\cdot)$, and such that $\phi(\bar{\boldsymbol{y}}_\xi(s)) = (s, 0, 0)$. We choose s to be the arc-length identifying the point on the desired path at minimum distance from the quadrotor center of mass. The coordinates w_1 and w_2 express the distance between the quadrotor center of mass and the point on $\bar{\boldsymbol{y}}_\xi(\cdot)$ identified by s.

Consistently, the first component of $\phi(\cdot)$ is defined as

$$\phi_1(\boldsymbol{y}) := \arg\min_{\sigma \in \mathbb{R}} \|\boldsymbol{y} - \bar{\boldsymbol{y}}_\xi(\sigma)\|^2.$$

The minimizing s is unique provided that $\bar{\boldsymbol{y}}_\xi(\cdot)$ is locally a non-intersecting C^2 curve with non-vanishing $\bar{\boldsymbol{y}}'_\xi(\cdot)$ and that $\boldsymbol{y}(t)$ is close to $\bar{\boldsymbol{y}}_\xi(\cdot)$ for all t. In order to construct the other components of $\phi(\cdot)$, let us consider the Serret–Frenet frame, which origin has $\bar{\boldsymbol{y}}_\xi(s)$ as coordinates, defined by the basis $\{\overrightarrow{\mathbf{t}}(s), \overrightarrow{\mathbf{n}}(s), \overrightarrow{\mathbf{b}}(s)\}$. The vectors $\overrightarrow{\mathbf{t}}$, $\overrightarrow{\mathbf{n}}$, $\overrightarrow{\mathbf{b}}$ are respectively the tangent, normal and bi-normal vectors and they are defined, with components in the inertial frame, as $\bar{\boldsymbol{t}}(s) := \bar{\boldsymbol{y}}'_\xi(s)$, $\bar{\boldsymbol{n}}(s) := \bar{\boldsymbol{y}}''_\xi(s)/\bar{k}(s)$, $\bar{\boldsymbol{b}}(s) := \bar{\boldsymbol{t}}(s) \times \bar{\boldsymbol{n}}(s)$, where $\bar{k}(s) := \|\bar{\boldsymbol{y}}''_\xi(s)\|$ is the curvature of $\bar{\boldsymbol{y}}_\xi(\cdot)$ at s. The position of the quadrotor center of mass $\boldsymbol{y}$ can be written as

$$\boldsymbol{y} = \bar{\boldsymbol{y}}_\xi(s) + \bar{R}_{SF}(s)\boldsymbol{d}, \tag{22}$$

where the rotation matrix $\bar{R}_{SF} = (\bar{\boldsymbol{t}}\ \bar{\boldsymbol{n}}\ \bar{\boldsymbol{b}})$ maps vectors with components in the Serret–Frenet frame into vectors with components in the world inertial frame and $\mathbf{d} := (0\ w_1\ w_2)^T$ is the vector from the origin of the Serret–Frenet frame to the origin of the body frame, with components in the Serret–Frenet frame. Using the Eq. (22), we define the remaining components of the function $\phi(\cdot)$ as

$$\phi_2(\boldsymbol{y}) := \bar{\boldsymbol{n}}(s)^T(\boldsymbol{y} - \bar{\boldsymbol{y}}_\xi(s)), \tag{23}$$

$$\phi_3(\boldsymbol{y}) := \bar{\boldsymbol{b}}(s)^T(\boldsymbol{y} - \bar{\boldsymbol{y}}_\xi(s)). \tag{24}$$

With the function $\phi(\cdot)$ in hand, we can provide a change of coordinates from the state $\boldsymbol{x}$ to the coordinates $(s, \boldsymbol{w})$, where $s \in \mathbb{R}$ is the *longitudinal coordinate* and $\boldsymbol{w} \in \mathbb{R}^{11}$ is the vector of *transverse coordinates* with ith transverse coordinate, $i \in \{1, \ldots, 11\}$, denoted by w_i. The longitudinal and transverse coordinates as function of the system state are defined as

$$\begin{array}{llll} s \;\; := \phi_1(h(\boldsymbol{x})), & w_3 := \varphi - \bar{\varphi}_\xi(s), & w_6 \;\; := v_1 - \bar{v}_{1\xi}(s), & w_9 \;\; := p - \bar{p}_\xi(s), \\ w_1 := \phi_2(h(\boldsymbol{x})), & w_4 := \theta - \bar{\theta}_\xi(s), & w_7 \;\; := v_2 - \bar{v}_{2\xi}(s), & w_{10} := q - \bar{q}_\xi(s), \\ w_2 := \phi_3(h(\boldsymbol{x})), & w_5 := \psi - \bar{\psi}_\xi(s), & w_8 := v_3 - \bar{v}_{3\xi}(s), & w_{11} := r - \bar{r}_\xi(s). \end{array}$$

Furthermore, we define

$$\begin{aligned} \boldsymbol{u}_w &:= \boldsymbol{u} - \bar{\boldsymbol{u}}_\xi(s), & \gamma_{1w} &:= \gamma_1 - \bar{\gamma}_{1\xi}(s), \\ f_w &:= f - \bar{f}_\xi(s), & \gamma_{2w} &:= \gamma_2 - \bar{\gamma}_{2\xi}(s), \\ & & \gamma_{3w} &:= \gamma_3 - \bar{\gamma}_{3\xi}(s). \end{aligned} \tag{25}$$

The change of coordinates is used in order to write the standard quadrotor system (6–17) in $(s, \boldsymbol{w})$ coordinates. Deriving (22) with respect to time we have

$$\dot{\boldsymbol{y}} = \bar{\boldsymbol{y}}'_\xi \dot{s} + \bar{R}'_{SF} \dot{s}\, \mathbf{d} + \bar{R}_{SF} \dot{\boldsymbol{d}}. \tag{26}$$

Further on, for simplicity of notation, the dependency by s is omitted and all the bar terms are evaluated with respect to s. The term on the left side of Eq. (26) is

$$\dot{\boldsymbol{y}} = R\boldsymbol{v}, \tag{27}$$

according to the definition of $\boldsymbol{y}$ and the Eq. (1). The first term on the right side is

$$\bar{\boldsymbol{y}}'_\xi \dot{s} = \bar{R}_{SF} [\dot{s}\ 0\ 0]^T \tag{28}$$

according to the definition of the tangent vector $\bar{\boldsymbol{t}}$. Furthermore we have

$$\bar{R}'_{SF} \dot{s} = \bar{R}_{SF} \begin{bmatrix} 0 & -\bar{k}\dot{s} & 0 \\ \bar{k}\dot{s} & 0 & -\bar{\tau}\dot{s} \\ 0 & \bar{\tau}\dot{s} & 0 \end{bmatrix}, \tag{29}$$

where $\bar{\tau} := \bar{\boldsymbol{n}}\bar{\boldsymbol{b}}'$ is the torsion of $\bar{\boldsymbol{y}}_\xi(\cdot)$ at s. The expression (29) can be derived from the Serret–Frenet formulas [18] $\bar{\boldsymbol{t}}' = \bar{k}\bar{\boldsymbol{n}}$, $\bar{\boldsymbol{n}}' = -\bar{k}\bar{\boldsymbol{t}} + \bar{\tau}\bar{\boldsymbol{b}}$, $\bar{\boldsymbol{b}}' = -\bar{\tau}\bar{\boldsymbol{n}}$, using the definition of $\bar{R}_{SF}$. Multiplying both sizes of the Eq. (26) times $\bar{R}_{SF}^T$, using (27), (28), (29) and the coordinate transformation from $\boldsymbol{x}$ to $(s, \boldsymbol{w})$, we obtain

$$\begin{aligned} \dot{s} &= \bar{\boldsymbol{t}}^T R\boldsymbol{v}/(1 - \bar{k}w_1), \\ \dot{w}_1 &= \bar{\boldsymbol{n}}^T R\boldsymbol{v} + \bar{\tau}\dot{s}w_2, \\ \dot{w}_2 &= \bar{\boldsymbol{b}}^T R\boldsymbol{v} - \bar{\tau}\dot{s}w_1, \end{aligned} \tag{30}$$

where $\boldsymbol{v} = [(w_6 + \bar{v}_{1\xi})\ (w_7 + \bar{v}_{2\xi})\ (w_8 + \bar{v}_{3\xi})]^T$ and R is given by (5) with $\varphi = w_3 + \bar{\varphi}_\xi$, $\theta = w_4 + \bar{\theta}_\xi$ and $\psi = w_5 + \bar{\psi}_\xi$. Equation (30) are equivalent to Eqs. (6–8) but they are only functions of the longitudinal coordinate s, the transverse coordinates w_1, w_2 and the state trajectory $\bar{\boldsymbol{x}}_\xi(\cdot)$. Furthermore, Eqs. (9–17) can be expressed as function of s, $\boldsymbol{w}$ and $\bar{\boldsymbol{x}}_\xi(\cdot)$, using the coordinate transformation from $\boldsymbol{x}$ to $(s, \boldsymbol{w})$ and Eq. (25). Taking the time derivative of $x_i = w_{i-1} + \bar{x}_{i\xi}$, the equations $\dot{x}_i = f_i(\boldsymbol{x}, \boldsymbol{u})$ can be written as

$$\dot{w}_{i-1} = f_i(\bar{\boldsymbol{x}}_{\xi r} + \boldsymbol{w}_r, \boldsymbol{u}_w + \bar{\boldsymbol{u}}_\xi) - \bar{x}'_{i\xi}\dot{s}, \quad \forall i = 4, \dots 12, \tag{31}$$

where $\bar{\boldsymbol{x}}_{\xi r}$ is the vector containing from the 4th to the 12th component of $\bar{\boldsymbol{x}}_\xi$ and $\boldsymbol{w}_r$ is the vector containing from the 3rd to the 11th component of $\boldsymbol{w}$.

The system (30 and 31) can be expressed in a form such that the independent variable is the longitudinal coordinate s, rather than the time t. We parameterize the transverse coordinates using s, i.e., $w_i(t) = \bar{w}_i, \ \forall i = 1, \dots, 11$, and by using the chain rule, we compute the time derivatives of the transverse coordinates as $\dot{w}_i = \bar{w}'_i\dot{s}$. Thus, Eqs. (30 and 31) can be written as the *transverse dynamic system*

$$\begin{aligned} \bar{w}'_1 &= (\bar{\boldsymbol{n}}^T R\boldsymbol{v})/\dot{s} + \bar{\tau}\bar{w}_2, \\ \bar{w}'_2 &= (\bar{\boldsymbol{b}}^T R\boldsymbol{v})/\dot{s} - \bar{\tau}\bar{w}_1, \\ \bar{w}'_{i-1} &= f_i(\bar{\boldsymbol{x}}_{\xi r} + \bar{\boldsymbol{w}}_r), \bar{\boldsymbol{u}}_w + \bar{\boldsymbol{u}}_\xi)/\dot{s} - \bar{x}'_{i\xi}, \quad \forall i = 4, \dots 12, \end{aligned} \tag{32}$$

where $\dot{s} = \bar{\boldsymbol{t}}^T R\boldsymbol{v}/(1 - \bar{k}\bar{w}_1)$, $\boldsymbol{v} = [(\bar{w}_6 + \bar{v}_{1\xi}) \ (\bar{w}_7 + \bar{v}_{2\xi}) \ (\bar{w}_8 + \bar{v}_{3\xi})]^T$ and R is given by (5) with $\varphi = \bar{w}_3 + \bar{\varphi}_\xi$, $\theta = \bar{w}_4 + \bar{\theta}_\xi$ and $\psi = \bar{w}_5 + \bar{\psi}_\xi$.

The system (32) is a nonlinear control system with state $\bar{\boldsymbol{w}} \in \mathbb{R}^{11}$ and input $\bar{\boldsymbol{u}}_w \in \mathbb{R}^4$, for which s-varing control laws can be developed in order to regulate the transverse state $\bar{\boldsymbol{w}}$ to zero. We compute such control law solving a linear quadratic regulator (LQR) problem.

Let us consider the transverse linearization

$$\bar{\boldsymbol{w}}' = \bar{A}_T(s)\bar{\boldsymbol{w}} + \bar{B}_T(s)\bar{\boldsymbol{u}}_w,$$

i.e., the linearization of the transverse dynamic system (32). We design a feedback matrix $\bar{K}(\cdot)$ that asymptotically stabilizes the transverse linearization by solving a linear quadratic regulator problem. If the transverse linearization is exponentially stabilized by an s-varing linear state feedback, $\bar{\boldsymbol{u}}_w = -\bar{K}(s)\bar{\boldsymbol{w}}$, then the nonlinear feedback

$$\boldsymbol{u} = \bar{\boldsymbol{u}}_\xi(s) - \bar{K}(s)\boldsymbol{w} \tag{33}$$

exponentially stabilizes the maneuver $[\boldsymbol{x}_\xi]$ for (6–17) [14].

The controller in (33) can be rewritten by exploiting the dependence of s from the system output ($s = \phi_1(\boldsymbol{y})$) as

$$\boldsymbol{u} = \bar{\boldsymbol{u}}_\xi(\phi_1(\boldsymbol{y})) - \bar{K}(\phi_1(\boldsymbol{y}))\boldsymbol{w}.$$

The above expression highlights the *nonlinear feedback* structure of the proposed maneuver regulation controller. In particular, the feedforward term and the feedback matrix are nonlinear functions of the system state (the output portion).

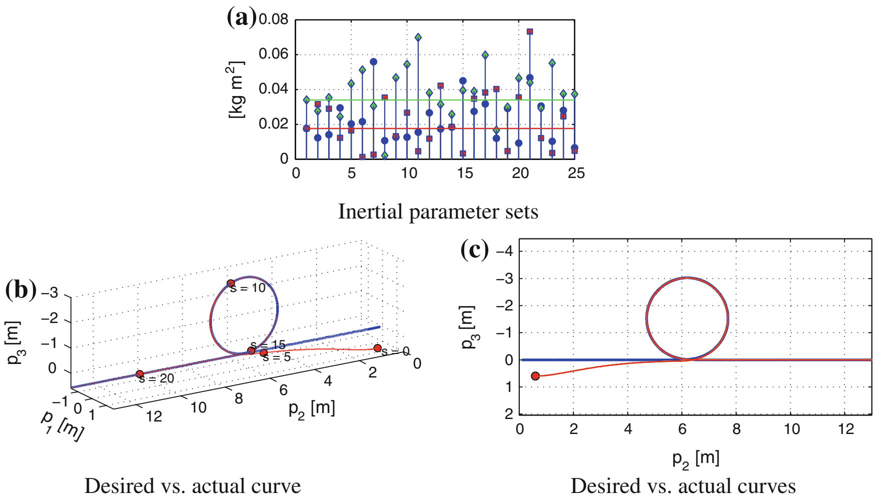


Fig. 2 **a** Values of the inertia matrices, used to perform the Monte Carlo simulation. The *blue*, *red* and *green* markers represent the terms j_x, j_y, y_z, respectively. **b**, **c** Closed loop trajectories for the different values of inertia matrices

4 Numerical Validation

This section provides two groups of simulations that are meant to be a preliminary step for experimental tests. The *first group* objective is to test the performance and robustness in the stabilization of an aggressive maneuver in presence of error in the initial conditions and uncertainty in the inertia matrix parameters; while the goal of the *second group* is to test the robustness on the same maneuver with respect to noise and saturation of the four propellers forces.

For both groups, we choose a *barrel roll* maneuver in the $p_2 - p_3$ plane with a constant velocity profile for which the quadrotor is subject to “significant” accelerations. The quadrotor center of mass is required to move along a circle of diameter $d = 3\,\mathrm{m}$ at a speed $v = 8\,\mathrm{m/s}$. The desired maneuver is designed so that the roll angle φ goes from 0° to 360° (i.e., the quadrotor performs a complete flip). As a reference for the nominal inertial parameters of the model we used the data-sheet values of the customized MK-Quadro[2] that is described in [19], i.e., $m = 0.749\,\mathrm{kg}$, $j_x = 0.0176\,\mathrm{kg\,m^2}, j_y = 0.0177\,\mathrm{kg\,m^2}, j_z = 0.034\,\mathrm{kg\,m^2}$.

In the *first group* of simulations we conduct a Monte Carlo analysis aimed at testing the robustness against imperfect initial conditions and inertial parameters. The desired barrel roll path is represented with a blue curve in Fig. 2b, c. In all the $25 + 1$ simulations of this first group the quadrotor starts 1.04 m distant from the nominal initial position, outside the $p_2 - p_3$ plane. In particular, each component

[2]www.mikrokopter.de.

of the position vector is perturbed respectively of 0.60 m from the nominal one. We opted for using the nominal mass of the MK-Quadro in every simulation. The reason is that in real applications the quadrotor mass is easily computable with high accuracy and reliability either using a scale or through simple hovering calibration. On the other hand, calibration of the inertia matrix requires a sophisticated measurement equipment and can easy become outdated if the geometrical configuration of the internal masses changes over time, e.g., whenever the battery is mounted in a slightly different place. In real applications, this is equivalent to have a random noise on the nominal parameters. In one simulation we used the nominal values for j_x, j_y, j_z while in each of the other 25 we used a different quadrotor model with set of values for j_x, j_y, j_z obtained by randomly perturbing the nominal values according to a normal distribution where the standard deviation is $\frac{0.0102}{3}$ kg m^2 (i.e., 95 % of the samples are within $(j_x \pm j_z, j_y \pm j_z, j_z \pm j_z)$ being j_z the maximum among j_x, j_y, j_z). Notice that we still consider negligible the off-diagonal terms of the matrix w.r.t. the diagonal ones, which is a reasonable assumption in practice. The 25 normally distributed samples of the 3 diagonal inertia coefficients are represented in Fig. 2a, where the blue, red and green markers represent the perturbed terms j_x, j_y, j_z, respectively. The nominal values are instead represented with 3 horizontal lines (notice that j_x and j_y are almost overlapping). In each of the $25 + 1$ simulations the maneuver regulation controller defined in (33) employes always the same $\bar{\boldsymbol{u}}_\xi(s)$, and $\bar{K}(s)$ that are computed using the nominal values of j_x, j_y, j_z. Notice that also the desired maneuver is designed considering the nominal value. In this way we can test the robustness of the controller against model parameter uncertainties. Finally, the diagonal weight matrices employed in the LQR problem in order to compute the feedback matrix $\bar{K}(\cdot)$ are $Q = \mathrm{diag}(100, 100, 9, 9, 9, 10, 10, 10, 3, 3, 3)$ and $R = \mathrm{diag}(0.1, 0.1, 1, 0.1)$ referred to the state $\bar{\boldsymbol{w}}$ and the input $\bar{\boldsymbol{u}}_w$, respectively.

Figure 2c presents the projection, on the $p_2 - p_3$ plane, of the desired position trajectory (thick blue curve), the closed loop position trajectory starting from the perturbed initial condition with nominal parameters (thick red curve), and the closed loop trajectories starting from the perturbed initial condition for all the 25 sets of perturbed inertial parameters (thin colored curve), which are actually indistinguishable from the red curve. After the initial transient, all the curves quickly converge to the desired curve right before the actual barrel roll maneuver starts. It is worth noting that even if the desired task is defined in the $p_2 - p_3$ plane, the perturbed initial condition is taken outside this plane and thus the resulting maneuver involves the whole dynamics, as can be seen from Fig. 2b.

In Fig. 3a, c, e we show the desired roll, pith and yaw angles compared to ones achieved during the 26 closed loop trajectories (here we use the same color convention described before). The three components of the (body frame) linear velocities are instead plotted in Fig. 3b, d, f still with the same color convention as before. All the quantities are plotted with respect to the arc-length σ. As already pointed out, the maneuver regulation, as opposed to classical path following techniques, is able to satisfy additional constraints like assigning orientation and linear velocity. In fact, the plots clearly show how, after the transient phase these additional constraints are fulfilled. Robustness with respect to the perturbation of the model parameters

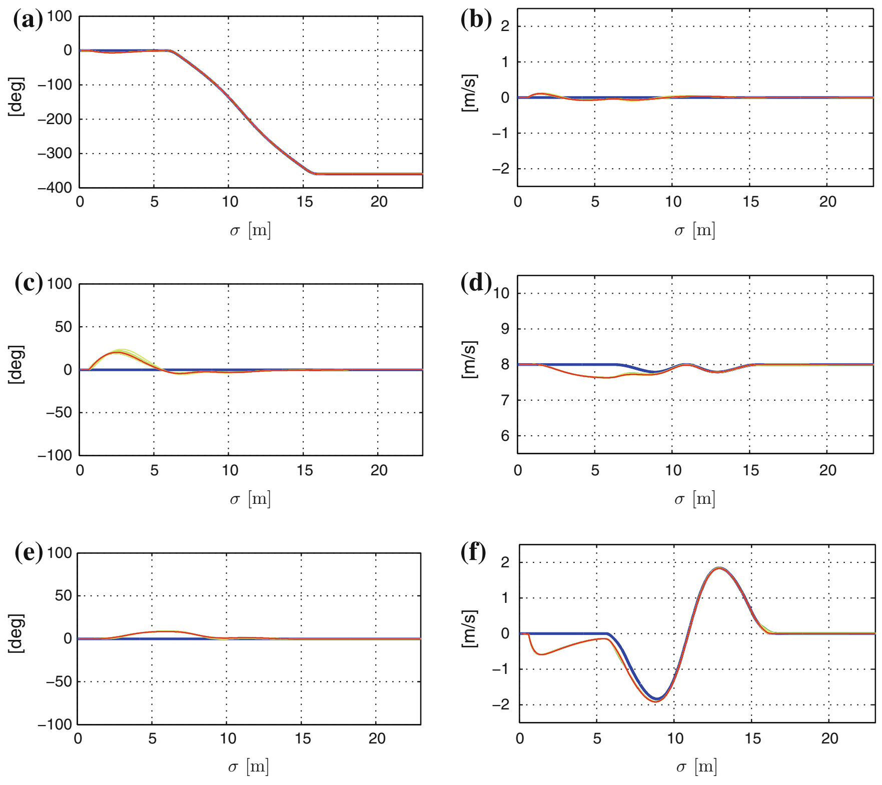


Fig. 3 Attitude angles and velocity components for the Monte Carlo simulation. *Thick blue lines* are used for the desired maneuver. *Thick red lines* are used for closed-loop maneuver with nominal parameters. All the other lines refer to the closed-loop maneuvers with perturbed parameters. **a** Roll angles φ. **b** Linear velocity components v_1. **c** Pitch angles θ. **d** Linear velocity components v_2. **e** Yaw angles ψ. **f** Linear velocity components v_3

is also manifest from the fact that all the 26 closed-loop trajectories are almost indistinguishable, despite the fact that the inertial parameters differ from the ones used in the controller design.

Control inputs of the first group of simulations are shown in Fig. 4. Differently from the plots presented so far, these plots show a different behavior of the torques γ_1,γ_2, and γ_3 across the 26 closed-loop simulations of this group. This happens because the proposed controller automatically adapts to the perturbation of the inertial parameters in order to track the desired trajectory. For example, the torque γ_2 in Fig. 4c, which is responsible for the flip of the roll, is either smaller or larger in order to automatically compensate for smaller or larger values of the perturbed j_y, respectively. Notice how all the thrusts show an almost indistinguishable behavior instead. This is manly due to the fact that the mass is not perturbed, for the reasons explained before, across the 26 simulations.

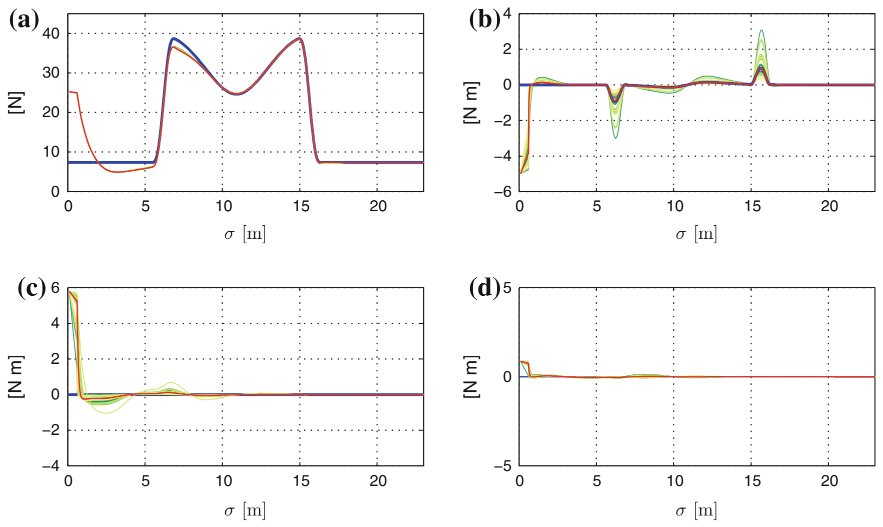


Fig. 4 Control inputs for the Monte Carlo simulation. *Thick blue lines* are used for the desired maneuver. Same color convention as in Fig. 3. **a** Thrust f. **b** Torque γ_1. **c** Torque γ_2. **d** Torque γ_3

In the *second group* of simulations we show how the proposed maneuver regulation controller is able to ensure good performances also in case of input noise and saturations, which are common in real physical systems. In order to further validate the controller with a realistic simulator we apply the controller on SwarmSimX [20], a quadrotor simulator whose fidelity has been already validated several times with respect to the test-bed described in [19] (see, e.g., [20] for a numerical comparison between the real quadrotor and the simulated one). We encourage the reader to watch the video attachments corresponding to these simulations at http://homepages.laas.fr/afranchi/robotics/?q=node/168.

The desired maneuver is the same barrel roll trajectory that has been used before but with a negligible initial error, since we want to evaluate here the sole effect of input noise and saturation. As it happens in the real world, we apply the actual noise and saturations on each single force produced by the 4 propellers. Denoting those forces with $f_1^p, f_2^p, f_3^p, f_4^p$ we have the well-known relation [3]:

$$\begin{bmatrix} f \\ \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ l & 0 & -l & 0 \\ -k & k & -k & k \end{bmatrix} \begin{bmatrix} f_1^p \\ f_2^p \\ f_3^p \\ f_4^p \end{bmatrix} \tag{34}$$

where l is the distance of the propeller from the center of mass and k is a suitable constant that mainly depends on the propeller shape.

The model parameters used in the simulations are the one of the customized MK-Quadro, i.e., mass and inertia matrix described before and $1 = 0.30\,\text{m}$. We apply a Gaussian noise with standard deviation equal to 0.1 N to each propeller. Three different cases are then considered: (i) no saturation; (ii) 8.5 N saturation, and (iii) 7 N saturation. These values are consistent with the capabilities of the MK-Quadro. Considering that (see, e.g., Fig. 4a) the barrel roll maneuver needs a 40 N peak of total thrust (i.e., see (34), 10 N per propeller on average), the chosen saturations correspond to have about 15 and 30 % less than the needed average total thrust.

Figure 5 shows the nominal force (thin black line) and actual forces commanded to each of the four propellers in the three saturation cases (blue, red, and green colored lines, respectively). Presence of noise and saturation is clear form the plots.

Figure 6 shows the projection on the $p_2 - p_3$ plane of the desired position maneuver (black line) and the three closed-loop actual position trajectories obtained for the three saturation cases (same color coding as the propeller forces). As expected,

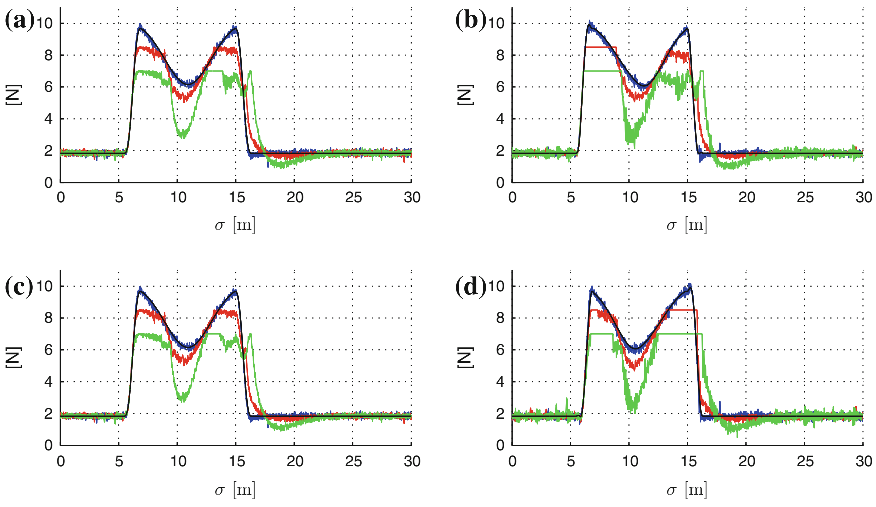
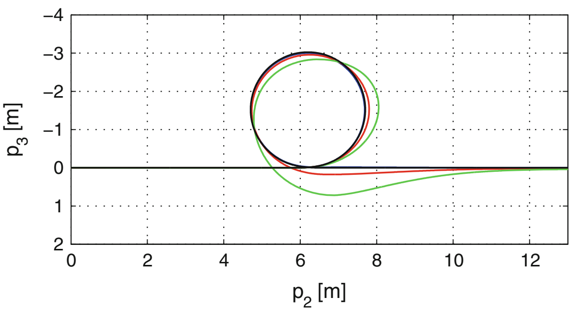


Fig. 5 Effect of the noise and saturation on the force produced by each propeller. **a** Force f_1^p. **b** Force f_2^p. **c** Force f_3^p. **d** Force f_4^p

Fig. 6 Effect of the force/torque saturation of each propeller on the actual trajectory



deformation of the trajectory is more pronounced for the 7 N saturation value than the 8.5 N case. However, the controller is capable of maintaining a stable behavior in all the four cases. A similar trend is observable on the evolution of the three velocities components shown in Fig. 7b, d, f as a function of the arc-length σ.

On the contrary, the achieved closed loop attitude angles are almost indistinguishable from the desired one, see Fig. 7a, c, and e. The different behavior can be explained in following way. The maximum achievable γ_1 when the force of the propeller is saturated at 7 N is, from (34), equal to $2l \cdot 7\,\text{N} = 4.2\,\text{Nm}$ which is higher than the maximum nominal torque γ_1 needed by the maneuver, which is visible in Fig. 4b (blue plot). Same discussion holds for the other torques. This shows how the controller is able to let the saturation only affect the quantities whose degradation is unavoidable (position and velocity in this case).

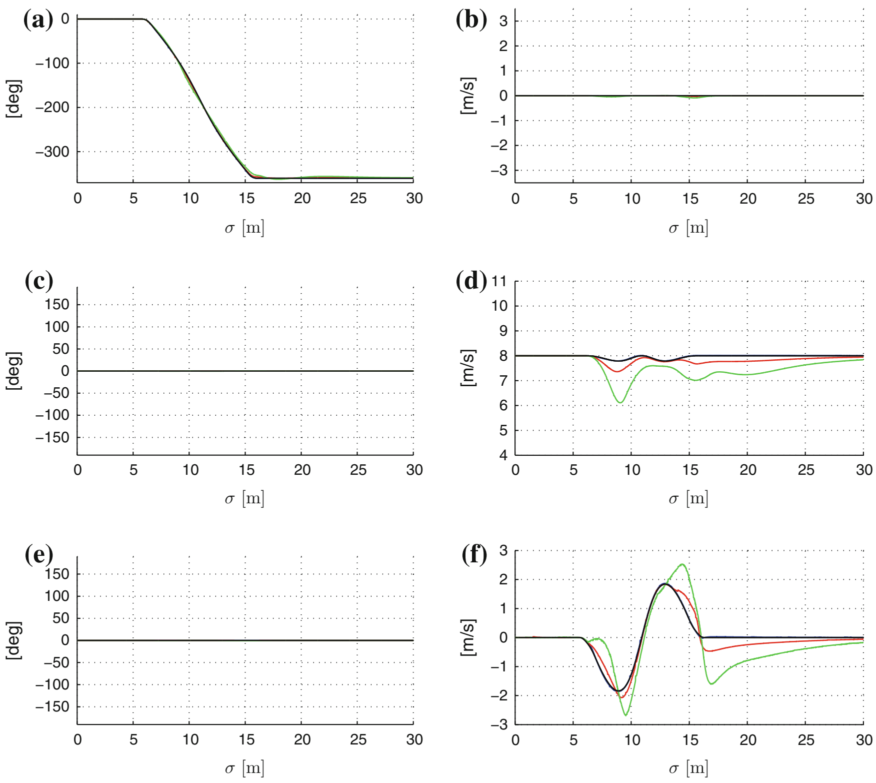


Fig. 7 Effect of the force/torque saturation on the attitude angles and linear velocity. **a** Roll angle φ. **b** Linear velocity v_1. **c** Pitch angle θ. **d** Linear velocity v_2. **e** Yaw angle ψ. **f** Linear velocity v_3

5 Conclusion

In this paper we have developed an LQR based maneuver regulation controller in order to make a quadrotor UAV perform three dimensional aggressive maneuvers. This controller overcome the drawbacks that affect both path following and trajectory tracking. We have shown how our maneuver regulation controller is robust with respect to the uncertainty of model parameters and input saturations. As a preliminary step to test the controller on the real quadrotor we have performed numerical computations on a quadrotor simulator with good physical fidelity. Given the promising performances of the controller we plan to test the controller on a real quadrotor platform.

Acknowledgments The work of G. Notarstefano and S. Spedicato is partially supported by the project SOCIAL-ROBOTS under the program “5 per mille per la ricerca”.

References

1. Mistler, V., Benallegue, A., M’Sirdi, N.K.: Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback. In: Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication, pp. 586–593. Paris, September (2001)
2. Madani, T., Benallegue, A.: Control of a quadrotor mini-helicopter via full state backstepping technique. In: Proceedings of the IEEE Conference on Decision and Control, pp. 1515–1520. San Diego, CA, December (2006)
3. Lee, T., Leoky, M., McClamroch, N.H.: Geometric tracking control of a quadrotor UAV on SE(3). In: Proceedings of the IEEE Conference on Decision and Control, pp. 5420–5425. Atlanta, GA, December (2010)
4. Mellinger, D., Kumar, V.: Minimum snap trajectory generation and control for quadrotors. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2520–2525. May (2011)
5. Gavrilets, V., Frazzoli, E., Mettler, B., Piedmonte, M., Feron, E.: Aggressive maneuvering of small autonomous helicopters: a human-centered approach. Int. J. Robot. Res. **20**(10), 795–807 (2001)
6. Abbeel, P., Coates, A., Quigley, M., Ng, A.Y.: An application of reinforcement learning to aerobatic helicopter flight. Adv. Neural Inf. Process. Sys. **19**, 1–8 (2007)
7. Rong, X., Ozguner, U.: Sliding mode control of a quadrotor helicopter. In: Proceedings of the IEEE Conference on Decision and Control, pp. 4957–4962. San Diego, CA, December (2006)
8. Raffo, Guilherme V., Ortega, Manuel G., Rubio, Francisco R.: Path tracking of a UAV via an underactuated control strategy. European J. Control **17**(2), 194–213 (2011)
9. Cowling, I.D., Yakimenko, O.A., Whidborne, J.F., Cooke, A.K.: A prototype of an autonomous controller for a quadrotor UAV. In: Proceedings of the European Control Conference (2007)
10. Cabecinhas, D., Cunha, R., Silvestre, C.: Rotorcraft path following control for extended flight envelope coverage. In: Proceedings of the IEEE Conference on Decision and Control Held Jointly with the Chinese Control Conference, pp. 3460–3465. Shanghai, December (2009)
11. Roza, A., Maggiore, M.: Path following controller for a quadrotor helicopter. In: Proceedings of the American Control Conference, pp. 4655–4660. Montreal, QC, June (2012)
12. Akhtar, A., Waslander, S.L., Nielsen, C.: Path following for a quadrotor using dynamic extension and transverse feedback linearization. In: Proceedings of the IEEE Conference on Decision and Control, pp. 3551–3556. Maui, HI, December (2012)

13. Mellinger, D., Michael, N., Kumar, V.: Trajectory generation and control for precise aggressive maneuvers with quadrotors. Int. J. Robot. Res. **31**(5), 664–674 (2012)
14. Saccon, A., Hauser, J., Beghi, A.: A virtual rider for motorcycles: Maneuver regulation of a multi-body vehicle model. IEEE Trans. Control Sys. Technol. **21**(2), 332–346 (2013)
15. Notarstefano, G., Hauser, J., Frezza, R.: Computing feasible trajectories for control-constrained systems: The PVTOL example. In Proceedings of the IFAC Symposium on Nonlinear Control Systems. Pretoria, SA, August (2007)
16. Notarstefano, G., Hauser, J.: Modeling and dynamic exploration of a tilt-rotor VTOL aircraft. In Proceedings of the IFAC Symposium on Nonlinear Control Systems. Bologna, September (2010)
17. Guenard, N., Hamel, T., Mahony, R.: A practical visual servo control for a unmanned aerial vehicle. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1342–1348. Roma, April (2007)
18. Setterlund, M.P.: Geometric-based Spatial Path Planning. UMI Microform (2008)
19. Franchi, A., Secchi, C., Ryll, M., Bülthoff, H.H., Robuffo Giordano, P.: Shared control: Balancing autonomy and human assistance with a group of quadrotor UAVs. IEEE Robot. Autom. Mag. Special Issue on Aerial Robotics and the Quadrotor Platform. **19**(3), 57–68 (2012)
20. Lächele, J., Franchi, A., Bülthoff, H.H., Robuffo Giordano, P.: SwarmSimX: Real-time simulation environment for multi-robot systems. In: Noda, I., Ando, N., Brugali, D., Kuffner, J.J. (eds.) 3rd International Conference on Simulation, Modeling, and Programming for Autonomous Robots. Lecture Notes in Computer Science, vol. 7628, pp. 375–387. Springer, Heidelberg (2012)

Towards Modeling Real-Time Trust in Asymmetric Human–Robot Collaborations

Anqi Xu and Gregory Dudek

Abstract We are interested in enhancing the efficiency of human–robot collaborations, especially in "supervisor-worker" settings where autonomous robots work under the supervision of a human operator. We believe that trust serves a critical role in modeling the interactions within these teams, and also in streamlining their efficiency. We propose an operational formulation of human–robot trust on a short interaction time scale, which is tailored to a practical tele-robotics setting. We also report on a controlled user study that collected interaction data from participants collaborating with an autonomous robot to perform visual navigation tasks. Our analyses quantify key correlations between real-time human–robot trust assessments and diverse factors, including properties of failure events reflecting causal trust attribution, as well as strong influences from each user's personality. We further construct and optimize a predictive model of users' trust responses to discrete events, which provides both insights on this fundamental aspect of real-time human–machine interaction, and also has pragmatic significance for designing trust-aware robot agents.

1 Introduction

In this paper, we consider methods that aim to increase the efficiency of human–robot teams, by optimizing the robot's level of performance with respect to an objective function reflecting human satisfaction: *trust*. The specific class of human–robot collaborations that we address are those with supervisor-worker relationships, where the human oversees and delegates work to a robot "worker", and also has the ability to take over control momentarily to correct the robot's mistakes when necessary. We are motivated to develop techniques to simultaneously improve the quality of the work performed by the robot, and also reduce the human supervisor's task load demands. We believe that *trust* is the key underpinning towards realizing these objectives,

A. Xu (✉) · G. Dudek
School of Computer Science, McGill University, Montreal, Canada
e-mail: anqixu@cim.mcgill.ca

G. Dudek
e-mail: dudek@cim.mcgill.ca

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_7

namely by enabling the robot to sense and adapt to the human's intentions through trust assessments, and by encouraging the building of the human's level of trust in the robot leading to the intrinsic disposition towards task delegation.

This paper describes our latest contributions in quantifying and predicting real-time changes in the human's level of trust in the robot during interaction. This is inspired by our previous research [20] on modeling and making use of human–robot trust within a tele-robotics setting, where a human operator supervises a remotely-located robot that is autonomously tracking terrain boundaries using visual processing. Whereas our previous work characterized the level of trust that the robot *deserves* based on logical reasoning about its task performance, we take a more data-driven approach in this work. Concretely, we attempt to predict the *actual* degree of trust the user has in the robot, from moment to moment during interaction, by studying experience-based metrics as well as human factors that give rise to subjective trust attribution.

We report on a user study that collected empirical human–robot interaction data in reaction to different controlled events. In particular, this study logged all experiences and actions occurred during short interaction sessions, and also elicited questionnaire responses reflecting each operator's mental state when interacting with the robot. The questionnaires were designed to quantify a number of factors that are known to influence human–robot trust, based on existing literature. We also present a descriptive analysis of the resulting dataset that establishes several quantitative characteristics of users' trust responses to different events, which are both logical and consistent with prior research. We further propose a parametric model for predicting reactive changes in the user's trust, and evaluate this novel model's generalizability and ability to predict the real-time progression of human–robot trust.

2 Background

Our work is inspired by an extensive literature across diverse disciplines observing the critical role played by trust in human teams. Trust is a very rich concept in the modeling of human behavior, and it is subject to a multitude of interpretations under different contexts, such as within a society, an organization, or a mutual relationship [16]. Within a mutual human–robot team, trust encompasses two major elements:

- the degree of trust: a quantifiable subjective assessment towards another individual;
- the act of trust: the decision and behavior of relying upon an individual's abilities or services.

In this work, we focus solely on quantifying the *degree of trust*. This measure can then be applied to mechanisms that encourage a human to adopt the *act of trust*, as shown in our prior work [20].

2.1 Related Work

Studies of trust in Human–Robot Interaction (HRI) historically evolved out of a multi-decade literature investigating the interaction between humans and automation. Several measurement scales of a human's degree of trust in computer automation have been previously proposed and evaluated [14, 17]. Other groups have quantified how trust varies with respect to the performance and error rate of the human–robot team [4, 7, 10], the nature of these failures [2], and the user's mental load [7].

Lee and Moray [15] developed a temporal model for characterizing human trust within a human–automation team setting. Our work shares several similarities with the authors' approach, and further extends analysis into the human–robot domain.

Among the earliest studies of trust in HRI, Hall [11] formulated a binary trust measure assigned to each state of the world, and devised an update mechanism for trust based on the robot's experiences. Freedy et al. [9] investigated the effects of mixed initiative robot control on a user's trust within a military tele-robotics setting. Hancock et al. [12] carried out a meta-analysis of empirical results from the HRI trust literature, and established quantitative estimates of various factors influencing trust across different interaction domains. Yagoda [21] gathered trust assessments from a broad audience by showing videos of human–robot interactions and eliciting users' trust responses through an online crowd-sourcing framework. Arkin et al. [1] studied aspects of trust and deception in a tele-robotics context. Our research shares various commonalities with all of these works, in the formulation, instantiation, elicitation, and evaluation of human–robot trust.

Our study of human–robot trust is most similar to the work by Desai et al. [5, 6], which carried out a multitude of investigations on trust within a search-and-rescue tele-robotics setting. In particular, the authors quantified the effects on trust of diverse interaction-level factors, including the level of autonomy, degree of robot reliability, amount of situational awareness, etc. Likewise, in this work we begin with a descriptive approach for studying human–robot trust, though at a finer time scale of the interaction experience. We then expand on the analysis further to extend towards a predictive model for the user's real-time trust assessments.

2.2 Trust Characterization

Several studies have highlighted the influences of various factors on the degree of a human's trust in a robot (e.g. [6, 12, 15]). These factors include:

- human's *demographic attributes*: e.g. age, gender, occupation;
- human's *attitudes and experiences*: e.g. propensity to trust robots, prior experience with robots and with task setting;
- human's perception of *robot attributes*: e.g. adaptability, benevolence;
- robot's *task performance*: e.g. internal automation failures, task errors;
- attributes of the *interaction setting*: e.g. communication quality, task complexity.

A key distinction among these factors is their *bases of trust*, which can be categorized into two main classes: certain trust factors relate to notions of the robot's competence, such as its task accuracy and consistency. These differ conceptually from factors related to the trustees intentions, pertaining for instance to the robots willingness and benevolence. Following the majority of research in robotics and automation, our work will take intention-centric bases of trust for granted, and assume that the robots designers did not include deceptive behaviors into its programming.

Measures of trust have been quantified using a number of different formats in the literature, including a binary representation [11], a continuous bounded measure [9, 15], and a multi-dimensional measure [14, 17]. Each representation has its own merits and drawbacks, and there is no "true" or perfect format unfortunately since trust is fundamentally a non-observable construct. In this paper, we quantify human–robot trust as 1-D bounded continuous measure, which enables both ease of user feedback and the application of standard statistical analysis techniques.

2.3 Interaction Context

Our research revolves around a team that is comprised of an autonomous robot and a human supervisor, collaborating on a common task. Ideally, the software agent governing the robot's autonomous behaviors is responsible for carrying out the bulk of the workload, while the operator predominantly monitors of the task progression. When the autonomous agent makes a mistake however, the human can actively intervene and provide corrective help by overriding the robot's commands. Given the nature of this supervisor-worker relationship, we assume that the human's intervening commands will *always supersede* those generated by the autonomous agent.

We have chosen to study vision-guided navigation as our primary application domain. Concretely, our human–robot setting consists of a human operator sharing control with an autonomous vision-based agent [19] over an aerial vehicle, while being tasked to track different terrain boundaries, such as coastlines and roads. Visual navigation tasks are appealing because humans are naturally inclined to solve them robustly and without effort. In addition, the necessary complexity in autonomous solutions (e.g. [6, 8]) warrants the need of trust. Finally, these setups are relevant to a wide variety of different robot platforms and application contexts.

Within our boundary tracking framework, the human operator is presented with a graphical interface showing the live camera feed from the robot, as seen in Fig. 1. The autonomous agent's internal state is overlaid on top of this view, in the form of the currently-tracked boundary curve and line fit, as well as the current steering command. These overlays provide transparency to the autonomous agent's sensing and control processes, and therefore help the user understand the robot's behaviors more clearly. In addition, whenever the boundary tracking algorithm fails to detect the boundary, the user can also readily perceive such faults by the absence of the boundary-related overlays.

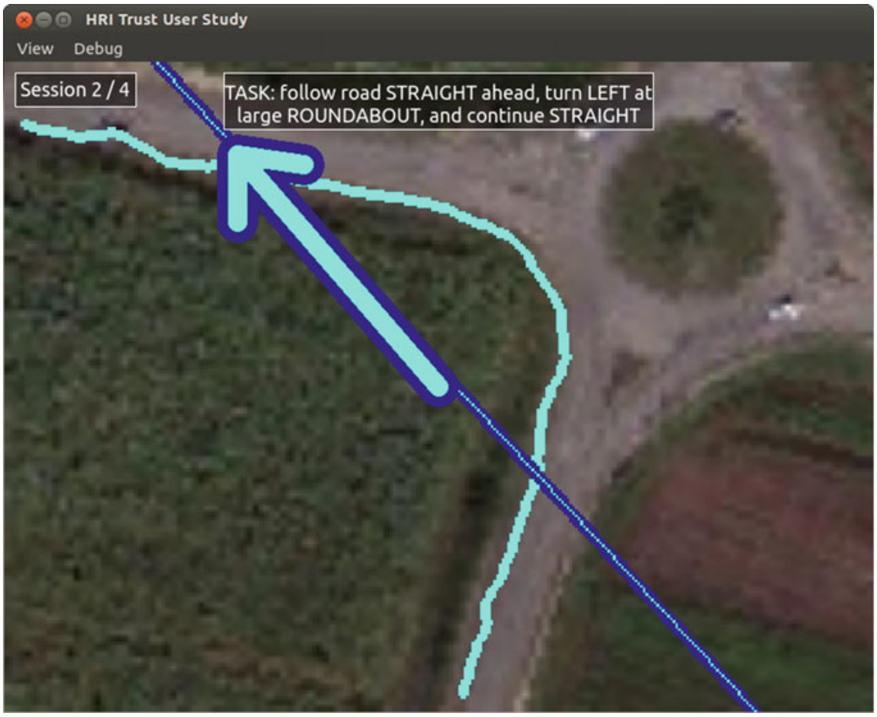

HRI Trust User Study
View
Debug
Session 2 / 4
TASK: follow road STRAIGHT ahead, turn LEFT at large ROUNDABOUT, and continue STRAIGHT

Fig. 1 Our boundary tracking framework provides an interface that overlays the autonomous tracker's internal state representation on a real-time display of the robot's camera view. Additional text overlays pertain to our human–robot interaction study, which are discussed in Sect. 3

The operator can take over control of the vehicle at any time, by holding down the mouse cursor over the camera display in the desired steering direction; these user interventions are reflected by a change in the steering arrow's color. The autonomous tracker is always running and its internal visualizations are also continuously displayed on-screen, even during periods of manual control. This allows the user to perceive when the agent has regained the tracked target, and thus also when to return control back to the autonomous system.

3 Methodology

We developed a human–robot interaction study to collect empirical data towards the analysis and modeling of real-time human–robot trust. In this study, users interacted with our autonomous boundary tracking agent to control a simulated aerial vehicle. Our simulation framework generates a bird's-eye camera feed of a non-holonomic fixed-wing aerial robot, based on real satellite imagery.

Although our boundary tracker has been previously deployed on both fixed-wing aerial robots [19] and quadrotors [20], we chose deliberately to target an aerial robot simulation framework for our user study in contrast. This setup allowed us to administer specific interaction experiences to different study participants in a *controlled* and *repeatable* manner, and also eliminated pragmatic concerns such as battery levels and fluctuations in environment conditions.

3.1 Event-Centric Perspective

We perceive a given period of human–robot interaction experience as a sequence of *discrete events*, where each event corresponds to a salient change in the state of the robot and/or the environment. Examples of such events within our visual navigation setting include a sustained period of internal failures in the boundary tracking algorithm, or a strong gust that pushes the aerial robot sideways and causes it to lose track of the target boundary. By measuring the human's *change in trust* in reaction to different types of events, we can quantify the progression of human–robot trust at a small time scale, namely short periods of interaction experience centered around each event. This event-centric view also differentiates our investigations from the majority of studies in the literature, which have characterized impacts on trust from aggregated interaction experiences on longer-term time scales (e.g. [6, 9, 21]).

Concretely, our user study is comprised of a number of short interaction sessions (<60 s each), where in each section the task is to follow a straight road for about 30 s till an intersection, make a predetermined turn, and then continue following the new path. We used different road segments and varied the turn directions in each session, in order to introduce diversity in the interaction experience and also prevent potential learning effects. The autonomous boundary tracker is capable of following sides of roads proficiently, but lacks the ability to switch between multiple target boundaries at intersections. Participants in our study were explicitly informed of this limitation prior to commencing the interaction sessions.

In this paper we focus specifically on events corresponding to robot failures, i.e. periods of decreased reliability in the robot's task performance. These drops in reliability are achieved by changing the parameter settings of our boundary tracking algorithm into a *low-reliability* state, where the autonomous agent poorly tracks roadside boundaries and also exhibits frequent internal failures in a non-predictable manner. We devised the following *event scenarios*, by programmatically toggling between reliability modes at different times during the course of each session:

- `Baseline`: the boundary tracking agent is set to high-reliability state throughout the entire session;
- `PoorStart`: the agent starts in the low-reliability state for 10 s and is then switched into the high-reliability setting, prior to the intersection,

- `RobotFault`: the agent is momentarily toggled into low-reliability state for a 10-s period in the middle of tracking the first road segment;
- `Limitation`: the agent is switched into low-reliability state at the road intersection, and then switched back into high-reliability state after 10 s.

We believe that robot operators typically behave in a *rational* manner, and will attribute blame following robot misbehaviors differently based on the cause of each failure. Our event scenarios are designed to elicit trust changes following different types of failures, namely poor initial tuning of the autonomous agent (`PoorStart`), algorithmic failure without any noticeable external cause (`RobotFault`), and failure due to limitations in the robot's programming (`Limitation`).

3.2 Trust Factors

The main purpose of our trust modeling work is for an autonomous robot to have the capability of predicting the human's trust in real-time during interaction. Starting from the rich corpus of factors that have been shown in the literature to influence human–robot trust, we exclude factors that are ill-defined, and those that are not possible for the robot to obtain, either via direct observation or by querying the human operator. In addition, since our research context assumes that the robot is always well-intentioned and is never adversarial, we also exclude all intention-based factors from our investigations. Therefore, our study is designed to collect interaction and user feedback data for quantifying the following *experience-based* trust factors:

- the robot's task performance (i.e. distance between robot and tracked target);
- the autonomous agent's internal failure rates;
- the frequency of interventions from the human operator;

In addition, our questionnaires elicit the following *assessment-based* factors:

- a pre-experiment survey: user demographics, general attitudes, and prior experience with robots and remote control (RC) tasks (following [5]);
- post-session questionnaires: assessments of the robot's and user's task performances, as well as the robot's perceived robustness and adaptability;
- a debriefing questionnaire: experiment-wide task load assessments (via Raw TLX [13]), and post-hoc updates on trust propensity towards autonomous robots.

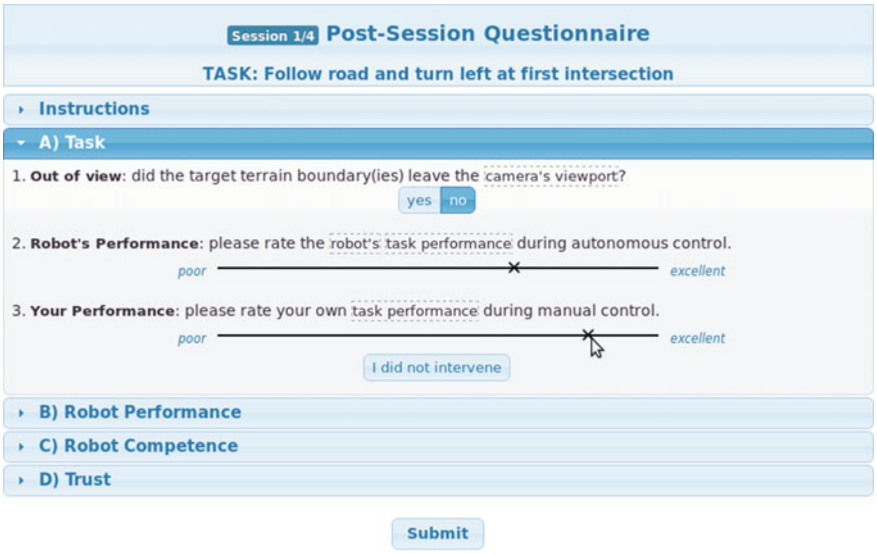
This study design highlights the important characteristic that human–robot trust is dependent on factors at *different time scales*. In particular, we expect responses to both the survey and debriefing questionnaires to be constant throughout the entire study, in contrast to per-session user assessments. These experience-based factors are summarized by statistics aggregated over the entire duration of each session, as well as within a short window of time following an event. The former set of measures reflect a cumulative characterization of the interaction experience during each session, whereas the post-event window-level statistics quantify immediate reactions from both the user and the robot following a discrete event.

3.3 User Assessment Elicitation

In addition to the various questionnaires for gathering assessment-based trust factors, we also asked users to indicate their degree of trust in the robot both before and after each session. The majority of these user queries employed the Visual Analogue Scale (VAS) [18] to elicit unipolar ("Likert-like") and bipolar responses. The VAS format, i.e. a continuous scale without tick marks (Fig. 2), was chosen for its superior metric properties over N-point discrete scales [18]. We have iterated over the design of our questionnaires, in order to mitigate common sources of biases [3].

3.4 Structure of Interaction Study

Figure 3 depicts the different phases in our interaction study, beginning with the survey questionnaire. A set of tutorial slides then provided explanations of the interaction


Session 1/4 Post-Session Questionnaire

TASK: Follow road and turn left at first intersection

Instructions

A) Task

1. **Out of view**: did the target terrain boundary(ies) leave the camera's viewport?

yes [✓ no]

2. **Robot's Performance**: please rate the robot's task performance during autonomous control.

poor ——————×—— excellent

3. **Your Performance**: please rate your own task performance during manual control.

poor ————————×— excellent

I did not intervene

B) Robot Performance

C) Robot Competence

D) Trust

Submit

Fig. 2 Questionnaires in our study are comprised predominantly of user assessments using a continuous Visual Analogue Scale (VAS) [18] (i.e. continuous slider without tick marks) as answering format, as well as a few discrete-choice queries

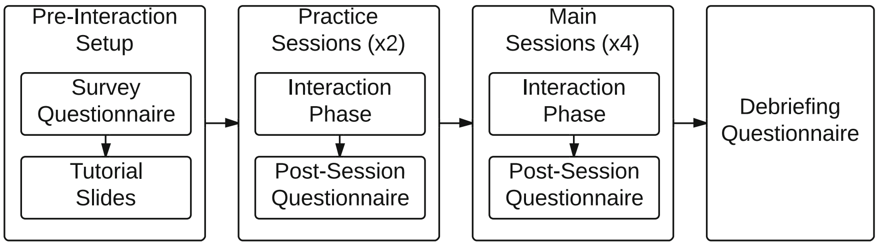


Fig. 3 The structure of our user study is designed around multiple controlled human–robot interaction sessions, which are supplemented by a number of questionnaires

and task context, the interface, as well as the robot's capabilities (i.e. configurable to track various types of boundaries) and limitations (i.e. incapability of changing tracked targets deliberately). The tutorial also explicitly asked users to assume that the autonomous robot was well-intentioned, and to therefore provide trust assessments based solely on the robot's performance and competence.

Next, participants interacted with boundary tracking robots during 6 distinct sessions, corresponding to 2 initial practice sessions, and followed by the 4 event scenarios in a random, counterbalanced order. Each session began by asking users to provide their initial trust assessments in the yet unseen robot, followed by the actual interaction phase, and ending with a post-session questionnaire.

In the first practice session, participants were instructed to get acquainted with the visual interface and robot controls during free roam, while tracking arbitrary terrain boundaries. The autonomous robot agent was toggled into its low-reliability state for short periods of time on several occasions, both to demonstrate robot failures as well as to implicitly prompt users to practice intervening and taking manual control of the vehicle. The second practice session consisted of a variant of the `Baseline` scenario, and acquainted participants with the road-following task objective by providing a demonstration of a typical interaction experience.

Following the 6 interaction sessions, the study concluded with a debriefing questionnaire that collected assessments of the aggregated interaction experience, as well as general feedback. This entire interaction study was completely automated, including the event triggers and data logging components.

A key aspect of this study design is that the user's trust assessments are affected by a full-fledged autonomous agent, operating within a simulated environment. This setup therefore provides conditions similar to a real world setting, while also enabling experimental control to ensure consistent and repeatable experiences.

4 Interaction Study Results and Analyses

We recruited 30 participants from the School of Computer Science at McGill University to participate in our interaction study. The user population is comprised of 24 males and 6 females, and included 11 undergraduate students, 13 graduate students, 2 professors, and 4 university personnel. Participants had varying degrees of experience operating and programming robots, although no user had prior interaction experience with our boundary tracking system.

The resulting dataset encompassed 60 practice session entries and 120 non-practice session entries. We carried out statistical analyses to investigate several key aspects of this human–robot trust dataset, including order effects resulting from our crossover session design, the effect of event scenarios on the amount of change in users' trust assessments, and the relative significance of various factors previously identified in the literature on the influence of real-time human–robot trust.

4.1 Session Order Effects and Properties of Pre-session Trust

In order to mitigate learning effects of the crossover study design from introducing unwanted biases, we explicitly emphasized, both during the tutorial and during pre-session trust elicitations, that the experience from each session should be assessed *independently* from previous ones, and that each session may encompass robots with different reliability levels, differing task objectives, as well as distinct environments.

Figure 4 shows the degrees of trust participants felt towards the boundary tracking robot prior to the start of each session. A repeated measures analysis of variance (rmANOVA) revealed no significant relationship between the mean pre-session trust values to the session ordering, $F(5, 145) = 0.30$ ($p = 0.91$), although there was a strongly significant effect from the different users, $F(29, 145) = 24.18$ ($p < 1e^{-16}$). We thus conclude that although trust assessments at the beginning of sessions varied among individual users, these measures were not noticeably affected by the session ordering.

Figure 4 also indicates that the experiment's population demonstrated slight positive bias for pre-session trust level that is above the uninformed prior response of 0.5. A one-way two-tailed Student's t-test revealed that the mean of the pre-session

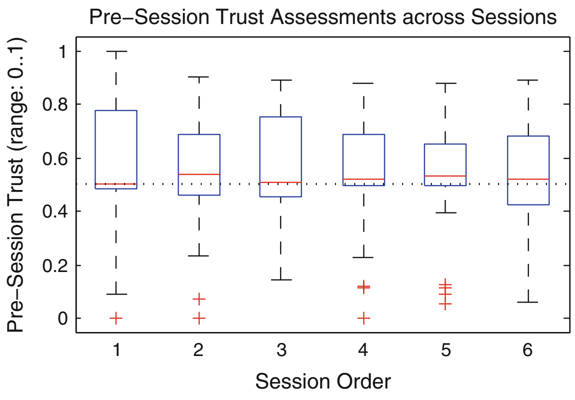


Fig. 4 Pre-session trust assessments from users are consistent across sessions and do not show any significant effects of the session ordering. These results from our study population also demonstrate slight positive bias in initial robot trust assessments, which are consistent with prior findings in the literature [6, 7]

trust assessments across all users and all sessions (including practice sessions) was significantly different from 0.5 ($p < 0.05$). This positivity bias is consistent with similar findings in other human–robot studies [6, 7].

4.2 Effects of Event Scenarios

Figure 5 depicts changes between pre- and post-session trust assessments, in response to different events during the 4 main sessions. Repeated measures ANOVA revealed significant effects on the mean amount of trust changes both due to events, $F(3, 87) = 16.61$ ($p < 1e^{-16}$), as well as due to users, $F(29, 87) = 2.35$ ($p = 1.22e^{-03}$). Post-hoc pairwise comparisons using the Tukey range test at the $p < 0.05$ level showed *non-significant differences* in trust changes between `Baseline` & `Limitation`, `PoorStart` & `RobotFault`, and `PoorStart` & `Limitation`.

Looking at the average user responses, the gain in trust in reaction to the high-performance boundary tracker settings of `Baseline` is logically expected, and similarly so are the losses in trust due to failures during the `PoorStart` and `RobotFault` scenarios. Although the difference in the amount of trust lost between `PoorStart` and `RobotFault` was not significant, Fig. 5 suggests that users reacted more leniently when the robot in `PoorStart` started with poor performance but then soon showed improvements, as opposed to when the initially-reliable robot in `RobotFault` unexpectedly failed to track a straight road.

In contrast, the typical response of *increase in trust* for the `Limitation` scenario may appear surprising at first, since the robot was programmatically switched into the low-reliability mode when it reached the intersection. Nevertheless, we believe

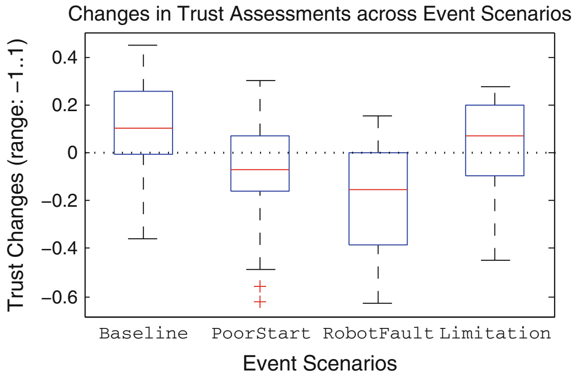


Fig. 5 Trust changes in response to different events are consistent with rational reactions based on causal attribution. In particular, the slight increase in trust in the `Limitation` scenario suggest that most users deliberately did not blame the robot for failing to execute a change in task objective, given known limitations in its programming

that users deliberately did not penalize these failures because they were actively conscientious that the autonomous boundary tracker lacked the capabilities of carrying out changes in the task objective. Therefore, the robot's momentary drop in reliability following the intersection can be interpreted in analogy to a switch from the `Baseline` to the `PoorStart` conditions, which is consistent with our post-hoc pairwise comparative analysis. In summary, these results indicate that overall, participants evaluated trust responses in a manner consistent to a rational mindset that both considered causal event assessments, and carried out deliberations with the robot's capabilities and limitations in mind.

4.3 Descriptive Analysis of Factors Influencing Trust Change

A backwards stepwise linear regression analysis was carried out to identify the most significant relationships between factors enumerated in Sect. 3.2, and changes to the user's trust level in response to different events. Starting from a full linear model involving all considered trust factors, the session order, and event scenarios, we applied stepwise regression using the Sum Squared Error (SSE) criterion, which iteratively removed insignificant factors (when $p > 0.1$) and re-introduced relevant factors (when $p < 0.05$). Interactions and high-order terms were disallowed to preclude spurious associations between factors at different time scales. Experience-based factors reflecting the immediate post-event reactions were computed over a 10-s window, corresponding to the duration of the pre-determined lapse into the low-reliability mode. Also, since no failure events occurred during the `Baseline` scenarios, event windows were chosen at random for the corresponding sessions.

Table 1 provides a summary of the stepwise regression results. Starting with 52 trust factors at different time scales, only 22 factors remained in the final model. Significant factors at the experiment-level time scale included demographic entries (e.g. age, occupation), prior expertise and attitudes (e.g. driving and robot control experience, willingness to use a self-driving car), as well as post-experiment assessments (e.g. measures of mental demand and performance). Session-level factors in

Table 1 Descriptive analysis summary of human–robot trust factors using stepwise regression

Factor categories (Initial factor count)	Final factor count	DF	Σ MS	Min. p	Avg. p	Max. p
Survey & debriefing assessments (24)	13	15	4.25	$<1e^{-10}$	<0.01	0.02
Post-session assessments (4)	2	2	1.27	$<1e^{-16}$	<0.01	0.02
Experience during session (12)	4	4	0.30	$<1e^{-2}$	0.02	0.04
Experience within 10 s window (12)	3	3	0.21	$<1e^{-3}$	0.27	0.79
Residual		84	0.01			

DF: degrees of freedom
Σ MS: combined mean sum of squares

the final model incorporated post-session user assessments of the robot's performance characteristics, as well as session-wide internal failure metrics. Finally, significant window-level factors corresponded to measures of short-term external task errors. Both the session order ($p = 0.60$) and event scenarios ($p = 0.20$) were excluded by the regression process. The final model showed excellent fit to the data, with a Root Mean Squared Error of $RMSE = 0.11$ and a goodness-of-fit of $R^2 = 0.83$.

We hypothesize that because the event scenario was categorical, it lacked metric precision in correlating to the quantitative amount of trust change, and was thus dropped in favor of other non-discrete window-level and session-wide metrics that characterized the interaction experience. More importantly, these experience-based factors were dwarfed in statistical significance compared to users' assessments, especially at the experiment-level time scale, as reflected by the aggregated Mean Sum of squares (MS) and average p-value statistics in Table 1. We therefore conclude that the evolution of real-time human–robot trust is dependent on each user's *personality* (e.g. expertise, beliefs, tendencies, and perceptions) more significantly than the *actual experiences* and their causalities during interaction.

5 Predictive Real-Time Trust Modeling

We now develop an initial model for predicting trust changes in real-time human–robot interactions, based on our statistical analyses above. This requires a critical change in methodology from our previous descriptive characterizations, which quantified the relationships between trust and its related factors using *all of the collected dataset*. In contrast, the main objective of predictive real-time trust modeling is to estimate event-centric trust responses during interactions with *potentially new users*, while having minimal or no prior knowledge about these users. This is achieved using the standard machine learning technique of Maximum Likelihood model parameter learning through cross-validation.

Section 4.3 previously revealed the strong dominance of personality-based factors over experience-based factors on event-centric human–robot trust. Unfortunately, such personalized information may not be available when the robot is interacting with a new user. Therefore, we excluded factors at the experiment-level time scale in this initial work towards predicting human–robot trust.

5.1 Parametric Event-Centric Trust Model

Our model for predicting changes in trust in reaction to interaction events is derived from the same stepwise regression approach previously used in our descriptive analysis. Specifically, event-based trust change $\Delta\mathbb{T}^{session} \in [-1..1]$ is quantified as a weighted linear sum (with weights ω) of both experience-based metrics at the post-event reactionary windowed time scale ($\mathbb{E}_i^{post-event}$) and at the event-centric session level ($\mathbb{E}_j^{session}$), as well as provided user assessments following each session ($\mathbb{A}_k^{session}$). This model is trained using supervised learning by computing the difference between the user's trust assessment before and after each session:

$$\begin{aligned} \Delta\mathbb{T}^{session}(W,Q) &= \frac{1}{Q}\left[round\left(Q\cdot\mathbb{T}^{post\text{-}session}\right) - round\left(Q\cdot\mathbb{T}^{pre\text{-}session}\right)\right] \\ &= \sum_{\forall i,j,k}\left(\omega_0 + \omega_i\,\mathbb{E}_i^{post\text{-}event}(W) + \omega_j\,\mathbb{E}_j^{session} + \omega_k\,\mathbb{A}_k^{session}\right) \end{aligned} \quad (1)$$

This trust model has two parameters: the post-event window duration W, and the quantization level Q for both the elicited pre-session and post-session trust assessments. The window duration parameter W is related to the temporal sensitivity in the predicted trust change, and allows our model to generalize to other HRI settings potentially, such as turn-based episodic human–robot interaction settings. In addition, within our specific domain of visual robot navigation tasks, W can be adjusted to optimize the predictability on trust changes from post-event experience-based metrics, such as internal robot failures and the rate of user interventions.

The trust response quantization parameter Q addresses a separate concern, namely that reported assessments within questionnaires may contain diverse sources of bias [3]. Some of these biases were addressed in our study design by using the Visual Analogue Scale (VAS) answer format, which exhibits desirable metric properties [18]. In conjunction, the Q parameter quantizes the user's trust responses in order to eliminate noise resulting from the variability in the exact pixel placement of selections on the VAS answer scale. Quantized trust responses are also re-normalized to facilitate comparisons between different quantization levels.

5.2 Parameter Learning

The predictive power of our trust model in Eq. 1 depends on the values of its parameters, namely the post-event window duration W, and the trust response quantization level Q. We used a Maximum Likelihood (ML) supervised learning and validation approach to determine optimal parameter settings. Concretely, we constructed a test set by isolating data from a random 20 % of users, and carried out parameter fitting using 6-fold cross-validation on the remaining 80 % of user data. We iterated over 11 window duration values of $W = [0.5, 1, 2, 3, 4, 6, 8, 10, 12, 15, 20]$ seconds, and across 13 trust quantization values, $Q = [3, 5, 7, 9, 11, 15, 21, 31, 51, 71, 101, 201, 501]$. Each stepwise regression model was built using the same procedure as described in Sect. 4.3. The accuracy of each model was then evaluated using the Root Mean Squared Error (RMSE) of the predicted trust changes in the cross-validation set.

In addition, we quantified the generalizability of the chosen model parameters by training and evaluating trust models on a variable-sized subset of data entries, after randomly excluding 20 % of users into a separate test set. Training-set and test-set RMSE values (i.e. $RMSE^{train}$ & $RMSE^{test}$) from multiple runs were aggregated to reflect the typical level of generalizability for our trust model.

5.3 Results and Discussion

Our parameter fitting procedure trained over 800 stepwise regression trust models. Among these, the model with the smallest $RMSE^{test}$ revealed optimal parameter values of $W = 2$ s and $Q = 31$ levels. Thus, within our boundary tracking task domain, metrics reflecting the robot's behaviors and the user's interventions within a 2-s post-event window was found to be most useful at predicting trust changes. In addition, the large magnitude of the selected trust quantization level $Q = 31$ indicates that this trust model has the potential to provide fine-scaled predictions of the quantitative change in the users' trust assessments.

Figure 6 depicts the progressions of the training-set and test-set errors averaged over 50 independent runs. The prediction errors of models built using the full training set were $\overline{RMSE}^{train}(96) = 0.13$ ($S.D. = 0.01$) and $\overline{RMSE}^{test}(24) = 0.19$ ($S.D. = 0.05$). The smooth asymptotic convergence in the training-set and test-set errors suggest that a significant portion of our training set was required to allow for generalization and avoid over-fitting. In addition, the small magnitude of $\overline{RMSE}^{train}$ and the gap after convergence between the training-set and test-set errors together indicate the presence of high variance in our learning technique. This implies that our

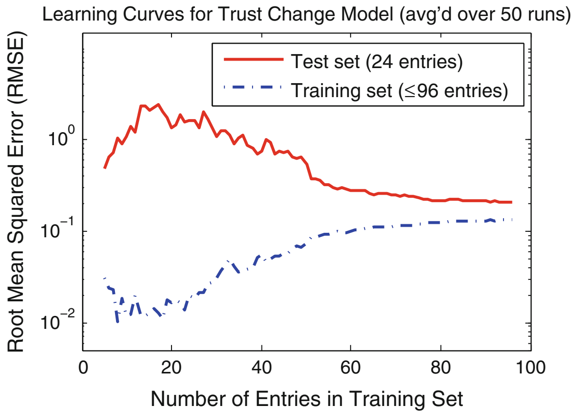


Fig. 6 Learning curves for our optimized trust model ($W = 2$, $Q = 31$), comparing the predictive error of different-sized training sets (up to 80 % of the entire dataset) and of a complementary test set, averaged over 50 independent runs. The asymptotic gap between the two error curves indicates that our learning algorithm exhibits high variance, which suggests that larger-sized datasets could further improve the final model's quality

(non-regressed) trust model structure has sufficient expressibility, and we would thus expect to gain further predictive power and generalizability by expanding the size of our training dataset.

Finally, by interpreting $\overline{RMSE}^{test}$ as the standard deviation in the typical prediction error in trust responses for novel users, we deduce that 95 % of times the predicted values differ from actual trust changes by ± 0.37 (recall that trust change values lie in $[-1..1]$). We acknowledge that the quantitative performance of our trained and regressed trust models in this work reflect only moderate levels of predictive power, especially compared to the numerical precision of the chosen trust quantization level, $1/Q \approx 0.03$. We suspect that a major source of the predictive error lies in the variability among different users, which has been consistently shown in our analyses to affect trust assessment, and therefore should be incorporated into our trust model in the future to further improve its predictive power.

6 Conclusions

In this work, we characterized key aspects of real-time trust in supervisor-worker human–robot teams, and illustrated these aspects concretely within a collaborative tele-robotics setting. We also carried out a controlled user study to collect both interaction experience and user assessment data, and whose results quantified different degrees of trust changes in reaction to various events during interaction. In particular, we found empirical support for the hypothesis that users in our interaction study typically behaved *rationally* and attributed trust changes based on the cause of each failure event. We further determined that the progression of human–robot trust was predominantly shaped by each user's *personality*, in comparison to the influence from the *actual experiences* in the interaction. Finally, we developed an initial, parametric model for predicting event-reactive trust changes within real-time continuous human–robot interactions, and empirically characterized its performance and generalizability.

We are currently working towards an extended version of our interaction study, which will target a wider user audience and a more elaborate set of event scenarios. We are also investigating ways to integrate personality-based factors into our real-time trust model, in order to further improve its predictive power. Separately, we are actively pursuing pragmatic applications of our real-time trust model to enable robots to intelligently adapt its behaviors based on different types of human interventions [20]. We anticipate that this work and its extensions will culminate into a robust real-time predictive trust model, which can then be applied to streamline the efficiency of human–robot collaborations.

Acknowledgments We would like to acknowledge the NSERC Canadian Field Robotics Network (NCFRN) for its funding support. We would also like to thank all of the participants who contributed to our user study.

References

1. Arkin, R.C., Ulam, P., Wagner, A.R.: Moral decision making in autonomous systems: Enforcement, moral emotions, dignity, trust, and deception. Proc. of the IEEE **100**(3), 571–589 (2012)
2. Bisantz, A.M., Seong, Y.: Assessment of operator trust in and utilization of automated decision-aids under different framing conditions. Ind. Ergon. **28**(2), 85–97 (2001)
3. Choi, B.C., Pak, A.W.: A catalog of biases in questionnaires. Preventing Chronic Disease **2**(1) (2005)
4. de Vries, P., Midden, C., Bouwhuis, D.: The effects of errors on system trust, self-confidence, and the allocation of control in route planning. Hum.-Comp. Studies **58**(6), 719–735 (2003)
5. Desai, M.: Modeling trust to improve human-robot interaction. Ph.D. thesis, Computer Science Department, U. Massachusetts Lowell (2012)
6. Desai, M., Medvedev, M., Vázquez, M., McSheehy, S., Gadea-Omelchenko, S., Bruggeman, C., Steinfeld, A., Yanco, H.: Effects of changing reliability on trust of robot systems. In: Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI'12), pp. 73–80 (2012)
7. Dzindolet, M.T., Peterson, S.A., Pomranky, R.A., Pierce, L.G., Beck, H.P.: The role of trust in automation reliance. Hum.-Comp. Studies **58**(6), 697–718 (2003)
8. Fong, T., Thorpe, C., Baur, C.: Collaboration, dialogue, and human-robot interaction. In: Proceedings of the International Symposium on Robotics Research (ISRR'01), pp. 255–266 (2002)
9. Freedy, E., DeVisser, E., Weltman, G., Coeyman, N.: Measurement of trust in human-robot collaboration. In: Proceedings of International Symposium on Collaborative Technologies and Systems (CTS'07), pp. 106–114 (2007)
10. Gao, F., Clare, A., Macbeth, J., Cummings, M.: Modeling the impact of operator trust on performance in multiple robot control. In: Proceedings of AAAI Spring Symposium: Trust and Autonomous Systems (2013)
11. Hall, R.J.: Trusting your assistant. In: Proceedings of the 11th Knowledge-Based Software Engineering Conference, pp. 42–51 (1996)
12. Hancock, P., Billings, D., Schaefer, K., Chen, J., De Visser, E., Parasuraman, R.: A meta-analysis of factors affecting trust in human-robot interaction. Hum. Factors: J. Hum. Factors Ergon. Soc. **53**(5), 517–527 (2011)
13. Hart, S.G.: NASA-Task Load Index (NASA-TLX); 20 years later. In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 50, pp. 904–908 (2006)
14. Jian, J.Y., Bisantz, A.M., Drury, C.G.: Foundations for an empirically determined scale of trust in automated systems. Int. J. Cogn. Ergon. **4**(1), 53–71 (2000)
15. Lee, J., Moray, N.: Trust, control strategies and allocation of function in human-machine systems. Ergonomics **35**(10), 1243–1270 (1992)
16. McKnight, D.H., Chervany, N.L.: The meanings of trust. Tech. rep, U. Minnesota (1996)
17. Muir, B.M.: Operators trust in and use of automatic controllers in a supervisory process control task. Ph.D. thesis, U. Toronto (1989)
18. Reips, U.D., Funke, F.: Interval-level measurement with visual analogue scales in internet-based research: VAS generator. Behav. Res. Methods **40**(3), 699–704 (2008)
19. Xu, A., Dudek, G.: A vision-based boundary following framework for aerial vehicles. In: Proceedings of the IEEE/RSJ International Conference on International Robots and Systems (IROS'10), pp. 81–86 (2010)
20. Xu, A., Dudek, G.: Trust-driven interactive visual navigation for autonomous robots. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'12), pp. 3922–3929 (2012)
21. Yagoda, R.E., Gillan, D.J.: You want me to trust a ROBOT? the development of a human-robot interaction trust scale. Soc. Robot. **4**(3), 235–248 (2012)

Optimal Control for Viscoelastic Robots and Its Generalization in Real-Time

Sami Haddadin, Roman Weitschat, Felix Huber, Mehmet Can Özparpucu, Nico Mansfeld and Alin Albu-Schäffer

Abstract Inspired by the elasticity contained in human muscles and tendons, viscoelastic joints are designed with the aim of imitating human motions by exploiting their ability to mechanically store and release potential energy. This distinct feature makes elastic robots especially interesting to the application of optimal control principles, as generating such motions is not possible by data-driven paradigms. In particular, reaching peak velocities by using the stored energy in the springs is of great interest, as such capabilities might open up entirely new application domains. In this paper, we review our results on solving various optimal control problems for elastic joints and full scale robot arms, as well as the experimental validation. Clearly, solving optimal control problems for highly nonlinear full robot dynamics is feasible nowadays only numerically, i.e. offline. In turn, optimal solutions would only contribute a clear benefit for real tasks, if they would be accessible/generalizable in real-time. For this, we developed a framework for executing near-optimal motions of elastic robot arms in real-time. In contrast to existing approaches, we use *dynamically optimal* motions (i.e. offline solutions of optimal control problems) as given learning input and then apply generalization via Dynamic Movement Primitives (DMPs). With this approach, we intend to overcome the well-known problems of optimal control and data-driven learning with associated generalization: being offline and

S. Haddadin (✉)
Institute of Automatic Control, Leibniz Universität Hannover,
Appelstr. 11, 30167 Hanover, Germany
e-mail: sami.haddadin@irt.uni-hannover.de

R. Weitschat · F. Huber · M.C. Özparpucu · N. Mansfeld · A. Albu-Schäffer
RMC, DLR e.V., Münchner Str. 20, 82234 Wessling, Germany
e-mail: roman.weitschat@dlr.de

F. Huber
e-mail: felix.huber@dlr.de

M.C. Özparpucu
e-mail: mehmetcan.ozparpucu@dlr.de

N. Mansfeld
e-mail: nico.mansfeld@dlr.de

A. Albu-Schäffer
e-mail: alin.albu-schaffer@dlr.de

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_8

being suboptimal (In fact, data-driven approaches can only be applied if the solution is already quite obvious for the human teacher. In case of highly nonlinear problems these "intuitive" initial solutions are typically not available.), respectively.

1 Introduction

Humans are capable of highly dynamic motions such as throwing or kicking. This is mainly possible due to their ability to store and release potential energy in their elastic musculoskeletal system in combination with inertial energy transfer between the rigid parts of the body. In robotics, however, intrinsic compliance is widely regarded as a clear drawback, degrading the associated quality measures such as repeatability and accuracy. Planning robot trajectories for rigid systems is one of the largest fields of robotics research [2, 7, 18, 20]. Since rigid robots are usually supposed to execute purely geometric tasks, the roles were distributed such that a low-level controller ensures accurate tracking behavior, while trajectory generation may remain on kinematic abstraction level. Another approach to generate trajectories originates from the learning community, where the *learning-by-demonstration* (LbD) paradigm is very common. Typically, a desired motion is taught to a robot kinesthetically [4, 23]. Alternatively, human motion tracking is used as a trajectory reference. A very popular, simple, and yet powerful scheme to efficiently encode these trajectories is the Dynamic Movement Primitives (DMPs) approach [16]. Apart from generalized replication of motions, DMPs allow also for incorporation of disturbance forces (for collision avoidance or retraction) [27]. In summary, there exist numerous successful motion generation approaches. However, a major drawback of the mentioned schemes is that they work maximally up to kinodynamic level. Therefore, they can never fully exploit the (often nonlinear) inherent dynamic capabilities of a robot.[1] As industrial robotics is mainly concerned with path tracking at maximum speed, several researchers considered this problem for rigid robots within the mathematical framework of optimal control (OC) [5, 25, 28, 29]. However, due to various inherent limitations and drawbacks of rigid actuation, a new class of robots that are equipped with intrinsically elastic actuation (or more general Variable Stiffness Actuation (VSA)) was developed over the last decade [1, 3, 30]. Its potential capabilities are expected to clearly outperform classical actuation by means of dynamic performance, mechanical robustness, and energy efficiency. At the same time the associated paradigm shift towards rather complex flexible joint mechanics introduced some underlying fundamental research questions to be answered. Most importantly, one wants to solve the problem of how to optimally store and release elastic energy such that it enables the execution of explosive motions. This question got thoroughly analyzed for 1 degree-of-freedom (DoF) elastic and VSA in [10, 12–15, 22], again

[1]Please note that in case of LbD we do not refer to a subsequent optimization according to some cost function after the trajectories were obtained.

with the tools from OC. However, apart from the analytic solutions for several basic cases, full scale systems with their nonlinear input and state constraints could only be solved numerically [6, 10, 11, 21], i.e. offline. Therefore, even though we know elastic robotic systems can indeed produce motions never seen before, we cannot access them in real-time. This brings us to the main contributions of our paper: First, we summarize our results on optimal excitation of elastic robot arms to generate explosive motions. For this, we present some interesting theoretical insights for the 1DoF case and experiments with the DLR Hand-arm system (*Hasy*). Then, we outline our approach to overcome the limitations of OC and LbD by combining them into a single Optimal Motion Framework (OMF). Specifically, we

1. use **dynamically** optimal trajectories coming from solving complex nonlinear optimal control problems as learning input,
2. encode them into an optimized dynamical system,
3. and use a metric based on the cost function or geometric distance for selecting a near-optimal parameter set in real-time for generalization.

Somewhat related approaches can be found in [17, 19]. Our approach discriminates from these existing techniques in several terms: First, we consider elastic systems for the first time. Apart from being 4th order systems, they have several additional constraints and nonlinearities to be considered. Furthermore, we combine entirely model-based optimal control[2] with generalization algorithms. Finally, we introduce the estimated cost function coming from the optimal learning input as an underlying generalization metric.

The paper is organized as follows. We review some of our results on how to optimally use (variable) joint elasticity for execution of high speed motions in Sect. 2. Section 3 describes our concept of merging prototypical optimal control problems with DMPs to finally generalize to unforeseen problems. Simulations and experiments support the validity of the approach. Finally, Sect. 4 concludes the paper.

2 Optimal Control for Viscoelastic Joint Robots

2.1 *Basics of Optimal Control*

Let us consider the class of systems that is described by a set of first order differential equations $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t))$, where $\mathbf{x}$ denotes the state vector and $\mathbf{u}$ the control input, respectively. The initial state is denoted by $\mathbf{x}(t_0) = \mathbf{x}_0$, the final constraints are $\phi(\mathbf{x}(t_f), t_f) = \mathbf{0}$, and the set of path constraints is $\mathbf{c}(\mathbf{x}(t), \mathbf{u}(t), t) \leq \mathbf{0}$. Solving an

[2]In contrast to pure nonlinear optimization, OC offers the tools to give stronger hints for global optimality of a solution.

optimal control problem aims at finding the control input $\mathbf{u}^*$ that minimizes a given optimality criterion

$$\min_{\mathbf{u}(t)} \mathcal{J} = h(\mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} g(\mathbf{x}(t), \mathbf{u(t)}, t)dt, \tag{1}$$

with $h(\mathbf{x}(t_f), t_f)$ being the terminal cost and $\int_{t_0}^{t_f} g(\mathbf{x}(t), \mathbf{u(t)}, t)dt$ the running cost. Essentially, the optimality criterion denotes the task to be accomplished and may take various forms.

2.2 Visco-Elastic Joint Dynamics

The dynamics of a full VSA arm can be described by the following set of equations

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}_J \tag{2}$$

$$B\ddot{\boldsymbol{\theta}} + \boldsymbol{\tau}_J = \boldsymbol{\tau}_m \tag{3}$$

$$\boldsymbol{\tau}_J = f(\boldsymbol{\phi}, \boldsymbol{\sigma}) + D_J(\dot{\mathbf{q}} - \dot{\boldsymbol{\theta}}), \tag{4}$$

where $\boldsymbol{\theta}$ and $\mathbf{q}$ are the motor and link side position, respectively, and σ is the position of the stiffness actuator. Note that σ is treated constant in this paper when considering the dynamics of the *Hasy* system in Sect. 2.4. $M(\mathbf{q})$, $C(\mathbf{q}, \dot{\mathbf{q}})$, $\mathbf{g}(\mathbf{q})$, and B are the link side inertia matrix, the centrifugal and Coriolis matrix, the gravity torque, and the motor inertia. $\boldsymbol{\tau}_m$ denotes the torque acting through the positioning motor. The vector $\boldsymbol{\tau}_J$ is the elastic joint torque, which is a nonlinear function of deflection $\boldsymbol{\phi} = \boldsymbol{\theta} - \mathbf{q}$, its derivative $\dot{\boldsymbol{\phi}}$, and $\boldsymbol{\sigma}$. Generally, $\boldsymbol{f}(\boldsymbol{\phi}, \boldsymbol{\sigma})$ is chosen to have $\frac{\partial f(\phi,\sigma)}{\partial \phi} > 0$ (increasing) and $\frac{\partial^2 f(\phi,\sigma)}{\partial^2 \phi} > 0$ (convex). The damping part of the joint torque due to joint damping D_J is assumed to be of additive linear form. If $f(.) = K_J(\mathbf{q} - \boldsymbol{\theta})$, then one typically calls the design Series Elastic actuation (SEA), with K_J being the positive definite, diagonal, and constant joint stiffness matrix. Please note that we typically assume that it is possible to solve (4) for $\boldsymbol{\theta}$, given $\boldsymbol{\sigma}$.

For a single joint with constant joint elasticity and damping (see Fig. 1) the considered system equations (2)–(4) simplify to

$$M\ddot{q} + D_J(\dot{q} - \dot{\theta}) + K_J(q - \theta) = 0, \tag{5}$$

$$B\ddot{\theta} + D_J(\dot{q} - \dot{\theta}) + K_J(q - \theta) = \tau_m. \tag{6}$$

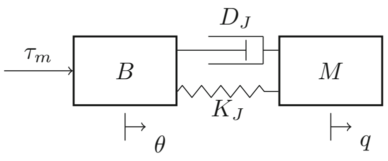


Fig. 1 1-DoF visco-elastic joint

In the following, we shortly review some of our results on solving OC problems for several 1-DoF cases analytically. Thereafter, numerical and experimentally verified solutions for *Hasy* are described.

An overview of the problems we solved so far is depicted in Fig. 2. For example, elastic 1-DoF joints with constant elasticity and damping that are controlled via motor velocity, acceleration, or torque (grouped as SEA) were considered in [14, 22]. Furthermore, different motor models and nonlinear joint torque/deflection curves have been analyzed in [12, 13]. For the VSA case, we derived analytical solutions for bounded input motor and stiffness/damping control in [15]. For the

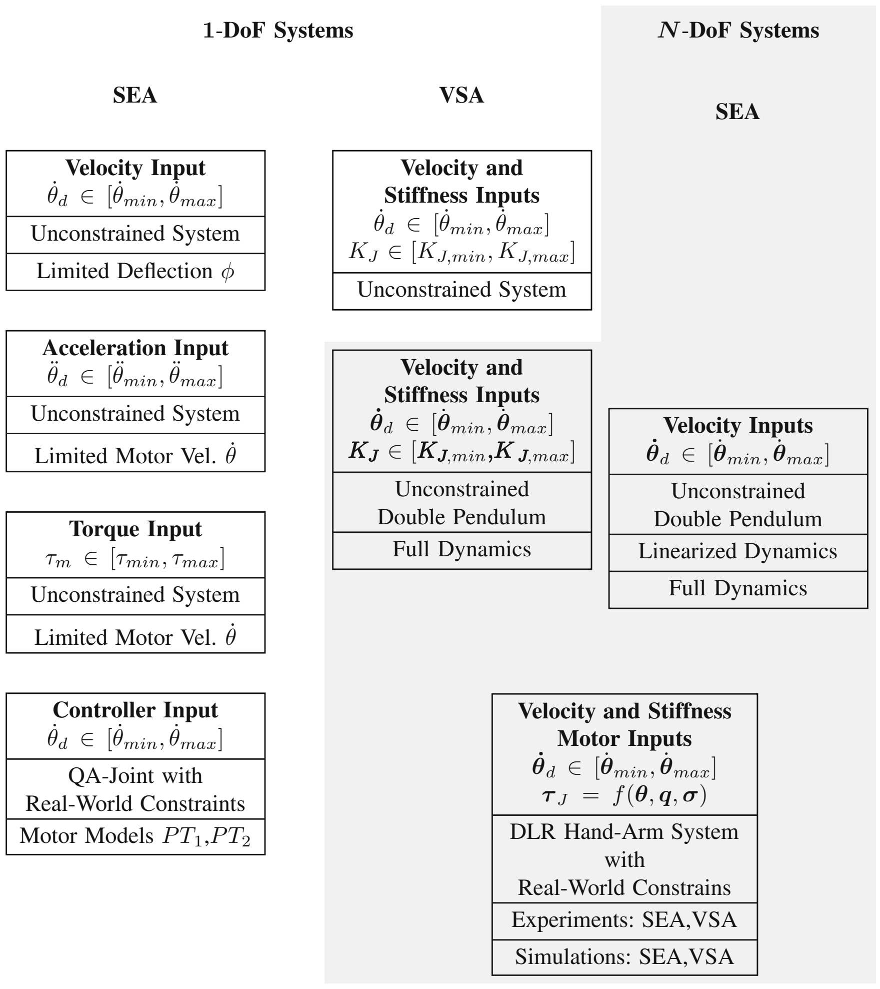

1-DoF Systems
N-DoF Systems
SEA
VSA
SEA
Velocity Input $\dot{\theta}_d \in [\dot{\theta}_{min}, \dot{\theta}_{max}]$
Unconstrained System
Limited Deflection ϕ
Acceleration Input $\ddot{\theta}_d \in [\ddot{\theta}_{min}, \ddot{\theta}_{max}]$
Unconstrained System
Limited Motor Vel. $\dot{\theta}$
Torque Input $\tau_m \in [\tau_{min}, \tau_{max}]$
Unconstrained System
Limited Motor Vel. $\dot{\theta}$
Controller Input $\dot{\theta}_d \in [\dot{\theta}_{min}, \dot{\theta}_{max}]$
QA-Joint with Real-World Constraints
Motor Models PT_1,PT_2
Velocity and Stiffness Inputs $\dot{\theta}_d \in [\dot{\theta}_{min}, \dot{\theta}_{max}]$ $K_J \in [K_{J,min}, K_{J,max}]$
Unconstrained System
Velocity and Stiffness Inputs $\dot{\boldsymbol{\theta}}_d \in [\dot{\boldsymbol{\theta}}_{min}, \dot{\boldsymbol{\theta}}_{max}]$ $\boldsymbol{K_J} \in [\boldsymbol{K}_{\boldsymbol{J},min}, \boldsymbol{K}_{\boldsymbol{J},max}]$
Unconstrained Double Pendulum
Full Dynamics
Velocity Inputs $\dot{\boldsymbol{\theta}}_d \in [\dot{\boldsymbol{\theta}}_{min}, \dot{\boldsymbol{\theta}}_{max}]$
Unconstrained Double Pendulum
Linearized Dynamics
Full Dynamics
Velocity and Stiffness Motor Inputs $\dot{\boldsymbol{\theta}}_d \in [\dot{\boldsymbol{\theta}}_{min}, \dot{\boldsymbol{\theta}}_{max}]$ $\boldsymbol{\tau}_J = f(\boldsymbol{\theta}, \boldsymbol{q}, \boldsymbol{\sigma})$
DLR Hand-Arm System with Real-World Constrains
Experiments: SEA,VSA
Simulations: SEA,VSA

Fig. 2 Considered OC problems for visco-elastic joint robots

N-DoF case, the effects of full dynamics, variable stiffness, and finally full robot dynamics, incorporating all relevant input and state constraints, as well as nonlinear torque-deflection curves, were obtained [11]. The 1-DoF velocity controlled motor dynamics ($u_1 = \dot{\theta} = \dot{\theta}_d$) with constant linear or nonlinear joint elasticity and constrained deflection give interesting insights into the problem, as explained next.

2.3 Time-Optimal Control for 1-DoF

The most important real-world state constraint to be considered when generating optimal control policies is the maximum angular deflection ϕ_{max}. Exploiting the maximum potential joint energy $E_{pot_{max}} = \frac{1}{2} K_J \phi_{max}^2$ to obtain the maximum additional velocity gain $\Delta\dot{q}_{max}$ is the primary goal. The maximum link velocity one can achieve in principle is

$$\dot{q}_{max} = u_{1max} + \omega\phi_{max} = \dot{\theta}_{max} + \omega\phi_{max} = \dot{\theta}_{max} + \Delta\dot{q}_{max}, \tag{7}$$

with $\omega = \sqrt{\frac{K_J}{M}}$ being the system's eigenfrequency. Figure 3 shows the constructed switching curves that can be obtained by bringing the problem into an ellipsoidic form and finding the optimal motor velocity $u_1 = \dot{\theta}_d$, which is of bang-bang type except for a possible singular arc [12]. The most important result is that the particular solution (number of switching cycles and presence of singular arc) depends on the energy ratio e_{SL}, which is defined as

$$e_{SL} := \frac{E_{pot_{max}}}{E_{kin}} = \left(\frac{\omega\phi_{max}}{\dot{\theta}_{max}}\right)^2. \tag{8}$$

Equation (7) can now be written as

$$\dot{q}_{max} = \dot{\theta}_{max}(1 + \sqrt{e_{SL}}). \tag{9}$$

The switching curve indicates that the motor needs to reverse its direction of speed each time the link speed grows more than two times the motor speed, i.e.

$$n_c = \left\lceil \frac{\dot{q}_{\max}}{2\,\dot{\theta}_{\max}} \right\rceil = \left\lceil \frac{1 + \sqrt{e_{SL}}}{2} \right\rceil, \tag{10}$$

where n_c is the number of motor cycles. In Fig. 3 (upper left) $e_{SL} = 11$ is chosen, i.e. three intervals starting with u_{1max} exist. If ϕ_{max} is lowered, the trajectory may intersect the switching curve $\mathcal{S}$. Figure 3 (upper right) visualizes this case, which occurs only if $\phi_{max} < \frac{u_{1max}}{\omega}$, respectively $e_{SL} < 1$. Keeping $u_1 = u_{1max}$, once ϕ_{max} is reached would lead to a violation of the constraint. It is therefore optimal to follow the constraint in this singular case.

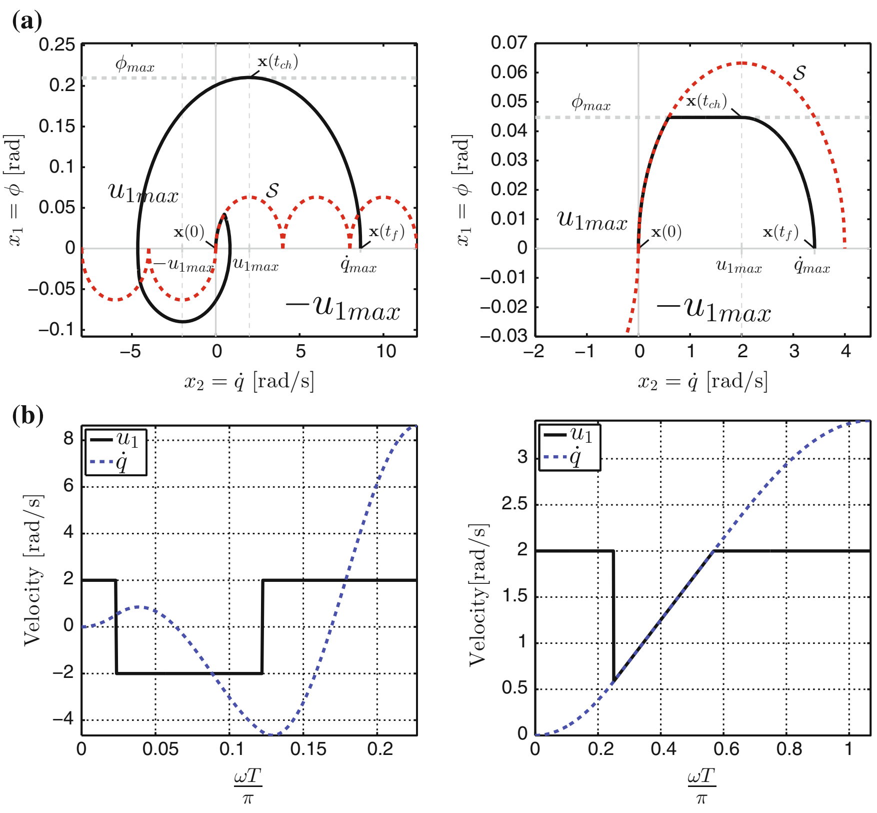


Fig. 3 Time optimal control strategy ($M = 0.1\,\text{kgm}^2$, $K_J = 100\,\text{Nm/rad}$, $u_{1max} = \dot{\theta}_{max} = 2\,\text{rad/s}$). **a** Phase plot ($\dot{q}$, ϕ). **b** Motor Velocity $\dot{\theta}$ and Link Velocity $\dot{q}$

Interesting to notice is that the results for constant K_J can be generalized to nonlinear $f(.)$. From an energy point of view, also the maximum kinetic energy of a nonlinear elastic joint $E_{kin_{max}}$ depends on the maximum elastic energy and the maximum motor velocity. In other words, (9) is still valid with e_{sl} now being defined as

$$e_{sl} = \frac{E_{pot_{max}}}{E_{kin}} = \frac{2\int_0^{\phi_{max}} \tau_J(\phi)\mathrm{d}\phi}{M\dot{\theta}^2_{max}} = \left(\frac{\dot{\phi}_{max}}{\dot{\theta}_{max}}\right)^2 . \tag{11}$$

Figure 4 shows phase plots of these trajectories for different e_{sl} values of the DLR QA-Joint, which has highly nonlinear joint torque characteristics [8]. e_{sl} is increased by lowering $\dot{\theta}_{max}$, while keeping $E_{pot_{max}}$ and thus ϕ_{max}, $\dot{\phi}_{max}$ constant. Since $|\dot{\phi}(0)| = u_{1max}$ and switching of u_1 changes $\dot{\phi}$ by $2u_{1max}$, the relative velocity $\dot{\phi}_{nc}$, representing $\dot{\phi}$ obtained with n_c motor cycles is bounded from above:

$$\dot{\phi}_{n_c} \leq (2n_c - 1)\dot{\theta}_{max}. \tag{12}$$

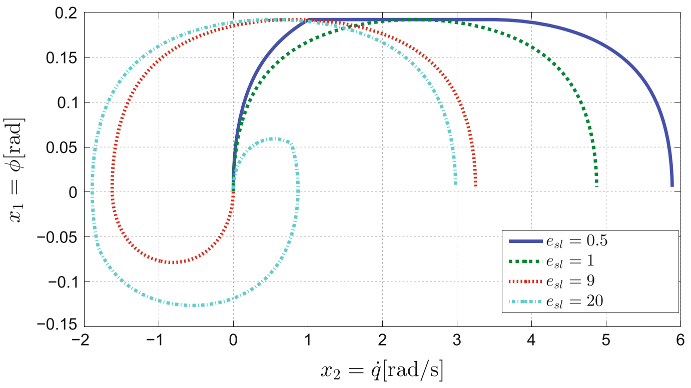


Fig. 4 Time optimal control for the DLR QA-Joint [8] $\left(u_1 = \dot{\theta}, a_s \approx 26.34, b_s \approx 28.62, \phi_{max} = \sigma = 11^o, \dot{\phi}_{max} \approx 2.44 \frac{rad}{s}\right)$

Consequently, the link velocity after n_c cycles satisfies $\dot{q}_{n_c} \leq 2n_c\dot{\theta}_{max}$, yielding a similar relation for n_c as in (10). For instance, with a single switching of the motor ($n_c = 2$), $\dot{q}_{n_c}$ can theoretically reach $4\dot{\theta}_{max}$ at most. Figure 4 shows that this velocity is indeed obtained in a time-optimal way by one switching of the motor at $\phi = 0$, when $e_{sl} = 9$ or equivalently $\dot{q}_{max} = 4\dot{\theta}_{max}$. Note that for $e_{sl} > 9$, the maximum velocity $\dot{q}_{max}$ cannot be obtained with one switching anymore. Next, we discuss key results for the nonlinear N-DoF.

2.4 *Optimal Control for N-DoF*

For the nonlinear N-DoF case, it is not possible anymore to derive the desired final state by physical reasoning anymore. Thus, the OC problem has e.g. to be stated in the form of maximizing link side velocity at given final time t_f: $J(\boldsymbol{u}) = -\vartheta(\boldsymbol{x}(t_f)) = -v_{TCP}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = -\mathbb{J}(\mathbf{q})\dot{\mathbf{q}}$, with $\mathbb{J}$ being the end-effector Jacobian.[3] The following numerical optimal trajectories are obtained with the nonlinear optimal control solver GPOPS [24].

2.4.1 Simulation

Figure 5 depicts results for the variable stiffness[4] case without state constraints. Basically, the solution is of bang-bang type for both, velocity and stiffness input,

[3]Please note that we explicitly are not interested in an integral energy-type cost term (except for using it for slight regularization) that intends to minimize the energy of the task. In fact, the problem we consider rather aims at maximizing the energetic state of the robot at a certain time instant.

[4]For this analysis we consider the stiffness K_J to be a direct control input, instead of being an implicit relation coming from the torque-deflection curve and $\boldsymbol{\sigma}$, meaning to chose $\boldsymbol{\tau}_J = K_J(t)(\boldsymbol{\theta} - \mathbf{q})$.

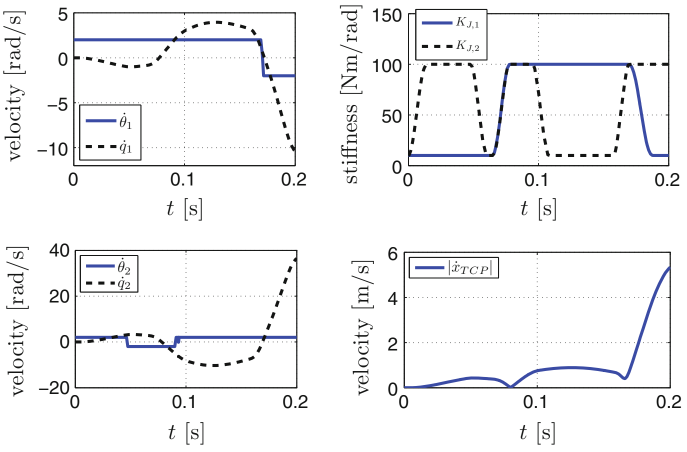


Fig. 5 Optimal solution for maximum link side velocity with constrained variable stiffness input and full nonlinear dynamics

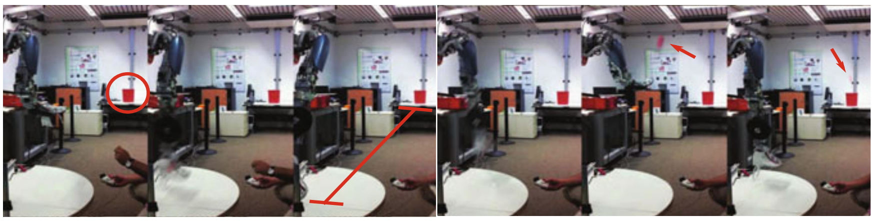


Fig. 6 Throwing experiment with *Hasy*

respectively. Following observations can be made. For $t \in [0.1\,\mathrm{s}\ \ 0.15\,\mathrm{s}]$ link 2 "decouples" ($K_{J,2} = K_{J,\min}$) in order to swing back without "being held" by the otherwise high joint torque. Then, during final acceleration, the rapid deceleration of the first link (the motor velocity switches its sign) causes an inertial energy transfer into the distal part. In addition to this effect, the stiffness behavior for $t \in [0.15\,\mathrm{s}\ 0.2\,\mathrm{s}]$ contributes an additional energetic input for link 2: Joint 2 switches to maximum stiffness $K_{J,2} = K_{J,\max}$ for rapid acceleration due to a jump in elastic joint force. Shortly hereafter the first joint "decouples" at $t \in [0.15\,\mathrm{s}\ \ 0.18\,\mathrm{s}]$ for preventing unnecessary "pulling torque" to act on link 2. At final time, the link side velocity in each joint exceeds the respective motor velocity multiple times.

In the following experiment (Fig. 6), the goal is to throw a ball as far as possible by maximizing the throwing distance d_{Bin}. This optimal control problem is of course strongly related to the already considered maximum velocity problem.

2.4.2 Experiment

The experiment uses a shoulder and the elbow joint at fixed stiffness preset σ^*. The ball launches in the fifth image at an angle of 45° and scores into the bin, which is placed at $\approx$5.5 m distance from the robot base.

Figure 7 shows measurements vs. simulations for $\dot{\theta}_{max} = 2$ rad/s in both joins. The upper left plot depicts $\boldsymbol{\theta}$ and $\mathbf{q}$, the upper right one $\dot{\mathbf{q}}$, the lower left one the absolute Cartesian velocity (which reaches 3.4 m/s at launch), and the lower right ϕ. Simulation and experiment are in well accordance, except for some additional oscillatory behavior in simulation, as we do not consider the (rather low) friction of the real system. If we assumed a stiff robot driving at maximum motor velocity $\dot{\theta} = 2$ rad/s, its maximum Cartesian velocity would be $\approx$2.2 m/s, i.e. the elastic robot is $\approx$50 % faster, see Fig. 7 (lower left).

In the following section, we review our optimal motion framework (OMF), which enables the execution of near-optimal motions in real-time.

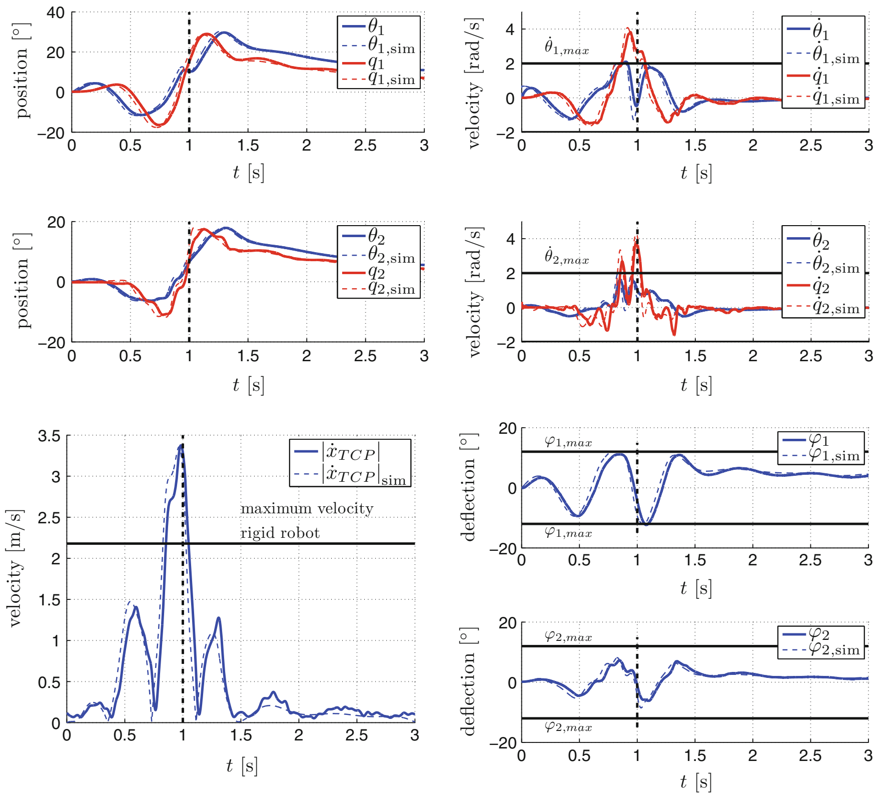


Fig. 7 Experimental performance of the optimal throw with *Hasy*

3 Optimal Motion Framework

3.1 Prototypical Optimal Control Problems

By nature, the motion generation problem is infinitely large and generally poorly defined in the sense of what a desired motion should exactly achieve. Essentially, it is idle to find a general optimal control problem that inherently contains all possible instantiations and thus captures the essence of motion. Therefore, we pragmatically resolve this dilemma by introducing *prototypical optimal control problems*. These are sought to find an optimal control input $\mathbf{u}^*$ that generates a distinct type of qualitative motion for a given robotic system and potentially a secondary system the robot is associated to (e.g. an object to be manipulated, which is associated to a second target dynamics $\mathbf{z}$). The following classification aims at grouping motion behaviors according to their "higher-level" target. We coarsely distinguish between *reaching type* and *tracking type*. The motion type, the constraints $\mathbf{c}(\mathbf{x}(t), \mathbf{u}(t), t)$, and the task vector $\boldsymbol{\xi}$ (containing, e.g., the goal location $\mathbf{q}_f$ (*reaching type*) or the bin location $\mathbf{x}_{Bin}$ in case of throwing (*implicit target type*)) define the respective optimal control problem to be solved. More details on our classification from Fig. 8 can be found in [31]. Please note that we outline the OMF on joint level for sake of clarity. Next, the embedding of optimal trajectories into DMPs and the generalization step are reviewed.

3.2 Encoding

Figure 9 compares the overall structure of our OMF (lower) to classical learning approaches (upper). We start from a given set of m task vector instantations ($\{\boldsymbol{\xi}_k\}$) with the respective cost function and associated constraints. Basically, we solve the prototypical OC problem for a certain grid of the task vector. Solving the optimal

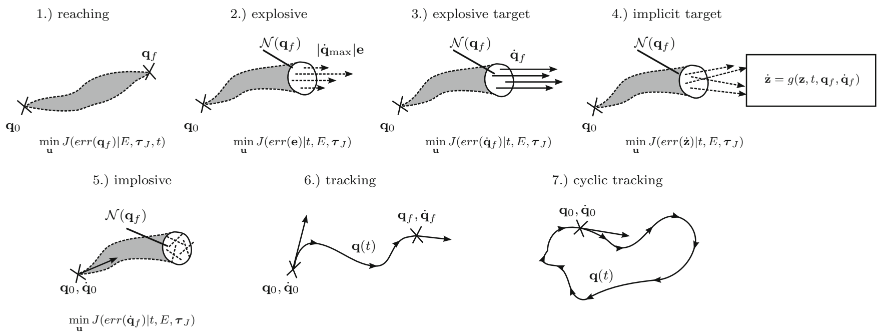

1.) reaching
$\mathbf{q}_f$
$\mathbf{q}_0$
$\min_{\mathbf{u}} J(err(\mathbf{q}_f)|E, \boldsymbol{\tau}_J, t)$
2.) explosive
$\mathcal{N}(\mathbf{q}_f)$
$|\dot{\mathbf{q}}_{\max}|\mathbf{e}$
$\mathbf{q}_0$
$\min_{\mathbf{u}} J(err(\mathbf{e})|t, E, \boldsymbol{\tau}_J)$
3.) explosive target
$\mathcal{N}(\mathbf{q}_f)$
$\dot{\mathbf{q}}_f$
$\mathbf{q}_0$
$\min_{\mathbf{u}} J(err(\dot{\mathbf{q}}_f)|t, E, \boldsymbol{\tau}_J)$
4.) implicit target
$\mathcal{N}(\mathbf{q}_f)$
$\dot{\mathbf{z}} = g(\mathbf{z}, t, \mathbf{q}_f, \dot{\mathbf{q}}_f)$
$\mathbf{q}_0$
$\min_{\mathbf{u}} J(err(\dot{\mathbf{z}})|t, E, \boldsymbol{\tau}_J)$
5.) implosive
$\mathcal{N}(\mathbf{q}_f)$
$\mathbf{q}_0, \dot{\mathbf{q}}_0$
$\min_{\mathbf{u}} J(err(\dot{\mathbf{q}}_f)|t, E, \boldsymbol{\tau}_J)$
6.) tracking
$\mathbf{q}_f, \dot{\mathbf{q}}_f$
$\mathbf{q}(t)$
$\mathbf{q}_0, \dot{\mathbf{q}}_0$
7.) cyclic tracking
$\mathbf{q}_0, \dot{\mathbf{q}}_0$
$\mathbf{q}(t)$

Fig. 8 Grouping motion behaviors into *reaching type* (1-5) and *tracking type motions* (6-7)

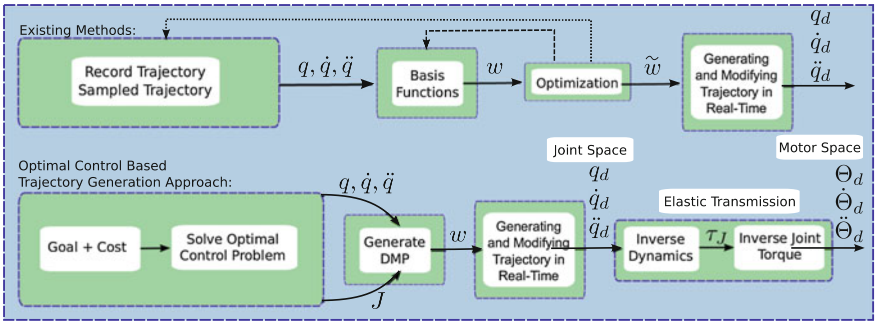

Existing Methods:
Record Trajectory Sampled Trajectory
$q, \dot{q}, \ddot{q}$
Basis Functions
w
Optimization
$\tilde{w}$
Generating and Modifying Trajectory in Real-Time
q_d $\dot{q}_d$ $\ddot{q}_d$
Joint Space
Motor Space
Optimal Control Based Trajectory Generation Approach:
Goal + Cost
Solve Optimal Control Problem
$q, \dot{q}, \ddot{q}$
J
Generate DMP
w
Generating and Modifying Trajectory in Real-Time
q_d $\dot{q}_d$ $\ddot{q}_d$
Elastic Transmission
Inverse Dynamics
τ_J
Inverse Joint Torque
Θ_d $\dot{\Theta}_d$ $\ddot{\Theta}_d$

Fig. 9 Towards an optimal motion framework. w denotes weights of a suitable basis

control problem numerically yields sampled trajectories defined by link side position $\mathbf{q}_k^*(t_i)$, velocity $\dot{\mathbf{q}}_k^*(t_i)$ and acceleration $\ddot{\mathbf{q}}_k^*(t_i)$ in joint or operational space. Deploying these trajectories into a second order differential equation (DMP)

$$\mathbf{f}_k^*(t_i) = -\tau^2\ddot{\mathbf{q}}_k^*(t_i) + \kappa(t_i)(\mathbf{q}_k^*(\tau) - \mathbf{q}_k^*(t_i)) - D\tau\dot{\mathbf{q}}_k^*(t_i) \tag{13}$$

with a total movement duration τ, a stiffness factor $\kappa(t)$, and damping D we obtain a force-based trajectory $\mathbf{f}_k^*(t_i)$ (see e.g. [9]). The dynamical force $\mathbf{f}_k^*(\mathbf{v})$, which is a function of a canonical system $\mathbf{v}$, is then encoded into an appropriate approximative basis, involving several optimization steps. The approximative force is denoted $\mathbf{f}_{\approx}(\mathbf{v})$, see e.g. [9]. Thereafter, we may generalize the motion along $t \in [0 \cdots t_f]$ for different goals $\mathbf{g}(\boldsymbol{\xi})$ by solving

$$\mathbf{q}(t) = \frac{1}{\tau^2}\int_0^{t_f}\int_0^{t_f} \mathbf{f}_{\approx}(\mathbf{v}) + \kappa(t)(\mathbf{g}(\boldsymbol{\xi}) - \mathbf{q}) - D\tau\dot{\mathbf{q}}\ \mathrm{d}t\ \mathrm{d}t + \mathbf{q}(0) \tag{14}$$

Note that the goal configuration $\mathbf{g}$ depends on the task vector $\boldsymbol{\xi}$. From (14) we obtain position, velocity, and acceleration in real-time, which can be inserted into (2). The motor trajectories $\boldsymbol{\theta}(t)$ can now be obtained by solving (2) for $\boldsymbol{\tau}_J$ and then apply (4). However, this simple generalization approach works only well for new goals that are close to the initial instantiations $\mathbf{g}_k(\boldsymbol{\xi}_k)$. Next, a modified generalization procedure is introduced that solves this problem.

3.3 Generalization

In order to make use of the fact that "close" trajectories (DMPs) presumably encode more relevant information for a new goal, distance-based weighting is applied for the interpolation step. For the reaching movement generalization, we may apply the method from [26] and get for each joint l the interpolated weights

$$\mathbf{w}_l^*(\boldsymbol{\xi}_g) = \frac{\sum\limits_{\forall k:\sigma_k \leq \delta} \mathbf{w}_l^*(\boldsymbol{\xi}_k)\sigma_k^{-1}}{\sum\limits_{\forall k:\sigma_k \leq \delta} \sigma_k^{-1}}. \tag{15}$$

Equation (15) is a sum of kernel weights multiplied with the inverse of the geometric or cost-based distances between the previously learned and the new goal. σ_k is defined as

$$\sigma_k = ||\boldsymbol{\eta}_g - \boldsymbol{\eta}_k|| + \epsilon, \tag{16}$$

where $\boldsymbol{\eta}_g$, $\boldsymbol{\eta}_k$ denote the new goal and k-th task vector ($\boldsymbol{\xi}_g$, $\boldsymbol{\xi}_k$) or goal cost function value (J_g, J_k), respectively. ϵ prevents division by zero. Subsequently, some simulations and experiments obtained with the OMF for reaching and throwing movements are outlined.

3.4 Simulations and Experiments

3.4.1 Reaching Movements

Based on energetically optimal solutions of the reaching movement problem (find an optimal path between inital and final configuration), near-optimal reaching motions can now be executed in real-time for varying goals. For this, a grid of optimal solutions with minimal required energy is calculated over the entire workspace, see Fig. 10. The upper left plot depicts sampled optimal reaching movements for the robot workspace. The 2DoF robot in stretched out configuration is indicated in black. The color value indicates the value of J for every trajectory, i.e. the consumed energy for the motion. The lower left plot shows a close up around $\mathbf{y} = (0.4, 0.95)^T$. The optimal trajectories that are used for the learning step are indicated by the crosses in the corner. The generalized DMP motion towards the new goal within the rectangle is compared against the according optimal motion, which was **not** used for learning. It was generalized inside this section by weighting kernel parameters with inverse distance weighting (16). The two upper right plots show optimal and DMP trajectories. The randomly generalized trajectory are compared with the optimal trajectory (checked afterwards). Furthermore, the lower two plots on the right show the difference between geometric and cost-based weighting. Figure 11 provides a comparison between OC trajectories, DMPs with geometric weighting (average $J_{err} = (\sum\limits_{i=1}^{N} J_{DMP_i} - J_{OC_i})/\sum\limits_{i=1}^{N} J_{OC_i} = 2.66\,\%$), cost-based generalization (average $J_{err} = 4.61\,\%$), and a common 5th order polynomial interpolation (average $J_{err} = 48.03\,\%$). To sum up, the distance-based generalization yields the best results for the examined task. However, in particular for strong nonlinearities in the cost manifold we believe that the cost-based metric can e.g. serve as a first filter

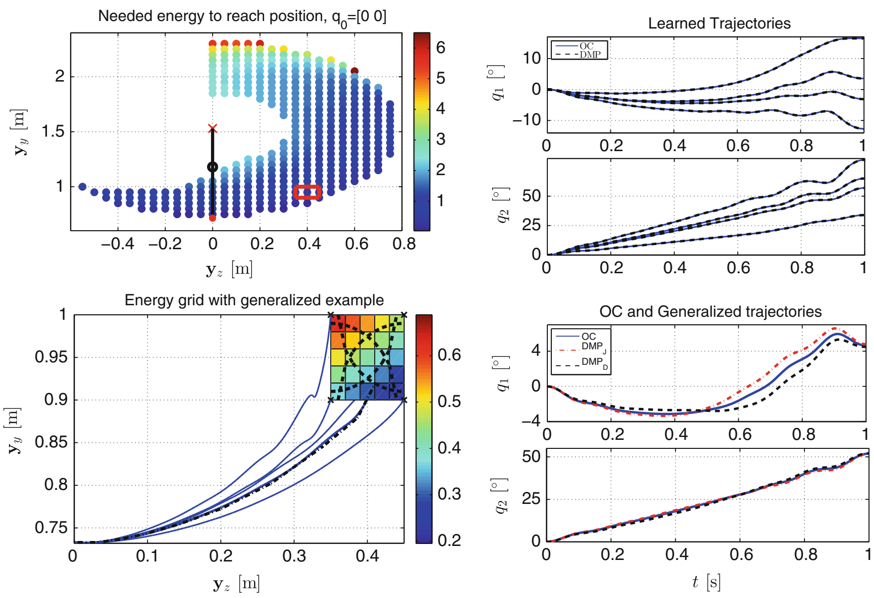


Fig. 10 Learning and generalization of optimal reaching movements

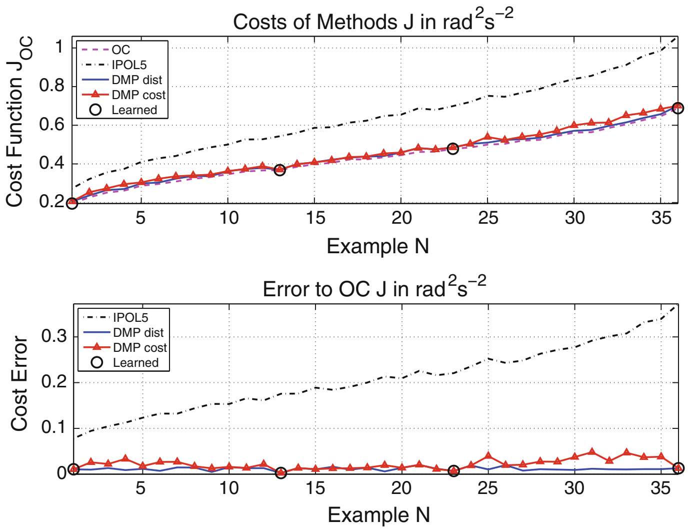


Fig. 11 Comparison of cost functions for OC, DMP reconstruction, and polynomial interpolation

for rejecting samples too far away in the cost sense. However, this is left for future investigation.

Next, we discuss the application of the OMF to execute optimal explosive motions for *Hasy* system in real-time.

3.5 Throwing with the DLR Hand-Arm System

Three trajectories for different throwing distances ($\{\boldsymbol{\xi}_i\} = \{[d_{Bin,i} \;\; 0 \;\; 0]^T\}$) were obtained by solving the according optimal control problems. Non-optimized goals are simulated for comparison between desired and resulting goal positions, see Fig. 12. It depicts the full movement consisting of the throwing task and the subsequent stopping motion (2-phase OC problem). These trajectories show the comparison between online generated and optimized solutions. Again, the generalized motions are very close to the optimal solution. Opening the hand for releasing the ball is triggered at $t = 1$ s. For validating the approach experimentally, the task is to throw a ball into a bin that is manually placed by a human at unforeseen target locations. With a Microsoft Kinect the according target distance d_{Bin} is measured. This parameterizes the OMF instantaneously such that the robot can execute the throw. Figure 13 depicts a photo series of three different bin distances. The success rate to hit the bin is high (in average 80 %). Failures depend mainly on whether the ball can be placed repetitively in the hand. This, however, requires some training by the operator.

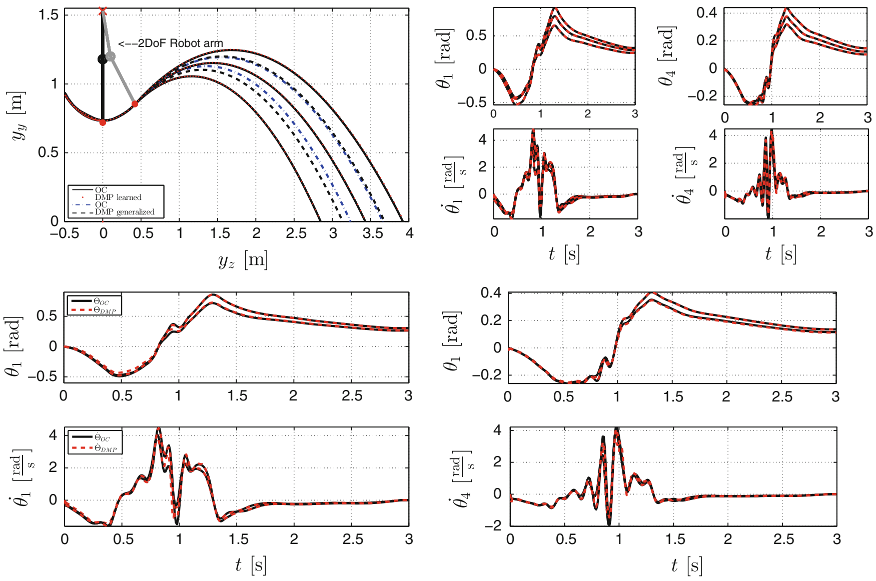


Fig. 12 Learning and generalization of optimal throwing movements. Optimal and generalized throwing trajectories in Cartesian space with flight trajectory of the ball (*top left*). Learned trajectories and velocities in joint space with resulting optimal motor trajectories and velocities (*four upper right*). Generalized throwing trajectories and velocities in joint space with resulting motor trajectories in comparison with optimal trajectories (*lower four*)

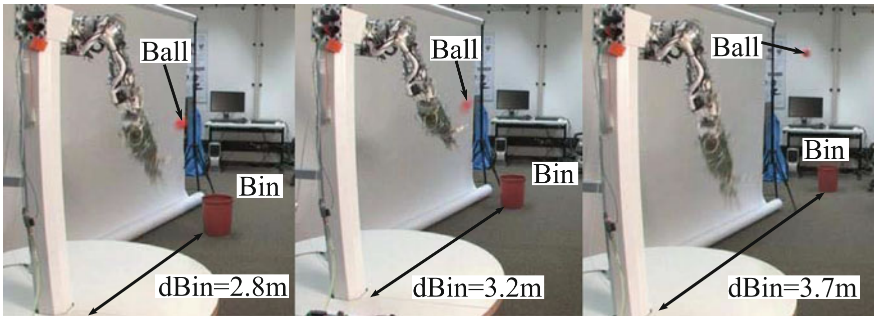


Fig. 13 Throwing sequence for varying target (bin). The bin distance d_{Bin} is measured with a Kinect sensor. All trials were successful scores

4 Conclusion

In order to to exploit the novel capabilities of intrinsically elastic arms and thus enable robots to achieve human-like dynamic performance, it is essential to understand and make use of their inherent flexible dynamics. In this paper we unveiled some of the underlying properties of VSA arms and how to optimally store and release elastic energy in their joints such that highly dynamic motions become possible. Due to the significant nonlinearity and complexity of the problem, candidate trajectories cannot be simply generated with the help of data-driven approaches (e.g. LbD). Instead, rigorous model-based problem formulation have to be applied and optimal solutions to be found with appropriate numerical solvers. In turn, one has to accept the cost of larger computational efforts for finding a solution (in particular when considering all relevant real-world constraints and the highly nonlinear flexible dynamics). This, however, prevents real-time use of the controls/trajectories, making them an insightful result but not applicable to real-world problems in dynamic environments. In order to solve this dilemma, we introduced an approach for learning optimal motions and generalizing them in real-time. The basic idea is to compute a sample set of optimal trajectories that are encoded into DMPs for subsequent generalization in terms of metric-based weight interpolation. The developed motion framework performs a variety of near-optimal motions in real-time and was validated in simulation and experiment on the anthropomorphic DLR robot *Hasy*.

Acknowledgments This work has been partially funded by the European Commission's Sixth Framework Programme as part of the project SAPHARI under grant no. 287513.

References

1. Albu-Schäffer, A., Eiberger, O., Grebenstein, M., Haddadin, S., Ott, C., Wimböck, T., Wolf, S., Hirzinger, G.: Soft robotics: From torque feedback controlled lightweight robots to intrinsically

compliant systems. IEEE Robot. Autom. Mag.: Spec. Issue Adapt Compliance/Var. Stiffness Robot. Appl. 15(3):20–30 (2008)
2. Biagiotti, L., Melchiorri, C.: Trajectory Planning for Automatic Machines and Robots. Springer, Berlin (2008)
3. Bicchi, A., Tonietti, G.: Fast and soft arm tactics: Dealing with the safety-performance trade-off in robot arms design and control. IEEE Robot. Autom. Mag. **11**, 22–33 (2004)
4. Billard, A., Calinon, S., Dillmann, R., Schaal, S.: Robot programming by demonstration. Handbook of Robotics, Chap 59 (2008)
5. Bobrow, J.E., Dubowsky, S., Gibson, J.S.: Time-optimal control of robotic manipulators along specified paths. Int. J. Robot. Res. **4**(3), 3–17 (1985)
6. Braun, D., Howard, M., Vijayakumar, S.: Exploiting variable stiffness in explosive movement tasks. In: Robotics: Science and Systems (2011)
7. Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G., Burgard, W., Thrun, S., Kavraki, L.E.: Principles of Robot Motion: Theory, Algroithms, and Implementation. MIT Press, Cambridge (2005)
8. Eiberger, O., Haddadin, S., Weis, M., Albu-Schäffer, A., Hirzinger, G.: On joint design with intrinsic variable compliance: Derivation of the DLR QA-joint. IEEE International Conference on Robotics and Automation, pp. 1687–1694 (2010)
9. Gams, A., Petric, T., Zlajpah, L., Ude, A.: Optimizing parameters of trajectory representation for movement generalization: robotic throwing. In: Proceedings of IEEE 19th International Workshop on Robotics in Alpe-Adria-Danube Region, pp. 161–166 (2010)
10. Garabini, M., Passaglia, A., Belo, F.A.W., Salaris, P., Bicchi, A.: Optimality principles in variable stiffness control: the vsa hammer. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3770–3775 (2011)
11. Haddadin, S., Huber, F., Albu-Schäffer, A.: Optimal control for exploiting the natural dynamics of variable stiffness robots. In IEEE International Conference on Robotics and Automation, pp. 3347–3354 (2012)
12. Haddadin, S., Krieger, K., Mansfeld, N., Albu-Schäffer, A.: On impact decoupling properties of elastic robots and time optimal velocity maximization on joint level. In IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5089–5096 (2012)
13. Haddadin, S., Weis, M., Wolf, S., Albu-Schäffer, A.: Optimal control for maximizing link velocity of robotic variable stiffness joints. In: Proceedings of IFAC World Congress, pp. 6863–6871 (2011)
14. Haddadin, S., Laue, T., Frese, U., Wolf, S., Albu-Schäffer, A., Hirzinger, G.: Kick it with elasticity: requirements for 2050. Robot. Auton. Syst. **57**, 761–775 (2009)
15. Haddadin, S., Özparpucu, M.C., Albu-Schäffer, A.: Optimal control for maximizing potential energy in a variable stiffness joint. In IEEE Conference on Decision and Control, pp. 1199–1206 (2012)
16. Ijspeert, J.A., Nakanishi, J., Schaal. S.: Learning rhythmic movements by demonstration using nonlinear oscillators. In IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 958–963 (2002)
17. Jetchev, N., Toussaint, M.: Trajectory prediction: learning to map situations to robot trajectories. In International Conference on Machine Learning, pp. 449–456 (2009)
18. Kavraki, L.E., Svestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. **12**(4), 566–580 (1996)
19. Lampariello, R., Nguyen-Tuong, D., Castellini, C., Hirzinger, G., Peters, J.: Trajectory planning for optimal robot catching in real-time. In IEEE International Conference on Robotics and Automation, pp. 3719–3726 (2011)
20. LaValle, S.M., Kuffner, J.J Jr.: Randomized kinodynamic planning. In IEEE Internatuinal Conference on Robotics and Automation, pp. 473–479 (1999)
21. Mettin, U., Shiriaev, A.: Ball-pitching challenge with an underactuated two-link robot arm. In: Proceedings of IFAC World Congress, pp. 1–6 (2011)

22. Özparpucu, M.C., Haddadin, S.: Optimal control for maximizing link velocity of a visco-elastic joint. In IEEE/RSJ Int. Conference on Intelligent Robots and Systems, pp. 3035–3042 (2013)
23. Park, D-H., Hoffmann, H., Pastor, P., Schaal, S.: Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. IEEE-RAS International Conference on Humanoid Robots, pp. 91–98 (2008)
24. Rao, A.V., Benson, D.A., Darby, C.L., Patternson, M.A., Francolin, C., Sanders, I., Huntington, G.T.: Algorithm 902: Gpops, a matlab software for solving multiple-phase optimal control problems using the gauss pseudospectral method. ACM Trans. Math. Softw. **22–60**(2), 39 (2010)
25. Shin, K.G., McKay, N.D.: Minimum-time control of robotics manipulators with geometric path constraints. IEEE Trans. Autom. Control **30**(6), 531–541 (1985)
26. Stark, A.: A Study of Business Decisions under Uncertainty: The Probability of the Improbable. Ph.D thesis, Boca Racon, USA Florida (2010)
27. Stulp, F., Oztop, E., Pastor, P., Beetz, M., Schaal, S.: Compact models of motor primitive variations for predictible reaching and obstacle avoidance. In IEEE-RAS International Conference on Humanoid Robots, pp. 589–595 (2009)
28. Todorov., Li, W.: A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In: Proceedings of the 2005 American Control Conference, pp. 300–306 (2005)
29. Stryk, O.V., Schlemmer, M.: Optimal control of the industrial robot manutec r3. Computational Optimal Control. International Series of Numerical Mathematics 115
30. van Ham, R., Sugar, T., Vanderborgth, B., Hollander, K., Lefeber, D.: Compliant actuator designs: Review of actuators with passive adjustable compliance/controllable stiffness for robotic applications. IEEE Robot. Autom. Mag. **16**(3), 81–94 (2009)
31. Weitschat, R., Haddadin, S., Huber, F., Albu-Schäffer, A.: Dynamic optimality in real-time: A learning framework for near-optimal robot motions. In IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5636–5643 (2013)

K-Redundant Trees for Safe and Efficient Multi-robot Recovery in Complex Environments

Golnaz Habibi, Lauren Schmidt, Mathew Jellins and James McLurkin

Abstract This paper presents a self-stabilizing distributed algorithm to recover a large number of robots safely and efficiently in a goal location. Previously, we designed a distributed algorithm, called DMLST, to recover robots [1]. Our approach constructed a maximum-leaf spanning tree for physical routing, such that interior robots remained stationary and leaf robots move. In this paper, we extend our approach to k-DMLST recovery that provides k-connectivity in the network, meaning that each robot is connected to the goal location through *k* trees. This redundancy provides stronger network connectivity by reducing the probability of losing the parent during recovery. We also propose an efficient navigation algorithm for the motion of robots which guarantees forward progress during the recovery. k-DMLST recovery has been tested and compared with other methods in simulation, and implemented on a physical multi-robot system. A basic recovery algorithm fails in all experiments, and DMLST recovery is not successful in few trials. However, k-DMLST recovery efficiently recovers more than 90 % of robots in all trials.

1 Introduction

Many practical applications of multi-robot systems, such as search-and-rescue, exploration, mapping and surveillance require robots to disperse across a large geographic area. Often overlooked the need to *recover* the robots to a goal location after

G. Habibi (✉) · L. Schmidt · J. McLurkin
Rice University, Houston, TX, USA
e-mail: gh4@rice.edu

L. Schmidt
e-mail: lss4@rice.edu

J. McLurkin
e-mail: jmclurkin@rice.edu

M. Jellins
Purdue University, West Lafayette, IN, USA
e-mail: mjellins@gmail.com

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_9

the application is complete. In our previous work, we designed a self-stabilizing distributed algorithm, called DMLST, to safely and efficiently recover large populations of robots with limited sensing in complex and unstructured environments [1].

We define *safety* as the ability to recover the robots while providing a guarantee that we do not leave any behind. We define *efficiency* as the comparison between actual and optimal execution time. The execution time is defined as the time that is required for all the robots to reach the goal location. We desire a *self-stabilizing* distributed algorithm; an algorithm that produces a desired configuration, i.e. a recovered group of robots, from any connected configuration in a bounded time.

The DMLST recovery algorithm works by building a *maximum-leaf spanning tree*, with the robot at the goal location as the root. Robots who are not leaves remain stationary, forming a stable, connected routing tree that the leaf robots use to move towards the goal location. Our assumption in DMLST recovery was that network communication failures between the stationary robots were rare. In practice, this was not the case, and a network failure can leave an entire subtree with no connection to the root robot. [2]

In this paper, we construct k-redundant trees, which provide k independent paths from each robot to the root of the trees, illustrated in Fig. 1. Should a network connection fails in the stationary routing tree, each robot will still have $k - 1$ edges to use for navigation while the connection is down. This k is a user-specified parameter, increasing k provides more reliable recovery, but at the cost of efficiency, as more robots need to remain stationary and act as routing node.

We also present an enhancement to our navigation algorithm from our previous results. As leaf robots navigate to the source, certain network configurations could

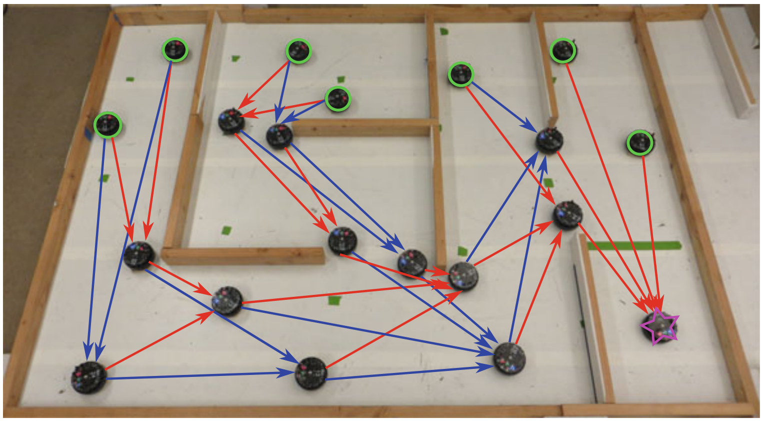


Fig. 1 Two redundant trees for 20 robots in a maze environment. *Red lines* show the first tree and *blue lines* show the second tree. *Arrows* indicate the direction from child to parent. The star highlights the root, or source, of the tree while *green circles* indicate the mobile leaf nodes. Leaf robots are mobile, others remain stationary. Using two redundant trees produces a reliable and robust recovery algorithm

create a liveness fault—the leave robots could get stuck between two parents and make no progress towards the root. Our improved algorithm takes a user-supplied parameter to determine when to move towards one parent or another, which eliminates this liveness fault while still producing efficient navigation paths and reducing physical interference from inter-robot collisions.

The paper is organized as follows: The related work are presented in Sect. 2. Section 3 presents our communication assumptions and model of robots. We propose *k* spanning trees in Sect. 4 and briefly describe DMLST algorithm. Section 5 explains k-DMLST recovery and angle-based navigation. Section 6 analyzes the features of k-DMLST recovery algorithm. We present experimental results in Sect. 7. The paper is concluded in Sect. 8.

2 Related Work

There are some studies in multi-robot system recovery. Batalin [3] used stationary nodes as navigation guides, but these stationary nodes remained stationary and were not recovered. Lavalle [4] used a simple quantized control law, a group of agents with limited sensing achieved rendezvous and gathered in a location [4]. However robots gathered anywhere in the space and there is no specific location defined as a goal location for the recovery. Moreover, this work does not use the realistic model for sensors. Li and Rus [5] developed a distributed algorithm for sensor networks to guide robots to a target, but it did not guarantee inter-robot connectivity during the recovery. Previously, we have designed DMLST algorithm to recover all robots at the goal location [1]. However, some robots lost their connection with their parents and could not arrive the goal location. This paper is an extension of DMLST algorithm by providing k-connectivity during recovery. In this method, each robot is connected to the goal location through k-redundant trees which makes the stronger connectivity during recovery. We propose a construction similar to k-connected k-dominating set (k-CDS) as a backbone to balance efficiency and fault tolerance [6]. There are several studies in k-connected k-dominating set [6, 7], but to the best of our knowledge, none of them have been applied for multi-robot systems. We are motivated by k-edge connected k dominating set to construct a fault tolerant structure. A graph is k-edge-connected if it remains connected whenever fewer than k edges are removed. To build a safe and efficient recovery algorithm, we propose a stronger construction than k-edge connected in which each node is k-edge-connected.

3 Model and Assumptions

We assume that the robots network starts in a dispersed, but connected state. There are no assumptions on the size or shape of the network or of the environment. The communication network is an undirected unit disk graph, $G = (V, E)$. Each robot is modeled as a vertex, $u \in V$, where V is the set of all robots and E is the set of all robot-to-robot communication links. The neighbors of each robot $u \in V$ are the set of robots within communication range r of robot u with a line-of-sight connection, denoted $N(u) = \{v \mid \{u, v\} \in E\}$. Each robot sits at the origin of its local coordinate system, with the $\hat{x}$-axis aligned with its current heading. Each robot can measure the angle with respect to its own $\hat{x}$-axis to each of its neighbors.

Each robot is modeled as a small disk with a pose which is defined as $u.pose = (x, y, \theta)$. The robot has a differential drive and can rotate in place and translate up to a maximum speed v_{max}. When the robots collides with the environment or other robots, obstacle sensors detect those collisions and low-level behaviors maneuver the robot away from any collision. We do not model the physical interference caused by the collisions with robots in simulations [8], hence the dimension of the robot is ignored in simulation. However, this interference affects the experimental results in Sect. 7 and is further discussed there.

Algorithm execution occurs in a series of synchronous *rounds*. A synchronizer allows us to model an asynchronous distributed system as a synchronous distributed system [9]. This synchronizer simplifies analysis and is straightforward to implement in a physical system [10]. At the end of each round, every robot u broadcasts a message, *u.message*, that is received by all of its neighbors $v \in N(u)$. Since its neighbors are doing the same, robot u will also receive a *v.message* from each neighbor $v \in N(u)$ by the end of each round. All of the pseudocode described below runs at the end of each round; after the robot has received messages from all neighbors, but before it transmits its own message. Each message contains a tuple of integers of the form: $(u.id, u.hop, u.ChildrenCount, u.SelectedList, u.Stationary)$. Each robot u has a unique id, $u.id$. A distinguished goal robot, R_G, is located at the goal location. It transmits a broadcast flood throughout the network, and the field $u.hop$ contains the number of hops robot u is from the source of the broadcast message. Using these hops, we define $N_h(u) \equiv \{v \in N(u)|v.hop = u.hop\}$, $N_{h-1}(u) \equiv \{v \in N(u)|v.hop < u.hop\}$ and $N_{h+1}(u) \equiv \{v \in N(u)|v.hop > u.hop\}$. The definitions of the other fields will be described in Sect. 4, but here we note that the size of each field is at most $log_2 n$, i.e. the number of bits required to identify a robot. This produces a total message of constant size.

4 K-DMLST and K-Redundant Spanning Trees

We extend our DMLST algorithm [1] to generate k-redundant spanning trees with keeping the maximum number of possible leaves. In our approach, we call k-DMLST, each node selects k parents from N_{h-1} with maximum number of children. If there are $|N_{h-1}| < k$ for a node, the node selects all of its N_{h-1}.

k-DMLST algorithm (Algorithm 1) calls DMLST function at most k times (line 5). DMLST returns *Selectedparent* in each iteration $t = 1, \ldots, k$, which is added to *SelectedList*, and removed from *CandidateList*. *CandidateList* is initially set to N_{h-1}. Each robot shares its *SelectedList* with its neighbors. Robots in *SelectedList*

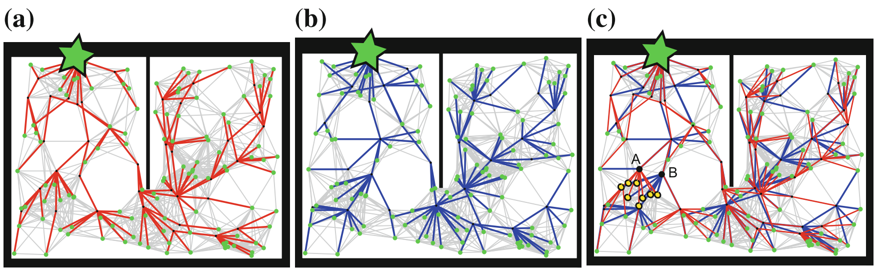


Fig. 2 Redundant trees: The network edges are in *gray* and the source is a *green star*. **a** The first tree with edges (*red*) and leaves (*green*). **b** The second tree with edges (*blue*) and leaves (*green*). **c** The two trees from a and b are combined. Robots in *yellow color* select robot A and robot B as their parents in the first tree and the second tree respectively

become *internal* nodes, and the rest become *leaves* of the trees. Our goal is to build k trees with the maximum number of *leaves* to increase the number of moving robots during recovery. This produces an approximation to a Maximum Leaf Spanning Tree. Figure 2 shows two redundant trees in the network of robots in a complex environment. Robots in yellow color selects robots A and B as their parents in the first and the second tree respectively.

Algorithm 1 $SelectedList = KDMLST(u, N(u))$

```
1: Do forever
2: $ParentCount \leftarrow 0$
3: $CandidateList \leftarrow N_{h-1}(u)$
4: while $(ParentCount < k) \wedge (CandidateList \neq \emptyset)$ do
5:     $SelectedParent \leftarrow DMLST(u,CandidateList)$
6:     add $SelectedParent$ to $SelectedList$
7:     Remove $SelectedParent$ from $CandidateList$
8:     $ParentCount{+}{+}$
9: end while
```

DMLST function is a main part of k-DMLST that we have proposed in our previous work [1]. The following section briefly explains DMLST algorithm.

4.1 Distributed Maximum Leaf Spanning Tree Algorithm

DMLST algorithm contains three main tasks which happen concurrently in the network of robots $G = (V, E)$: (1) Robot R_G broadcasts the message to build an ad-hoc network and all robots compute their *hops* value, the depth of the tree [9]. (2) Each robot $u \in V$ selects a parent *SelectedParent* from N_{h-1} in a fashion intended to maximize the number of children per parent. Having two or more $\in N_{h-1}(u)$ with the maximum number of children, the robot chooses the parent with minimum *id* with some probability p. Otherwise, the robot remains UNDECIDED and waits for the next round. This allows information from its neighbors' decisions to reach robot u and breaks the symmetry. (3) Each robot $u \in V$ reads all the messages, *v.message*, from its neighbors $v \in N(u)$ and calculates *u.ChildrenCount*, which is the number of children that have actually selected robot u in *SelectedList*, and UNDECIDED children. *u.ChildrenCount* of robot u is broadcasted to its neighbors at the end of the round.

4.2 K-Redundant Trees Properties

K-DMLST allows each robot with hop h to select k parents with hop $h-1$ if $|N_{h-1}| \geq k$. There is an exception for robots with hop $h = 1$ which select only the source as their parent. As Fig. 3 shows, K-DMLST algorithm generates at least k independent paths from each node with hop h to the source. To explain that, we draw a tree rooted from node A with hop $= h$, which is in layer $= h$ and ended in layer $= 1$. The edges in this tree indicates the links between a robot and its selected parent. Therefore, we construct a tree with branching factor of k, This tree generates $k \leq p \leq k^h$, where p is the number of independent paths from node A to the layer 1. As illustrated, there are at least k independent paths from an arbitrary node A in layer h to the source in a k-DMLST structure.

To measure the reliability of k-redundant trees, we compute the probability of network failure $F(c, k)$ as a function of k and c in Eq. 1, where k is the number of redundant trees and c is the probability of the failure of an edge between a node and its parent:

$$F(c, k) = c^k \tag{1}$$

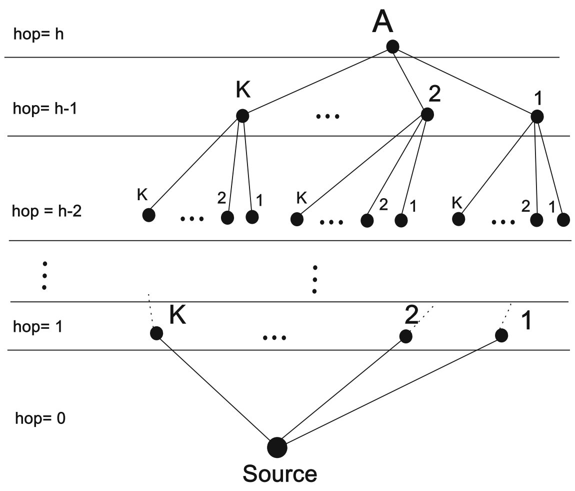


Fig. 3 k-redundant paths from robot A with hop h to the source. *Horizontal lines* show the depth of network which we call layer, Node A selects k parents with hop $h-1$ in layer $h-1$. Since each node selects k parents in a *lower layer*, a tree rooted from A is generated. There exists similar tree for each nodes in the network, except for nodes in layer 1, which select only the source as their parent

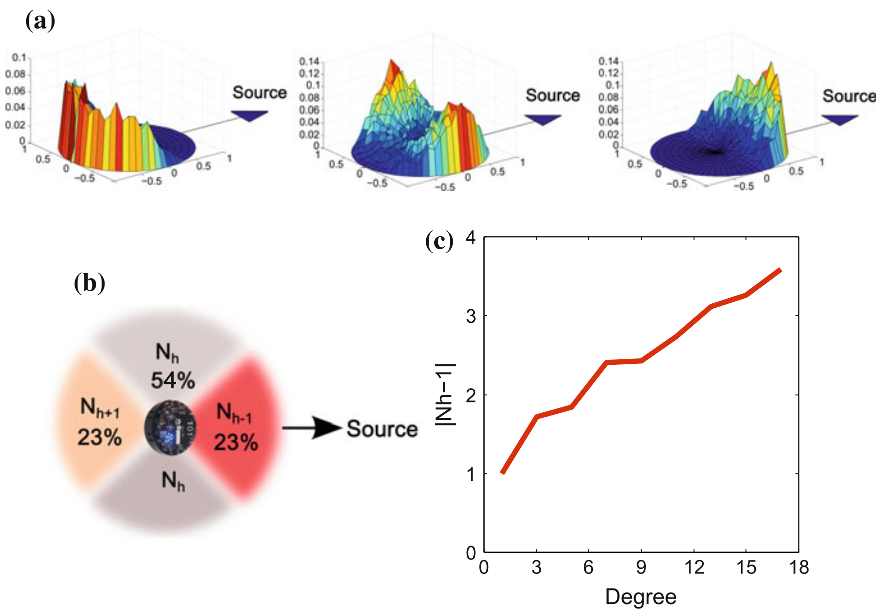


Fig. 4 **a** The average distribution of N_{h+1}, $N_h$and N_{h-1} from *left* to *right* for 12 networks of 1000 robots. The Communication range is 1 with the distribution of neighbors based on bearing to the source. **b** The distribution of all neighbors with their percentage. **c** The size of N_h against the average degree for the same networks

By using k-redundant trees, the probability of network connectivity failure decreases by increasing k. However, it is true only when each node has $|N_{h-1}| \geq k$, to be able to generate at least k-redundant trees. Otherwise, the number of distinct redundant trees is less than k and the connectivity cannot be strengthened by increasing k. Not all networks can provide this condition. Figure 4a shows that the distribution of neighbors is based on the robot's bearing to the source. As Fig. 4b, c illustrate, the average $|N_{h-1}|$ is 23 % of all neighbors. Therefore, $|N_{h-1}| \geq k$ requires a network with $\Delta \geq \frac{k}{0.23}$, where Δ is the average degree of the network. As we discuss in Sect. 6, it is possible to construct a network that supports desired k-redundant trees by multi-robot recovery.

Although k-redundant trees increase fault tolerant, The number of leaves decreases by increasing the number of redundancy k. The more parents that are selected, the less leaves produced. When k goes to infinity, only nodes on the boundary of the network can be leaves, and all others are selected as parents and become internal nodes. Figure 5a shows leaves converges to the robots in the green shaded area when k goes to infinity leaves, where r is the communication range in a unit disk graph. This estimation is only true when the network density is infinity and no voids in the network. For a circular environment with the source in the center, the number of leaves, l, converges to the number of nodes at the annular ring of the circle (Fig. 5b):

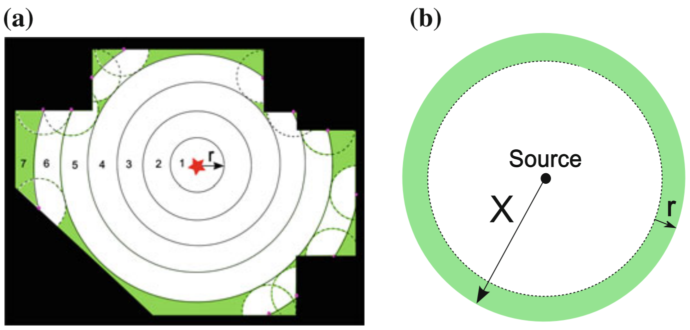


Fig. 5 **a** For K = infinity leaves are converging to the green shaded area in an illustrated environment. Internal nodes in the white area, the source is indicated by a *red star*. *r* is the communication range. The hop layers and communication areas are in *solid* and *dashed circles* respectively. Number of hops is also shown. Some nodes are shown in *purple* to show that how robots in the boundary select all their neighbors as parents. **b** Leaves tends to in a Annular ring area with radius *r* for k = infinity, X and r are the radius of environment and communication range respectively

$$\lim_{k\to\infty}\frac{l(n,k)}{n}=\frac{\pi X^2-\pi(X-r)^2}{\pi X^2}=\frac{2Xr-r^2}{X^2} \tag{2}$$

where, r is the communication range, X is the radius of environment, n is the network size. By substituting $X = cr$, where c is a real positive value, we can simplified the Eq. 2 as follows:

$$\lim_{k\to\infty}\frac{l(n,k)}{n}=\frac{2c-1}{c^2} \tag{3}$$

Figure 6a shows the percentage of number of leaves for different k-redundant trees in a circle environment with the radius of $X = 4.7r$ and $\varDelta = 100$. This figure illustrates how the percentage of leaves decreases by increasing k, The number of leaves becomes 0.36 for $k = 101$. Since the degree is not infinity, this result is a little lower than our estimation which is 0.3803. The stretch of the network also affects on the result. The stretch s is defined as the ratio of the transmitted path to the shortest path [11, 12] and $s \geq 1$ for a random graph. For $s > 1$ the depth of the network exceeds the radius of network and forces some robots to be selected as the parents of the leaves in the stretched boundary. Therefore, the number of leaves is significantly decreases. We decreased the stretch from 1.12 to 1.04 by removing the nodes with hop 6 for same network of Fig. 6a. Figure 6b illustrates the leaves are mostly in an annular ring for $k = 101$. Figure 6c shows the percentage of leaves increases by increasing the degree of the network such that it tends to 1 when the degree goes to infinity. When $|N_{h-1}| \geq k$ for each robot, a newly added robots does not need to be selected as parent and becomes a leaf. if k is limited, there exists a certain $\varDelta$ after which all added robots become leaves and the percentage of leaves is ultimately one when $\varDelta$ goes to infinity. Figure 6c also shows the rate of increasing the number of leaves reduces when k increases.

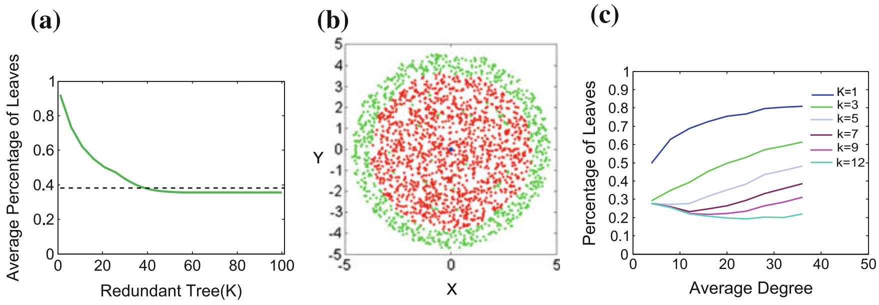


Fig. 6 **a** The average percentage of leaves versus the number of trees (*solid*), $1 \leq k \leq 101$, networks radius is $X = 4.7r$ and $\Delta = 100$ is compared with the estimated number of leaves (*dashed*) in an annular ring. **b** The same network with leaves in *green* and internal nodes in *red*, source is at the center in *blue*. **c** The average percentage of leaves for $4 \leq \Delta \leq 36$, 12 networks for each degree

5 DMLST Recovery Algorithm

As robots move towards the source and trees are updated, internal robots become leaves and move towards the goal location. In this way, the number of moving robots increases so that all the robots move toward the source and ultimately reach the goal location. The proposed recovery algorithm depends on frequent updates of the broadcast communication tree. We assume that the trees are updated faster than the robots' motions [13]. This assumption will also enforce our hardware experiments to prevent disconnection. Algorithm 2 shows how robots become stationary or start moving during execution of the recovery algorithm.

We have proposed a mid-angle navigation to decrease the collision during the recovery [1]. However, the proposed navigation did not guarantee forward progress during robots navigation, because a navigating robot may not find a new parent when it reaches the mid-angle point between two *NavigationGuides* and gets in stuck and never be recovered.

Algorithm 2 k-DMLST Recovery algorithm

```
1: Do forever
2: BroadCast messages from $R_G$ and compute hops
3: Construct trees by k-DMLST algorithm
4: for each Robot $u$ do
5:     if $u.ChildrenCount = 0$ then
6:         $u.Stationary \leftarrow 0$
7:         $AngleBasedNavigation(u)$
8:     else
9:         $u.Stationary \leftarrow 1$
10:        Robot $u$ stops moving
11:    end if
12: end for
```

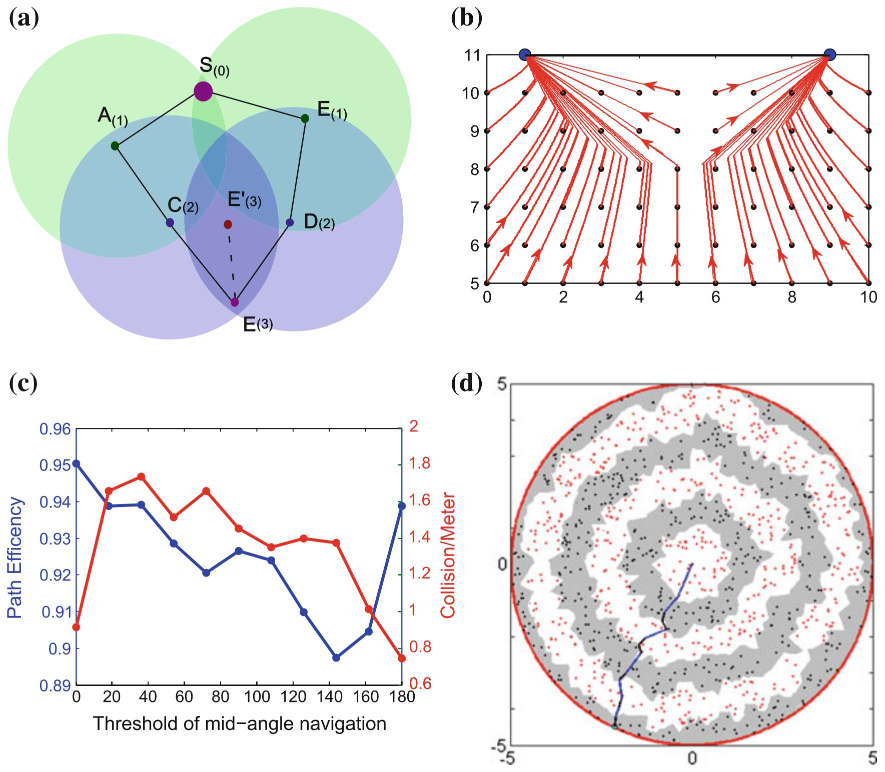


Fig. 7 **a** Mid-angle navigation fault: a navigating robot starts from position E and stops in the mid-angle point E' between its two guides C and D. *Solid lines* show the connectivity between robots in the network, navigation path is shown in *dash line*. There is no more progress afterwards in E', because none of robots A and B are visible from the navigating robot. **b** The path flow of robots (*black circles*) when they move towards two guides (*blue circles*) with angle-based navigation, threshold is $\alpha = 0.7\pi$ rad. **c** The path efficiency and the number of collisions per meter during the angle-based navigation for different values of threshold α. **d** Angle-based navigation path for a robot that starts moving from the boundary of a circle (initial position in *blue circle*, 6 hops away from the source) and moves to the source (at center of the *circle*) through 1000 robots. $\alpha = 0.7\pi$ rad, mid-angle navigation path in *black* and direct navigation is in *blue*. *Gray* and *white* regions represents the position of robots with even and odd hops respectively

Fig. 7a shows a case in which a robot in position E' stops moving because it cannot find the new parents. Moving directly towards the parent ensures forward progress, because the navigating robot ultimately finds the parent with lower hop when orbiting around its current parent. However, the number of collisions between parents and children increases. To improve the efficiency of the navigation, we propose an algorithm which combines mid-angle navigation and direct motion towards the parent. We define α as the angle threshold for switching between these two states, meaning that a navigating robot leaves *mid-angle* state to *direct* navigation when the angle between the robot and its two guides reaches the threshold α. Figure 7b shows

Algorithm 3 Angle-based Navigation

```
1: Do forever
2: if $|N_{h-1}| > 1$ then
3:     $SelectedGuide \leftarrow$ two random guides from $N_{h-1}$
4: else
5:     $S \leftarrow N_h$
6:     if $S \neq \emptyset$ then
7:         $SibGuide \leftarrow \{S_0 \in S, bearing(S_0, SelectedGuide)$ is minimum$\}$
8:         Add $SibGuide$ to $SelectedGuide$
9:     else
10:        $SelectedGuide \leftarrow N_{h-1}$
11:    end if
12: end if
13: if $SelectedGuide.size > 1$ then
14:    $GuideAngle \leftarrow$ angle between $u$ and $SelectedGuide$
15:    if $GuideAngle < \alpha$ then
16:        mid-angle navigation: Turn towards the bisector of $GuideAngle$
17:    else
18:        direct navigation: Turn towards $SelectedGuide$
19:    end if
20: else
21:    Turn towards $SelectedGuide$
22: end if
```

the navigating paths towards two *NavigationGuides* by using angle-based navigation with $\alpha = 0.7\pi$ rad.

Algorithm 3 describes angle-based navigation with forward progress guarantee. The navigation state is switched from mid-angle to direct navigation in line 13. Using angle-based navigation improves the efficiency of recovery while the number of collisions decreases comparing to direct navigation. We have to trade-off between the number of collisions and the path efficiency. Path efficiency is defined as the ratio of the shortest path to the navigated path. We define time efficiency as the ratio of optimal time, or the time to travel the shortest path, to the navigating path. A Collision decreases time efficiency by reducing the speed of the robot by $\frac{1}{10}$ at the collision. We measure the number of collisions to measure the time efficiency.

The result in Fig. 7c shows that path efficiency decreases when the threshold changes from $\alpha = 0$ rad, i.e. direct navigation to $\alpha = \pi$ rad, i.e. mid-angle navigation. However, the number of collisions, does not change by varying α. We select $\alpha = 0.7\pi$ rad to achieve a good efficiency while the number of collisions is tolerable (Fig. 7d).

6 Discussion

6.1 Correctness of DMLST Recovery Algorithm

We show the correctness of the recovery algorithm in three terms: safety, forward progress and self-stabilization.

Safety: Previously, we have shown a DMLST recovery has safety [1]. k-DMLST is an extension of DMLST has not only all properties of DMLST algorithm, but also has stronger structure in that each robot has k stationary parents. This constraint guarantees connectivity even if communication between a robot with $k - 1$ of its parents fails.

Progress: In order to guarantee forward progress of our algorithm, we need to show two things: first, our trees always has at least one moving robot; second, moving robots are navigating towards the source. Leaves are moving robots, and it follows from the definition of a tree that there is always at least one leaf. k trees have also at least one leaf which is a robot with maximum hop value. Demonstrating motion in the correct direction requires us to show that robots with fewer hops in the tree are geometrically closer to the source. The proof in Li'work [5] shows in this type of geometric, distributed BFS algorithm cannot have a local minimum in the number of hops, or basins of attraction for navigating robots. Therefore, robots in such approaches with fewer hops are always closer to the source, and always move toward the source. Our angle-based navigation always guarantee forward progress by the combination of moving to the point between two guides with same or fewer hop and moving directly to the guide with fewer hop which are closer to the source, meaning that the navigating robots always move towards the source never moves away from the goal location.

Self-Stabilization: k-DMLST includes k DMLST algorithms. In our previous work, we have shown that DMLST is self-stabilizing [1]. Therefore, k-DMLST is a self-stabilizing algorithm, meaning that starting from any arbitrary configuration, or adding any a disturbance in position, state or population, the algorithm will eliminate the effects of this perturbation after a predictable number of communication rounds [14].

6.2 Generating k-Redundant Trees by Recovery

To construct k-redundant trees, we need to build a network such that there is enough neighbors to satisfy $|N_{h-1}| \geq k$ for each robot to select k parents. We build this network by using multi-robot recovery. As leaves move towards the source, each node has more neighbors and eventually $|N_{h-1}| \geq k$ to generate k-redundant trees. Figure 8 shows how the number of selected parents converges to $k = 3$ as the recovery progresses. This figure also illustrates the deviation of selected parents is decreased to zero as the recovery proceeds so that at a certain time all robots select three parents and $k = 3$ redundant trees are produced. We implement $k = 3$ redundant trees in a real multi-robot recovery system in Sect. 7.

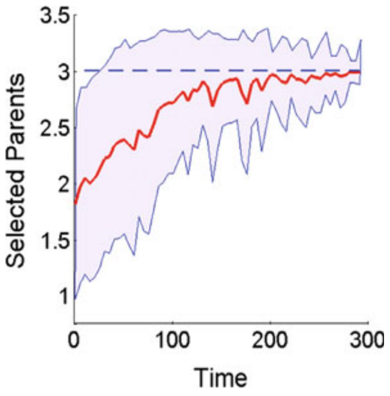


Fig. 8 The average of number of selected parents with its standard deviation during the recovery process. The number of selected parent for each robot converges to 3, as desired, during recovery. This illustration ignores counting robots that are one hop away from the source and select the source as their parent. At time t = 292 s, all robots sees the source and select R_G as their parent. Recovery is finished at time t = 360 s

7 Experiments on Robot Hardware

We have tested k-DMLST algorithm for k = 0, k = 1, and k = 2 on real robots in a maze-like environment (Fig. 9). Once the goal robot is selected as a source, the k-redundant tree is generated and leaf robots start moving to the goal location. In $k = 0$, called basic recovery, it does not mean there is not any tree. The broadcast tree is generated, but the parents do not wait for their children, i.e. all robots move toward their N_{h-1}. As illustrated in Fig. 9, the basic recovery algorithm is not able to cope with the corners in this environment, so some robots become disconnected during recovery trials. In $k = 1$ DMLST recovery, or simply DMLST recovery, only one tree is generated and each robot has at least one stationary parent [1]. DMLST is usually able to move all robots to the goal location with an occasional robot on the edge of the network being left behind. For better performance, each robot's bump sensors help it move away from environmental walls and other robots.

While some robots become stationary parents and wait for their children to move, a child may encounter an obstacle and move out of communication range to avoid physical contact. The DMLST network with $k = 2$ is more likely to maintain connectivity at these critical points in the environment by providing k parents for each robot and therefore successfully more robots are recovered. Figure 10a also compares the performance of k-DMLST recovery for $k = 0, 1, 2$ with the ideal shortest path recovery in the same experiment. We have also plotted the percentage of recovered robots versus the time in which last robot is recovered in this experiment (Fig. 10b). The result is also summarized in Table 1, which confirms that increasing k decreases the speed of recovery, *time efficiency*. Instead, the percentage of recovery, i.e. recovery accuracy, increases. In fact, we have to trade-off between time accuracy and recovery

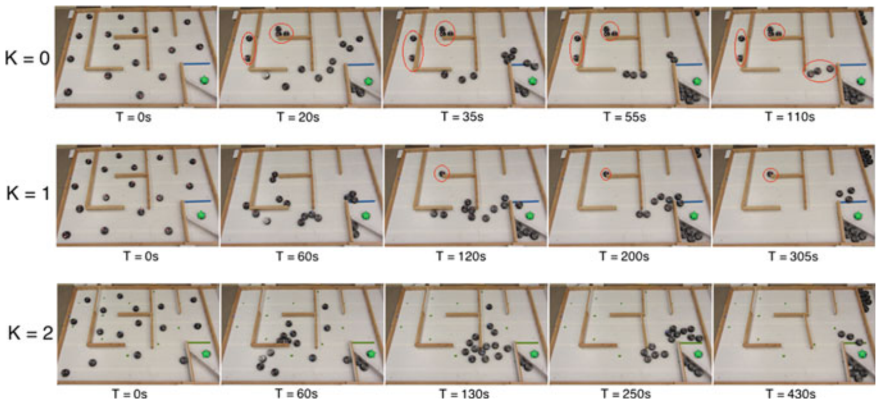


Fig. 9 Screen shots from real experiments for robot recovery. The goal location for all figures is in *green color*. The maze environment has two corners with 180°. Area dimensions: 183 cm × 233 cm, Corridors: 53 cm wide with lengths of either 61 or 102 cm for short and long segments respectively, Robot diameter: 11 cm, robot communication range: 100 cm. Each robot measures the angle with its neighbor with a limited resolution of pi/8. First row (*left* to *right*) Basic recovery; The network becomes disconnected along the left corridor and before both 180° turns. These 8 lost robots are shown in a red ellipse. Second row (*left* to *right*) DMLST Recovery; The network remains connected around all but one corner, and 16 of 17 robots are able to recover to the goal location. Third row (*left* to *right*) $k = 2$ DMLST Recovery; All robots are able to successfully navigate the environment and are recovered at the source

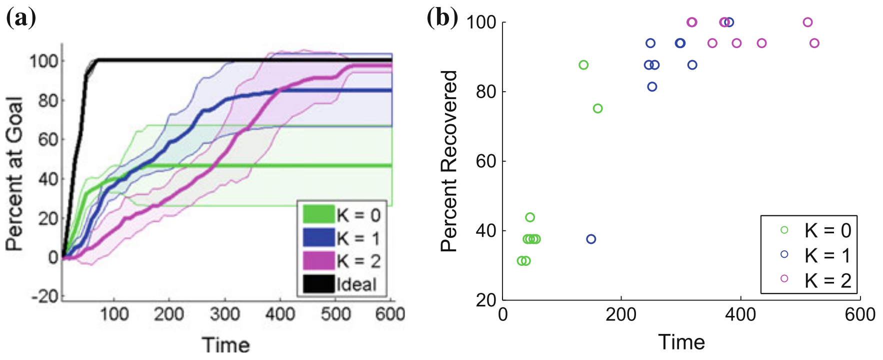


Fig. 10 Real experimental result: **a**The comparison of ideal (*black*), basic (*green*), DMLST (*blue*), and k-redundant DMLST (*magenta*) recovery performance in a maze environment for 9 trials. The percentage of robots that reach the goal location is illustrated against recovery time. Ideally, all robots recover along the shortest path to the source, as indicated by the 100 % recovery rate of the *black curve*. **b** The scatter of recovery data for the same experiment of (**a**). The percentage of robots that reach the goal location is illustrated against the time the last robot has been recovered

Table 1 Safety and time efficiency of recovery algorithms

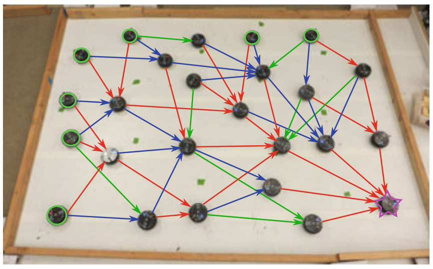
Recovery method	Recivery time (s)	Unsuccessful trials (%)	Accuracy (%)	
			Average	std
Basic	125	100	46.5	20.3
DMLST	380	77.8	84.7	18.5
2-DMLST	520	44.5	97.2	3.3

accuracy to have an acceptable safety and efficiency in a recovery. We also define *successful trial* as an experiment in which 100 % of robots are recovered.

To confirm k-redundant trees are generated during a recovery, 25 robots were placed in an open environment with desired number of selected parents $k = 3$ (Fig. 11a). The number of selected parents by each robot was collected over radio for each communication round. As Fig. 11a, initially there are not three redundant trees. As illustrated in Fig. 11b, the standard deviation approaches zero with time as the average number of selected parents converges to 3, indicating that all robots have selected k parents and k-redundant trees are generated.

Though more successful, the k-redundant DMLST recovery required more time to complete due to the reduced number of leaves. The errors encountered during recovery were largely caused by physical interference between robots, especially around corners where parents and children tend to clump. This problem was minimized by implementing a parent bump behavior that allows a child to push its parent slightly out of the way upon collision. However, the 180° corners in this environment still present a challenge to network connectivity, and are the worst-case scenario for this kind of recovery.

(a)



(b)

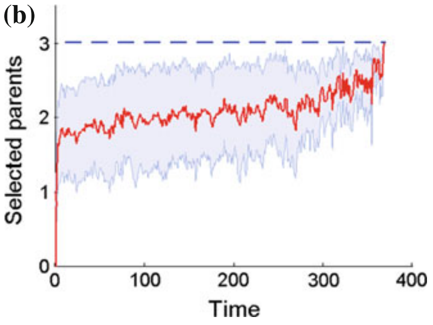


Fig. 11 **a** Approximately Three Redundant Trees for 25 robots in an open environment. *Red* edges highlight the first tree, while *blue* edges show $k = 2$ redundancy and *green* show $k = 3$ redundancy. **b** Real experimental result: Average number of selected parents versus time for 25 robots over 12 trials. Approaches $k = 3$, indicating that k-redundant trees were produced

8 Conclusion

We have proposed k-DMLST algorithm which is an extension of DMLST algorithm to generate *k*-redundant trees for strengthening the connectivity. k-DMLST algorithm achieves a higher connectivity than DMLST during the recovery. However, the number of moving robots decreases. We have also proposed an angle-based navigation algorithm to navigate robots efficiently through the created spanning trees while minimizing the number of collisions. The recovery algorithm has three properties: forward progress, safety and self-stabilizing. Extensive simulation results and hardware experiments have demonstrated the effectiveness of our approach, even in the worst-case environments, and show clear improvement over previous approaches. While basic recovery failed in all trials and DMLST recovery succeeded in most of trials, k-DMLST recovery was quite successful to recover more than 90 % of robots in all trials. 10 % of failure was due to hardware issues and physical interference. Only one case of failure happened because of rare double communication failure.

References

1. Habibi, G., Mclurkin, J., Maximum-Leaf Spanning Trees for Efficient Multi-Robot Recovery with Connectivity Guarantees, In: Proceedings of the Symposium on Distributed Autonomous Robotic Systems, pp. 1–14 (2012)
2. Muriel, M., Finn, S.G., Barry, R.A., Gallager, R.G., Fellow, L.: Redundant Trees for Preplanned Recovery in Arbitrary Vertex-Redundant or Edge-Redundant Graphs, **7**(5), pp. 641–652 (1999)
3. Batalin, M., Sukhatme, G., Hattig, M.: Mobile robot navigation using a sensor network, In: Proceedings of IEEE International Conference on Robotics and Automation, ICRA '04, pp. 636–641 Vol. 1 (2004) [Online] http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1307220
4. LaValle, S.M., Liberzon, D.: Rendezvous without coordinates. In: Proceedings of 47th IEEE Conference on Decision and Control, pp. 1803–1808 (2008). [Online] http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4739343
5. Li, Q., Rus, D.: Navigation protocols in sensor networks. ACM Trans. Sensor Netw. **1**(1), 3–35 (2005)
6. Dai, F., Wu, J.: On constructing k-connected k-dominating set in wireless ad hoc and sensor networks. J. Parallel Distrib. Comput. **66**(7), 947–958 (2006)
7. Li, Y., Yin, R., Liu, H., Hao, X.: A Reliable Connected Dominating Set Algorithm in Wireless Sensor Networks Related Work, vol. 6, pp. 2583–2592 (2012)
8. Maratic, Interference as a Tool for Designing and Evaluating Multi-Robot Controllers. In: Proceedings of AAAI-97. AAAI Press, pp. 637–642 (1997)
9. Lynch, N.A.: Distributed Algorithms. Morgan Kaufmann Publishers Inc., San Francisco (1996)
10. McLurkin, J.: Analysis and implementation of distributed algorithms for Multi-Robot systems, Ph.D. thesis, Massachusetts Institute of Technology (2008)
11. Enachescu, M., Wang, M.: Reducing Maximum Stretch in Compact Routing, no. 0339262, pp. 977–985 (2008)
12. Kleinrock, L., Silvester, J.: Optimum transmission radii for packet radio networks or why six is a magic number. In: Conference Record, National Telecommunications Conference, Birmingham, Alabama, pp. 4.3.2–4.3.5, December 1978

13. McLurkin, J.: Measuring the accuracy of distributed algorithms on Multi-Robot systems with dynamic network topologies. In: Proceedings of the International Symposium on Distributed Autonomous Robotic Systems (DARS) (2008)
14. Dolev, S.: Self-Stabilization. The MIT Press, Cambridge (2000)

Adaptive Inter-Robot Trust for Robust Multi-Robot Sensor Coverage

Alyssa Pierson and Mac Schwager

Abstract This paper proposes a new approach to both characterize inter-robot trust in multi-robot systems and adapt trust online in response to the relative performance of the robots. The approach is applied to a multi-robot coverage control scenario, in which a team of robots must spread out over an environment to provide sensing coverage. A decentralized algorithm is designed to control the positions of the robots, while simultaneously adapting their trust weightings. Robots with higher quality sensors take charge of a larger region in the environment, while robots with lower quality sensors have their regions reduced. Using a Lyapunov-type proof, it is proven that the robots converge to locally optimal positions for sensing that are as good as if the robots' sensor qualities were known beforehand. The algorithm is demonstrated in Matlab simulations.

1 Introduction

Multi-robot systems have the capacity to carry out large scale tasks efficiently. However, in order to be practical in real-world settings, multi-robot systems should be robust to the deficiencies of individual robots. In this work we consider the problem of decentralized coverage control in the case when different robots may have different, but unknown, sensing qualities. We propose an online, adaptive method to compensate for the relative differences in sensing quality using only information from the robots' sensor readings. The robots estimate a "trust weighting" online, which they use to adjust their sensing load.

The necessity for adaptive trust can be illustrated through several examples. First, consider a situation in which a group of robots is deployed over a region to take pictures following some disaster, such as an earthquake or building collapse. The

A. Pierson (✉)
Boston University, Boston, MA 02215, USA
e-mail: pierson@bu.edu

M. Schwager
Stanford University, Stanford, CA 94305, USA
e-mail: schwager@stanford.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_10

quality of sensors may degrade differently, for example, due to dust or cracks on the camera lens. Our algorithm accounts for the sensor quality and adapts trust weightings accordingly. Even in the most benign situations, sensor creep can occur causing uneven degradation in sensing performance. As sensor creep occurs in the group, adaptive trust weightings adjust for the lower-performing robots and increases the overall integrity of the group data collected.

A common strategy for coverage control, first proposed by Cortés et al. is based on Voronoi tessellations of the environment [2, 3]. This strategy drives all robots to the centroids of their Voronoi cells, also referred to as the move-to-centroid controller. It is known from previous research that the centroidal Voronoi configuration has optimal properties for minimizing distances to points [4], as well as applications in data compression [5]. Other extensions have been proposed with the weighted Voronoi cell where the weightings account for differences in agent performance and sensor qualities. Pavone et al. illustrated that using weighted Voronoi diagrams, also known as Power Diagrams, the different cell weights allow for different agents to take on varying sensing responsibility [12]. Another method considers the sensing radius as the Voronoi weighting [13], which is useful in a heterogeneous group of robots. Another application defined the weight as a measure of energy-efficiency of a robot, allowing the group to compensate for low-energy robots [9]. Marier et al. have used the Voronoi weightings to quantify sensor health of each robot, assigning low-performing robots smaller areas of coverage and higher sensing costs [10, 11].

While there is a wide variety of existing research on weighted Voronoi cells with respect to robot performance, most assume the correct weightings are known a priori. In contrast, our work proposes an algorithm to adapt trust weightings online using only comparisons between a robot's sensor measurements, and those of its neighbors. We integrate a measure of sensor discrepancy into a cost function for the group, and use this to derive an adaptation law for each robot to change its trust weightings online, while simultaneously performing a Voronoi based coverage control algorithm. We prove that the system converges to a local minimum of the cost function using a Lyapunov proof. The weightings serve as an adaptive way to assess trust between agents and improve the overall sensing quality of the group.

2 Problem Set-Up

Consider a set of n robots in a bounded, convex environment $Q \subset \Re^2$. A given point in Q is denoted q, and let the position of the ith agent be $p_i \in Q$. Prior coverage control algorithms use the standard Voronoi partition of the environment. Let $\{V_1, \dots, V_n\}$ be the Voronoi partition of Q, with each cell satisfying the Voronoi definition

$$V_i = \{q \in Q \mid \|q - p_i\| \leq \|q - p_j\|, \quad \forall j \neq i\}.$$

For our work, we use the weighted Voronoi partition, also known as the Power Diagram, with each weighting w_i serving as the trust weighting for robot i. Let

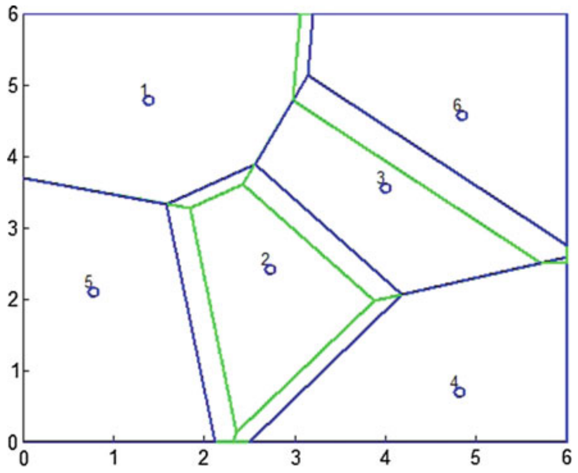


Fig. 1 The regular (*blue*) and weighted (*green*) Voronoi cell for a six-robot configuration. Here, robot 2 has a lower weighting than the other robots and robot 6 has a higher weighting, which is reflected in the changes in cell area

$\{W_1, \ldots, W_n\}$ be the weighted Voronoi partition of Q, with each cell satisfying

$$W_i = \{q \in Q \mid ||q - p_i||^2 - w_i \leq ||q - p_j||^2 - w_j, \quad \forall j \neq i\}. \tag{1}$$

The trust weighting for robot i is w_i, and it has the effect of increasing or reducing the size of its associated Voronoi cell. Figure 1 illustrates the differences between a weighted and standard Voronoi cell.

In Fig. 1, both the regular and weighted Voronoi cells are drawn to illustrate the effect of the weightings on the Voronoi cell boundaries. As shown, the trust weighting of robot two is lower than its neighbors, giving it a decreased cell area. Conversely, the trust weighting of robot six is higher, giving it the increased area.

For our bounded region Q, we also define an integrable function $\phi : Q \to \Re_{>0}$ to represent the areas of importance in the environment. Areas with large values of $\phi(q)$ are more important than those with small values, and all the robots have knowledge of this function. When the robots do not know this function, techniques have been developed to learn it online from sensor data [16]. We also introduce the sensing function $\gamma_i(p_i, q)$ to model the relationship between sensor health and data sensed by the robot. This function uses a quadratic approximation to model how the health affects a sensor reading. Unlike $\phi(q)$, $\gamma_i(p_i, q)$ may take different values by different robots looking at the same point. For example, if robot i uses a camera for sensing, $\gamma_i(p_i, q)$ may be the brightness of the pixel looking at q while the robot is located at p_i. Robot i's camera positioned at p_i may have a different value for some point q than robot j's camera positioned at p_j.

2.1 Locational Optimization

Before introducing our problem formulation, we will state the basic nomenclature and results from Locational Optimization. A complete discussion can be found in [3]. We can formulate a cost function for the sensing network over the area Q as

$$\mathscr{H}(p_1, \dots, p_n) = \sum_{i=1}^{n} \int_{V_i} \frac{1}{2} \|q - p_i\|^2 \phi(q) dq. \tag{2}$$

Note that sensing cost increases as robots move further away from high values of $\phi(q)$. Intuitively, a low value of $\mathscr{H}$ would indicate a good configuration of the robots for coverage of the environment. Two other useful quantities to define from this formulation are the mass and centroid of a Voronoi region V_i, respectively, as

$$M_{V_i} = \int_{V_i} \phi(q) dq, \text{ and } C_{V_i} = \frac{\int_{V_i} q\phi(q) dq}{M_{V_i}}.$$

Given that $\phi(q)$ is strictly positive, both M_{V_i} and C_{V_i} are analogous to physical masses and centroids of the Voronoi cells. Although there is a complex dependency between robot position and the geometry of the Voronoi cells, a surprising result from locational optimization [4] is that

$$\frac{\partial \mathscr{H}}{\partial p_i} = -\int_{V_i} (q - p_i)\phi(q) dq = -M_{V_i}(C_{V_i} - p_i). \tag{3}$$

Equation (3) implies that the critical points of $\mathscr{H}$ correspond to the configurations in which all robots are located at the centroid of their Voronoi cell, or $p_i = C_{V_i}$ for all i. Critical points can either correspond to local minimum, maximum, or saddle points. Cortés introduced a gradient-based controller that is guaranteed to drive the robots to the critical points corresponding to local minimum [3]. The controller we use here also has this property. We restrict ourselves to only considering local minima of $\mathscr{H}$ since global optimization of (2) is known to difficult (NP-hard). Thus, when we refer to optimal coverage configurations, we mean locally optimal configurations. Variations on the control law which attempt to find global minima through exploration are discussed by Salapaka et al. [14] and Schwager et al. [15].

Our formulation introduces trust weightings for each agent as an additional optimization variable. These weightings are used in calculating the weighted Voronoi cell, also known as the Power Cell, for each agent, given in (1). We still wish to formulate this as a locational optimization problem, and use a modified cost function written

$$\mathscr{W}(p_1, \dots, p_n, w_1, \dots, w_n) = \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \left(\|q - p_i\|^2 - w_i \right) \phi(q) dq, \tag{4}$$

where W_i is the robot's weighted Voronoi cell, and w_i is the robot's individual trust weighting. Note that this is almost identical to the formulation in (2), except the integral is calculated over the weighted Voronoi cell instead of the standard Voronoi cell. Additionally, we have added w_i to the integrand, giving it the same form as the weighted Voronoi cell definition (1).

Similar to the original cost function, we can also define the mass and centroid of the weighted Voronoi cell, respectively, as

$$M_{W_i} = \int_{W_i} \phi(q)dq, \text{ and } C_{W_i} = \frac{\int_{W_i} q\phi(q)dq}{M_{W_i}}.$$

From this, we can take the partial derivative of the cost function with respect to agent positions, and we find

$$\frac{\partial \mathscr{W}}{\partial p_i} = -\int_{W_i} (q - p_i)\phi(q)dq = -M_{W_i}(C_{W_i} - p_i), \tag{5}$$

which implies that critical points of $\mathscr{W}$ will also correspond to robots positioned at the centroids of their weighted Voronoi cells [10]. Using (5), we will introduce a controller similar to the Cortés controller that only moves the robots towards the local minima.

2.2 Robot and Sensor Model

In this section, we describe our model for the dynamics of the robots and the quality of the sensor. First, we assume that the robots have integrator dynamics, where

$$\begin{aligned} \dot{p}_i &= u_{i,1}, \text{ and} \\ \dot{w}_i &= u_{i,2}. \end{aligned} \tag{6}$$

Here, $u_{i,1}$ is the control input to the robot, and $u_{i,2}$ is an adaptation law for the weightings. We can equivalently assume there are low-level controllers in place to cancel existing dynamics and enforce (6). We also assume that the robots will be able to communicate with their neighbors and share information about data sensed. The communication network is defined as an undirected graph in which two robots share an edge of the graph if they share Voronoi cell boundaries. This is also known as the Delaunay graph. We can then write the set of neighbors for any robot i as $\mathscr{N}_i := \{j | V_i \cup V_j \neq 0\}$. Additionally, robots are able to compute their own weighted Voronoi cells, as defined by (1), which is a common assumption in the literature [3, 10, 14].

To model the effect of sensor health on performance, we consider a specific form for the sensor function $\gamma_i(p_i, q)$. This relates the impact of health as a quadratic approximation of the actual function near points of comparison. While in practice it is not necessary to know γ_i, for our convergence proofs we assume that γ_i can be approximated by

$$\gamma_i(p_i, q) = -\alpha \left(\|q - p_i\|^2 - h_i \right), \tag{7}$$

where h_i is some health offset indicative of sensor performance and α is some scaling factor. Note that this equation for $\gamma_i(p_i, q)$ shares a similar structure with the weighted Voronoi cell definition (1). It is not necessary for the robots to know h_i or α for a given sensor so long as $\gamma_i(p_i, q)$ can be measured from the robot's sensor. For example, imagine that $\gamma_i(p_i, q)$ conveys pixel brightness captured from a camera. While the robot may not know the camera health, it is still capable of obtaining pixel brightness. The α and h_i variables shape the approximation of how the health affects the quality at some point q from p_i. We believe this is a valid model for sensor quality, as the performance of sensing some point q decreases as q moves away from the sensor at p_i. This also allows different sensors to have a different performance. We also note that $\gamma_i(p_i, q)$ can be extended to any 2D sensor model, not just cameras. Another example to consider is an implementation where the robots have lidar sensors to map an environment topography, and $\gamma_i(p_i, q)$ is the elevation measurement at point q for the robot's sensor positioned at p_i.

3 Decentralized Control

The main goals of our work are to (1) drive the robots to an optimal coverage configuration in the environment and (2) adjust trust weightings to account for variations in sensing performance. To accomplish these goals, we propose one control law to change the positions of the robots and one adaptation law to change the weightings of the robots. We will then prove that both of these control laws will drive the robots to converge asymptotically to a stable equilibrium configuration corresponding to a local minimum of the sensing cost function.

With respect to the position controller, we will use the control law

$$\dot{p}_i = u_{i,1} = k_p(C_{W_i} - p_i), \tag{8}$$

where k_p is a positive proportional gain constant, and C_{W_i} is the centroid of the weighted Voronoi cell. This controller is commonly referred to as the move-to-centroid control law, first proposed by Cortés [3] and extended and modified in [1, 10, 16]. While the original control law used the unweighted Voronoi cell centroid, C_{V_i}, it does not impact the performance of the controller to use the weighted Voronoi centroid, C_{W_i} [10, 13].

For the weightings, we propose a new adaptation law

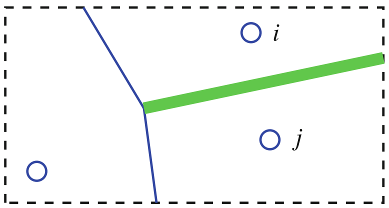


Fig. 2 For neighbors i and j the *green line* highlights their shared Voronoi cell boundary. In the weightings adaptation law, sensing data is compared along points in this boundary

$$\dot{w}_i = u_{i,2} = \frac{k_w}{M_{W_i}} \sum_{j \in \mathcal{N}_i} \left(\int_{b_{ij}} \gamma_i(p_i, q_c)dq - \frac{\int_{b_{ij}} \gamma_i(p_i, q_c)dq + \int_{b_{ij}} \gamma_j(p_j, q_c)dq}{2} \right) \tag{9}$$

where k_w is a positive proportional gain constant, and b_{ij} is the cell boundary line between neighboring agents i and j. Essentially, this compares values of sensing data between two neighbors over shared points along their boundaries. Figure 2 illustrates the shared boundary.

The control law $u_{i,1}$ is referred to as the positional controller, and the control law $u_{i,2}$ is the weightings adaptation law. The behavior of the system with these control laws is formalized in the following theorem.

Theorem 1 *Using the positional control law* (8) *and the weightings adaptation law* (9)*, the robots converge to an asymptotically stable local minimum of the sensing cost function* $\mathscr{W}(p_1, \ldots, p_n, w_1, \ldots, w_n)$ (4)*. Furthermore, the positions of the robots satisfy*

$$\|p_i - C_{W_i}\| \to 0 \quad \forall \ i \in n, \tag{10}$$

and the weightings satisfy

$$(w_i - w_j) \to (h_i - h_j) \quad \forall \ i, j. \tag{11}$$

Proof We will first show that the controllers drive the system to stable equilibria by using the cost function $\mathscr{W}$ as a Lyapunov function candidate. We will then introduce a new Lyapunov function to show that the adaptation law for the weightings, $u_{i,2}$, also drives the weightings to the set defined in (11).

Consider our cost function $\mathscr{W}$ in (4) as a Lyapunov-like function. Taking the time derivative of this function yields

$$\dot{\mathscr{W}} = \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \, \dot{p}_i + \sum_{i=1}^{n} \int_{W_i} \frac{1}{2} \phi(q) dq \, \dot{w}_i.$$

We can break this expression into two parts as

$$\dot{\mathscr{W}}_1 = \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \, \dot{p}_i, \quad \dot{\mathscr{W}}_2 = \sum_{i=1}^{n} \frac{1}{2} M_{W_i} \dot{w}_i.$$

Plugging in our adaptation law $u_{i,2}$ (9) for $\dot{w}_i$, $\dot{\mathscr{W}}_2$ simplifies as

$$\begin{aligned}\dot{\mathscr{W}}_2 &= \sum_{i=1}^{n} \frac{1}{2} M_{W_i} \frac{k_w}{M_{W_i}} \sum_{j \in \mathscr{N}_i} \left(\frac{\int_{b_{ij}} \gamma_i(p_i, q) dq - \int_{b_{ij}} \gamma_j(p_j, q) dq}{2} \right) \\ &= \sum_{i=1}^{n} \frac{k_w}{4} \sum_{j \in \mathscr{N}_i} \int_{b_{ij}} \left[\gamma_i(p_i, q) - \gamma_j(p_j, q) \right] dq \\ &= 0.\end{aligned}$$

Now consider $\dot{\mathscr{W}}_1$. By plugging in our controller $u_{i,1}$ (8) for $\dot{p}_i$, the time derivative of the cost function becomes

$$\begin{aligned}\dot{\mathscr{W}} = \dot{\mathscr{W}}_1 &= \sum_{i=1}^{n} \int_{W_i} (q - p_i)^T \phi(q) dq \left[k_p (C_{W_i} - p_i) \right] \\ &= \sum_{i=1}^{n} -k_p M_{W_i} \| C_{W_i} - p_i \|^2 \leq 0. \end{aligned} \tag{12}$$

Using La Salle's Invariance Principle [8], the robots converge to the largest invariant set such that $\dot{\mathscr{W}} = 0$. From (12), when $p_i = C_{W_i}$ for all i, then $\dot{\mathscr{W}} = 0$. From our control law (8), when $p_i = C_{W_i}$, $\dot{p}_i = 0$ for all i, therefore the centroidal Voronoi configuration $p_i = C_{W_i} \; \forall \; i$ is the largest invariant set. By La Salle's, the robots converge to the centroidal configuration, proving (10) from Theorem 1.

In order to prove (11) from Theorem 1, consider a second Lyapunov-like function,

$$\mathscr{V} = \sum_{i=1}^{n} \frac{1}{2} \| w_i - h_i \|^2$$

with time derivative

$$\begin{aligned}\dot{\mathscr{V}} &= \sum_{i=1}^{n} (w_i - h_i)^T \dot{w}_i \\ &= \sum_{i=1}^{n} (w_i - h_i)^T \frac{k_w}{2 M_{W_i}} \sum_{j \in \mathscr{N}_i} \int_{b_{ij}} \left[\gamma_i(p_i, q) - \gamma_j(p_j, q) \right] dq.\end{aligned}$$

To simplify this expression, we notice from (7) that

$$\gamma_i(p_i, q) - \gamma(p_j, q) = -\alpha \left(\| q - p_i \|^2 - h_i - \| q - p_j \|^2 + h_j \right).$$

However, we are evaluating point q at the cell boundary, so it will satisfy (1)

$$\|q - p_i\|^2 - w_i = \|q - p_j\|^2 - w_j.$$

Combining these expressions, we find

$$\gamma_i(p_i, q) - \gamma(p_j, q) = -\alpha\left(w_i - w_j - h_i + h_j\right)$$

The difference in sensing quality is constant between two neighboring robots along the boundary b_{ij}. Thus, when we plug this into our expression for $\dot{\mathscr{V}}$, we obtain

$$\begin{aligned}\dot{\mathscr{V}} &= \sum_{i=1}^{n}(w_i - h_i)^T \frac{k_w}{2M_{W_i}} \sum_{j\in\mathscr{N}_i} -\alpha\left(w_i - w_j - h_i + h_j\right)\int_{b_{ij}} dq \\ &= \sum_{i=1}^{n}(w_i - h_i)^T \frac{\alpha k_w}{2M_{W_i}} \sum_{j\in\mathscr{N}_i} \left(w_j - h_j - w_i + h_i\right) d_{ij} \qquad (13)\end{aligned}$$

where d_{ij} is the length of the boundary b_{ij}. It is advantageous to re-write this expression in matrix form. To do so, we will define

$$\tilde{w} = \begin{bmatrix} w_1 - h_1 \\ \vdots \\ w_n - h_n \end{bmatrix}, \qquad M^{-1} = \begin{bmatrix} \frac{1}{M_{W_1}} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{1}{M_{W_n}} \end{bmatrix}, \text{ and}$$

$$L = \begin{bmatrix} \ddots & & L_{ij} \\ & \sum_{j\in\mathscr{N}_i} d_{ij} & \\ L_{ij} & & \ddots \end{bmatrix}, \quad L_{ij} = \begin{cases} -d_{ij} & \text{for } j \in \mathscr{N}_i \\ 0 & \text{otherwise} \end{cases}.$$

Hence from (13) we can write the derivative of the Lyapunov function in matrix form as

$$\dot{V} = -\alpha k_w \tilde{w}^T M^{-1} L \tilde{w}.$$

M^{-1} is a diagonal matrix of positive entries and L is the weighted Laplacian of the neighbor graph, which is known to be positive semi-definite [6, 7]. It can be shown that the product $M^{-1}L$ is positive semi-definite, which allows us to state

$$\dot{\mathscr{V}} \leq 0$$

To complete the proof, we use La Salle's Invariance Principle to find the largest invariant set such that $\dot{\mathscr{V}} = 0$. The invariant set is defined as when $\tilde{w}$ is in the null space of L. From graph theory, we know this occurs when $\tilde{w}$ is a vector of identical entries, i.e. $\tilde{w}_i = \tilde{w}_j$ for all neighbors. This can also be written as the set of all w_i such that

$$w_i - h_i = w_j - h_j \quad \forall\, i, j \in n$$

or

$$(w_i - w_j) = (h_i - h_j) \quad \forall\, i, j \in n,$$

proving (11) from Theorem 1. □

Remark 1 This proof shows that using the controller $u_{i,2}$ (9), our weightings converge to a set of values relating the robot trust weightings back to sensing performance. Overall, the convergence of the weightings implies they will reach static values, which in conjunction with the move-to-centroid controller means that the robots will find final locations in the environment. Although changing the weightings creates a change in boundaries and thus a change in the cell centroids, the weightings eventually converge to an invariant set, which means the positions of the robots will eventually reach their centroids.

Remark 2 Theorem 1 guarantees convergence to a relative difference between the weightings and the health factor, not the direct health value. This is as expected, since from our problem setup, trust is a relative notion among neighboring agents with no external authority. Additionally, weighted Voronoi cell boundaries are calculated from a relative difference (1), as any constant offset would be canceled out on either side.

Remark 3 The convergence of the robots to locally optimal locations in the environment is as good as if the correct robot trust weightings were known beforehand. If the weightings are correct, it implies all robots will agree in compared sensing data values. In this case, $\dot{w}_i$ goes to zero (9), while the positional controller $\dot{p}_i$ remains the same (8).

Remark 4 One simplification of the weightings adaptation law is to compare the sensing values between neighbors at any subset of points in b_{ij}, including a single point, instead of across the entire boundary. The motivation to compare sensing functions at fewer points, as illustrated by Fig. 3, is that it may be quicker and computationally easier than the boundary calculation, albeit less robust. Corollary 1 shows that this simplification of any subset of points still maintains convergence of the weightings to an invariant set, as well as convergence of the location of the robots to their centroids.

Corollary 1 *The claims of Theorem 1 also hold true for the adaptation law*

$$\dot{w}_i = u_{i,3} = \frac{k_w}{2M_{W_i}} \sum_{j\in\mathcal{N}_i} \big(\gamma_i(p_i, q_c) - \gamma_j(p_j, q_c)\big)\,, \tag{14}$$

where q_c is any point in b_{ij}.

Proof Using (14) in place of the previous adaptation law (9) and noting that the weighted graph Laplacian becomes the normal graph Laplacian [6], the same proof and arguments hold from Theorem 1. □

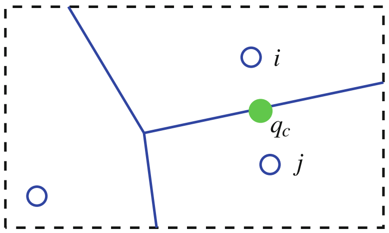


Fig. 3 For neighboring robots i and j, the weighted midpoint q_c (*green*) lies along the shared Voronoi boundary (*blue*)

4 Simulation Results

Simulations were carried out in a Matlab environment. The controllers in (8) and (14) were applied to a group of $n = 10$ robots. Riemann sums were used to approximate integrals in calculating the controllers, the weighted Voronoi cell masses and centroids, and the cost function (4). The environment Q was defined as a square region. The information density function $\phi(q)$ was defined as constant in Scenario A and B, and as a sum of two Gaussian functions, with peaks in the upper right quadrant and lower left quadrant in Scenario C.

All robots are initialized with random positions, and three scenarios are included in this paper. Scenario A starts with equal weightings but unequal sensing performance. Scenario B starts with unequal weightings but equal sensing performance. Scenario C starts with randomized weightings and health factors for all agents.

4.1 Scenario A

In this scenario, all robots start with equal trust weightings. However, robot $i = 2$ has a lower sensor health h_2, which implies it is not performing at its expected ability. The initial weighting and health values are

$$w_0 = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0],$$
$$h_0 = [1.0, 0.2, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0].$$

Over the course of the simulation we notice that the weighting w_1 decreases as a result of its lower health. Figure 4 shows a comparison between final configurations with and without the adaptive trust weightings, while Fig. 5 shows the global sensing cost and trust weightings over time.

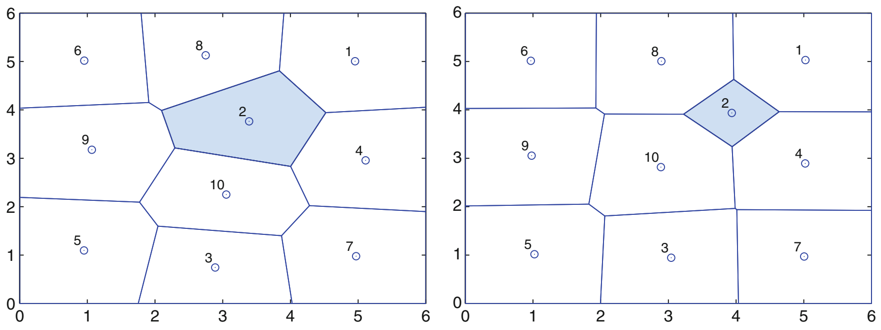


Fig. 4 Final configurations without adaptive weightings (*left*) and with adaptive weightings

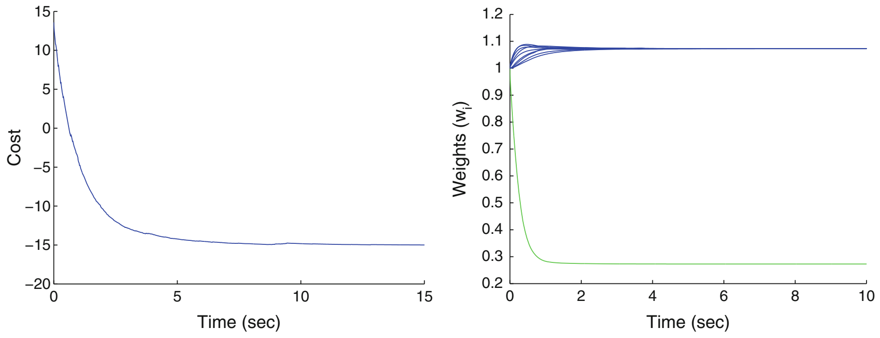


Fig. 5 *Left* Global sensing cost of the system over time. *Right* Convergence of weighting values over time. Values for w_2 are shown in *green*, while the rest of the group is shown in *blue*

We can see in Fig. 4 that without adaptive trust weightings, the lower-health robot is able to get an equal share of the environment sensing load. With the adaptive weightings, its contribution is reduced to improve the overall group quality. To verify the system converges as predicted by Theorem 1, see Fig. 5.

The values of the global cost function decrease over time, a result of the move-to-centroid controller $u_{i,1}$. In addition, by looking at the new weighting values, we see that the weight w_2 did indeed drop in value, while the other weights remained equal. The final values for the weightings taken after 100 s were

$$\begin{aligned} w_f &= [1.07, 0.27, 1.07, 1.07, 1.07, 1.07, 1.07, 1.07, 1.07, 1.07], \\ w - h &= [0.07, 0.07, 0.07, 0.07, 0.07, 0.07, 0.07, 0.07, 0.07, 0.07]. \end{aligned}$$

Consistent with our predictions, the difference between w_i and h_i is equal in value for all robots. Note that the values of w_i do not converge to the exact values of h_i, only the difference.

4.2 Scenario B

To illustrate the converse of Scenario A, we create a situation in which the robots are performing equally, but robot $i = 3$ has been initially assigned a lower trust weighting than the rest of the group. We show that the weightings will eventually converge to the same value when the sensing healths are equal. Initially, the weightings and sensing healths were assigned to be

$$w_0 = [1.0, 1.0, 0.2, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0],$$
$$h_0 = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0].$$

As we have stated previously, it is not necessary in practice to know the sensor health h_i, but we will assign these values in simulation to show functionality. With all of the sensing health values set equal, we expect that the weightings will coverage to equal values as well.

Figure 6 shows the final configurations with and without adaptive weightings. By using the adaptive weightings, the incorrectly-assigned trust weighting is corrected at robot three is given a more-equal share of the sensing load. Figure 7 shows the cost plot and the weightings over time.

From the plot of the weightings, we see that w_3 moves towards the other weights over time. The final weightings values after 100 s were

$$w_f = [0.92, 0.92, 0.92, 0.92, 0.92, 0.92, 0.92, 0.92, 0.92, 0.92],$$
$$w - h = [-0.08, -0.08, -0.08, -0.08, -0.08, -0.08, -0.08, -0.08, -0.08, -0.08].$$

As predicted, with all health values equal, the weightings converge to equal values. Note that even though the health hasn't changed, the values of the weightings decrease. From Remark 2, we know that the decrease is not important, so long as the difference between all agents is equal. It makes no difference on the Voronoi configuration if the weightings are all 0.92 or 1.0, because the relative difference is the same.

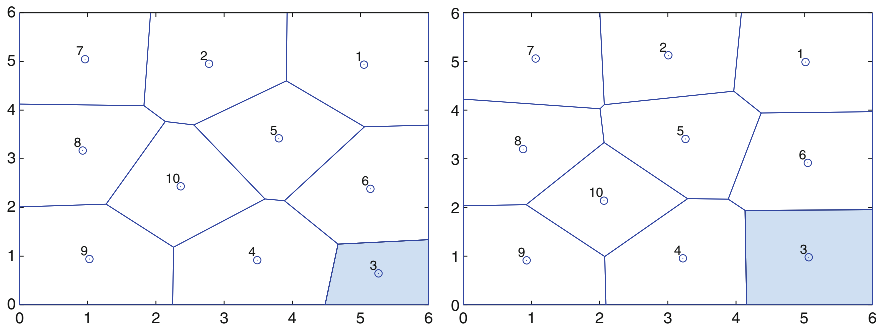


Fig. 6 Final configurations without adaptive weightings (*left*) and with adaptive weightings

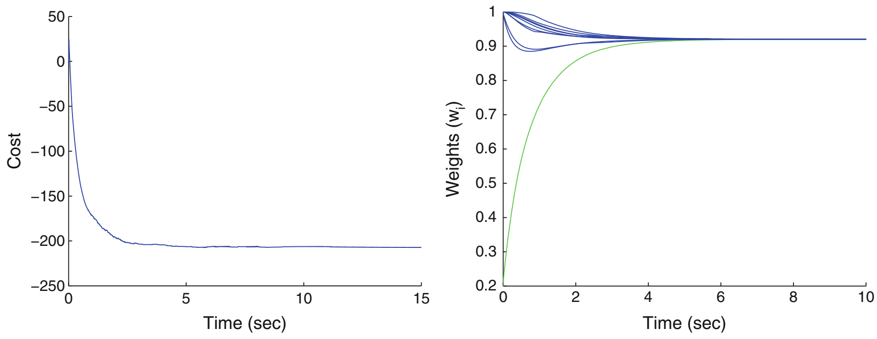


Fig. 7 *Left* Global sensing cost of the system over time. *Right* Convergence of weighting values over time. Values for w_3 are shown in *green*, while the rest of the group is shown in *blue*

4.3 Scenario C

The previous two scenarios used simple initial values to show in detail how the weightings adapt over time. In this scenario, the weightings and the sensing health factors were initialized as random numbers drawn from the uniform distribution over [0,1] to illustrate more complex functionality. The initial values were:

$$w_0 = [0.66, 0.63, 0.29, 0.43, 0.02, 0.98, 0.17, 0.11, 0.37, 0.20],$$
$$h_0 = [0.49, 0.34, 0.95, 0.92, 0.05, 0.74, 0.27, 0.42, 0.55, 0.94].$$

Similar to before, the simulation was run in Matlab, with initial and final configurations shown in Fig. 8. We expect that our weightings controller will drive the difference $w_i - h_i$ to equal values amongst the group while still maintaining coverage control. From Fig. 8, we see the algorithm is able to accomplish a centroidal Voronoi configuration from randomized initial positions.

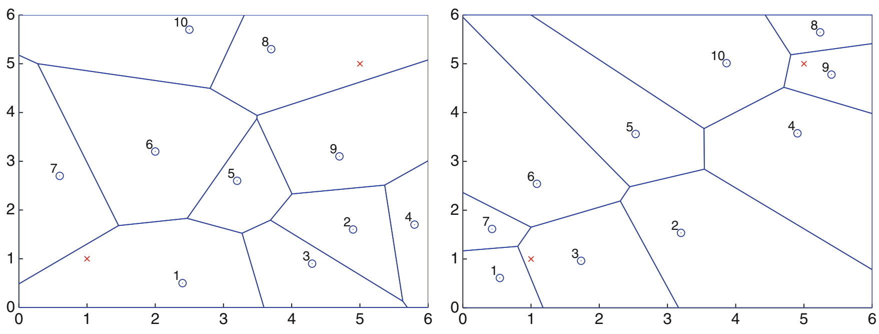


Fig. 8 Initial (*left*) and final configurations of the robots

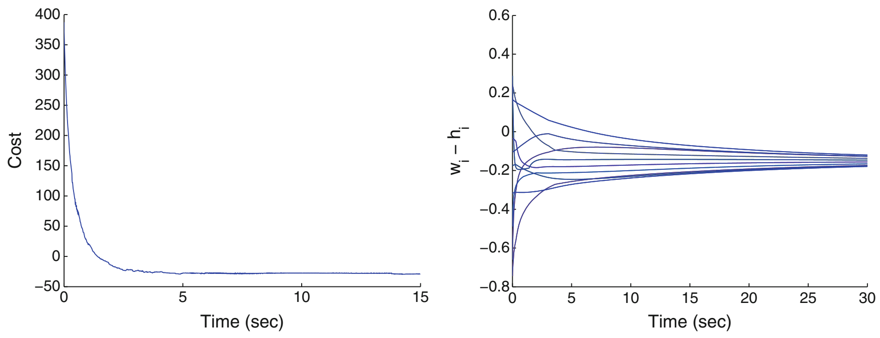


Fig. 9 *Left* Global sensing cost of the system over time. *Right* For each agent, the difference $w - h$ is plotted, showing convergence to a common value

To verify that this indeed is a better final position, we can see the cost function decreases to a final value in Fig. 9. We also observe that the final values of $w - h$, taken after 100 s, are

$$w - h = [-0.15, -0.15, -0.15, -0.15, -0.15, -0.15, -0.15, -0.15, -0.15, -0.15].$$

These values show convergence to the common invariant set described in Theorem 1. Figure 9 shows that all of these weightings relative to their respective health factor converge over time. Note that this plot is the relative difference $w - h$, which is different than what is plotted in Scenario A and B.

We plot the difference $w - h$ to better illustrate the convergence, as plotting the actual values of the weightings would seem random and erratic. From this, we see all values coming into agreement, as predicted by (11). In conjunction with the decreasing cost function, we can verify that the final position is locally optimal and stable.

5 Conclusion

In this paper we have described a method for quantifying robot-to-robot trust in a multi-robot coverage control application. Specifically, we allow the robots in the group to compare values of data sensed with their neighbors, and using an adaptive control law, adjust their weightings to better account for their performance. These trust weightings adjust the placement of the weighted Voronoi boundaries between neighboring robots. By controlling the weights on each Voronoi cell, we are able to adjust a robot's cell size relative to its neighbors, which is analogous to creating trust relationships between neighboring robots. The weightings adaptation law was proven to converge to an asymptotically stable invariant set, which is shown to be as good as knowing the health factors directly. The positional controller was similar to positional controllers in previous works in that it moved robots towards the centroid of their Voronoi cells.

Our method can be used to incorporate the robot sensor degradation into the overall decentralized algorithm while maintaining stability and performance. This will provide robustness in real-world coverage control applications. First, it can adjust for variations in the degradation of sensing performance caused by external factors, such as dust on a camera lens. Second, it provides robustness against sensor creep over long periods of time. Finally, in applications where there may be malicious robots in the group, these trust weightings can provide insight into identifying and mitigating against malicious robots. Future extensions of this work aim to further investigate the relationship of trust and adversarial robots. This paper provides the framework for quantifying trust, but additional steps are still needed to ensure that the impact caused by adversarial robots is limited. Another extension is to apply this concept of trust to applications beyond coverage control, such as multi-agent mapping, target tracking, or search.

Acknowledgments This work was supported in part by ONR grant N00014-12-1-1000, and by a Clare Boothe Luce Fellowship. We are grateful for this financial support.

References

1. Breitenmoser, A., Schwager, M., Metzger, J.C., Siegwart, R., Rus, D.: Voronoi coverage of non-convex environments with a group of networked robots. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 4982–4989. IEEE (2010)
2. Cortés, J.: Coverage optimization and spatial load balancing by robotic sensor networks. IEEE Trans. Autom. Control **55**(3), 749–754 (2010)
3. Cortes, J., Martinez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. IEEE Trans. Robot. Autom. **20**(2), 243–255 (2004)
4. Drezner, Z.: Facility Location: A Survey of Applications and Methods. Springer Series in Operations Research. Springer, New York (1995)
5. Du, Q., Faber, V., Gunzburger, M.: Centroidal Voronoi tessellations: applications and algorithms. SIAM Rev. **41**(4), 637–676 (1999)
6. Godsil, C., Royle, G.: Algebraic Graph Theory. Graduate Texts in Mathematics. Springer, New York (2001)
7. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1990)
8. Khalil, H.: Nonlinear Systems. Prentice Hall PTR, Upper Saddle River (2002)
9. Kwok, A., Martinez, S.: Energy-balancing cooperative strategies for sensor deployment. In: 46th IEEE Conference on Decision and Control, pp. 6136–6141. IEEE (2007)
10. Marier, J.S., Rabbath, C.A., Léchevin, N.: Optimizing the location of sensors subject to health degradation. In: Proceedings of the American Control Conference (ACC), pp. 3760–3765. IEEE (2011)
11. Marier, J.S., Rabbath, C.A., Léchevin, N.: Health-aware coverage control with application to a team of small UAVs. IEEE Trans. Control Sys. Technol. **21**, 1719–1730 (2012)
12. Pavone, M., Arsie, A., Frazzoli, E., Bullo, F.: Equitable partitioning policies for robotic networks. In: IEEE International Conference on Robotics and Automation, ICRA'09, pp. 2356–2361. IEEE (2009)
13. Pimenta, L., Kumar, V., Mesquita, R.C., Pereira, G.: Sensing and coverage for a network of heterogeneous robots. In: 47th IEEE Conference on Decision and Control, CDC 2008, pp. 3947–3952. IEEE (2008)

14. Salapaka, S., Khalak, A., Dahleh, M.: Constraints on locational optimization problems. In: Proceedings of the 42nd IEEE Conference on Decision and Control, vol. 2, pp. 1741–1746. IEEE (2003)
15. Schwager, M., Bullo, F., Skelly, D., Rus, D.: A ladybug exploration strategy for distributed adaptive coverage control. In: IEEE International Conference on Robotics and Automation, ICRA 2008, pp. 2346–2353. IEEE (2008)
16. Schwager, M., Rus, D., Slotine, J.J.: Decentralized, adaptive coverage control for networked robots. Int. J. Robot. Res. **28**(3), 357–375 (2009)

Part II
Design

AIST Humanoid Robotics Challenge

Kazuhito Yokoi

Abstract This paper presents the challenges assigned to humanoid robots by National Institute of Advanced Industrial Science and Technology (AIST). Since the Ministry of Economy, Trade and Industry of Japan initiated a research and development (R&D) project on humanoid robotics, AIST has developed platforms for humanoid robotics research and conducted R&D on humanoid robots applications. Particular attention has been developed to the requirements of industries using the platforms. Following TEPCO's Fukushima Daiichi Nuclear Power Station accident in 2011, DARPA started the DARPA Robotics Challenge (DRC) to promote innovation in robotic technology for disaster-response operations. In the DRC, participating robots are assigned a series of challenging tasks that indicate their suitability for disaster response. This paper presents AIST Humanoid Challenge related the DRC-selected tasks.

1 Introduction

Building machines that resemble humans is not only an interesting scientific challenge but also a practical engineering endeavor. With physical form resembling humans, humanoid robots present as potential proxies or assistants of humans performing human-oriented tasks in real world environments. Currently, humanoid robotics is extensively researched in universities, research institutes, and companies around the world. Among these, National Institute of Advanced Industrial Science and Technology (AIST) has been actively developing Humanoid Robotics for more than ten years.

Why is our research focused on humanoids? Many non-humanoid robots, such as Roomba, da Vinci, and Quince, have proven useful. However, we believe that humanoid robots possess five features that render them especially suitable for performing human tasks. These features are listed below.

K. Yokoi (✉)
National Institute of Advanced Industrial Science and Technology,
AIST Central 2, 1-1- Umezono, Tuskuba 305-8568, Japan
e-mail: kasuhito.yokoi@aist.go.jp

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts
in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_11

- The human-like shape of humanoid robots evokes emotional feelings in humans.
- Humanoid robots can work in environments designed for humans.
- Humanoids can expand their ability by using machines and equipment currently used by humans.
- Humanoid motions can be easily interpreted and predicted by humans.
- Humanoids can perform diverse tasks.

To realize these features, AIST has developed a humanoid robotics platform (HRP). The HRP series includes humanoid robot HRP-2, HRP-3, HRP-4 and software platforms OpenHRP and Choreonoid. We have undertaken several trials. The first of the above features has been demonstrated on HRP-2 and HRP-4C. Regarding the second feature, the human-like shape and functionality of humanoids is compatible with working environments that are dangerous to human workers, for example, where there is contamination risk from radiation, virus, or chemical materials. The third feature is realized in Robonaut 2 which operates alongside human astronauts in the International Space Station and HRP-1S which drives industrial vehicles similarly to human operators. The forth feature has been demonstrated in HRP-2 which has cooperated with a human expert as a novice. Regarding the fifth feature, the Humanoid Robotics Project (HRP) of Ministry of Economy, Trade and Industry (METI) of Japan, has allocated four different tasks to the same humanoid robot hardware, each implemented by different software.

Despite the above attempts, we have yet to realize a heavy-duty humanoid robot that can operate in real hazardous environments. The TEPCO's Fukushima Dai-ichi Nuclear Power Station accident in 2011 is expected to be completely decommissioned more than 40 years from now. Thus, we should begin minimizing human exposure to the radioactive environment by mid-to-long-term research and development of robotics and automation technology.

The DARPA Robotics Challenge (DRC) initiated in 2012, promotes innovative robotic technologies for disaster-response operations. DRC teams are developing robotic systems that will successfully complete a series of challenging tasks; Vehicle, Terrain, Ladder, Debris, Door, Wall, Valve, and Hose, that are relevant to disaster response.

The following section discusses the AIST Humanoid Challenge related tasks selected by the DRC.

2 AIST Humanoid Challenges

2.1 *Driving an Industrial Vehicle*

Substantial advantages are gained when humanoids operate machines generally used by human beings. Specifically, humanoids can operate the machine without requiring substantial modifications [1]. Furthermore, like human beings, humanoids can extend their operational skills to other machines. Given the strong demand for unmanned industrial vehicles that can drive in hazardous areas, we selected driving industrial vehicles as our target task [15]. Using a tele-operated humanoid robot to operate an industrial vehicle confers several advantages. When a tele-operated humanoid robot is placed in the cockpit of an industrial vehicle, the vehicle instantaneously becomes tele-operated. A tele-operated humanoid robot can not only operate the vehicle, but can also exit the vehicle and perform ancillary activities, such as checking the vehicle. In this task, we assessed whether the humanoid robot could operate typical classes of industrial vehicles such as lift trucks and backhoes.

In the first set of experiments, the tele-operated humanoid robot was employed as a proxy driver of a lift truck. The humanoid robot was set in a standing posture as shown in Fig. 1 [4].

Second, we challenged the humanoid robot to drive a backhoe, the most typical construction machine, using the following three technologies:

- the "protection technology" which protected the humanoid robot against shock and vibrations of its operating seat and against the influences of the natural environmental influences such as rain and dust;
- "full-body operation control technology" which controlled the humanoid robot's total body movements while walking. The robot was installed with autonomous control capability to prevent it from toppling; and
- the "remote control technology" which instructed the humanoid robot to perform total body movements under remote control. The remote control tasks were executed by the remote control system.

When tele-operating a humanoid robot in various situations, it is often desirable to change the tele-operation method to one that ensures easy operation. Therefore, we introduced three operational modes, namely, riding, sitting down and manipulating levers, which are selected by the robot depending on its task.

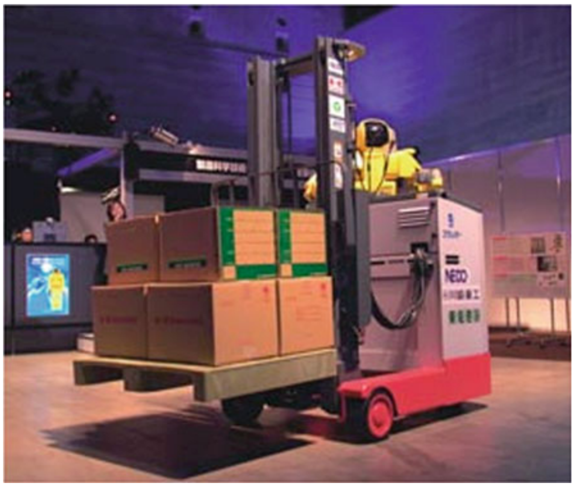


Fig. 1 Tele-operated humanoid robot driving a lift truck

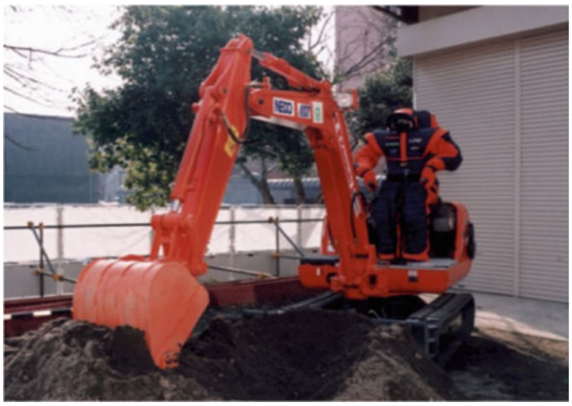


Fig. 2 Tele-operated HRP-1S driving the backhoe

1. The playback mode: The robot is governed by a reference motion pattern generated by the program off-line. This mode is adopted for seating. Note that the seating motion pattern is easily reproduced by setting the height of the seat.
2. The manual mode: The robot is manually tele-operated by the remote control device. Both the master arm device and the master foot device can be used. The arm and leg control is applied after seating and head control.
3. The semi-automatic mode: The operator directs a partial command of a whole body motion, and the robot automatically calculates the rest of the motion. This mode reduces the responsibility of the operator in the manual operation and is applied during walking toward the backhoe.

Remote operation tasks, comprising a sequence of riding, driving, and excavating, were successfully executed by the tele-operated humanoid robot (Fig. 2).

2.2 Travel Across Uneven Terrain

When the humanoid walks across uneven terrain, its swinging foot may unexpectedly strike the terrain. Such unexpected contact occurs in three ways. During a preplanned landing, when a swinging foot contacts the terrain early in the single support phase, it experiences a high horizontal impact force that may topple the robot halfway through the walking. Therefore the swinging leg must be able to absorb an unexpected contact force. Moreover, the foot contact becomes a physical constraint. During such a collision, the robot must also re-plan its foot trajectory. The other unexpected contacts of the swinging foot occur immediately preceding and following the preplanned landing time. In these cases, the instant a contact is detected, the swing motion is halted by a pattern generator and the walking phase immediately switches from single support phase to double support phase.

To prevent contact-induced tripping and balance loss, we counteract horizontal collisions of the swinging foot during walking. Robust bipedal walking across uneven terrain was realized by collaboration between the reactive pattern generation of the center-of-gravity/zero-moment-point (COG-ZMP) controller and the swinging motion of the robot. Two primary procedures of this system are (1) Set the impedance gains of the feet to appropriate values for the walking phase, (2) Update the desired landing position and immediately convert it to a COG pattern which stores the action as a detecting/releasing contact [9]. Balance control is improved by Capture Point (CP) control, whose controller essentially behaves as a conventional balance controller. We analyze the transfer function of the balance controller and introduce a new state variable which integrates the CP and the ZMP to truncate the long-term offset of both controls [10].

The validity of the proposed balance controller was verified through experiments involving the humanoid robot HRP-2. Snapshots of the walking experiment at 1 s intervals are shown in Fig. 3. The robot must walk between square blocks (area = $(8 \times 8)\,\mathrm{cm}^2$; height = 1 cm) and hard rubber cylinders (diameter = 10 cm; height = 1 cm) are randomly placed on the floor. In this experiment, the preplanned step length was 0.2 m and the walking cycle was 0.8 s. All feedback loops were run

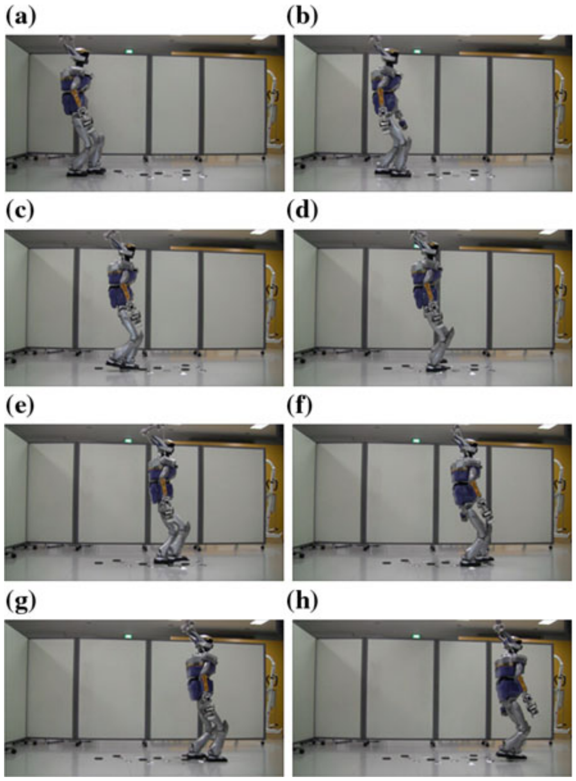


Fig. 3 Photographic sequence of HRP-2 walking on uneven terrain. **a** t = 4.0 s, **b** t = 5.0 s, **c** t = 6.0 s, **d** t = 7.0 s, **e** t = 8.0 s, **f** t = 9.0 s, **g** t = 10.0 s, **h** t = 11.0 s

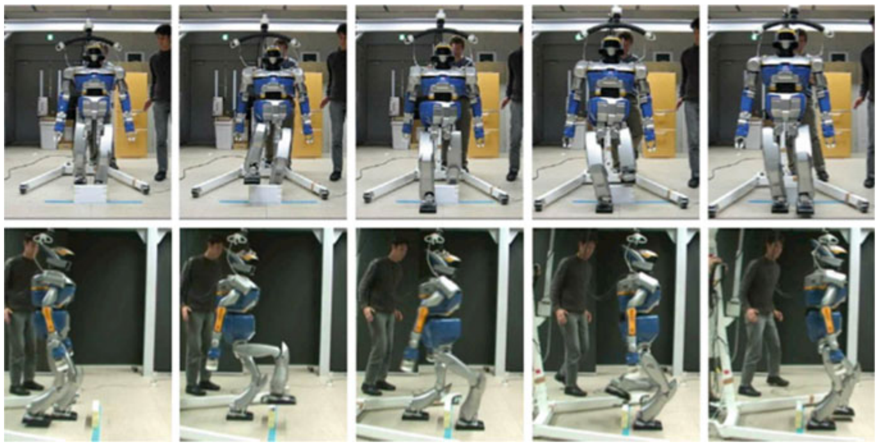


Fig. 4 Photographic sequence of HRP-2 stepping over an obstacle 15 cm high and 5 cm wide (increased to 18 cm height and 11 cm wide when safety boundaries are imposed). The images are captured at 0.64 s intervals [14]

at 1 ms intervals. The support phase was synchronized with the reaction force, and transited from a single to a double support phase during contact.

Unlike wheeled or crawling robots, legged robots can walk over objects. This ability is conferred in HRP-2 by implementing dynamic motions rather than a quasi-static approach. In this implementation, the robot can negotiate obstacles without needing to cease its motion, and can more rapidly negotiate higher obstacles. The method, based on the work of Kajita et al. [6], determines the required CoM height trajectory for stepping over large obstacles. It also implements a dynamic pattern generator for stepping over large obstacles [14].

In Fig. 4, the robot is photographed stepping over an object 15 cm high and 5 cm wide (with height and area safety boundaries of 3 cm and $(2 \times 3)\,\mathrm{cm}^2$ respectively).

2.3 *Removing an Obstacle in the Path*

If the path of the humanoid robot is blocked by an insurmountable, the robot should remove the object. Small lightweight objects are effectively removed by carrying. Pickup and replacement tasks are commonly performed by humans, generally without difficulty. However, if the humanoid robots are to successfully perform the same tasks, the hand reaction forces require appropriate consideration. We attached force sensors to the wrists and ankles of the humanoid robot, enabling the robot to stably carry an object without knowing the mass or COG position of the object. To prevent the humanoid robot from falling while carrying an object, we modified the position of the waist depending on the magnitude of the force applied to the hands.

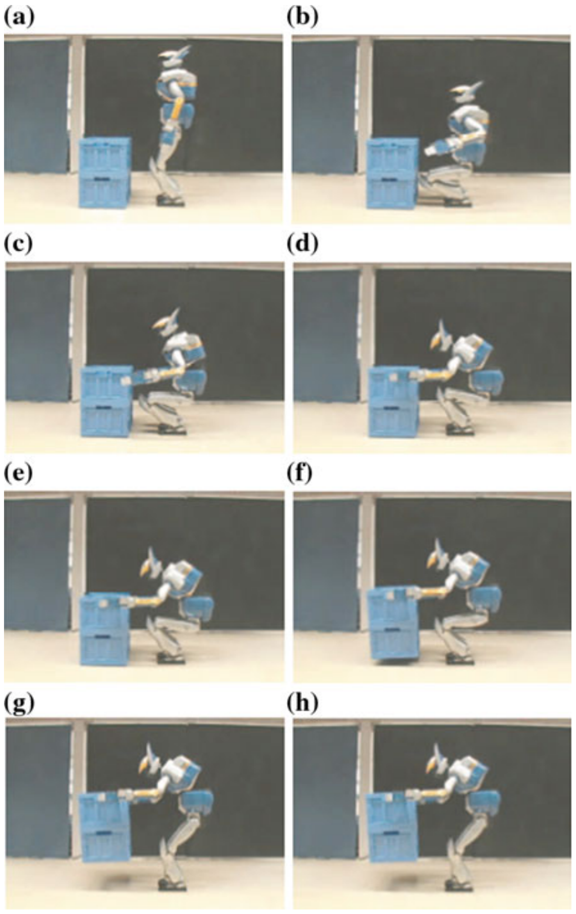


Fig. 5 Photographic sequence of HRP-2 carrying an 8.5 kg object [2]. **a** t = 0.0 s, **b** t = 3.0 s, **c** t = 6.0 s, **d** t = 9.0 s, **e** t = 12.0 s, **f** t = 15.0 s, **g** t = 18.0 s, **h** t = 21.0 s

The position of the COG of the object within the horizontal plane is now identifiable from the force sensor information. From the identified physical parameters of the object, a walking pattern is generated [2]. A series of snap-shots of this experiment is shown in Fig. 5. HRP-2 first squats down (Fig. 5a and b), captures an object (Fig. 5c, d and e), and raises it (Fig. 5f, g and h). The weight of the carried object was 8.5 kg.

Large heavy objects are effectively removed by pushing. To achieve stable pushing by a humanoid robot, regardless of the objects mass, the arms are controlled by the impedance control and foot placement is planned in real-time according to the result of object manipulation [3]. The heavier the object, the slower the humanoid robots walking pace when pushing the object. This method was realized by separating the pushing phase from the stepping phase. In the pushing phase, the force applied to the tip of the robots arms is controlled without stepping. The pushing effort determines the length of each step. Snapshots of HRP-2 pushing a 10 kg table are shown in Fig. 6. Although the motion of the table was disturbed between 12 and 20 s, HRP-2

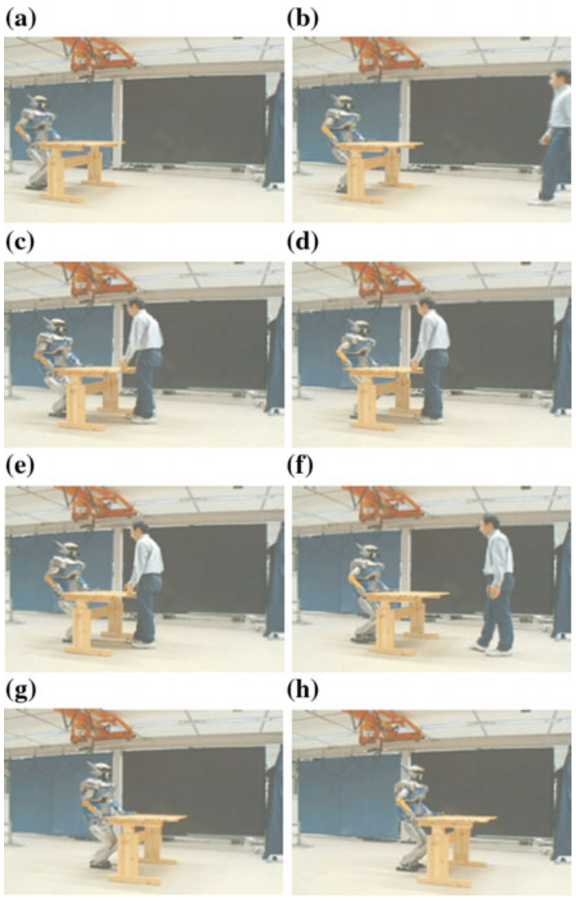


Fig. 6 Photographic sequence of HRP-2 pushing a 10 kg table [3]. **a** t = 0.0 s, **b** t = 10.0 s, **c** t = 12.0 s, **d** t = 14.0 s, **e** t = 18.0 s, **f** t = 22.0 s, **g** t = 32.0 s, **h** t = 42.0 s

maintained balance by adaptively changing its gait pattern depending on the amount of pushing required.

2.4 Opening a Door and Walking Out

A door presents a strong barrier to mobile and flying robots directed to enter a room. When the humanoid robot encounters a door, the following conditions should be satisfied.

- One hand of the humanoid robot can reach the door knob;
- the humanoid robot can overcome the reaction force exerted by the door;
- no part of the humanoid robot except the arm should contact the door; and
- the humanoid robot avoids the wall surrounding the door.

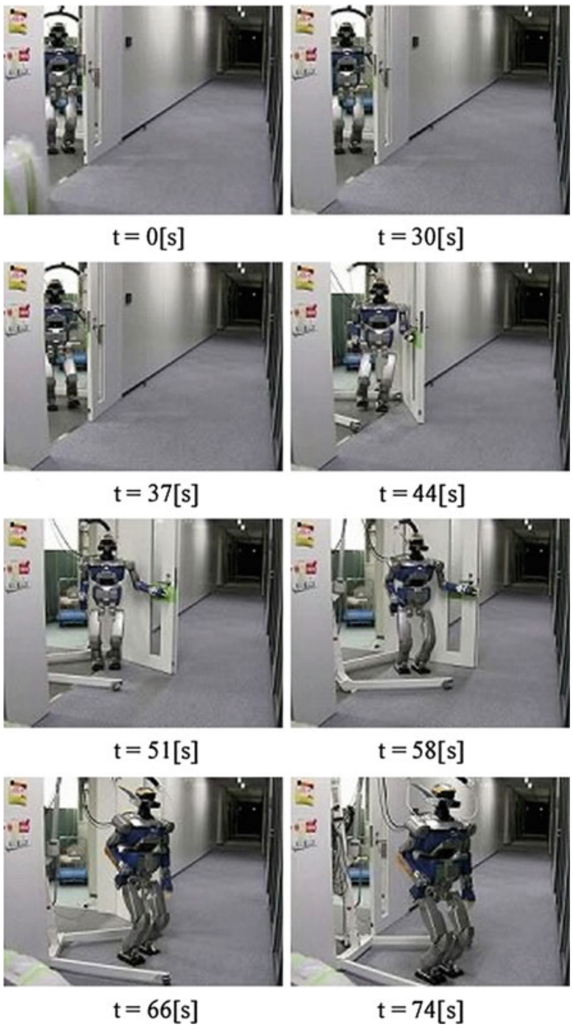


Fig. 7 Photographic sequence of HRP-2 opening a door and exiting a room [12]

The above conditions were satisfied by a planning method that calculates the positions and orientations of the humanoid robot enabling the robot to select a suitable set of these parameters [11]. Figure 7 displays snapshots of HRP-2 opening a door and exiting the room.

2.5 *Using a Power Tool to Fasten a Bolt*

Humans manipulate conventional tools by hand mechanisms. Since the hand of HRP-2 is a one degrees-of-freedom (DOF) gripper, HRP-2 cannot use a power tool with a

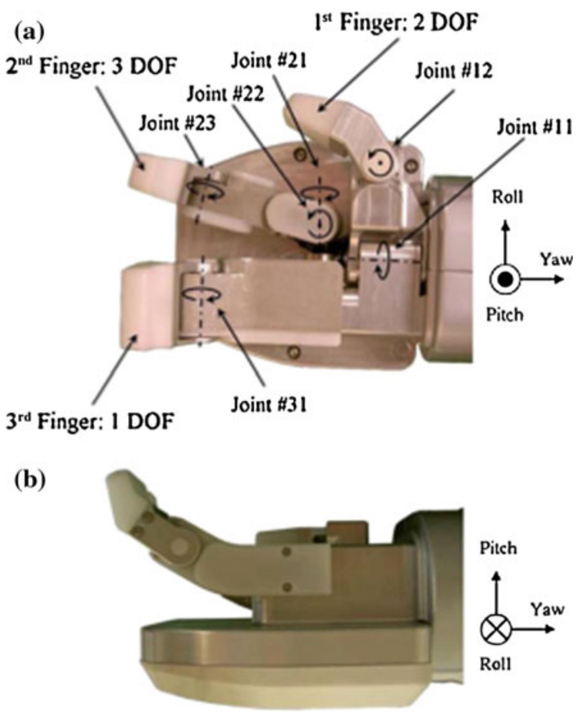


Fig. 8 HRP-3 three-fingered hand with 6 DOF [7]. **a** Pitch axis view. **b** Roll axis view

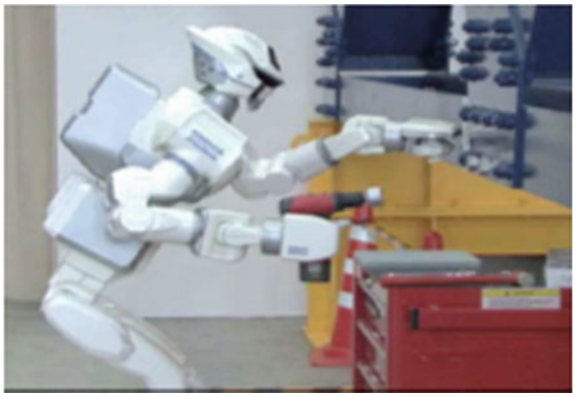


Fig. 9 Demonstration of the use of a power tool [7]

push type switch or a trigger. The hand of HRP-3 was designed to both grasp objects and use an electrical driver equipped with a trigger. Figure 8 shows the developed hand of HRP-3, which has three fingers and a total of 6 DOF.

Figure 9 demonstrates the humanoid robot engaged in bridge construction work. In this figure, HRP-3 leans against the bridge using the left hand and fastens a bolt with an electric drill grasping in the right hand. It is assumed that HRP-3 cannot move closer to the bridge. This motion was generated by applying the whole body operation method to the humanoid robot, which also enables the robot to overcome environmental obstacles [5]. From this demonstration, we confirmed that HRP-3 can potentially manipulate power tools.

2.6 Reaching a Valve in a Complex Environment

Since a humanoid robot has many DOF, complete tele-operation is difficult using ordinary input devices, especially in highly constrained environments. To tackle this problem, we developed a reaching-motion planning and execution framework tailored for exploration missions by human-operated humanoid robots in hazardous environments such as nuclear plants. This framework offers low-level but practical autonomy, allowing the robot to plan and execute simple tasks, such as reaching a target object, within a reasonable timeframe. The efficiency of this framework benefits the human operator who can then maneuver the robot without waiting several minutes for the planning results. The efficiency of this action is improved by two procedures. First, a reaching motion is rapidly planned by solving inverse kinematics to approximate the mass distribution and kinematic structure. If the robot is working in environments not completely known, the proposed planner can access measured voxel maps. The second procedure executes the planned path while compensating the approximation error in real time without violating other constraints.

Simulations confirmed that the HRP-2 humanoid in an environment constrained with pipes takes approximately one second to plan its reaching motion. Figure 10 presents snapshots of the executed reaching motion. The right arm passes through the narrow passage between a pipe and its lower body and reaches the target position. The planned path consists of nine configurations. Snapshots in Fig. 11 show a reaching

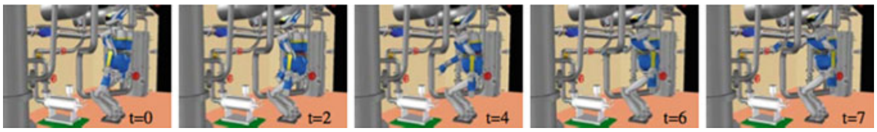


Fig. 10 Reaching motion executed by the *right arm* [8]

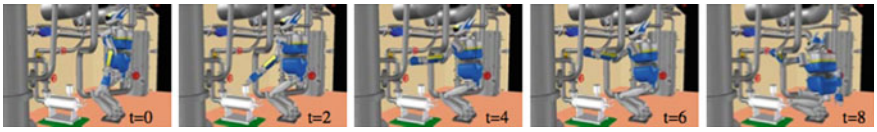
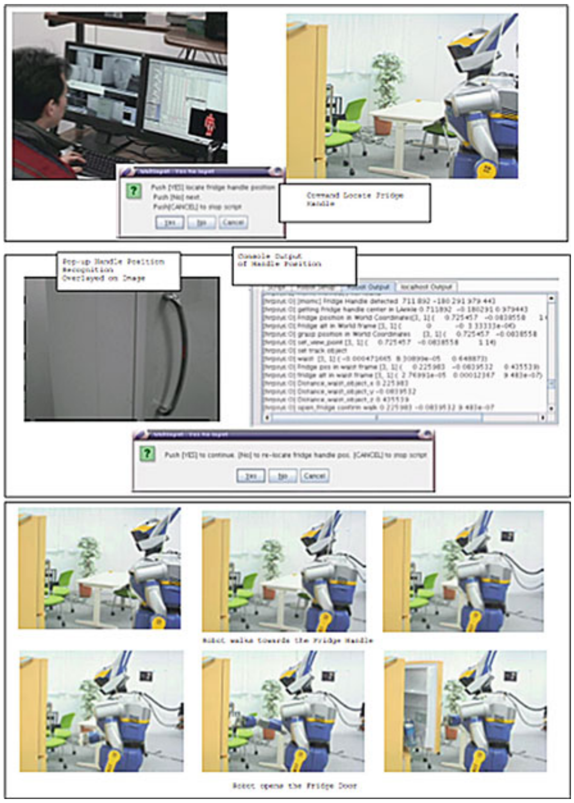


Fig. 11 Reaching motion executed by the *left arm*. The robot lowers its body to avoid its *left shoulder* colliding with a pipe [8]

motion by the left arm. As evident in the figure, the robot lowers its body to avoid collisions between its left shoulder and a pipe. The planned path consists of 14 configurations.

3 Behavior-Level Operation System for Humanoid Robots

Among the AIST Humanoid Challenges discussed above, "Driving an industrial vehicle" and "Using a power tool to fasten a bolt" were performed by a tele-operated humanoid robot. The remaining tasks were autonomously achieved by the humanoid robot. Although tele-operation systems enable the operator to continuously guide the robots motion, the benefits of such high flexibility may be offset by excessive time consumption. The operator will become overtired if he/she must constantly guide the robot even through routine, commonly-performed motions. The most efficient system would provide the operator with a discrete behavior-level control option. To realize this option, we have constructed perceptual and motional behaviors; locating


Command Locate Fridge Handle
Pop-up Handle Position Recognition Overlayed on Image
Console Output of Handle Position
Robot walks towards the Fridge Handle
Robot opens the Fridge Door
Yes
No
Cancel

```
Push [YES] locate fridge handle position
Push [No] next.
Push[CANCEL] to stop script
```

```
[hrpinput 0] [momc] Fridge Handle detected 711.892 -180.291 979.443
[hrpinput 0] getting fridge handle center in Lwaist 0.711892 -0.180291 0.979443
[hrpinput 0] fridge position in World Coordinates[3, 1]( 0.725457 -0.0838558 1
[hrpinput 0] fridge att in World frame [3, 1]( 0 -0 3.33333e-04)
[hrpinput 0] grasp position in World Coordinates [3, 1]( 0.725457 -0.0838558
[hrpinput 0] set_view_point [3, 1]( 0.725457 -0.0838558 1.14)
[hrpinput 0] set track object
[hrpinput 0] waist [3, 1]( -0.000471665 8.30899e-05 0.648873)
[hrpinput 0] fridge pos in waist frame [3, 1]( 0.225983 -0.0839532 0.435539)
[hrpinput 0] fridge att in waist frame [3, 1]( 2.76991e-05 0.00012367 9.483e-07)
[hrpinput 0] Distance_waist_object_x 0.225983
[hrpinput 0] Distance_waist_object_y -0.0839532
[hrpinput 0] Distance_waist_object_z 0.435539
[hrpinput 0] open_fridge confirm walk 0.225983 -0.0839532 9.481e-07
```

```
Push [YES] to continue. [No] to re-locate fridge handle pos. [CANCEL] to stop script
```

Fig. 12 Cooperation between operator and humanoid robot in locating and confirming a fridge handle position [13]

objects, deciding whether a volume is reachable, reaching for objects, and opening and closing doors. These behaviors facilitate online operations of humanoid robots.

The proposed behavioral-level operation was assessed on a simple task; removing a can from a refrigerator. This operation is depicted in Figs. 12 and 13. Throughout the task, the operator and the robot cooperated in the following sequence:

- The operator sent a “locate fridge handle” command by clicking the command dialogue box.
- The robot located the fridge handle in its view.
- The recognition result appeared in a pop-up image, and the walking distance required to approach to the handle appeared in the console output.
- The operator confirmed the recognition result was correct, and triggered walking by clicking on the command dialogue.
- The robot generated walking motion towards the fridge handle.
- The operator sent an “open fridge” command by clicking the command dialogue box.
- The robot generated an opening fridge motion by reaching toward the handle and opening the fridge door.
- The operator sent a “take can” command by clicking the command dialogue box.
- The robot generated a “take can” motion by reaching toward the can and grasping it.

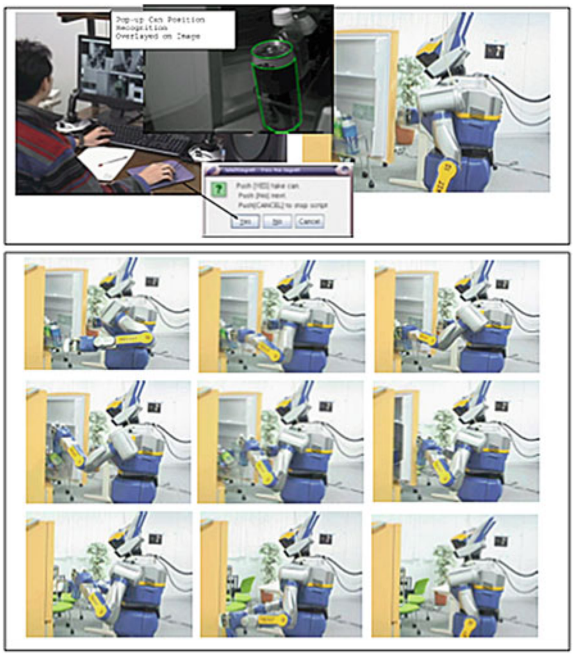


Fig. 13 Cooperation between operator and humanoid robot in taking a can from the fridge with the *left hand* [13]

- The operator sent a “pickup can” command by clicking the command dialogue box.
- The robot generated the motions required to pick up the can and close the fridge door.

4 Conclusion

As the DRC Finals approach, the competence of the humanoid robots is expected to approximate that of a two-years-old child. The robots should autonomously obey simple commands such as “Clear the debris in front of you” or “Close the valve”. More complex tasks sequenced from chains of simpler tasks will require guidance from human operators. Nonetheless, disaster response robots show great promise in mitigating the effects of future disasters.

Currently, the perceptional and motion behaviors implemented in our behavior-level operation system are insufficient for humanoid disaster response robots. Developing a sufficiently sophisticated system is a lofty but not unattainable goal. We believe that robotics technology will support decommission of the TEPCO’s Fukushima Dai-ichi Nuclear Power Station.

Acknowledgments The author would like to thank sincerely all the past and current members of the Humanoid Research Group, Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology and CNRS-AIST UMI3218/CRT JRL.

References

1. Ambrose, R.O., Aldridge, H., Askew, R.S., Burridge, R., Bluethman, W., Diftler, M.A., Lovchik, C., Magruder, D., Rehnmark, F.: ROBONAUT: NASA’s space humanoid. IEEE Intell. Sys. J. **15**, 57–63 (2000)
2. Harada, K., Kajita, S., Saito, H., Morisawa, M., Kanehiro, F., Fujiwara, K., Kaneko, K., Hirukawa, H.: A humanoid robot carrying a heavy object. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1724–1729 (2005)
3. Harada, K., Kajita, S., Kanehiro, F., Fujiwara, K., Kaneko, K., Yokoi, K., Hirukawa, H.: Real-time planning of humanoid robot’s gait for force controlled manipulation. IEEE/ASME Trans. Mechatron. **12**(1), 53–62 (2007)
4. Hasunuma, H., Kobayashi, M., Moriyama, H., Itoko, T., Yanagihara, Y., Ueno, T., Ohya, K., Yokoi, K.: A tele-operated humanoid robot drives a lift truck. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2246–2252 (2002)
5. Hasunuma, H., Harada, K., Hirukawa, H.: The tele-operation of the humanoid robot-whole body operation for humanoid robots in contact with environment. In: Proceedings of the IEEE-RAS International Conference on Humanoid Robots, pp. 333–339 (2006)
6. Kajita, S., Kanehiro, K., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., Hirukawa, H.: Biped walking pattern generation by using preview control of zero-moment point. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1620–1626 (2003)

7. Kaneko, K., Harada, K., Kanehiro, F., Miyamori, G., Akachi, K.: Humanoid robot HRP-3. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2471–2478 (2008)
8. Kanehiro, F., Yoshida, E., Yokoi, K.: Efficient reaching motion planning and execution for exploration by humanoid robots. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (2012)
9. Morisawa, M., Kanehiro, F., Kaneko, K., Kajita, S., Yokoi, K.: Reactive biped walking control for a collision of a swinging foot on uneven terrain. In: Proceedings of the IEEE-RAS International Conference on Humanoids, pp. 768–773 (2011)
10. Morisawa, M., Kajita, S., Kanehiro, F., Kaneko, K., Miura, K., Yokoi, K.: Balance control based on capture point error compensation for biped walking on uneven terrain. In: Proceedings of the IEEE-RAS International Conference on Humanoids, pp. 734–740 (2012)
11. Nakamura, T., Arisumi, H., Yokoi, K.: Passing through a door by a humanoid robot. In: Proceedings of the SICE System Integration Division Annual Conference, pp. 452–455 (2009)
12. Nakamura, T., Arisumi, H., Yokoi, K.: Passing through a door by a humanoid robot—Experimental verification. In: Proceedings of the JSME Annual Conference (2009)
13. Neo, E.S., Sakaguchi, T., Yokoi, K., Kawai, Y., Maruyama, K.: A behavior level operation system for humanoid robots. In: Proceedings of the IEEE-RAS International Conference on Humanoid Robots, pp. 327–332 (2006)
14. Stasse, O., Verrelst, B., Vanderborght, B., Yokoi, K.: Strategies for humanoid robots to dynamically walk over large obstacles. IEEE Trans. Robot. **25**(4), 960–967 (2009)
15. Yokoi, K., Kobayashi, M., Mihune, H., Hasunuma, H., Yanagihara, Y., Ueno, T., Gokyu, T., Endou, K.: A tele-operated humanoid operator. Int. J. Robot. Res. **25**(5–6), 593–602 (2006)

A Scripted Printable Quadrotor: Rapid Design and Fabrication of a Folded MAV

Ankur M. Mehta, Daniela Rus, Kartik Mohta, Yash Mulgaonkar, Matthew Piccoli and Vijay Kumar

Abstract Robotic systems hold great promise to assist with household, educational, and research tasks, but the difficulties of designing and building such robots often are an inhibitive barrier preventing their development. This paper presents a framework in which simple robots can be easily designed and then rapidly fabricated and tested, paving the way for greater proliferation of robot designs. The Python package presented in this work allows for the scripted generation of mechanical elements, using the principles of hierarchical structure and modular reuse to simplify the design process. These structures are then manufactured using an origami-inspired method in which precision cut sheets of plastic film are folded to achieve desired geometries. Using these processes, lightweight, low cost, rapidly built quadrotors were designed and fabricated. Flight tests compared the resulting robots against similar micro air vehicles (MAVs) generated using other processes. Despite lower tolerance and precision, robots generated using the process presented in this work took significantly less time and cost to design and build, and yielded lighter, lower power MAVs.

A.M. Mehta (✉) · D. Rus
Massachusetts Institute of Technology, Cambridge, USA
e-mail: mehtank@csail.mit.edu

D. Rus
e-mail: rus@csail.mit.edu

K. Mohta · Y. Mulgaonkar · M. Piccoli · V. Kumar
University of Pennsylvania, Philadelphia, USA
e-mail: kmohta@seas.upenn.edu

Y. Mulgaonkar
e-mail: yashm@seas.upenn.edu

M. Piccoli
e-mail: piccoli@seas.upenn.edu

V. Kumar
e-mail: kumar@seas.upenn.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_12

1 Introduction

Robotic systems by their very nature can be highly capable and versatile, providing enhanced actuation and automation abilities to enable otherwise untenable activities. Robots can be useful by themselves [1], or can assist in conducting unrelated research [2]. Robots are also useful in furthering educational goals [3].

However, the tightly coupled nature of mechanical, electrical, and software subsystems in a robot often demand sophisticated engineering skills to design and build an appropriate device. Computer-aided design (CAD) packages can be expensive and arcane, and numerous such tools are often required. The varied toolsets and design environments cause unique specialized robots to be created from the ground up for each application, with little design reuse across projects.

High prototyping costs and turnaround times are also impediments to robot development. Conventional fabrication techniques for mechanical and electrical components typically require trained technicians operating sophisticated machinery, and thus take significant investments of time and money. Recent developments in 3D printing technologies have done much to ameliorate such effects, allowing a vast variety of solid parts to be fabricated in a home environment. Nonetheless, such 3D printers still take hours to produce parts, making incremental improvements and rapid turnaround untenable.

Ultimately, robot creation currently tends to remain an expert's domain. In order for robotics to become prevalent, then, the entire process from conception to construction needs to be modified.

A key element to simplify the design process is to enable modular design. By breaking up a robot into functional subsystems, new designs can be made from previously developed and tested components of other robots. These modules can span disciplines—robotic subsystems necessarily involve mechanical, electrical, and software components. As more robots get designed in the system, the corpus of available components will grow; eventually robot development can transition to making high level functionality decisions and designing by connecting components from libraries.

Alternate fabrication processes are also necessary to ease robot development. As design often involves repeated testing with incremental improvement, a quick and cheap method of rapidly iterating prototypes can greatly increase the number of build-test-refine cycles, resulting in more varied and successful robots.

With these goals in mind, a new process was developed for rapid robot development. A quadrotor micro air vehicle (MAV) system was selected as a simple but nontrivial instance of a robotic specification. Its mechanical structure is basic enough to concisely demonstrate the modular code-based design, while its resulting behavior and performance is rich enough to demonstrate the relevance of this process.

Mechanical airframes were designed using a set of scripts in the Python programming language, then fabricated by folding a sheet of cut plastic. The flight control board and software were designed using existing library modules. The completed MAV was then flown to analyze it's performance. Data from these flights were compared across similar robots created using alternate design and fabrication techniques to characterize the scripted printable process.

The contributions presented in this paper include:

- the new scripted design platform and printed folding fabrication process for producing mechanical structures,
- a specific lightweight, low cost, and rapidly designed and manufactured MAV generated using them, and
- a comparison of these to other conventional processes and MAVs.

In particular, this process outshines other robot design methodologies by:

- allowing extensive *modular design*, to the level of simply connecting existing modules to generate a final robot,
- requiring easily available *low cost* software, hardware, and raw materials,
- fabricating the designs in considerably *shorter time*,
- producing *lighter* yet robust mechanical structures,
- enabling *rapid iteration* through the use of scripted design.

This paper begins in Sect. 2 with a discussion on the various rapid fabrication methods available for creating mechanical structures, and compares them to the folded plastic process that is the focus of this work. Section 3 follows by outlining methods for designing mechanical structures—existing CAD programs are compared against the scripted design methodology developed herein. The modular design of the control system, now common in hardware and software development, is explained in Sect. 4. Section 5 describes the setup in place to characterize the robots and presents the data from those flight tests. The conclusion in Sect. 6 examines the results presented in this work.

2 Rapid Fabrication Processes

A wide variety of conventional manufacturing processes can be used to generate custom parts to meet arbitrary size, weight, and strength requirements of robotic components. However, these generally require considerable time, expense, and expertise. Rapid fabrication processes trade off precision and specificity for ease of use and build speed; some of these are described below. In particular, a process is proposed in which precision cut plastic sheets are folded to realize desired 3D geometries. This has the benefit of producing extremely light structures in a small fraction of the time required by other methods.

2.1 3D Printing

3D printing is an additive manufacturing technique that has gained widespread popularity for producing rapid prototypes. It owes this popularity in large part to the

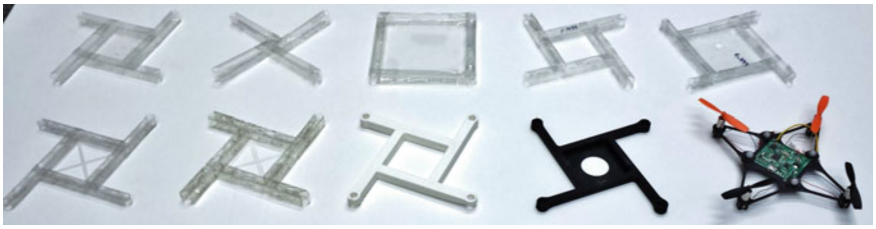


Fig. 1 A fleet of quadrotor frames weighing 4–15 g were manufactured using 3D printing technologies and folding. Source materials included ABS, PLA, photopolymer plastics, and polyester

relatively low cost of raw materials and hobbyist tabletop machines and to the ease of producing a part from a CAD specification when compared to traditional machining practices. However, 3D printing can still take on the order of hours per part. Furthermore, 3D printing is not always cost effective.

This method was used to produce a number of quadrotor frames, some of which are shown in Fig. 1. An industrial-grade printer was used to fabricate a frame out of acrylonitrile butadiene styrene (ABS), a common thermoplastic, a hobbyist grade printer was used to make frames out of polylactic acid (PLA), another common source material, while a desktop printer was used to make frames with a photopolymer. Fabrication of these structures took between one to three hours. The parts ranged in weight from 6g to upwards of 15g, based on the configuration options set during manufacture. The lightest part was nothing more than a hollow shell, with a single layer of plastic forming the surface of the geometry (at 0 % infill).

2.2 Plastic Sheet Cutting

Precision cutting can be used to generate arbitrary 2D parts from potentially inexpensive solid plastic sheets. This method has distinct benefits over other machining methods; though laser cutters are more expensive than the cheapest 3D printers, higher precision is achievable at a lower cost than professional grade printers or conventional machining. Alternately, desktop paper or vinyl cutters can accomplish a similar task for a fraction of the price. Design and machining requires simpler CAD tools and demands less skill from a designer. And fabrication is fast—parts generally take on the order of minutes to cut. However, precision cutting is limited to directly producing strictly 2D designs.

2.3 Origami-Inspired Folding

Inspired by the traditional Japanese art of origami, folding is an efficient method of creating 3D structures from planar fabrication processes such as the sheet cutting

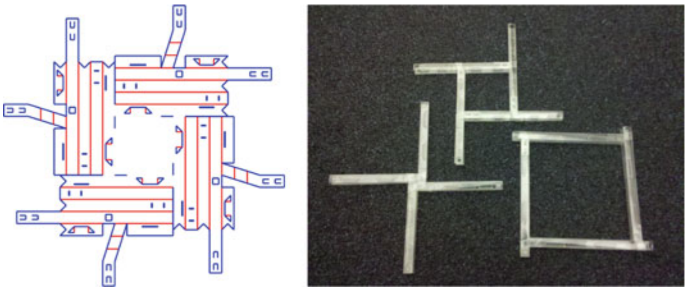


Fig. 2 The folded frames on the right are made from design files generated by the developed Python package like the one on the left. A laser cutter or desktop vinyl cutter operates on a sheet of plastic, cutting along the *blue lines* and perforating along the *red lines*

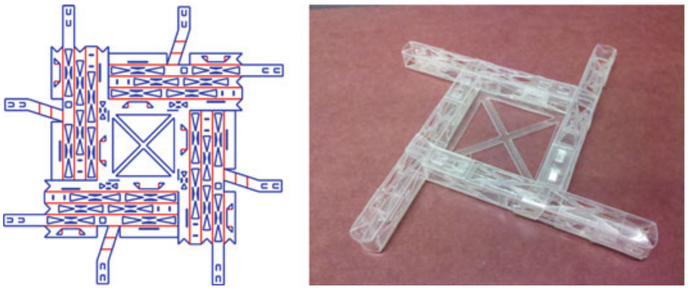


Fig. 3 Automated scripts can quickly and easily apply complex modifications to the generated designs, such as the addition of lattice shaped speed holes on each solid wall of the quadrotor frame

described above. Using 2D processes such as cutting or laser machining, folding patterns can be formed on a thin flat substrate similar to creasing a sheet of paper to create a tendency to fold along these creases. The resulting perforation lines can be manually or automatically [4] folded in both mountain and valley directions. This approach produces printed 2D precursors that are subsequently folded into rapidly and easily fabricated 3D robots [5–7]. Quadrotor frames designed using the Python package described below can be seen in Figs. 2 and 3. These designs were cut out of sheets of polyester (PE) film using a commercial laser cutter. Arbitrary thickness film could be used: the lightest frames, made using 0.005” thick stock with a latticed design, weighed less than 3g; the strongest structures, made with unbroken 0.010” film, weighed just under 8g.

3 Mechanical Design

Robot mechanical design involves generating 3D geometries that meet structural, kinematic, and dynamic requirements for specific robot abilities. There are a number of software tools that engineers can use to aid in this process; some programs are compared below. The new work presented in Sect. 3.3 is a Python-based scripted design package, developed to provide a more beginner-friendly tool that can allow for powerful automation, modular design, and widespread sharing and reuse of components and scripts.

3.1 Commercial CAD Software

Among professionals in the robotics community, mechanical design is most often done using commercial CAD packages such as SolidWorks and AutoCAD. SolidWorks is known for its user-friendly interface and 3D capabilities, while AutoCAD is often used for its command-line power at 2D drawings. Both programs are highly featureful when it comes to making sophisticated designs, but are very expensive. They also require significant processing power and graphics hardware to operate.

In addition to their high cost, the underlying proprietary nature of these tools inhibits widespread sharing or collaborative design. Binary source files inhibit the effective use of sophisticated distributed version control systems to fork and merge designs and changes. Furthermore, though scripting options exist in such programs, they are not well developed; rather than being able to automate repetitive design decisions, most tasks are left to the user to implement.

For this work, SolidWorks was used to design quadrotor frames to be 3D printed. The design was quickly implemented from scratch by an experienced designer, and the entire geometry was manually drawn and each dimension was individually constrained. Though dimensional changes were easy to implement, larger scale configuration modifications required starting a completely new design.

3.2 OpenSCAD

OpenSCAD is a free open source cross-platform 3D CAD program. It takes C-style text-based programs as input to generate 3D geometries by hierarchically applying primitive operations to basic shapes. With a low cost to entry given both the free program and minimal hardware requirements, it has become popular among hobbyist designers. The text-based source enables widespread sharing and modification, as demonstrated in online communities such as Thingiverse [8]. It also enables simple reconfiguration by rearranging the modular definitions of constituent elements. However, though scripted, it is primarily focused on object generation, with only basic automation options. Nonetheless, again due to its text-based input, other scripting languages can be used to automate OpenSCAD program generation.

3.3 Python

Python is a user-friendly scripting language with an extensive collection of modules. With a large community of users, it has been used to interface to CAD programs, as well as generate CAD designs directly. In this work, a Python package was developed to directly generate 2D design files for the origami-inspired folded plastic sheet fabrication process.

This package leverages hierarchical design principles to build a library of building blocks from which to generate mechanical designs. Much like OpenSCAD, operations on basic geometries are included in the package, from which scripts can assemble more complicated geometries. However, because Python is a fully featured programming language, and the internal representation of the designed geometries are fully available in userspace, many design tasks can be fully automated. This greatly simplifies the modification and extension of existing designs, even by non-expert designers.

The basic unit of design for folded robots is the face, a polygon that represents a flat continuous section of the source sheet. Faces can be joined at folded edges to make 3D structural elements. For example, a beam is formed by an array of rectangular faces as shown in Listing 1. These elements form the basic building blocks in the package's library. Other building blocks include additional structural elements such as various polyhedra to form bodies, wheels, etc.; as well as combining forms such as tabs and slots, hinges, and pleats.

Listing 1 A simple beam is formed by joining a number of rectangles along folded edges

```
class Beam(Drawing):
  def __init__(self, length, thickness, shape=3):
    Drawing.__init__(self)

    r = Rectangle(thickness, length)
    self.append(r.times(shape, 'e3', 'e1', 'r', FOLD))
```

Complicated mechanical design can then be reduced to hierarchically combining simpler building blocks with appropriate joints. Though this package focuses on the folded plastic process, a similar approach can include other fabrication methods.

A quadrotor frame is a simple illustrative example that can clearly demonstrate this design process. The frame itself is hierarchically composed of submodules:

- a belt comprises a rectangular face with tabs and slots for mounting,
- a motor mount includes a belt along with notches for the motor to sit against,
- an arm is created by attaching the motor mount to the end of a beam.

The quadrotor frame can then be generated by symmetrically joining four such motor arms. Example code demonstrating this composition can be seen in Listing 2.

Listing 2 Modular hierarchical design of the constituent motor arms for a quadrotor

```
class Belt(Rectangle):
  def __init__(self, thickness, length):
    Rectangle.__init__(self, thickness, length)

    t = tabs.BeamTabSlot(thickness, thickness)
    self.attach('e2', t.tabs.times(2, 'e0', 'e2', 'mt', FLAT), 'mt0.e0', 'mt0',
        FLAT)
    self.slots = t.slots.times(2, 'e0', 'e2', 'ms', FLAT)

class MotorMount():
  def __init__(self, thickness, motor_diameter):
    self.belt = Belt(thickness, pi * motor_diameter/2.)
    self.notch = Face(((thickness, 0), (thickness/2., thickness/2.))).flip()

  def connect(self, beam):
    beam.attach('r0.e2', self.notch, 'e0', 'm0', FLAT)
    beam.attach('r1.e2', self.belt.slots, 'ms0.e0', 's0', CUT)
    beam.attach('r2.e2', self.notch, 'e0', 'm1', FLAT)
    beam.attach('r3.e2', self.belt, 'e0', 'mt', FLAT)

class MotorArm(Beam):
  def __init__(self, length, thickness, motor_diameter):
    Beam.__init__(self, length, thickness, 4)

    m = MotorMount(thickness, motor_diameter)
    m.connect(self)
```

This code is straightforward to generate—a user-friendly interface can automatically synthesize code like this from intuitive graphical input. Furthermore, once such components have been designed and committed to the library of parts, the details of their implementation are no longer relevant. New designs can be formed through combinations of these underlying building blocks. Existing designs can be easily reconfigured by adjusting the composition parameters of the constituent modules.

In addition to the modularity, another benefit of this method of design comes from its scripting abilities. An automated script was able to generate matching tabs and slots to attach the faces for each beam based on overall geometry. Another script was used to perforate each solid face with lattice-like speed holes, as shown in Fig. 3. Though such scripts take some time and expertise to develop initially, they can then be applied at will. In contrast to manual design tools that require repetitive placement of each feature, this process enables future designs to apply a short snippet of code to modify the entire design, regardless of complexity, as seen in Listing 3.

Listing 3 Scripting capabilities allow repetitive or complex modifications to be easily encapsulated and shared

```
import quad
import lattice

q = quad.Quad(radius = 75)
lattice.latticify(q)
```

4 Control System

4.1 *Electronic Hardware Design*

Similar to the Python package for mechanical design, the controller for these vehicles employed modular design to quickly develop highly customized circuits from an electrical design library of component modules. Expert schematic designers create self contained schematic sheets for various electronic subsystems using EAGLE PCB Design software; experienced PCB designers can create accompanying board layouts for these schematics. Complex boards can then be compiled from these libraries by inexperienced users by stitching the desired components together. The generated hardware designs can be sent out to commercial foundries to get manufactured. The result is indistinguishable from an expert designer's board yet consumes less time and skill to compile.

The quadrotor hardware comprises six library modules. An ARM Cortex M4 STM32F373 microprocessor forms the central controller, interfacing with an Atmel AT86RF212 900MHz Zigbee transceiver for wireless communication and an InvenSense MPU-6050 six degree of freedom MEMS gyro plus accelerometer for inertial sensing. A voltage regulator module regulates power to these components. Four instances of DC brushed motor drivers complete the flight control board. A USB connection to the microprocessor allows for programming and direct computer communication. Because both the USB and wireless blocks are used, an identical board can connect to a host computer, enabling bi-directional wireless communication without needing special basestation hardware. The final board is less than 15 cm^2 and can be made in a single layer process.

4.2 *Software*

The software driving a quadrotor controller needs to stabilize both its attitude dynamics as well as its position. The attitude controller needs a high update rate to handle the fast dynamics of the small frames; this then allows a position controller to run at a slower rate.

4.2.1 Onboard Controller

The low level software package complements the electronic hardware. A software library for each electronic module exists in the database. Their inclusion is done via preprocessor statements indicating the hardware's pin location. Future versions will extract this data from the schematic files automatically. Additional libraries are included in the same manner and range from LED manipulation to timers to brushless motor vector control. The quadrotor uses the Zigbee radio, IMU, serial packetization, and LED manipulation libraries, while the basestation uses the same libraries plus the USB and minus the IMU.

The microcontroller estimates its attitude at 2 kHz, rate limited by the IMU output data rate. A PD attitude controller runs on the microcontroller with target attitudes received over the radio from a basestation computer. Motor speeds are controlled by pulse-width modulation (PWM) commands to the motor drivers. In addition to maintaining stable hover, the autopilot wirelessly reports the measured inertial rates, calculated orientation, and commanded control inputs for data logging, post processing, and analysis.

4.2.2 Base-Station Software

The outer loop position control is implemented using a pose and yaw estimate generated at 100 Hz by an external motion capture system from Vicon Motion Systems [9]. The position controller is a simple PID controller as described in [10] with an integral terms added for the 3 axes.

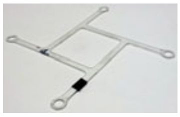
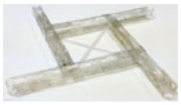
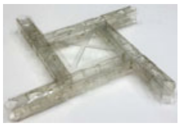
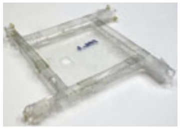
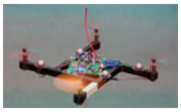
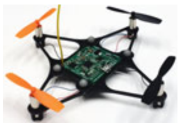
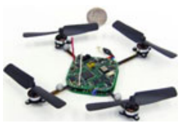
5 Flight Testing

In order to evaluate the relative merits of the fabrication methods, a common design for a quadrotor frame was realized in the different processes. An additional frame of similar dimensions, but a more traditional design was used as a control. These frames were then outfitted with the same controller and actuators and flown through a series of tests. The flight tests consisted of both autonomous hovering and controlled flight using the control system described above. A full summary of the frames tested in this work is presented in Tables 1 and 2. These fabricated devices were compared against a commercially available option, the NanoQuad from KMel Robotics [11].

Data collection for the flight parameters of the various models was conducted in two phases. Power consumption during hover was measured while tethered to a power supply, yet still carrying its battery, as seen in Fig. 4. To measure dynamic parameters, the power supply was disconnected and the battery was used. The data logged during the untethered flights included the position of the quadrotor and all feedback indicated in Sect. 4.2.1.

To ensure reasonable comparisons between the different frames, the vehicle controllers were modified for each frame. Frequently, quadrotors are described as a fourth order system (although they can go above fifth order), and are divided into two second order PD controllers. Separation of time scales allows us to do this if the inner controller is significantly faster than the outer. The inner, high speed, onboard PD controller controls vehicle attitude and outputs motor PWM commands. The outer, lower speed, offboard controller controls vehicle position and commands vehicle attitude. In both cases, there is negligible natural damping (from wind resistance) and no returning force around the hover condition in the horizontal directions. Thus,

Table 1 A comparison of the design of the various quadrotor frames built and flown

		Design process	Fabrication method	Material
(A)		SolidWorks	Laser Cut	1/16” Acrylic
(B)		Python script	Laser Cut + Fold	0.005” PE
(C)		Python script	Laser Cut + Fold	0.010” PE
(D)		Python script	Electronic Cut + Fold	0.005” PE
(E)		SolidWorks	Fused Deposition Modeling	ABS
(F)		SolidWorks	Stereolithography	Photopolymer
(G)		(N/A)	Purchased	Carbon fiber

all proportional and derivative effects come from the controllers alone. Examining the vehicle’s attitude behavior along one of the principal axis, after taking the Laplace transform, the simplified transfer function becomes:

$$\frac{Y(s)}{U(s)} = \frac{K_p + K_d s}{Is^2 + K_d s + K_p}$$

where K_p and K_d are the proportional and derivative gains, and I is the vehicle’s moment of inertia along that axis.

Table 2 A comparison of the performance metrics of the various quadrotor frames

	Fabrication time (min)	Frame mass (g)	Total mass (g)	Hover power (W)	Power to mass (W/g)	Hover STD			Rise time (s)
						X (cm)	Y (cm)	Z (cm)	
(A)	**1**	**3.9**	**38.9**	**9.9**	0.254	1.08	1.26	**0.05**	**0.344**
(B)	8	4.0	39.0	10.2	0.262	1.13	0.98	0.11	0.352
(C)	8	7.6	43.0	10.4	0.242	1.12	1.93	0.11	0.374
(D)	10	4.0	39.0	10.2	0.262	**0.66**	**0.66**	0.08	0.400
(E)	58	15.2	51.0	11.2	**0.220**	2.22	4.94	0.16	0.480
(F)	150	7.4	42.4	10.4	0.245	1.23	1.11	0.12	0.366
(G)	(N/A)		74.9	17.2	0.230	1.50	1.42	0.21	

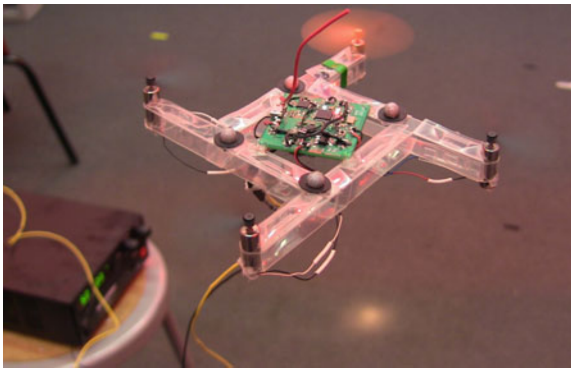


Fig. 4 A folded plastic quadrotor is tethered to a laboratory supply to measure power consumption during controlled hover

Solving the for the poles:

$$s = \frac{-K_d \pm \sqrt{K_d^2 - 4IK_p}}{2I}$$

we see that the poles converge at $K_d = \sqrt{4K_pI}$. This corresponds to critical damping, which was the target for the experiments. In reality, the coefficient of 4 does not yield critical damping due to sensor error, time delays, etc. The coefficient and K_p for the inner control loop were found manually while K_d was adjusted according to the formula above for each frame. The same process was repeated in the outer control loop, replacing I for the mass.

5.1 Comparison of Airframe Properties

Airframe comparisons were broken down into three categories: production, frame characteristics, and assembly. Production comparisons include fabrication time, labor time, material costs, and machine costs. Frame characteristics are mass, stiffness, and brittleness.

The fabrication time of 2D designs are markedly lower than their 3D counterparts. These range from 1 to 2 min using a laser cutter, and 4 min using a desktop electronic crafts cutter. Only frame (A) from Table 1 required no manual labor to construct the frame. The foldable frames can be folded by an inexperienced person without directions or tools in less than 30 min, while an experienced person can complete a frame in as little as 6 min. Depending on the 3D printer type, the removal of support material (scaffolding used during the printing process) takes a comparable time to folding, and can require more equipment like acid baths or high powered water guns. The folded frames are the cheapest material-wise. The 0.005" polyester frames, (B) and (D), cost roughly $0.13 USD when bought in quantity and the 0.010" frames, (C) are $0.25. They were cut on both $35,000 laser cutters and $250 electronic cutting machines with similar results. The 1/16" acrylic frame uses $0.75 of material, but cannot be cut with the low cost electronic cutter. The 3D printed ABS, (E), or PLA frames are $0.75 not including support material, which could bump it up to $1.00, and were printed on $30,000 machines. Hobby versions of this machine are now in the low $1000 range. The control frame, (F), cost $11 to print on a $20,000 machine. In contrast, the commercial option (G) costs on the order of $4000.

The frames' masses vary greatly, ranging from 3.9 to 15.2g. While in general, frame mass corresponds to frame rigidity, the folded frames have a small amount of play due to the clearances required to feed tabs through slots. The ideal solution appears to be to 3D print extremely light frames like the white frames found in Fig. 1. Unfortunately, 3D printers have a minimum thickness on the order of the polyester sheets used for folding. At the minimum thickness, each wall of the vehicle is a single layer thick, which is very brittle in the FDM style of 3D printing. Note that none of these frames were tested since their layers delaminated before any significant testing could be completed. Furthermore, frame (A) broke in a hard landing after flight testing was complete. Because it is flat, it is particularly weak in the vertical direction. Tubular or rod shapes with uniform materials like the folded polyester and the photopolymer did not experience this problem. These frames are also compliant without being brittle, contributing to their high tolerance to damage.

The stiff, 3D printed frames were quick and simple to assemble. Features like press fits for the motors not only kept the motors in place, but also aiming the motors in the proper direction. Utilizing the belts mentioned in Sect. 3.3, the folded frames easily accepted the motors. Care was required during assembly, however, since the play in the frames could cause the belt to grab the motor on an angle, resulting in motors facing several degrees off of vertical. Frame (A) had no mechanism for aligning motors as it is a thin, 2D design. Motors are aligned by eye and held in place using hot melt adhesive. The first flight of this frame resulted in uncontrollable yawing. Only after careful remounting was it a competitive frame.

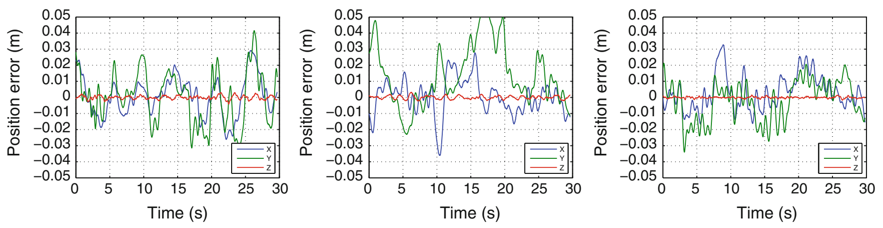


Fig. 5 The position error during hover for folded quadrotors, frame B (*left*), frame C (*middle*) and the laser-cut quadrotor, frame A (*right*) demonstrate performance comparable to commercial options

5.2 *Comparison of Flight Characteristics*

A plot of the hover performance of the folded frames (frame B and frame C) and a laser-cut frame (frame A) is shown in Fig. 5. The deviation of the position from the desired setpoint, summarized in Table 2, demonstrates hover performance better than or on par with that of research grade quadrotors [10, 12].

Though the full system mass was largely dominated by the control electronics and battery, the lighter frames generated by the cutting processes consumed notably less power in hover. With a lighter control platform (such as the 2 g controller from [13]), this benefit would be further magnified.

However, the folded frames were also less efficient, consuming more power per unit mass than those produced by conventional technologies. The relaxed tolerances of the folded structures resulted in misaligned motors; these then generated antagonistic thrust, increasing the required ratio of power consumption to effective lift. Furthermore, the more compliant frames leached energy from lift into the bending modes of the frame.

The impact of the of motor misalignment is further evident from the control signals to the four quadrotor motors. Figure 6 shows the motor PWM values during hover for the same quads as in Fig. 5. Though the position errors are similar for these quadrotors, the relative motor PWM values vary significantly. The large difference

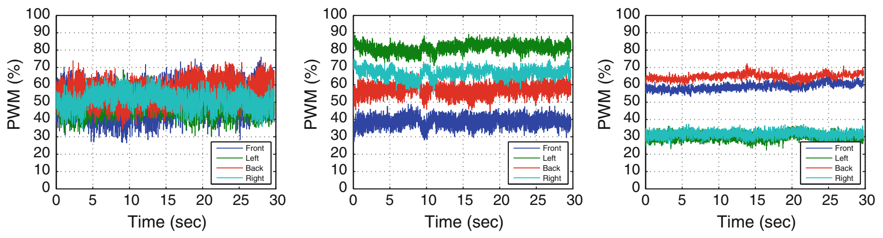


Fig. 6 Motor PWM during hover for folded quadrotors, frame B (*left*), frame C (*middle*) and the laser-cut quadrotor, frame A (*right*)

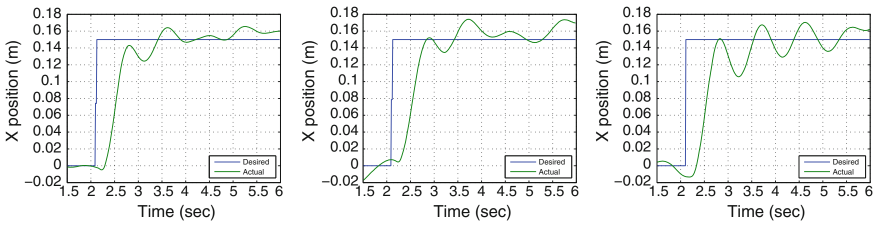


Fig. 7 The step response for a 0.15 m step in X-direction for frame B (*left*), frame C (*middle*) and frame A (*right*) demonstrates the agility and accuracy of the airframes. The position error about the setpoint can be further improved with a custom designed trajectory for each frame

in commanded output between the four motors is due to the torques required to compensate for the parasitic moments generated by off-axis alignment.

To compare the dynamic response of the different frames, we gave a step input of 0.15 m in the X-direction and looked at the resulting position response. These are shown in Fig. 7 and summarized in Table 2. The lighter frames are more agile—the constant control effort exerted by the motors has a greater effect given lower mass. Frame A, being made of a more flexible 2D sheet, displays greater amplitude oscillations than the stiffer folded structures.

6 Conclusions and Future Work

This paper introduced a new system to simplify the process of robot design. To evaluate the system, both the process and the resulting robot must be considered. The data presented above (and summarized in Table 2) displayed notable advantages of this system along both fronts.

The combination of the folded plastic fabrication method with the Python script-based modular design environment significantly reduced both design cost (both of raw materials and necessary hardware and software tools) and time (to both generate and fabricate complete designs). Combined, these benefits greatly ease the viability of iterated design-build-test cycles, critical for robot design optimization. Along with the simpler design environment, this system can bring the process of robot creation into the hands of non-experts.

This work identified areas where this system can be improved. Most notably, future work will focus on simplifying the user interface. This system reduces complicated robot design down to assembling component modules; a user-friendly interface can confine engineering expertise exclusively to module generation, thus eliminating any specialized skills required to synthesize robots. The system will also be expanded to combine the modular design of the both mechanical and electrical subsystems into a combined framework, further simplifying the process.

The robots generated by this system display benefits over those generated by current processes. In the case of mobile robots and especially MAVs, minimizing system weight while maintaining robustness is a key design goal. It is in this that the folded plastic process particularly shines. The high strength, low weight mechanical frames enabled reliable, high performance MAVs comparable to existing technologies, though at the nominal expense of efficiency due to lower overall stiffness. These parameters can be tuned by system design and material selection, and so future design effort can work towards ameliorating compliance in the structure.

In addition to robot design, the system presented in this paper is also valuable in determining avenues of future robotics research. Conventional quadrotor control algorithms assume symmetric parallel motor axes; imperfect alignment and folding tolerances can lead to inefficient or poorly controlled flight. Though mechanical design effort can attempt to lessen such variations, an important orthogonal direction for future research revealed through this work is to develop robust control algorithms that allow robots to identify their dynamics and self-tune controller gains.

Supplemental Multimedia

A video of the work presented in this paper can be accessed from:
http://people.csail.mit.edu/mehtank/ISRR2013/ISRR2013.mp4

Acknowledgments This work was funded in part by NSF grants 1240383 and 1138967, for which the authors express thanks.

References

1. Forlizzi, J., DiSalvo, C.: Service robots in the domestic environment: a study of the roomba vacuum in the home. In: Proceedings of the 1st ACM SIGCHI/SIGART Conference on Human-Robot Interaction, ACM, 258–265 (2006)
2. Lauder, G.V.: Flight of the robofly. Nature **412**(16), 688–689 (2001)
3. AFRON design challenges—african robotics network (AFRON). Online. http://robotics-africa.org/afron-design-challenges.html accessed 01 Feb 2014
4. Felton, S.M., Tolley, M.T., Onal, C.D., Rus, D., Wood, R.J.: Robot self-assembly by folding: a printed inchworm robot. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), 277–282. IEEE (2013)
5. Onal, C.D., Wood, R.J., Rus, D.: Towards printable robotics: origami-inspired planar fabrication of three-dimensional mechanisms. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), 4608–4613. IEEE (2011)
6. Onal, C.D., Tolley, M.T., Koyanagi, K., Wood, R.J., Rus, D.: Shape memory alloy actuation of a folded bio-inspired hexapod. In: ATBio Workshop, IROS (2012)
7. Onal, C.D., Wood, R.J., Rus, D.: An origami-inspired approach to worm robots. IEEE/ASME Trans. Mechatron. **18**(2), 430–438 (2013)
8. Thingiverse—digital design for physical objects. Online. http://www.thingiverse.com/. accessed 01 Feb 2014
9. Vicon. Online. http://www.vicon.com/. Accessed 01 Feb 2014
10. Michael, N., Mellinger, D., Lindsey, Q., Kumar, V.: The GRASP Multiple Micro-UAV Testbed. IEEE Robotics and Automation Magazine **17**(3), 56 (2010)
11. Kmel robotics. Online. http://www.kmelrobotics.com. Accessed 01 Feb 2014

12. Kushleyev, A., Mellinger, D., Kumar, V.: Towards A Swarm of Agile Micro Quadrotors. Science and Systems. In: Proceedings of Robotics (2012)
13. Mehta, A.M., Pister, K.S.J.: Warpwing: a complete open source control platform for miniature robots. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 5169–5174. IEEE (2010)

The Solving by Building Approach Based on Thermoplastic Adhesives

Fumiya Iida, Liyu Wang, Luzius Brodbeck, Derek Leach, Surya Nurzaman and Utku Culha

Abstract While, in nature, changes of morphology such as body shape, size, and strength play essential roles in animals' adaptability in a variety of environment, our robotic systems today still severely suffer from the lack of flexibility in morphology which is one of the most significant bottlenecks for their autonomy and adaptability. With the ability to autonomously modify own body shapes or mechanical structures in surroundings, robotic systems could achieve a variety of tasks in flexible and simple manners. For this reason, we have been investigating technological solutions based on a class of unconventional material, the so-called Thermoplastic Adhesives (TPAs), with which the robots are able to construct their own body parts as well as connecting and disconnecting various mechanical structures. Based on our technological exploration so far, in this paper, we introduce the concept of "solving-by-building" approach, in which we consider how autonomous construction of mechanical parts can help robots to improve performances or to "solve" problems in given tasks. Unlike the conventional adaptive systems that can only learn motor control policies, the ability to change mechanical structures can potentially deal with a significantly more variety of problems. By introducing some of the recent case studies in our laboratory, we discuss the challenges and perspectives of the solving-by-building approach based on TPAs.

F. Iida (✉) · L. Wang · L. Brodbeck · D. Leach · S. Nurzaman · U. Culha
Bio-Inspired Robotics Lab, Institute of Robotics and Intelligent Systems, ETH Zurich, Leonhardstrasse 27, 8092 Zurich, Switzerland
e-mail: fumiya.iida@mavt.ethz.ch

L. Wang
e-mail: liyu.wang@mavt.ethz.ch

L. Brodbeck
e-mail: luzius.brodbeck@mavt.ethz.ch

D. Leach
e-mail: derek.leach@mavt.ethz.ch

U. Culha
e-mail: utku.culha@mavt.ethz.ch

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_13

1 Introduction

Every biological organism exhibits many different forms of growth and self-replication in some parts of its life cycle if not the entire life. While there are many variations, the biological growth can be usually associated with the changes of volume, size, shapes, and the other mechanical properties such as rigidity, and all of which are driven by the underlying mechanisms of developmental processes such as cell growth, cell differentiation, or morphogenesis. These processes are fundamental for the biological systems because they determine both micro- and macroscopic details of organisms in self-organizing manners. For example, every biological organism starts its life from a single cell which is repeatedly replicated itself into organs with multiple cells that function as bones, skins, blood, and nervous circuitry [9, 24].

The concept of self-organizing systems has been investigated for many decades because of its substantial roles in biological systems [21]. Originally started as the theoretical exploration of cellular automata, reaction diffusion systems, or simulated nervous systems, a number of different aspects of self-organization processes have been reproduced in the engineered systems [18]. In particular, during the last decade or so, the self-reconfigurable and self-assembling systems have been intensively studied for this purpose. These approaches generally employ mechatronic modules that can be programmed to autonomously connect and disconnect to each other such that shapes and structures can be flexibly modified on the fly according to given task-environment. [5, 6, 13, 25, 31, 32]. Similarly, more complex interactions between modules were also investigated to understand the notions of self-organization in structure formation, where stochastic physical interactions between mechatronic modules were exploited [12, 30].

More recently, there has been a growing interest in the use of soft and deformable materials to construct flexible robots [8, 11], and along this line of research, we have been also investigating an alternative technical solution to enhance the capability of modifying mechanical structures of robotic systems. The innovation lies in the use of soft material, the so-called Thermoplastic Adhesives (TPAs, also known as Hot Melt Adhesives, HMAs) which is integrated into a robotic platform for both free-body fabrication, as well as bonding assembly of mechanical parts. Because of the TPA's unique mechanical characteristics, our approach provides the robotic systems a substantially more flexible capability in changing their own body structures with more precision, complexity, and diversity.

Based on our achievements so far, the goal of this article is to introduce the concept of "solving-by-building" approach. Unlike the conventional adaptive robots that rely on motion learning to deal with a variety of task-environments, the ability to modify mechanical structures allows the systems to handle substantially more complex problems in given task-environments, although there are still a number of challenges ahead of us. By introducing some of the recent case studies in our laboratory, in this article, we discuss the challenges and perspectives of this approach. Note that this paper focuses only on the conceptual design principles of this approach, with a rather restricted description of technical details. For those interested, technical details can be found in the publications referred to in the sections below.

We structure the rest of this article as follows. First, in Sect. 2, we explain the basic concept and enabling technologies of the proposed framework. In Sect. 3, we discuss the fundamental technical challenges of this approach. And finally, Sect. 4 discusses the implications, perspectives and applications based on the obtained knowledge so far.

2 Solving by Building Approach

Building an autonomous system is highly complex processes in which many knowledge, know-hows, skills, and operations are involved. For example, developers need to know about material properties, fabrication methods of parts, electronics and control, and how to test the complete systems and improve the entire organization of the systems. For investigating self-organization of autonomous systems, it is essential to automate the building processes partially or entirely. Although it might look an unsolvable challenge, this section discusses how such a problem can be systematically investigated by introducing the concept of "solving-by-building" approach, and explains the core enabling technologies that we developed for this purpose.

2.1 Conceptual Framework

For building an autonomous systems, developmental processes can be usually generalized into a set of similar steps as illustrated in Fig. 1. Given a target task and environment, a developmental process starts with a step of body design, in which source materials are determined; materials are processed; parts are fabricated and assembled; and sensory-motor circuitry is implemented. Then, once the body design step is completed, the next step deals with control design where physics models are firstly determined; and then the control policy is developed. Essentially the body and control design steps should be sufficient to enable the target systems functional for the target task-environment, although, in reality, a number of design iterations are necessary by conducting the interaction analysis. Typically the developed system is tested in the real-world task-environment, and the physical system-environment interactions are analyzed. For example, when a robot manipulator is developed for pick-and-place tasks, the robot behaviors are analyzed with respect to stability and robustness of gripping forces of objects, or speed and energy expenditure of the entire sequences. The interaction analysis is essential to identify problems in task execution or to increase performances. The outcome of analysis is then feedback to improve body and/or control design.

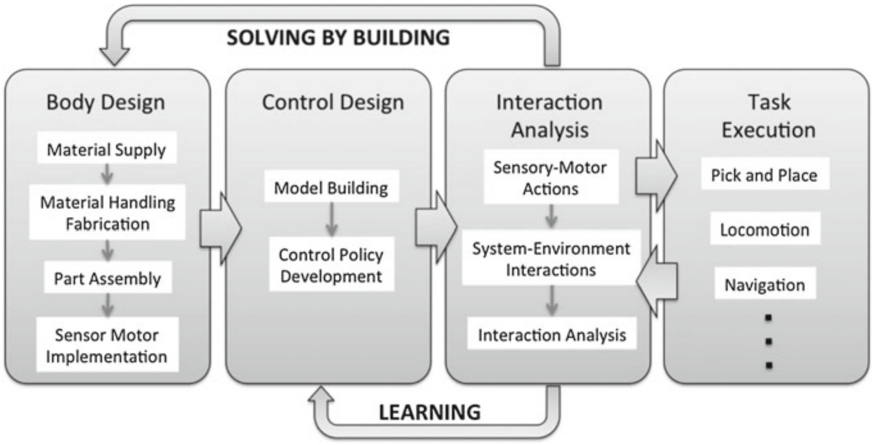


Fig. 1 Framework of the solving-by-building approach. A typical developmental process of autonomous systems is abstracted into four stages, i.e. Body Design, Control Design, Interaction Analysis and Task Execution. Usually human designers execute all these steps, but the robotics research of motor learning attempts to automate the process of control design (i.e. data obtained in interaction analysis can be used to improve control design). In contrast to the learning processes, the solving-by-building approach considers to automate the body design processes based on the feedback from interaction analysis

For reducing the amount of time and efforts in the system development processes, one can apply a "learning" mechanism which could automatically analyze physical system-environment interactions and autonomously adjust parameters related to physics models and/or control policies in the control design. The learning process can be typically generalized by an objective function that basically describes the target performance of the system in uncertain and dynamic task-environment, and the the outcome of analysis is feedback to adjust parameters in physics models and control policies.

In contrast, the motivation of the "solving-by-building" approach considers how the outcome of interaction analysis can be feedback to the body design step in Fig. 1. More specifically, in the solving-by-building approach, we also assume certain objective functions that can be used for quantitatively evaluation in the interaction analysis step, whereas the main difference from the learning mechanisms is to use the evaluation results to improve the processes in the body design step. With such a feedback mechanism, the systems should be able to broaden the adaptability in a broader range of uncertainties and changes in task-environment including those that the conventional learning mechanisms cannot deal with. In the rest of this article, we discuss how such feedback mechanisms can be practically applied to real-world robotic systems and identify the challenges and perspectives.

2.2 *Enabling Technologies*

The concept of solving-by-building approach is not a new one. In biology, the importance of morphological changes has been known for many decades, and in the field of complex systems and artificial life, there have been a number of "virtual robots" investigated for our in-depth understanding of the nature of adaptivity and autonomy [2, 14, 23]. However, there have not been much research activities that explored the technological means for this concept except for some case studies [1, 7, 17].

A real-world autonomous system is usually composed of a significant amount of different components, and automating the building processes requires considerable diversity in technologies. For a systematic investigation of this highly challenging problem, we have been focusing on one of the most fundamental yet technologically feasible aspects of building processes, which is the automated construction of passive mechanical structures. In nature, passive mechanical structures such as limbs, tendons, skins, hairs and nails are not generating forces and electric signals by themselves, but they are known to play significant roles in biological systems to be adaptive and autonomous. Automating the building processes of passive mechanical components is, however, not trivial if they have to be useful for robots without human intervention. As in the conventional self-reconfigurable systems that we introduced in Sect. 1, the morphological changes of robotic systems require so much mechatronic infrastructure for designing overall geometric shapes, moving parts, connecting and disconnecting components, for example. The use of mechatronic subsystems to achieve these self-reconfiguration processes requires complex hardware and control routines, and a simpler and more robust enabling technology seems to be necessary.

From this perspective, we have been investigating a simple technological solution, i.e. thermoplastic adhesives (TPAs), to be used for both formation of geometric shapes as well as connectivity control. TPA is a solvent-free polymer that exhibits two distinctive phases depending on the temperature [16]: At low temperatures typically below 60 °C, it is a solid plastic, while it becomes liquid at high temperatures over 140 °C. This transition is bi-directional and repeatable such that a solid TPA cube at room temperature, for example, can be heated up and deformed into arbitrary shapes, and the modified structure can sustain its shape by being cooled down. An important mechanical characteristic of TPA is its adhesion property that can be controlled through the bi-directional phase transition (Fig. 2). In the liquid phase, TPA is highly adhesive to almost any materials and it forms a large bonding strength as it is cooled down. Quantitatively, the bonding strength of TPA is approximately 1 MPa at room temperature, and it can be exponentially reduced by increasing temperature of the bonding surface [26, 28, 29] (Fig. 2).

We have developed a robotic manipulator that can exploit the TPA's mechanical characteristics for the demonstration of solving-by-building approach (Fig. 3). The manipulator is equipped with two specialized mechanisms at the end effector: First, the robot manipulator has the so-called thermo-connector equipped in the end-effector. The function of thermo-connector is heating and cooling of its flat surface

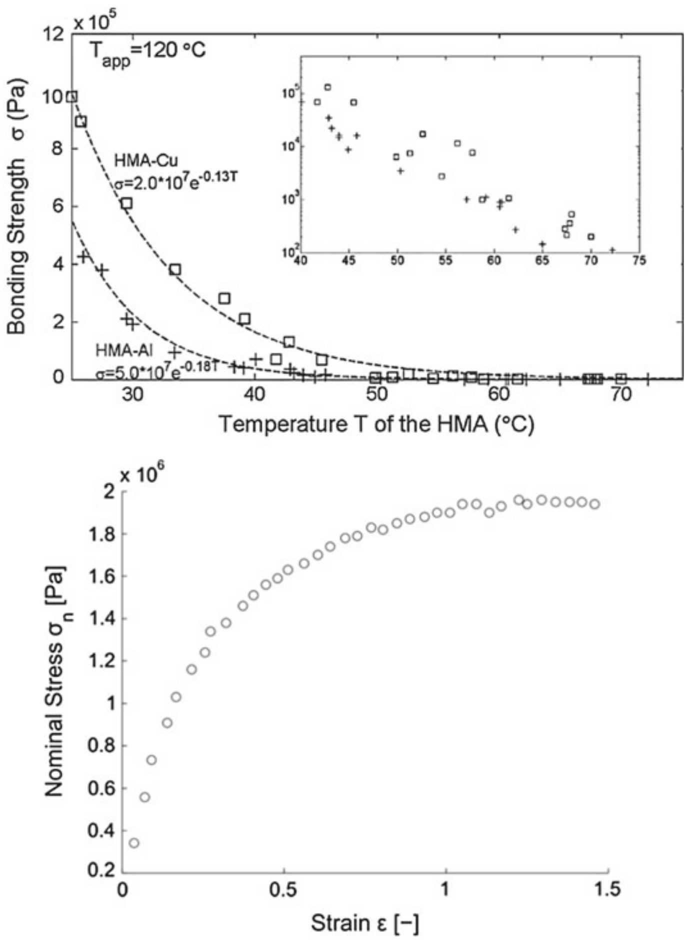


Fig. 2 Material properties of thermoplastic adhesives (TPAs) that we employed in our projects. (*Top figure*) Bonding strength of TPAs is exponentially decreased as the material temperature rises, thus a small change of temperature is sufficient to control adhesiveness in this material. Bonding strength, however, varies depending on the surface energy of bonding substrates, which is exemplified by copper and aluminum. Adapted from [29]. (*Bottom figure*) The stress-strain characteristics of TPAs is plotted under a room temperature. This figure indicates that TPAs are elastic but can take a stress up to 2 MPa

(i.e. the connection surface in Fig. 3) such that the surface can be used for liquifying and solidifying TPAs in physical contact. For example, if the surface temperature is above the melting point of TPAs, the surface can liquify the material on it, which can initiate a bonding of TPA to the surface by cooling the surface subsequently. Alternatively, a bonded TPA on the connection surface can be separated by re-heating the surface to the melting point. In order to control the surface temperature effectively, the connection surface is made of copper because of its characteristics of heat

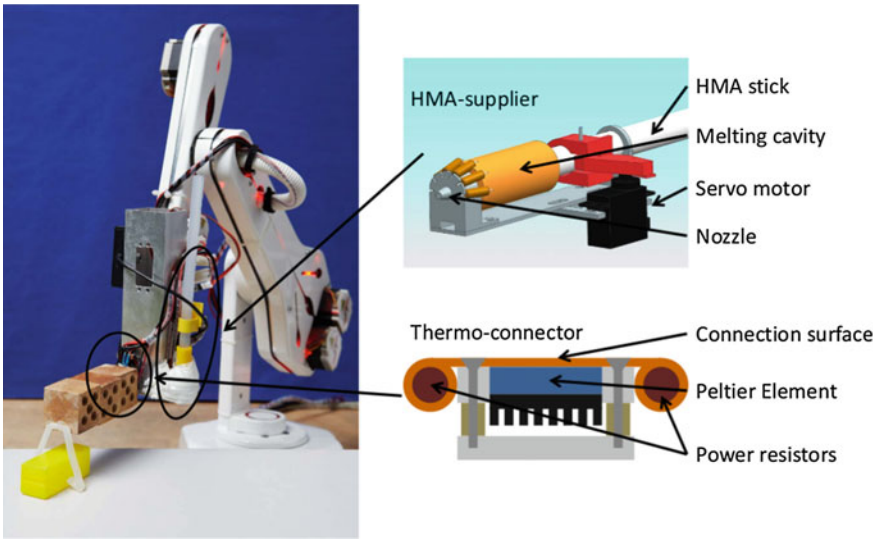


Fig. 3 An example of solving-by-building platform. (*Left figure*) The robotic manipulator that equips with the TPA handling devices (i.e. TPA Supplier and Thermo connector) at the end-effector. The robot base is a five-axis servo-controlled manipulator that is fixed on the ground and controlled by an external host computer. (*Right top figure*) A CAD design of TPA Supplier, which consists of melting cavity, nozzle and servo motor. A commercially available TPA stick is inserted into the melting cavity by the servomotor, and a liquified TPA material can be extruded from the nozzle. (*Right bottom figure*) A schematic design of the thermo connector, which consists of surface, Peltier element, and power resistors. Surface temperature of the connecting surface can be electrically regulated through Peltier element for cooling and the power resistors for heating. Adapted from [3, 4]

conductivity. An increase of temperature is achieved by power resistors attached the connection surface, and a cooling is done through a Peltier element which transfer the heat from the surface to heat sinks.

And the second mechanism is to utilize the adhesive property of TPA in the process of additive fabrication, in which variations of mechanical structures can be fabricated by placing a liquid-phase TPA flow and solidifying it [4] (Fig. 4). For example, when a thread is placed to form a spiral shape on a flat table, it cools down into a solid flat disk at room temperature, and an upright wall can be formed by accumulating a thread vertically. Although such additive fabrication processes usually employ other thermoplastics such as acrylonitrile butadiene styrene (ABS) or polylactic acid (PLA), we found it possible to use TPA for the same purpose by specifically developing a TPA handling device that regulates the flow of liquid-phase TPA under the kinematic control of a robot manipulator. So far our experimental setup is capable of continuously and smoothly extruding TPA thread with the minimum diameter of 1 mm, and the process is destabilized when fabricating with a smaller diameter. It is also important to mention that both the Young's modulus and tensile

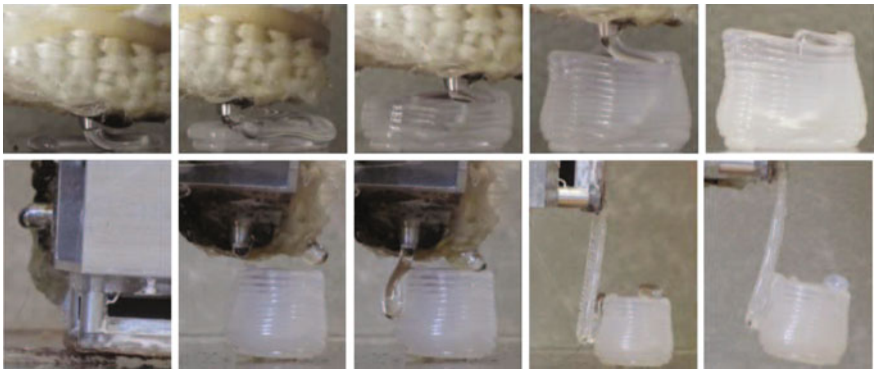


Fig. 4 Additive fabrication and part assembly based on thermoplastic adhesives. (*Upper figures*) The robotic manipulator equipped with the TPA supplier (Fig. 3) controls a TPA flow for additive fabrication. By regulating the TPA flow rate and speed of the end effector, the robot is able to construct a cup-like structure. (*Lower figures*) The same robotic manipulator operates a "bonding assembly" process: First the manipulator builds a stick-like structure by additive fabrication, then it is attached to thermo-connector. The manipulator then extrude a small amount of TPA on the cup-like structure such that the cup and the stick structures can be glued into one structure. Adapted from [26]

strength of TPA are as high as 10 MPa at room temperature, which allows to flexibly develop mechanical components with sensible structural strength.

3 Scalable Structure Formation

The additive fabrication technology is a promising solution for automatically constructing mechanical structures because it can construct almost any mechanical shapes as long as their geometries are within the reachable range of the manipulator [10]. Additive fabrication, however, is not sufficient to enable robots to achieve the solving-by-building approach, but there are still a few fundamental challenges to be solved. First, additive fabrication does not fully consider assembly/integration processes which are crucial in the body design of autonomous systems. In order to achieve the solving-by-building systems, the robots have to not only create variations of mechanical structures but also autonomously assemble or disassemble the parts into their own body plans. Second, there are severe limitations of the materials used in additive fabrication, and, as a result, it is not possible to generate variations of structures in terms of material properties. The material properties such as visco-elasticity, density, and friction are essential mechanical characteristics that make most of robots functional and useful, although they cannot be flexibly adjusted in the conventional additive fabrication devices. And third, additive fabrication does not allow structure construction in large scales, especially those larger than the size of their mother device [22]. The ability to construct larger structures is one of the unique function of living organisms which makes them functional and robust in uncertain and dynamic environments as exemplified in tree growth and multi-cellular organisms in nature.

To tackle these fundamental challenges, we have been focusing on the thermoadhesive property of TPAs to be used in the robotic systems, and investigating the implications of the technological solutions in the context of the solving-by-building approach. Interestingly, as explained in the case studies in this section, control of adhesiveness in robotic systems is a very powerful strategy that complements most of the aforementioned challenges in the conventional additive fabrication techniques.

3.1 Functional Structure Formation

The solving-by-building approach should be able to simplify the problems associated with robotic applications. In nature, we find numerous examples of this kind such as animals extending parts of their bodies to easily find and capture food, spiders and social insects constructing their nests, or other creatures growing hairs and antenna to increase sensitivity of receptors. What could be common requirements for such mechanical structures for our robotic systems, and how can we systematically investigate the technical challenges in our solving-by-building platforms? Since we restrict our research to construct only passive mechanical structures in this article, it is reasonable to consider how the constructed mechanical structures can cover desired lengths, areas, volumes while they could also sustain a certain amount of force and load.

Such mechanical structures need to satisfy two basic challenges: First the structures have to be fixated somewhere either in the environment or parts in the robot's own body, because floating structures are usually not very useful. For this reason, TPAs are an ideal material because it has the relatively large adhesion characteristics, and once a structure is constructed, it can be bonded to almost any solid connecting surfaces. And the second challenge is to design and construct mechanical structures that fulfill the size and strength requirements within the amount of given source materials. Essentially, the challenge here is to minimize the use of source materials while maximizing the size (i.e. lengths, areas, or volume of the structures of interest) and the strength (i.e. resistive load or pressure).

From this perspective, we have been exploring how our robot manipulator can autonomously construct minimum mechanical structures based on TPAs that can be, at the same time, fixated to the robot itself or to the given structures in environments. More specifically, as the first step of systematic investigations, we have developed a model based controller for the robot manipulator that is able to, on the one hand, make solid threads made of TPAs, and on the other, attach them onto the given frame structure (Fig. 5) [15]. For constructing a functional yet efficient structure that can support certain load on it, it is necessary to design the thread diameters along with the number of threads that are loaded, but we found that they can be realized based on a model based controller which considers physics models of extrusion process, as well as deformation of TPA threads under ambient cooling processes.

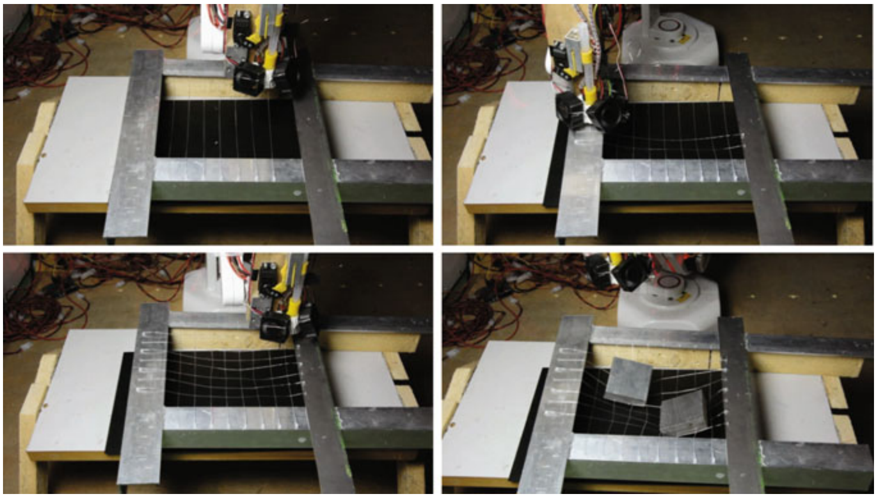


Fig. 5 Efficient construction of passive structures. The time-series photographs show how the robot manipulator can produce a set of TPA threads bonded onto an aluminum frame. The robot manipulator controls both its own axes and TPA Supplier such that it can keep the bonding of every thread to the frame while generating a constant diameter with certain tension. (*Bottom right figure*) At last, we tested the strength of the generated mesh structure by putting two aluminum blocks

The unique mechanical characteristics of TPAs can be also exploited for locomotion in a free 3D space. Inspired from the dragline locomotion of spiders, we developed a miniaturized TPA supplier such that it can be implemented onto a small and light-weight mobile robot that can locomote on the thread made by the robot itself (Fig. 6) [27]. This dragline locomotion is a typical use case of the solving-by-building concept, in which locomotion can be simplified and adaptable because of the building processes of mechanical structures. More specifically, aTPA thread that is made by the robot can be bonded to almost anywhere on a ceiling, and the robot can generate and extend the thread on the fly such that the robot can walk down by controlling the clamping mechanism at a certain speed. The robot is also able to vary the diameter of TPA thread flexibly between 1 and 5 mm depending on the payload and a safety factor required for the application. It is, however, important to mention that this particular application requires a dedicated holding mechanism that facilitates the control of thread formation during high-payload locomotion.

3.2 *From Additive Fabrication to Bonding Assembly*

In order for a solving-by-building robot to construct even larger structures, it is necessary to "ingest" source materials that are available within the robot's reach, and use the materials to construct the structure. Here we consider, for example, the

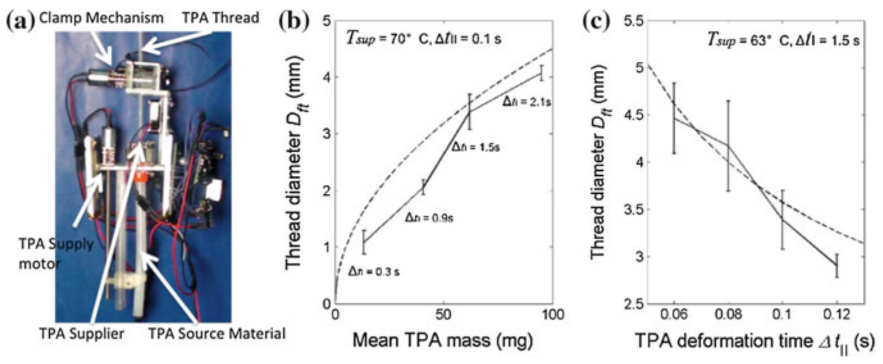


Fig. 6 Vertical locomotion with the solving-by-building approach. **a** A prototype of the mobile fabrication machine is equipped with a miniaturized TPA supplier and a motorized clamping mechanism. The TPA supplier generates a thread that is clamped by a holding mechanism to sustain the body weight of the robot. **b** Diameter controllability of the thread made by the robot. Diameter can be controlled by adjusting amount of TPA supplied (i.e. TPA mass regulated by the activation time Δt_I of the TPA supply motor). **c** Thread diameter can be controlled also by the amount of deformation time Δt_{II}, which indicate the activation time of the clamp mechanism. Adapted from [27]

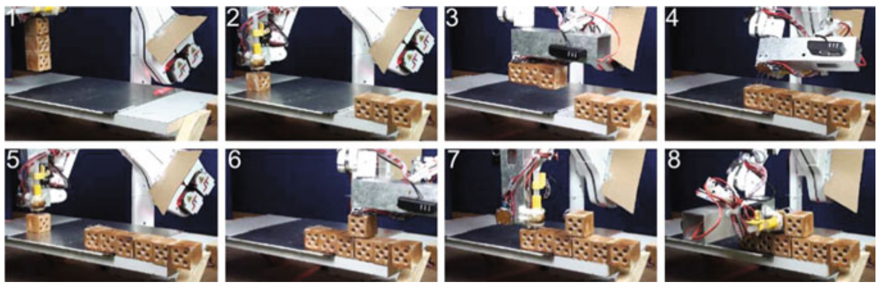


Fig. 7 A demonstration of the solving-by-building approach. The robot manipulator employs both the TPA supplier and thermal connector for picking and placing passive wooden cubes as well as bonding them together. A constructed stack is also shifted and bonded further with an additional stack in order to construct a long beam beyond the robot's own reach. Adapted from [3]

robot to visually (or blindly) identify the materials to be collected, manipulated, and assembled together to construct a structure, ideally tailored into a desired specific geometry. Figure 7 shows the demonstration of our robotic manipulator in which a long stick-like structure was constructed to extend the robot's own reach [3]. In this demonstration, we assumed that a number of passive wooden cubes are available within the reach of the manipulator, and considered how the robot is able to construct a single-supported beam structure by using the cubes. For this purpose, the robot needs to manipulate the cubes and bond them together by using the adhesive property of TPA.

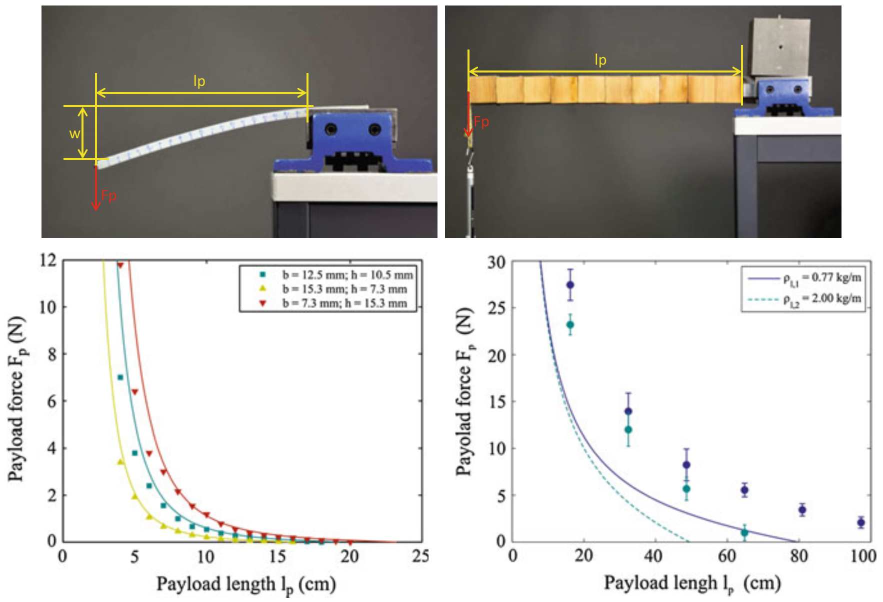


Fig. 8 Scalable extension of passive structures. Showing two types of extension: HMA only (*left figure*) and passive objects bonded by TPAs (*right figure*). Plots in lower figures show maximum payload forces F_p until failure for varying extension length. Failure modes differ: HMA only fails by bending ($w > w_max$), for extension with passive objects, bonding strength is critical. Partially adapted from [3]

This process of body extension that we call "bonding assembly" process is significantly more powerful than the additive fabrication process particularly when a larger structure needs to be constructed. Figure 8, for example, shows a simple comparison between the structured made by additive fabrication and bonding assembly processes with respect to the load exerted under single support [3]. Here we considered the beam is supposed to resist a payload F_p at the end, and compare lengths of beams that are able to support the payload. The experimental results in Fig. 8 indicates that, although the beam built by the additive fabrication can grow only up to 20 cm under no load, the beam made by the bonding assembly can reach further by at least four times.

This experimental results illustrated the complex nature of design problems for the solving-by-building approach based on thermoplastic adhesives. In general, adhesive materials such as hot melt in our system are usually softer (i.e. lower Young's modulus), thus harder to build larger structures, whereas stronger materials (i.e. higher modulus such as ABS and PLA, those typically used for additive fabrication) are, in contrast, not very adhesive to many types of bonding surfaces. An interesting insight we gained from the experiments above, however, is that the solving-by-building robots can take advantage of a soft adhesive material to construct larger structures when strong external materials can be manipulated. It is still an open question whether soft-adhesive material would be better than non-adhesive strong ones, but it is important to explore the degree to which soft-adhesive materials can be used to create a large variety of structures.

Given the basic understanding about the mechanical characteristics of the proposed approach, it is also important to investigate algorithmic aspects for scalable structure formation. For example, as briefly shown in Fig. 7, our robotic manipulator needs to execute a relatively complex set of actions in order to take advantage of adhesives for constructing larger structures, which can be significant constraints of scalability in structure formation. One of the most crucial constraints is originated from the fact that the manipulator has usually limited range of reaching of the end-effector. In fact, the example shown in Fig. 7 utilizes a specific set of actions that enable the system to construct a structure even if it grows beyond the reach of the robot manipulator.

To further investigate the algorithmic constraints of scalable structure formation, we have been investigating an optimization method that takes account of the physical constraints of robotic manipulators. For example, the simulation experiments shown in Fig. 9 consider a building task of a large structure that can bridge two distant locations (indicated by black and red squares in the figure). We implemented two physical constraints in this simulation, which are the location of a new cube to be bonded and

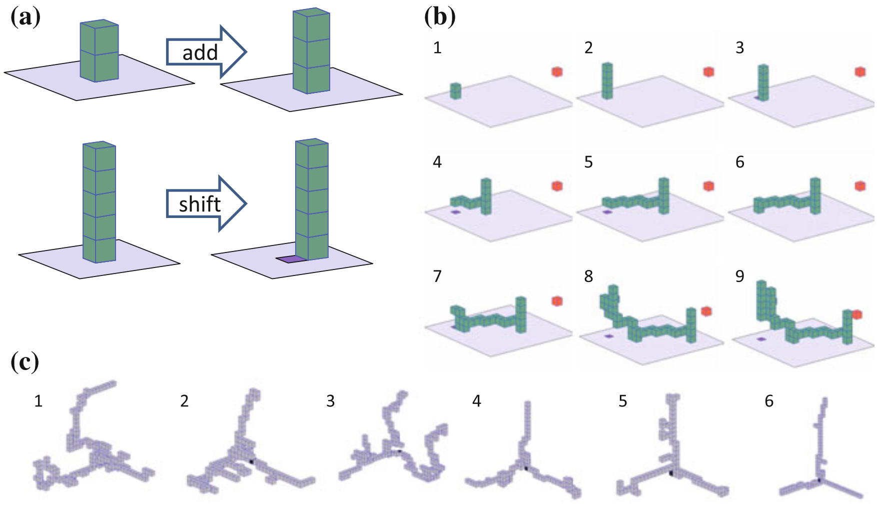


Fig. 9 Design exploration of passive structures by considering the constraints of the real-world robot manipulator. **a** Illustration of two basic operations, i.e. adding cubes and shifting the structure, which are used for the construction of structures. **b** An example structure that can be developed only by the two operations. A randomized optimization method was used to search a sequence of operations such that the resultant structure can bridge between the start and goal locations in a 3D space. We also applied a stability check algorithm to ensure that the structures don't fall down. **c** Other sample structures generated by variations of search algorithms. In this case study, we consider three target locations to be reached. C1–C3: No additional constraints. C4–C6: Additional stability constraints

the translational motions that the robot can generate against the entire constructed structures. Even with such simple physical constraints, we have demonstrated that a scalable structure formation can be possible as long as the manipulator can robustly execute these two types of actions.

4 Discussion and Conclusions

This article introduced the state-of-the-art technologies that help us understanding the notions of the solving-by-building approach. Conceptually the solving-by-building approach provides a new horizon for autonomous systems where we consider how an autonomous system can enhance its adaptability to uncertain and dynamic environment by modifying its morphologies such as shapes, sizes, and the other mechanical properties. Interestingly, there are many concrete engineering methods to automate some of the developmental processes of autonomous systems including the commercially available additive fabrication devices and their integrations to the other robotics technologies as outlined in this article. However, it is still a significant challenge to "close the loop": In order to provide feedback for autonomous improvement of body design and construction of autonomous systems, there are many issues to be tackled from both theoretical and technological standpoints.

Conceptually, it is important to consider the minimum set of requirements in a physically integrated system that is capable of generating a large variety of mechanical structures, since there is no universal material that can cover all different mechanical properties necessary for autonomous systems. In this sense, the investigation of adhesive properties seems to be a fairly reasonable approach because it enables 'ingestion' of a variety of external materials that have desired mechanical characteristics. The TPAs that we used in our projects seem to be a good choice from this perspective, because their bonding strength is reasonably large against almost all solid and dry bonding surfaces while it can be controlled with a relatively simple temperature control.

Scalability is one of the most significant challenges in our view: While there are much researches on the fabrication techniques on small-scale mechanical structures, it is largely unexplored how to construct larger ones, possibly larger than their mother devices. Unlike the biological cells that are capable of self-reproduction, our solving-by-building platforms cannot physically grow their own body sizes, thus special treatments are necessary for this purpose. In this article, we introduced two potential solutions (i.e. the case studies on mobile fabrication and robotic bonding assembly in Sect. 3), but it has not been fully clarified the degrees to which the proposed platforms can be scalable.

Finally, it is also important to explore more practical applications based on the concept and the new set of technologies. While adaptation of mechanical structures are expected to provide significant impact on many robotic applications such as transportation, manipulation, sensing, and exploration of unknown and uncertain environments, the technologies have to be further investigated in these contexts. For example, in our laboratory, we have been exploring how the adhesion control techniques can be applied to wall-climbing robots in unstructured environment, although there are many challenges remained in the low-level specifics such as rapid and efficient control of adhesiveness [20, 26, 28]. Also the solving-by-building approach can be applied for adaptive sensor morphology, with which a robotic system can autonomously adjust its range and sensitivity for force sensing [19]. As we gain more knowledge about the properties of thermoplastic adhesives and their handling technologies, we expect to have more new application opportunities in the near future.

Acknowledgments This research was supported by the Swiss National Science Foundation Professorship Grant No. PP00P2123387/1, the Swiss National Science Foundation through the National Centre of Competence in Research Robotics, and the ETH Zurich Research Grant ETH-23-10-3.

References

1. Adamatzky, A., Komosinski, M.: Artificial Life Models in Hardware. Springer, Berlin (2009)
2. Bongard, J.: Morphological change in machines accelerates the evolution of robust behavior. PNAS **108**(4), 1234–1239 (2011)
3. Brodbeck, L., Iida, F.: Robotic body extension for flexible and scalable reaching, under review
4. Brodbeck, L., Wang, L., Iida, F.: Robotic body extension based on hot melt adhesives. In: Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA 2012), pp. 4322–4327. St Paul, USA, 14–18 May 2012
5. Christensen, A.L., O'Grady, R., Dorigo, M.: Morphology control in a multirobot system. IEEE Robot. Automat. Mag. **14**(4), 18–25 (2007)
6. Diller, E., Pawashe, C., Floyd, S., Sitti, M.: Assembly and disassembly of magnetic mobile micro-robots towards deterministic 2-D reconfigurable micro-systems. Int. J. Robot. Res. **30**, 1667–1680 (2011)
7. Hara, F., Pfefer, R.: Morpho-functional Machines. Springer, Berlin (2002)
8. Iida, F., Laschi, C.: Soft robotics: challenges and perspectives. Procedia Comput. Sci. **7**, 99–102 (2011)
9. Johnston, T.D., Edwards, L.: Genes, interactions, and the development of behavior. Psychol. Rev. **109**(1), 26–34 (2002)
10. Jones, R., Haufe, P., Sells, E., et al.: RepRap: the replicating rapid prototyper. Robotica **29**, 177–191 (2011)
11. Kim, S., Laschi, C., Trimmer, B.: Soft robotics: a bioinspired evolution in robotics. Trends Biotech. **31**, 287–294 (2013)
12. Klavins, E., Ghrist, R., Lipsky, D.: A grammatical approach to self-organizing robotic systems. IEEE Trans. Autom. Control **51**(6), 949–962 (2006)
13. Kurokawa, H., Tomita, K., Kamimura, A., et al.: Distributed self-reconfiguration of M-TRAN III modular. Int. J. Robot. Res. **27**(3–4), 373–386 (2008)
14. Langton, C.G.: Artificial Life: An Overview. MIT Press, Cambridge (1995)
15. Leach, D., Wang, L., Reusser, D., Iida, F.: In situ thermoplastic thread formation for robot built structures, in preparation (2013)
16. Li, W., Bouzidi, L., Narine, S.S.: Current research and development status and prospect of Hot-Melt-Adhesives: a review. Ind. Eng. Chem. Res. **47**, 7524–7532 (2008)
17. Lipson, H., Pollack, J.: Automatic design and manufacture of robotic lifeforms. Nature **406**, 974–978 (2000)
18. Nolfi, S., Floreano, D.: Evolutionary Robotics: The Biology, Intelligence, and Technology of self-Organizing Machines. MIT Press, Cambridge (2000)

19. Nurzaman, S.G., Culha, U., Brodbeck, L., Wang, L., Iida, F.: Active sensing system with in situ adjustable sensor morphology, under revision (2013)
20. Osswald, M., Iida, F.: Design and control of a climbing robot based on hot melt adhesion. Robot. Auton. Syst. **61**(6), 616–625 (2013)
21. Pfeifer, R., Lungarella, M., Iida, F.: Self-organization, embodiment, and biologically inspired robotics. Science **318**, 1088–1093 (2007)
22. Revzen S, Bhoite M, Macasieb A et al.: Structure synthesis on-the-fly in a modular robot. In: Amato, N. (ed.) Proceedings of the 2011 IEEE/RSJ IROS, pp. 4797–4802 (2011)
23. Sims, K.: Evolving 3d morphology and behavior by competition. In: Brooks, R., Maes, P. (eds.) Proceedings of the Artificial Life IV, 28–39. MIT Press, Cambridge (1994)
24. Steinberg, M.S.: Reconstruction of tissues by dissociated cells. Science **141**(3579), 401–408 (1963)
25. Stoy, K., Brandt, D., Christensen, D.J.: Self-reconfigurable Robots: An Introduction. MIT Press, Cambridge (2010)
26. Wang, L., Brodbeck, L., Iida, F.: A parameterized-synthesis approach to the reconfiguration problem and its evaluation in pick-and-place, under review
27. Wang, L., Culha, U., and Iida, F.: Free-space locomotion with thread formation. In: The 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013), 3–7 November 2013, Tokyo Big Sight, Japan, accepted (2013)
28. Wang, L., Graber, L., Iida, F.: Large-payload climbing in complex vertical environments using thermoplastic adhesive bonds. IEEE Trans. Robot. in press (2013)
29. Wang, L., Iida, F.: Physical connection and disconnection control based on hot melt adhesives. IEEE-ASME Trans Mechatron (2012). doi:10.1109/TMECH.2012.2202558
30. White, P.J., Kopanski, K., Lipson, H.: Stochastic self-reconfigurable cellular robotics. In: Proceedings of IEEE International Conference on Robotics and Automation, 2888–2893 (2004)
31. Yim, M., Shen, W.M., Salemi, B., et al.: Modular self-reconfigurable robot systems: Challenges and opportunities for the future. IEEE Robot. Autom. Mag. **14**(1), 43–52 (2007)
32. Yun, S., Rus, D.: Optimal self-assembly of modular manipulators with active and passive modules. Auton. Robot. **31**(2–3), 183–207 (2011)

Slip Detection in a Novel Tactile Force Sensor

Raul Fernandez, Ismael Payo, Andres S. Vazquez and Jonathan Becedas

Abstract Tactile sensing improves the manipulation and grasping of unknown objects. It contributes to increase the knowledge of the environment and provides useful information to improve grasping control. The sensors traditionally used for tactile sensing emphasize in grasping object shape and force detection. However slip detection is also crucial to successfully manipulate an object. Several approaches have appeared to detect slipping, the majority being a combination of complex sensors with complex algorithms. In this paper, we present a simple, low cost and durable tactile force sensor and its use to slip detection via a simple but effective method based on micro-vibration detection. We also analyze the results of using the same principle to detect slip in other force sensors based on flexible parts. In particular, we also show the slip detection with: a flexible finger (designed by the authors) acting as a force sensor, the finger torque sensor of a commercial robotic hand (Barrett Hand), and a commercial 6-axis force sensor mounted in the wrist of a robot.

1 Introduction

Tactile sensing has had increasing interest since the 1980s [15], mainly in the fields of robotics and automation, with special interest on manipulation and grasping [4, 18] although with contributions in other fields like virtual reality and medicine, among others [3, 17].

R. Fernandez (✉) · I. Payo · A.S. Vazquez · J. Becedas
Department of Electrical, Electronics and Control Engineering,
University of Castilla-La Mancha, Ciudad Real, Spain
e-mail: raul.fernandez@uclm.es

I. Payo
e-mail: ismael.payo@uclm.es

A.S. Vazquez
e-mail: andress.vazquez@uclm.es

J. Becedas
e-mail: jonathan.becedas@uclm.es

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts
in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_14

Tactile sensing has traditionally emphasized in detecting contact forces and shape of a grasping object. However there are other aspects that contribute to increase the knowledge of the contact and improve the manipulation and grasping of an a priori unknown object. On line slip detection highly contributes to increase that knowledge and to improve manipulation [13]. Acquiring on line knowledge during manipulation is of special relevance if the environment in which these tasks are executed presents uncertainty [7].

Thus, avoiding slipping is crucial to successfully manipulate an object. However, specific tactics to detect and avoid the slip has not been extensively used, being the studies more focused in applying a sufficiently high and previously known force to avoid it [2]. Nevertheless, an early detection of slipping can contribute to control the contact forces to prevent it and successfully manipulate an uncertain object.

Several sensors have been tested in laboratory to detect slipping. In [10] two main approaches/types of sensors were defined: (i) Calculation of the friction coefficient using non dedicated and non specifically designed sensors. In this case, the detection of slipping is not accurate since the sensors indirectly measure the relative movement between the grasping surfaces. (ii) The design and construction of specific or dedicated sensors. There are two solutions: complex sensors designed for manipulation and grasping and a combination of different sensors (e.g. force/torque sensors combined with tactile arrays). These solutions increase the price and complexity of the system.

In the scientific literature we found different techniques to detect slip. In [13, 19], slip was detected in robotic fingers with skin acceleration sensors specially designed. The fingers were covered with foam rubber, and this covered with rubber skin. Between the foam and the rubber accelerometers were strategically disposed to measure the accelerations that the skin suffered when it was in contact with a slipping surface. They observed that low amplitude, high frequency vibrations were produced in the finger skin when the contact surface was slipping. In addition, in [19] a grasping force control to avoid slipping based in the estimation of the surfaces friction coefficients was implemented. However, accelerometers were really noisy and affected the measurements of small amplitude signals, so there was a systematic threshold under which the slipping could not be detected. Furthermore those sensors could not measure the location of the contact point, which is useful to grasp unknown objects. In [10] a designed 16×16 matrix of conductive rubber was used. Slip was detected by interpreting the deformation of the contact surface based on the change of the center of mass of the contact surface and by measuring the vibration produced in the skin when slip occurred. The resolution to localize the object was poor. In [14] a tactile array sensor was combined with a dynamic sensor to accurately detect the location of the object and the slipping effect by measuring the micro-vibrations caused by slipping. Other approaches use combinations of sensors to detect slip such as vision combine with tactile sensors [11] or silicon rubber based touch sensor combined with force sensors [12]. In [8] a conductive film sensor was used to detect slipping by studying the pressure load of the contact. In [16] a combination of tactile array sensors and force/torque sensors based on strain gauges was used for slip detection.

In consequence the slipping can be detected in three manners: (i) estimating the friction coefficient between the grasping surfaces, which is a not accurate indirect measurement. (ii) Analyzing the changes in the contact footprint over a tactile array sensor: changes in the shape of the contact with the object and changes in the pressure-force distribution. This strategy is not useful if the object is not rigid. (iii) Detecting micro-vibrations. This is a direct and reliable measure of the slipping, but of dynamic nature, so the dynamic sensors traditionally used, as accelerometers, are noisy and usually detect the slipping too late to respond to it and prevent it with a closed loop control law.

We have designed a simple non dedicated and low-cost tactile force sensor based in strain gauges to be used in different applications and be incorporated in robotic grippers. The sensor proposed is based on the strategically location of strain gauges in a elastic concentrator covered by a semi-cylindrical metallic capsule. Thus, the sensor plate is separated from the contact surface. This contributes to increase the life of the sensor and protect it from accidental and dangerous contacts (high temperature surfaces, accidental hit and cutting surfaces among others, in contrast to rubber or foam made surfaces).

The sensor was geometrically designed to detect the contact location with high accuracy. The strain gauges were disposed to be sensitive to the micro-vibrations produced in the contact surfaces when slip occurs. Strain gauges also present other advantages: high load range, high precision, high resolution, low hysteresis, high sensitivity and linear response (see [6]).

The paper is structured into 6 sections: Sect. 2 deals with the description of the tactile sensor; In Sect. 3 we present our slip detection algorithm; In Sect. 4 some experimental tests are shown; In Sect. 5 we present the slip detection with other strain-gauge based force sensors; And finally, in Sect. 6 the main conclusions of this work are summarized.

2 Description of the Tactile Sensor

The tactile sensor is composed of two structural parts as shown in Fig. 1a: (1) passive part (aluminium cover with semi-cylindrical shape) which comes into contact with the environment and (2) sensed part (elastic cylindrical beam of methacrylate) where some strain gauges are attached in strategic positions. The methacrylate beam is clamped both ends to the robotic fingertip. Figure 1b shows the sensor mounted in the fingertip of a Barret Hand and Fig. 1c shows a cross section of the sensor and their dimensions.

In the sensor model is assumed that the force F is applied in the plane Q (see Fig. 2a). The cover is clamped to the elastic beam at its midpoint as shown in Fig. 1c and therefore the applied force is transmitted to the elastic beam through this point. The applied force F can be decomposed into normal and tangential forces (F_n and F_t) relatives to the surface of the metallic cover as shown in Fig. 2b. In addition, the

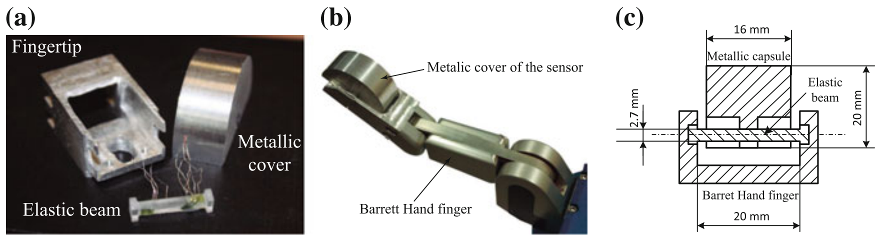
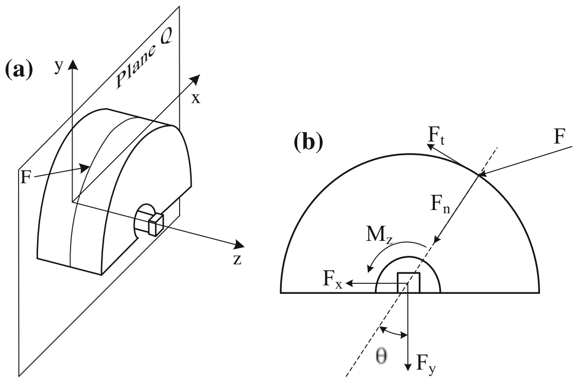


Fig. 1 **a** Sensor parts. **b** Sensor mounted on a BarrettHand finger. **c** Cross section of the sensor

Fig. 2 **a** Applied force. **b** Decomposition of force *F*



normal force can be decomposed into Cartesian coordinates relatives to the axis (X, Y) associated to the elastic beam as shown in Fig. 2b. The magnitude and the direction of the applied normal force F_n can be calculated with the following equations.

$$|F_n| = \sqrt{F_x^2 + F_y^2}; \qquad \theta = tg^{-1}\left(\frac{F_x}{F_y}\right) \tag{1}$$

Note that angle θ indicates the contact location. On the other hand, the tangential force generates a moment on the Z axis of the elastic beam. The magnitude of this force can be calculated as $F_t = M_z/r$ where r is the radio of the semi-cylindrical metallic cover.

Figure 3 shows the bending moments (M_x, M_y) and the torsional moment M_z transmitted to the elastic beam, caused respectively by F_x, F_y and F_t. These transmitted moments can be measured by using strain gauges placed strategically on the elastic beam. Figure 4 shows the location of the strain gauges used to measure each transmitted moment. Bending moments M_x and M_y are measured by two pairs of gauges (1–2 and 3–4, respectively) placed in opposition at the ends of the beam and the torsional moment M_z is measured by two gauges (5–6) placed to 45° with respect to the axial axis of the beam. With these configurations and an appropriate electrical connection to a Wheatstone bridge (2–gauge system), superimposed effects can be cancelled, measuring only the desired variable [9]. Therefore, the linear relation

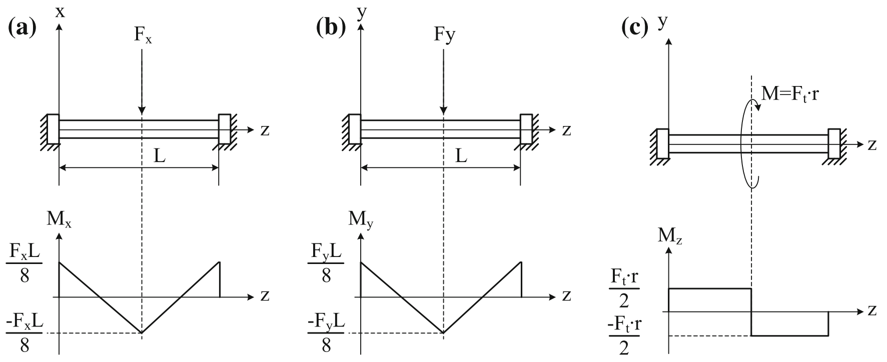
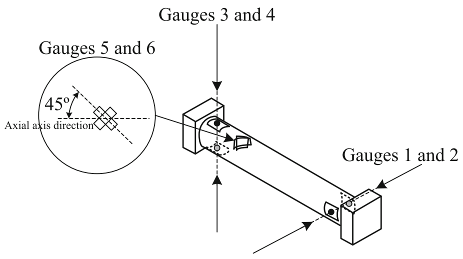


Fig. 3 **a** Bending moment M_x. **b** Bending moment M_y. **c** Torsional moment M_z

Fig. 4 Location of the strain gauges. Gauges 1 and 2 are used to measure M_x, gauges 3 and 4 are used to measure M_y and gauges 5 and 6 are used to measure M_z



between the gauge voltage signals and the applied forces can be finally expressed by the following three equations.

$$V_{g12} = k_x\,F_x; \quad V_{g34} = k_y\,F_y; \quad V_{g56} = k_t\,F_t \tag{2}$$

where k_x, k_y and k_t are constants that must be calibrated and determine the force sensor sensitivity (see Sect. 4.2).

3 Slip Detection Algorithm

When the surfaces of two objects come into slipping contact some structural vibrations of high frequencies arise in the objects. It has been tested in previous works (e.g. [10]) and it will also be demonstrate in the present study.

In order to detect a slipping contact in real time we have used an algorithm based in the discrete Fourier transform (DFT) of the strain gauges signals of the tactile sensor. This method allows to detect structural micro-vibrations of high frequencies in real time and therefore slipping contacts.

It is well known that the DFT of a signal φ is defined by the following equation.

$$\Phi(k) = \sum_{n=0}^{N-1} \varphi(n) e^{-i2\pi k \frac{n}{N}}; \quad k = 0, \ldots, N-1. \tag{3}$$

where N is the number of samples used. Instead of evaluating Eq. (3) directly, which requires $O(N^2)$ operations (O denotes an upper bound), we have used the Colley-Turkey algorithm [5]. This algorithm is a fast Fourier transform (FFT) that computes the same results in $O(N \log N)$ operations (lower computational cost).

The power spectrum of Φ can be calculated as

$$Pw(k) = \frac{\Phi(k)\, \Phi^*(k)}{N} \tag{4}$$

where $*$ denote the conjugate.

The detection method is summarized as follows: (1) the FFT of the signal Φ is computed each N samples eliminating the DC component; (2) the power spectrum of Φ is calculated in every iteration, obtaining the peak value and its frequency; (3) the peak value of the power spectrum is multiplied by its frequency; (4) finally, a slipping contact is detected when this later value exceeds a threshold defined empirically.

In order to detect slipping contact (associated to microvibration of high frequencies) in real time, the sampling frequency was chosen to be $f_s = 1\,\text{kHz}$ and the FFT was computed with the last $N = 64$ samples of the strain gauges signals. This means that the time needed to detect slipping is $N/f_s = 64\,\text{ms}$.

4 Experimental Results

The purpose of this section is the sensor calibration and to testing its behavior in order to measure forces, locate the contact points and detect slipping. First of all a brief description of the setup used for performing all experiments is introduced.

4.1 Experimental Setup

The force sensor was mounted on the fingertips of a robotic hand (BarretHand). The hand was attached to a 6DOF manipulator (model Stäubli RX-90). This manipulator was used to move the sensor toward an object and to apply a certain force. The motion control of this manipulator is not explained here because it is not the objective of this study. The strain gauges used (model Kyowa KFG) have a resistance of 120 ± 0.2 Ω and a gauge factor of 2.24 ± 1 and their dimensions are approximately 1 mm square grid. Commercial strain gauge amplifiers (model Vishay BA660) were used for conditioning the gauges signals. All gauges signals were read with a computer by means of a commercial data acquisition card (model NI PCIe-6363) and were processed in real time with the commercial data logging application LabView. The sampling frequency for real time data acquisition tasks was chosen to be 1 kHz.

4.2 Calibration of the Sensor

The three gauge signals of the sensor were calibrated by using the setup shown in Fig. 5. Forces F_x, F_y and F_t were applied on the sensor by using calibrated weights.

As said in Sect. 2, strain gauges 1 and 2 measure the applied force F_x. Figure 6a shows the voltage given by these gauges for different values of the applied force F_x. Several experiments were done for each value of the applied force. The range of the voltage values obtained are represented with vertical lines. A straight line with the independent term equal to zero has been fitted to the data by means of the square minimum method, obtaining the calibration value: $V_{g12} = 0.0694F_x$

The other two gauge signals which measure the forces F_y and F_t were calibrated following the same previous procedure. Figures 6b and 6c show, respectively, the voltage given by the gauges 3–4 and the gauges 5–6 for different values of the applied

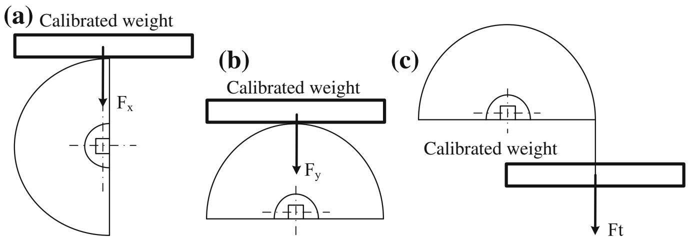


Fig. 5 Scheme of the experiments carried out to calibrate the gauge signals: **a** setup configuration to calibrate the gauge signal that measures the force F_x; **b** setup configuration to calibrate the gauge signal that measures the force F_y; **c** setup configuration to calibrate the gauge signal that measures the force F_t

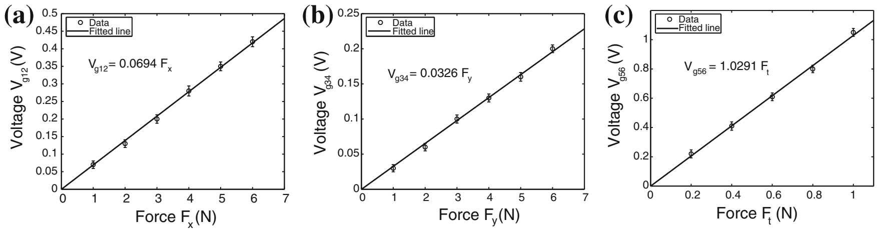


Fig. 6 Calibration of the gauge signal used to measure the force F_t

forces F_y and F_t. Straight lines was fitted in each case, obtaining the calibration values: $V_{g34} = 0.0326F_y$; $V_{g56} = 1.0291F_t$.

From the results it can be observed the different sensitivity of the signals. While the sensitivity of the signals V_{g12} and V_{g34} are of the same order, the sensitivity of the signal V_{g56} is significantly higher than previous. This is mainly caused by small misalignments of the strain gauges, but it is not a critical issue that prevents the operation of the sensor as it is solved by the calibration.

The resolution is defined as the smallest change detected in the measured variable. For this force sensor the resolution was 0.12 N for F_x, F_y and 0.01 N for F_t

4.3 Slipping Test

The objective of the test was to demonstrate the effectiveness of the tactile force sensor to measure the contact forces (F_n, F_t), locating the contact points, and detecting slipping. A steel table (flat surface) was chosen to be the contact object. The test was performed according to the following sequence: (1) the fingertip was moved toward the table until the sensor came into contact (2) the finger was rotated, allowing the semi-cylindrical surface of the sensor to roll on the table (rolling contact; (3) The finger was linearly moved producing the tactile sensor to slip on the surface of the table (slipping contact); (4) the finger was stopped. Figure 7 illustrates the aforementioned sequence.

Figure 8a shows the forces F_x, F_y and F_t, measured by the strain gauges during the experiment, and Fig. 8b shows the magnitude and the direction of the contact force F_n calculated according to Eq. (1). It is seen that the sensor came into contact with the table at $t = 2.43$ s. From $t = 2.43$ s to $t = 6.20$ s the cylindrical surface of the sensor rolled on the table. From $t = 6.20$ s to $t = 9.8$ s a slipping contact occurred. Finally, the finger stopped its movement at $t = 10$ s. Structural micro-vibrations of high frequencies caused by the slipping contact were detected better,

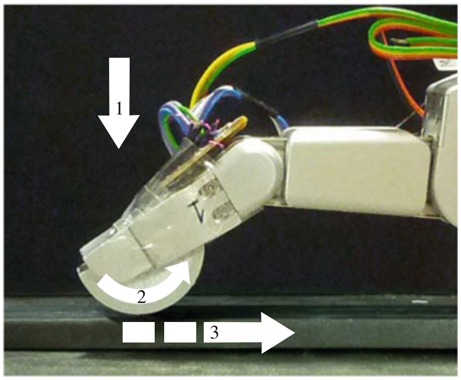


Fig. 7 Experimental setup with sensor

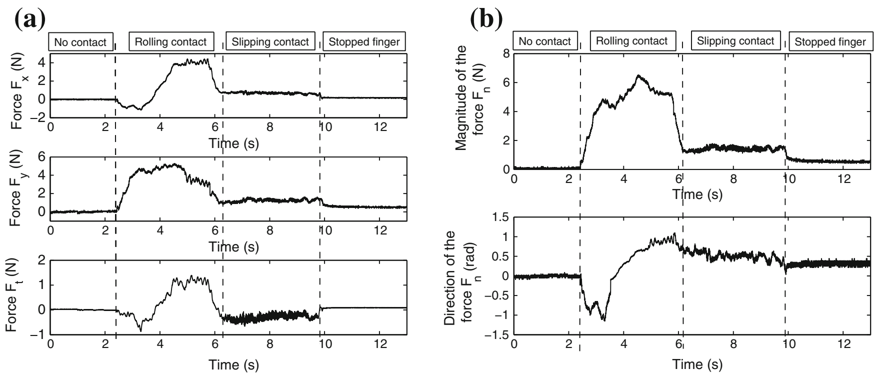


Fig. 8 **a** Forces F_x, F_y and F_t as a function of the time. **b** Magnitude and direction of the force F_n

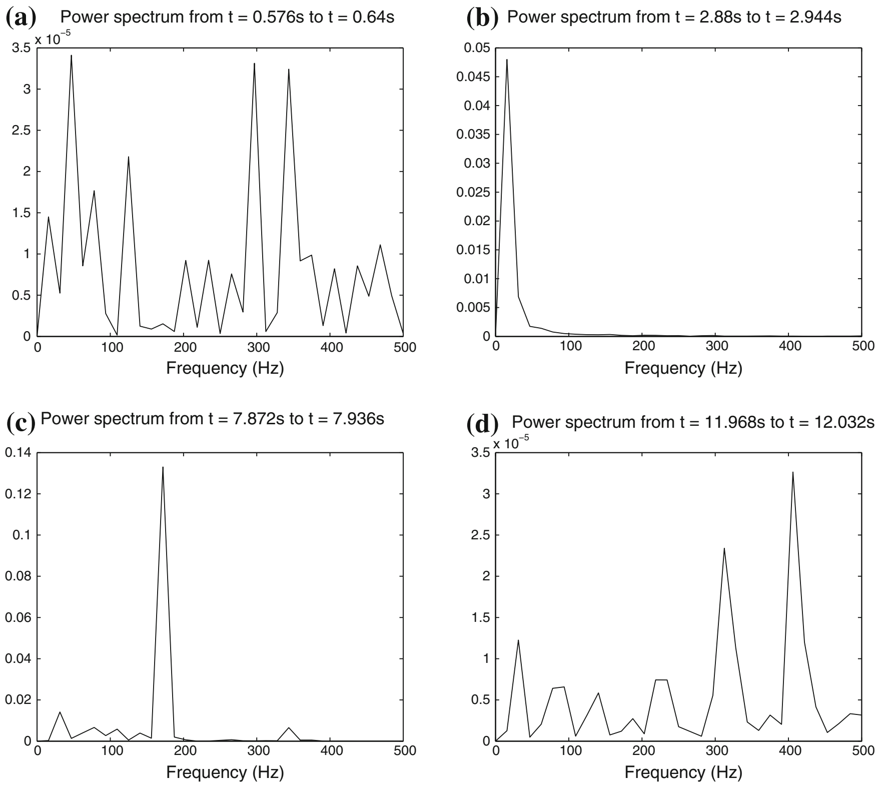


Fig. 9 Power spectrum: **a** before contact; **b** rolling contact; **c** slipping contact and **d** stopped sensor

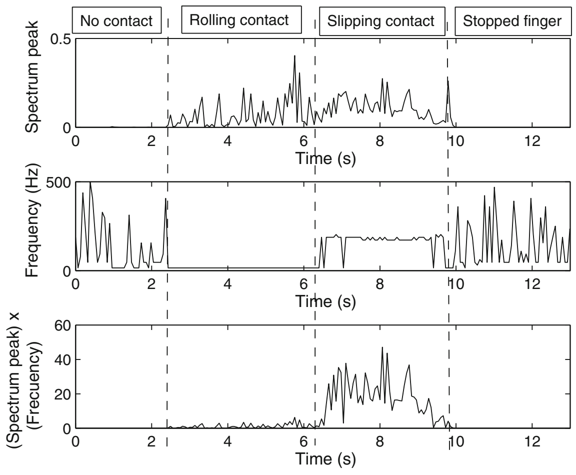
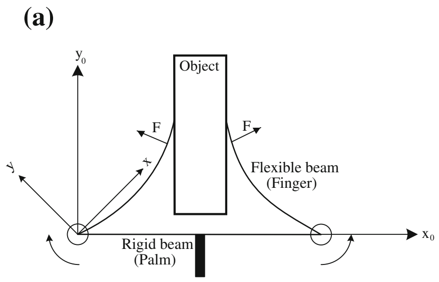


Fig. 10 Spectrum peak and its frequency as a function of the time

using the algorithm described in Sect. 3, in the signal which measured the tangential force F_t (this fact is plausible if we take into account that F_t is a friction force in this particular case). Figure 9 shows the power spectrum at four different instants of the signal which measured F_t: (a) before contact; (b) rolling contact; (c) slipping contact; (d) stopped finger. These spectrums represent the frequency content of this signal up to the Nyquist frequency. Figure 10 (top) and (middle) show, respectively, the peak value of the power spectrum and its frequency at each iteration ($N = 64$ samples, i.e. each 64 ms) and Fig. 10 (bottom) shows the product of both signals. It is observed that the slipping contact can be detected when this product exceeds a certain threshold (after performing many experiments this threshold was chosen to be 10). Note that high frequencies caused by noise in the signal (from $t = 0$ s to $t = 2$ s and from $t = 10$ s to $t = 13$ s as shown in Fig. 10 (middle)) and high spectrum peaks caused during the rolling contact as shown in Fig. 10 (top), are ignored by this method, detecting high frequencies associated to high spectrum peaks, two conditions which only appear with slipping contacts. Note also that the slip vibration frequency is about 180 Hz.

5 Slipping Detection with Other Strain-Gauge Based Force Sensors

In addition to previous analysis, in this section we analyze if the same slip detection procedure can be applied to other strain-gauge based force sensors. In particular we have tested 3 different sensors systems based on structural deformation.



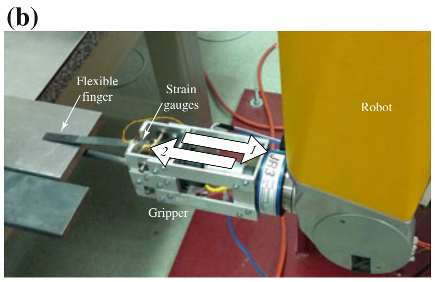


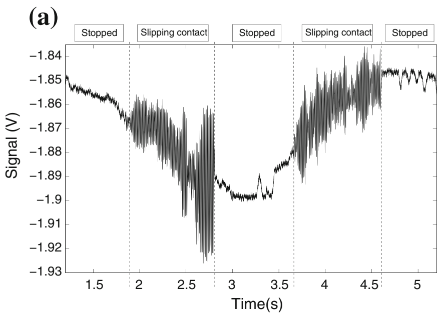
Fig. 11 **a** Gripper scheme. **b** Experimental setup with flexible gripper

5.1 *Flexible Finger Gripper Acting as a Force Sensor*

The type of grippers analyzed in this work is an underactuated mechanism constituted by rigid parts (palm) and flexible parts (flexible fingers) as shown in Fig. 11a. The flexible finger of the gripper shown in Fig. 11b can be used as force sensors [2]. A pair of strain gauge is placed in opposition on the base of each elastic finger as shown in Fig. 11b, each pair is connected to a Wheatstone bridge that is wired to a Vishay BA660 strain gauges amplifier to measure the deflection and therefore the applied force.

5.1.1 Experimentation and Discussion

The gripper was used as an end effector of a Stäubli RX90 robot as shown Fig. 11b. The contacted object was chosen to be a steel plate (flat surface) as shown in Fig. 11b. The test was performed according to the following sequence: (1) the fingers were slipped on the surface of the plate; (2) the fingers were slipped on the opposite direction. Figure 12 shows the gauge signal of one of the fingers, measured by the



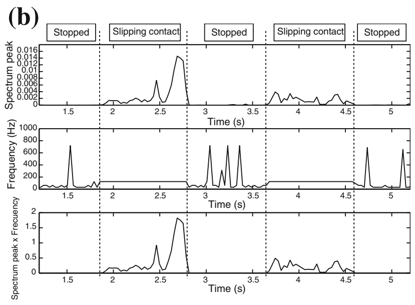


Fig. 12 **a** Gauge voltage Finger 1 as a function of the time. **b** Spectrum peak and its frequency as a function of the time (gripper)

strain gauges during the experiment. It is seen that the fingers slip in the plate from $t = 3.24$ s to $t = 4.65$ s. From $t = 4.65$ s to $t = 5.01$ s the gripper kept still. From $t = 5.01$ s to $t = 6.06$ s the fingers slip in opposite direction. Finally, the gripper was stopped at $t = 6.06\ s$.

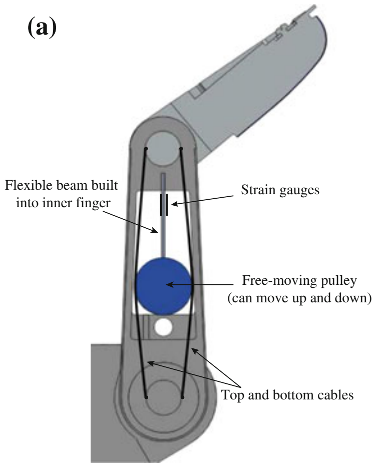
Based on the algorithm described in Sect. 3, we looked for high frequency vibrations. After many experiments the threshold to detect slipping was chosen to be 0.25. Structural microvibrations of high frequencies caused by the slipping contact were detected as can be seen in Fig. 12b. In this system the frequency of slipping vibration was detected at 125 Hz. We can conclude that the method proposed in Sect. 3 works for this sensor properly.

5.2 *Barrett Hand Torque Finger Sensor*

Each finger of the BarrettHand was mounted with a force sensing mechanism consisting of a flexible beam, a free-moving pulley, a pair of cables, and two strain gauges as it is shown Fig. 13a [1]. Basically, when a force is applied to the last phalange of the finger, the cables get tight which moves the pulley, bending the flexible beam built into inner finger. This deformation is measured by the strain gauges.

5.2.1 Experimentation and Discussion

The BarrettHand was used as an end effector of a Stäubli RX90 robot for these experiments. In the same way that in previous experiment, the contacted object was


(a)
Flexible beam built into inner finger
Strain gauges
Free-moving pulley (can move up and down)
Top and bottom cables

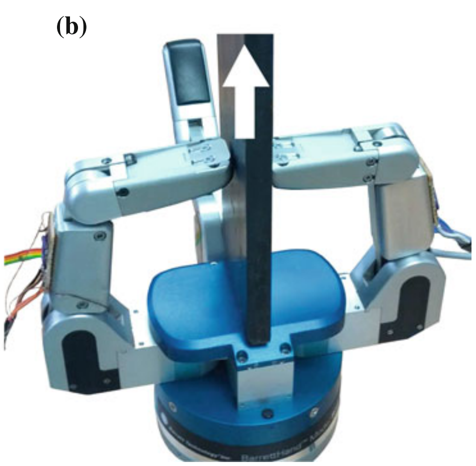


Fig. 13 **a** Cutaway diagram of finger reveals internal strain gages on BarrettHand. **b** Experimental setup with BarrettHand

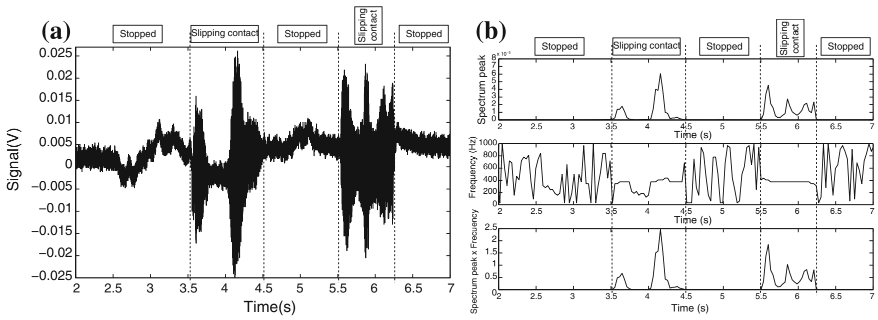


Fig. 14 **a** Gauge signal Finger 1 as a function of the time. **b** Spectrum peak and its frequency as a function of the time (BarrettHand)

chosen to be a steel plate as shown in Fig. 13b. The test was performed according to the following sequence: (1) the BarrettHand grasped the steel plate with two fingers; (2) the fingers were slipped on the surface of the plate until they lost contact with the object. Figure 14a shows the gauge signal of one finger during the experiment. It is seen that the fingers slipped on the plate from $t = 5.50$ s to $t = 6.25$ s.

Structural microvibrations of high frequencies caused by the slipping contact was detected with this sensor as we can see in Fig. 14b. In the Barrett system the frequency of slipping vibration was detected at 375 Hz. The frequency graph of Fig. 14b shows that this system have a lot high frequency noise, however the detection method still works properly.

5.3 *JR3 6-Axis Force/Torque Sensor*

Manipulator robots usually mount multi-axis force/torque sensors on the wrist that are very useful in manipulation tasks. Usually, these sensors mount, like in previously described sensors, strain gauges to measure deformations in order to calculate the loads applied. We have used in this work the model 67M25A3 of JR3 (Fig. 15) which

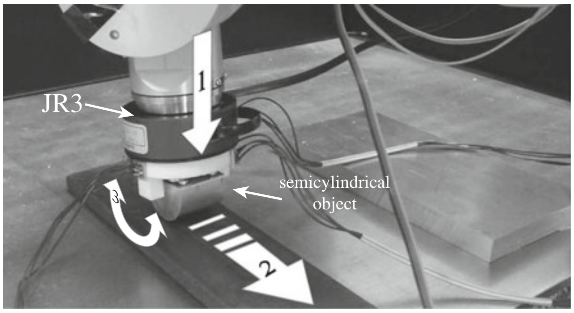


Fig. 15 Experimental setup with JR3

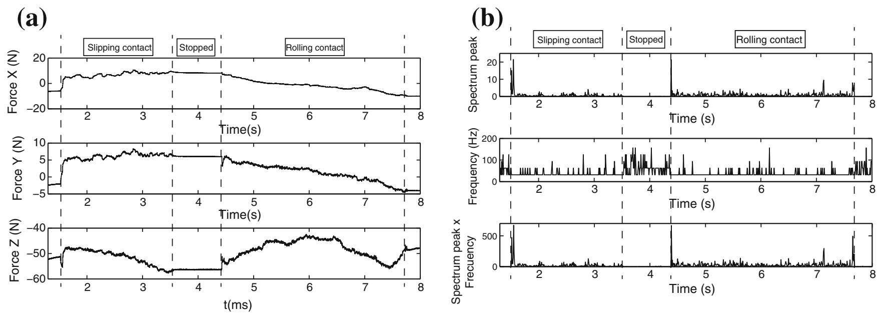


Fig. 16 **a** Forces in sensor JR3 as a function of the time. **b** Spectrum peak and its frequency on F_z as a function of the time

is able to measure in Fx and Fy with a standard measurement range of $100N$ and a digital resolution $0, 025N$ and in Fz with a standard measurement range of $200N$ and a digital resolution of $0.05N$.

5.3.1 Experimentation and Discussion

This sensor was mounted on the wrist of a Stäubli RX90 robot. As shown in Fig. 15, it was necessary to add a semicylindrical object as a end-effector to perform the following experiment: (1) the semicylindrical object was moved toward the table until it came into contact with it; (2) the object was slipped on the surface of the table (slipping contact); 3) the robot rolled the semi-cylindrical surface of the sensor on the table (rolling contact). Figure 16a. shows the forces F_x, F_y and F_z, measured during the experiment. It is seen that the object came into contact with the table at $t = 0.82$ s. From $t = 1.53$ s to $t = 3.55$ s a slipping contact occurred. From $t = 4.43$ s to $t = 7.75$ s the cylindrical surface of the object rolled on the table. Finally, the robot stopped its movement at $t = 7.75$ s.

Figure 16b shows the spectrum analysis of signal F_z, which, based on its magnitude, was considered as the best one to detect slipping. However, structural microvibrations of high frequencies were not detected. Basically, this is because of the relative amplitude and frequencies of the vibration modes when a slip occurs, which is not different in the rolling contact case. We have concluded that this is due to the high rigidity of this type of sensors.

6 Conclusions

We have presented a new force tactile sensor based on resistive strain gauges. The sensor can be mounted in the fingertips of a robotic hand easily, and is simple, durable and low cost. Its usefulness for measuring and locating forces has been confirmed with experimental tests in Sect. 4.2. We have also presented a simple algorithm, based on DFT, that allow us to detect slipping with our sensor in 64 ms as presented in Sect. 4.3

In addition, we have analyzed the usefulness of our method in other sensors based on strain gauges. We have concluded that the algorithm works properly in two of the three sensors analyzed. As a consequence of this study we can say that the slipping detection algorithm does not work properly for sensor which rigidity is high, because the amplitude and frequencies of their vibration modes cannot be distinguishable in slipping or rolling contacts.

Acknowledgments This work was supported by the Consejería de Ciencia y Tecnología of the Junta de Comunidades de Castilla-La Mancha, Spain, under Contract PPIC10-0202-8377, and by the European Social Fund.

References

1. Allen, P.K., Miller, A.T., Oh, P.Y., Leibowitz, B.S.: Integration of vision, force and tactile sensing for grasping. In Int. J. Intelligent Machines, Citeseer (1999)
2. Becedas, J., Payo, I., Feliu, V.: Two-flexible-fingers gripper force feedback control system for its application as endeffector on a 6 dof manipulator. IEEE Trans. Robot. **27**(3), 599–615 (2011)
3. Burdea, G.: Force and Touch Feedback for Virtual Reality. Wiley, New York (1996)
4. Cannata, G., Maggiali, M.: An embedded tactile and force sensor for robotic manipulation and grasping. In: 2005 5th IEEE-RAS International Conference on Humanoid Robots, pp. 80–85, (2005)
5. Cooley, J., Turkey, J.: An algorithm for the machine calculation of complex fourier series. Math. Comput. **19**, 297–301 (1965)
6. da Silva, J., de Carvallo, A., da Silva, D.: A strain gauge tactile sensor for finger-mounted applications. IEEE Trans. Instrum. Meas. **51**(1), 18–22 (2002)
7. Dollar, A., Jentoft, L., Gao, J., Howe, R.: Contact sensing and grasping performance of compliant hands. Automous Robot. **28**, 65–75 (2010)
8. Gunji, D., et al.: Grasping force control of multi-fingered robot hand based on slip detection using tactile sensor. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2605–2610 (2008)
9. Hoffmann, K.: Applying the wheatstone bridge circuit. Hottinger Baldwin Messtechnik GmbH (2001)
10. Holweg, E., et al.: Slip detection by tactile sensors: algorithms and experimental results. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 3234–3239 (1996)
11. Hosoda, K., Tada, Y., Asada, M.: Internal representation of slip for a soft finger with vision and tactile sensors. In: Proceedings of the International Conference on Intelligent Robots and Systems, pp. 111–115 (2002)
12. Kawamura, T., et al.: Measurement of slip, force and deformation using hybrid tactile sensor system for robot hand gripping an object. Int. J. Adv. Robot. Syst. **10**(83), 1–8 (2012)
13. Kawasaki, H., Komatsu, T., Uchiyama, K.: Dexterous anthropomorphic robot hand with distributed tactile sensor: Gifu hand ii. IEEE Trans. Mechatron. **7**(3), 296–303 (2002)
14. Lazzarini, R., Magni, R., Dario, P.: A tactile array sensor layered in an artificial skin. Proceedings of the IEEE International Conference on Intelligent Robots and Systems **3**, 114–119 (1995)
15. Lee, M., Nicholls, H.: Tactile sensing for mechatronics - a state of the art survey. Mechatronics **9**, 1–31 (1999)

16. Melchiorri, C.: Slip detection and control using tactile and force sensors. IEEE Trans. Mechatron. **5**(3), 235–243 (2000)
17. Sokhanvar, S., Packirisamy, M., Dargahi, J.: A multifunctional pvdf-based tactile sensor for minimally invasive surgery. Smart Mater. Struct. **16**, 989–998 (2007)
18. Tegin, J., Wikander, J.: Tactile sensing in intelligent robotic manipulation—a review. Ind. Robot. **32**, 64–70 (2005)
19. Tremblay, M., Cutkosky, M.: Estimating friction using incipient slip sensing during a manipulation task. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 429–434 (1993)

Concentric Tube Robots: The State of the Art and Future Directions

Hunter B. Gilbert, D. Caleb Rucker and Robert J. Webster III

Abstract Seven years ago, concentric tube robots were essentially unknown in robotics, yet today one would be hard pressed to find a major medical robotics forum that does not include several presentations on them. Indeed, we now stand at a noteworthy moment in the history of these robots. The recent maturation of foundational models has created new opportunities for research in control, sensing, planning, design, and applications, which are attracting an increasing number of robotics researchers with diverse interests. The purpose of this review is to facilitate the continued growth of the subfield by describing the state of the art in concentric tube robot research. We begin with current and proposed applications for these robots and then trace their origins (some aspects of which date back to 1985), before proceeding to describe the state of the art in terms of modeling, control, sensing, and design. The paper concludes with forward-looking perspectives, noting that concentric tube robots provide rich opportunities for further research, yet simultaneously appear poised to become viable commercial devices in the near future.

1 Introduction

Concentric tube robots, also known as active cannulas, are one of the smallest members of the broader family of continuum (i.e. continuously flexible) robots [58, 75, 79]. They are made from several tubes that are nested within one another concentrically (Fig. 1). These tubes are precurved and made of elastic material (usually superelastic nitinol). When the tubes are grasped at their respective bases, and linear insertion/retraction and axial rotation motions are applied, they interact elastically

H.B. Gilbert (✉) · R.J. Webster III
Vanderbilt University, Nashville, TN, USA
e-mail: hunter.b.gilbert@vanderbilt.edu

R.J. Webster III
e-mail: robert.webster@vanderbilt.edu

D.C. Rucker
University of Tennessee, Knoxville, TN, USA
e-mail: caleb.rucker@utk.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_15

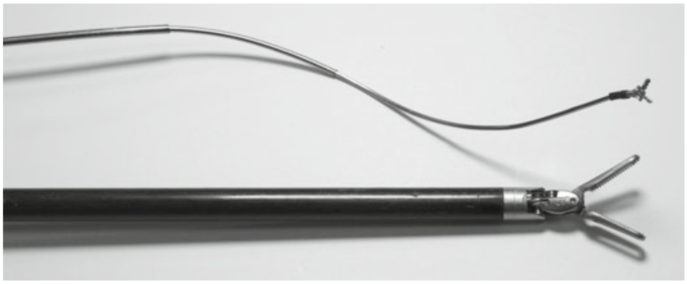


Fig. 1 A concentric tube robot next to a standard da Vinci laparoscopic tool

and make one another bend and twist. The net result is a needle-sized robot that can elongate and bend in a manner that has been likened to a miniature tentacle.

While it is possible that applications will be developed outside medicine in the future, to date the motivating applications for concentric tube robots have come exclusively from surgery and interventional medicine, and two distinct methods of use have been identified. The robot can act as a steerable needle or be used as a miniature teleoperated manipulator. In both contexts, the robot can enter the body in a variety of ways, including through the skin, through the vascular system, through a natural orifice, or through the ports in a rigid or flexible endoscope that is itself inserted into the body. In the trans-endoscope embodiment, concentric tube robots have been proposed for use in neurosurgery [16], transoral throat surgery [78], transoral lung biopsy and therapy delivery [45, 82], and transgastric surgery [81]. In the transvascular embodiment, concentric tube robots have been proposed for a variety of intracardiac procedures where they enter the heart through the vascular system [6, 30, 31, 76]. In the natural orifice embodiment, transnasal skull base [13] and transoral throat [78] applications have been proposed, and it is likely that surgeries through other natural orifices will be pursued in the future. In the percutaneous, needle-like embodiment, applications that have been suggested include fetal umbilical cord blood sampling [28], ultrasound guided liver targeting and vein cannulation [72], vascular graft placement for hemodialysis [7], thermal ablation of cancer [8, 11], prostate brachytherapy [77], retinal vein cannulation [85, 86, 89], epilepsy treatments [18], and general soft tissue targeting procedures [33, 43, 68].

Of all these applications, the two that have been studied most extensively are the cardiac applications of Dupont et al. and the endonasal applications of Webster et al. This includes the first ever use of a concentric tube robot in a live animal by Gosline et al. [26, 30]. It also includes the first insertion of a concentric tube robot into a human cadaver by Burgner et al. [10, 13]. Many researchers have also explored the use of concentric tube robots as steerable needles in a variety of phantom and ex vivo tissues, as discussed in the following subsection.

Use as Steerable Needles When cast as steerable needles, there are several ways concentric precurved tubes can be used. The term "steerable needle" typically refers to devices that harness tip-tissue interaction forces to steer [2, 55, 80]. Consistent with this, Salcudean et al. demonstrated a concentric tube design in which a small section of a circularly curved wire extends from an outer tube [52]. In this steering paradigm, as the needle is inserted into tissue with the wire held at a fixed angle and distance of deployment, tip-tissue interaction forces will cause the shaft of the needle to bend. Changes in the distance of wire deployment and the axial wire angle control the curvature and direction of bending. Loser adopted a different approach where needle shaft curvature could be controlled independently of needle insertion via two fully overlapping precurved tubes which could rotate with respect to one another [43]. With such a needle, a mid-insertion curvature change would cause forces to be applied to tissue along the entire needle shaft, deforming tissue in order to aim the needle towards the desired target.

In many steerable needle contexts, it is desirable to apply minimal deformation to tissue, and one wishes to maintain the needle's shaft exactly along the curved trajectory through which the tip has traveled. This is referred to as "follow-the-leader" insertion [29]. Early work, which neglected tube elastic interaction, implicitly assumed that concentric tube robots would automatically deploy in this manner [27]. However, the accumulation of experimental results and modeling advances soon showed that tube elastic interaction is typically significant. It also showed, perhaps counter intuitively, that concentric circularly precurved tubes do not achieve a circular conformation when axially rotated (see e.g. [66]). Both these factors make follow-the-leader deployment more challenging than it might at first seem.

However, a useful simple special case that can deploy in an exact follow-the-leader manner was identified early in the history of concentric tube robots. It consists of a device in which a circularly curved inner tube or wire extends from a straight outer tube. The earliest recorded use of this concept in a needle of which the authors are aware was in 1985 when the Mammalok product came to market [67], and the same basic concept has been employed many times since (e.g. [7, 8, 14, 21, 33, 51, 68, 72], among others). It is now known that both circular and helical tubes can deploy in a perfect follow the leader manner, with proper precurvature and actuation [29]. Although some special cases have been observed, the extent to which such results can be generalized beyond two tubes remains an open question.

Overall, an advantage of using concentric tube robots as steerable needles in comparison to most other kinds of steerable needles is that concentric tube robots do not rely on tissue forces to steer, meaning that their mechanical properties do not need to be perfectly matched to the properties of the tissue through which they pass. Moreover, they are one of only two steerable needle technologies that can follow the leader through both open and liquid filled cavities in addition to soft tissues (the other is tendon actuation [37]).

Use as Miniature Manipulators The basic idea of using a curved nitinol tube to deflect the tip of a manual laparoscopic tool was described in several references from the early 1990s [20, 49, 50, 69]. These apparently led to the commercial Roticulator (Covidien, formerly United States Surgical Corporation), which originally used a precurved nitinol tube [50], and remains on the market today with a precurved plastic tube as the bending element. The idea of a teleoperated robotic manipulator with multiple precurved tubes was developed independently and first proposed simultaneously by Sears and Dupont and by Webster, Okamura, and Cowan in 2006 [68, 81]. In this context, the device acts as a teleoperated slave robot in a manner conceptually similar to the patient side manipulator of the da Vinci system by Intuitive Surgical, Inc. These initial papers in 2006 began a period of rapid advancement in concentric tube modeling. This laid the foundation for the research reviewed in the remainder of this paper, much of which is aimed at making concentric tube robots useful in a master-slave context.

Development History The commercial Mammalok product mentioned earlier appears to be the earliest device incorporating concentric tubes and/or wires made from precurved nitinol [23]. Introduced in 1985, it was the first commercial nitinol device used in an interventional procedure, if orthodontic arch wires are excluded. In 1992 Melzer described the use of a curved tube to deflect a manual laparoscopic tool [49], and Cuschieri and Buess described a similar idea involving a telescoping curved dissection blade [20]. In 1995 Melzer and Winkel at Daum GmbH (Schwerin, Germany) developed the SMARTGuide, which was patented in 1995 and CE marked in 1996 [21]. In 1997 Melzer described the use of the SmartGuide in image-guided interventions [51]. In 2005, Loser used two counterrotated fully overlapping curved nitinol tubes to change the curvature of a needle he applied in an image-guided surgery setting [43]. Three groups (initially unaware of one another) then began simultaneous independent development of concentric tube robots, with first publications in 2005 and 2006 [27, 68, 81]. These publications and subsequent rapid modeling progress brought concentric tube robots to the general consciousness of the surgical robotics community.

Model development began with simple models, which were continually generalized via the incorporation of additional physical effects. The simplest possible model by Furusho et al. [27, 72] considered only geometry, assuming that every tube was infinitely stiff compared to all within it. Bending mechanics was included first by Loser for two fully overlapping tubes [43], and then by Webster et al. [81, 84] and Dupont et al. [25, 68] for general collections of tubes. Torsion was included first in straight sections of the device [81, 84], and then in curved sections with circular or general tube precurvatures [24, 25, 60, 61, 66]. External loading has been incorporated by considering the robot to be a single curved rod [42, 46], and more generally by describing the relative tube rotations induced by the external loads [42, 63, 65]. The above models have been used to enable teleoperation, and form the basis for for control, design, and sensing as discussed in subsequent sections. The following section describes the latest model [42, 63] in more detail.

2 Modeling

Model Formulation While there remains some activity in modeling, researchers appear to have more or less converged on a model which leverages the theory of special Cosserat rods to describe each component tube as a continuum which undergoes bending and torsion [25, 63]. Though future developments could potentially prove otherwise, at present it appears that these models have reached a "sweet spot," striking a balance between model complexity and accuracy.

All models to date neglect the effects of shear and axial extension of the rods, which are good assumptions for thin beams like the tubes in a concentric tube robot. The basic modeling approach is to write down a Cosserat rod equation for each tube, and then enforce concentricity by requiring all tubes to conform to the same curvature as a function of arc-length, leaving them free to rotate axially with respect to each other. This results in a system of differential equations with mixed boundary conditions. The boundary conditions at the base of the robot are the axial angles of the tubes, and the boundary conditions at the tip are internal moments that vanish because there is no material beyond the tip to support them. Note that after this mechanics problem is solved to determine tube axial tube angles along the robot, one must still integrate along the robot to determine the space curve of the robot itself. This model has been derived from both Newtonian equilibrium of forces and moments [25, 63] and energy minimization [24, 66], and the two approaches have been shown to be equivalent [24, 63]. Experimental testing of the model has shown that with calibration, mean error in the prediction of tip position can be as low as 1–3 % of overall arc length [25, 42, 63, 66].

External loading has been included in this modeling framework in two ways. One method is to consider the effects of loads on the model equations directly [42, 63]. A more computationally efficient, approximate way of handling external loads is to first solve the unloaded model and then treat the robot as a single curved rod that deforms under external loads [42, 47]. This approach does not model relative tube axial rotations induced by external loads, and whether or not the loss in accuracy is significant depends on the robot design and external loading conditions.

The models have also been extended to provide the differential kinematic maps for actuation (Jacobian matrix) and external loading (compliance matrix) [64, 88]. These maps have enabled resolved-rates-style algorithms for real-time control of concentric tube robots, as discussed further in Sect. 3. Additional factors like tube tolerances and friction have been explored (see Sect. 6), though not yet integrated into the modeling framework described above. A complicating factor in use of concentric tube robots, which is captured in the above modeling framework, is the presence of multiple solutions. Many potential cannula design choices will have bifurcations where solutions appear and disappear [25, 66, 84]. Rapid "snapping" may occur when tube actuation causes the robot to transition from one solution to another. Future analysis is needed to predict when and where snapping occurs.

Model Solution In contrast to the model formulation, no consensus has yet emerged as to the best way to evaluate concentric tube robot models. The model equations for two tubes with circular precurvature have been solved analytically using elliptic integrals [25, 66]. However, no analytical solutions have yet been found for robots consisting of more than two tubes, or for precurvature that varies with arc length. Hence, model equations are typically solved numerically.

Numerical implementations have most often used a “shooting” method, which iteratively adjusts unknown state values at one end of the robot until the boundary conditions are satisfied at the other end. This procedure can be performed either base-to-tip, guessing the unknown values at the proximal end and integrating to the distal end [62], or tip-to-base [42]. It was originally assumed that group preserving integration methods were required for the geometric integration [25, 66]. However, in practice we find that integration of the rotation matrix via standard explicit Runge–Kutta methods introduces numerical error that is negligible compared to kinematic error, as one would expect from the numerical examples in [22].

Simplifications to the model such as piecewise linearization enhance the speed of solution, and sensing the unknown proximal boundary conditions with torque sensors alleviates the need for root-finding techniques [87]. This choice does not necessarily find a solution that agrees with the distal boundary conditions, but the advantages may outweigh this drawback. There are many open questions in model evaluation. These relate to determination of which numerical methods are most efficient and numerically stable, the accuracy of various approximations, and the characterization of the “snapping” behavior (also known as “bifurcation”) mentioned earlier. All of these are discussed further in Sect. 6.

3 Control

Kinematic Control The main goal of kinematic control to date has been teleoperation. Three general frameworks have been proposed for kinematic control of concentric tube robots. The first involves precomputation of the model solutions over the entire workspace via one of the methods described in the previous section. To these solutions, an approximate forward kinematics model can be fit, such as a multidimensional Fourier series which is computationally efficient to invert via numerical root finding and can be evaluated at 1000 Hz [25]. The main advantages to this method are the consistent speed, suitability for real-time inverse kinematics, and the ability to identify numerical problems with solution of the model equations offline while the device is not performing a task. One disadvantage is that this method is unable to account for concentric tube robots which exhibit multiple solutions in the forward kinematics, which reduces the possible design space. Another is that the torsional effects of external loading cannot be considered.

A second general approach involves rapid solution of the model equations and computation of the manipulator Jacobian and compliance matrices [64], which current implementations in C++ can consistently do at a rate of 200–400 Hz [13]. This differential mapping is then used to update the actuator configuration iteratively to solve the inverse kinematics [10]. The advantages to this method are the ability to control robots which exhibit multiple solution behavior, the ability to immediately control new designs without precomputation or code changes, and the ability to control robots under known external loads. The main disadvantages are the increased programming effort required for fast model solution and Jacobian computation, and the lack of guarantees that the root-finding procedures of the forward kinematics computation will converge.

The third approach incorporates extra information provided by torque sensors at the tube bases. This eliminates the unknown boundary conditions at the proximal tube ends leading to rapid stable solution of the forward kinematics and the Jacobian [87, 88]. This has the advantage of retaining the full model structure, while simplifying the calculations and eliminating concerns about forward kinematics convergence. The disadvantages are that employing force sensors complicates actuation mechanism design and that system performance depends on sensor reliability.

Stiffness Modulation Mahvash and Dupont developed a model-based stiffness modulation approach, which varies the actuator positions in response to sensed tip displacements in order to display a desired force/displacement relationship at the tip of the robot [48]. This method was implemented with the torsionless model and was experimentally evaluated for a two-tube, three degree-of-freedom concentric tube robot that was well modeled by the torsionless model. Actuator current was used to estimate forces in the stiff axial direction of the robot.

Motion Planning Optimal motion planners can enable obstacle avoidance and generate actuation sequences needed to deploy along anatomical structures or to targets. Examples of prior work include planning paths around critical brain structures [44], through tubular anatomy such as the bronchi of the lung [45], and through the passages of the nasal sinuses [73]. Some of the first planners used simplified kinematic models employing circular arcs and were based on penalty methods that convert the constrained optimization problem of avoiding obstacles while maintaining a tip location into an unconstrained optimization problem [44, 45]. A different technique, termed Rapidly-Exploring Roadmaps, was first applied with the transmissional torsion model to find optimal plans [3], and later expanded to include the fully torsionally compliant kinematic model [73]. Lastly, we note that computational design and motion planning are highly interrelated problems and many of the above motion planners may be leveraged in tube parameter design. Conversely, many of the design algorithms discussed under "tube design" in Sect. 5 may be adaptable to motion planning.

4 Sensing

Image guidance is a critical part of many surgical procedures. These include teleoperated procedures where virtual fixtures [1] are used, as well as procedures where the concentric tube robot is used as a needle. One can use the mechanics-based model described in Sect. 2 to predict where the robot will be, provided that the procedure can tolerate errors of approximately 3 % of the arc length of the robot, and loads applied to the robot (if significant) are known. However, often this will not be the case, so real-time sensing and closed loop control will be required. An example of the use of visual feedback in tip position control was the use of a closed form Jacobian derived from the transmissional torsion model in [83]. In the remainder of this section we discuss the methods that are being investigated to sense the shape of the robot for such purposes, as well as methods to estimate applied loads. Force sensing based on deflection models will be discussed further in Sect. 6.

Image-Based Shape Sensing and Guidance Typically, when concentric tube robots are used as steerable needles, a control loop must be closed using medical image information. This can be done either using images in which one can see both the concentric tube robot and the target, or by registering the robot to the patient and image space. Croom et al. used self organizing maps to reconstruct the robot curve from stereo optical images [19]. This proved computationally expensive, and Burgner et al. developed a better approach using filtering and triangulation to measure robot shape using stereo fluoroscopy images [9]. Lobaton et al. also used fluoroscopy, and optimized view angles to reduce radiation dose, using statistical techniques to combine model and image information [40]. Ren et al. used a vesselness algorithm [56] and a tubular enhanced geodesic active contour algorithm [57] with 3D ultrasound images to detect curved surgical instruments. While shape sensing algorithms have not been employed specifically in computed tomography (CT) or magnetic resonance imaging (MRI), simple concentric tube robots with a straight outer tube and curved inner tube have been used with both MRI [18, 51, 70] and CT [33] images for open-loop targeting using the forward kinematic model. Similarly, surface-registered preoperative CT images were used open loop for targeting points in an anthropomorphic liver phantom [39]. Burgner et al. also used open-loop targeting with tracked 2D ultrasound, and Terayama et al. accomplished a similar objective by physically attaching the robot to the ultrasound probe [72]. As can be seen from all of the above references, substantial progress has been made toward use of concentric tube robots as image-guided steerable needles.

Magnetic and Fiber Optic Shape Sensing Standard, off-the-shelf magnetic tracking systems can be used for tip pose sensing, and also in principle to provide the pose at discrete points along the robot. These have been used by Mahvash and Dupont for stiffness modulation [47, 48], by Burgner et al. for image guidance [13], and by Xu et al. for model validation and evaluation of tracking performance [87, 88]. In principle, such sensing could be used in conjunction with the robot model to estimate the entire curve of the robot. Furthermore, given the recent interest in the surgical robotics community in fiber Bragg grating sensors in needles [35, 53, 59] and other optical sensing techniques (see e.g. [54]), one should expect to see fiber-based sensing used in conjunction with concentric tube robots in the near future.

Force Sensing Due to the inherent flexibility of the concentric tube robot, it will be useful to know the interaction forces between the robot and the environment for both accurate control and user feedback. A wide variety of force sensors have also been investigated in the context of minimally invasive surgical tools [54], but the only one that has been specifically designed for and applied to a concentric tube robot is a tip force sensor which measures force magnitude and contact angle based on electrical resistance of fluid-filled channels [5, 32]. The design of specialized force sensors is currently an active area of research with many opportunities for innovation, as discussed further in Sect. 6.

5 Design

There are three distinct aspects of concentric tube robot design. Perhaps the one that has received the most attention to date is the selection of tube properties (curvatures, lengths, diameters, number of tubes) appropriately based on application requirements. However, beyond this, one must also construct a suitable actuation unit that grasps the tubes at their respective bases and applies telescopic and axial rotation motions to each. Lastly, one must design the surgical end effectors necessary to accomplish the surgical objective.

Tube Design Optimal selection of tube properties has been the focus of substantial research, and was discussed in the earliest papers on use of concentric tube devices as robotic manipulators [68, 81], which provided ways to determine maximum curvatures and idealizations intended to facilitate design intuition. Since then, a number of authors have investigated algorithms for optimal tube design, using a variety of models and assumptions. Anor et al. planned piecewise constant curvature paths through the brain ventricles for choroid plexus cauterization [4]. Bedell et al. employed the torsionally rigid model and circular precurvatures to design tubes which minimize curvature and overall length while respecting anatomical constraints in cardiac surgery [6]. Torres et al. used circular precurvatures with the torsionally compliant model to develop a rapidly-exploring random tree algorithm to create a design together with an actuator plan for collision-free insertion through a lung lumen [74]. Burgner et al. also used the torsionally compliant model and introduced volume-based coverage objective functions to design robots that are able to optimally cover a desired workspace with their tips [12, 13]. Building upon the ideas in these initial studies, there remains much room for advancement in optimal design algorithms, as discussed further in Sect. 6.

Actuation Unit Design Actuation units have only recently become a topic of interest in the concentric tube robot research community, with early papers simply showing photographs of actuation units with little discussion on their design [68, 83]. A differential drive is described in [78], although this has the drawback of requiring long holes to be drilled through screws. Modular bimanual (two arm) [13] and quadramanual (four arm) [71] robots designed for endonasal surgery have also been presented. Single-arm MRI-compatible designs using piezoelectric motors [70] and pneumatic cylinders [17] have been constructed and demonstrated in MRI environments. A highly compact actuation unit for controlling one curved tube deployed through an endoscope port was described in [16]. Another compact and inexpensive (potentially disposable) actuation unit using a spline screw for CT-guided procedures was described in [33]. Consideration has also been given to reusable actuation units. An autoclavable hand operated actuation unit design was presented in [11]. An autoclavable and biocompatible motorized actuation unit (with a bagging procedure for the motor pack) was described in [14], and applied to evacuation of intracerebral hemorrhages.

End Effector Design A number of innovative end effectors have been developed for concentric tube robots. Dupont et al. developed remarkable metal microelectromechanical systems (MEMS) end effectors specifically for concentric tube robots for cardiac tissue approximation and tissue resection [15, 30, 31]. Burgner et al. mounted a gripper from a flexible endoscopic tool to the tip of a concentric tube robot and also developed a curette end effector for endonasal surgery [13].

6 Future Directions

The results described in the preceding subsections illustrate the state of the art in concentric tube robots. While much is known, there remain many opportunities for future research, as discussed in the subsections below.

Open Topics in Design Many diverse *end effectors* are needed for various surgical objectives. The metal MEMS end effector designs pioneered by Dupont et al. provide an example of a promising fabrication technology for future end effectors, as well as creative designs for tissue approximation and dissection [30].

Research also remains to be done on clinically applicable *actuation units*, and both the disposable and reusable paradigms seem viable. An important consideration in design is the ability to grasp tubes as near the entry point into the body as possible to minimize torsional windup. Safety features such as quick tube retraction will also be useful, as will the ability to change tubes rapidly [14].

Design of *tube properties* is also an open area for future research. Materials other than nitinol may be useful in some contexts as demonstrated by the Roticulator. In steerable needle contexts, non-annular tube profiles have been proposed, but not physically demonstrated, as a means of preventing torsional deformation and hence facilitating follow-the-leader deployment [34]. If new methods could be developed to increase torsional stiffness relative to bending stiffness, this would reduce the tendency of the device to "snap", as well as transmissional torsional windup. In the future it may even be possible to change the curvature of each tube through external means such as tendons or novel actuators embedded in tube walls (see e.g. [38] for an example of a non-precurved nitinol tube device with embedded tendons). There is also a great deal of research to be done in optimal algorithms for designing tube precurvatures, stiffnesses, and numbers of tubes for a given surgical application. To date, there has been no mechanics-based planner in which the number of tubes is a design parameter (Anor et al. consider geometry only [4]), and no design algorithm has yet considered non-circular precurvatures.

Also, importantly, *snapping behavior* (also called bifurcation in some contexts) has yet to be comprehensively incorporated into design algorithms. Current methods to guarantee a snap-free design only apply to the special case of two fully overlapping tubes with zero transmission length. Additionally, a snap-free design may not always be the best choice, depending on the surgical application. Designs that include snapping behavior can use higher curvatures, and snapping can, in principle, be prevented by restricting the configuration space appropriately in software.

Lastly, some *design heuristics* have been suggested, but their limits and implications have yet to be fully explored, and one must be careful in attempting to generalize them. For example, Burgner et al. found that their use of available heuristics alone did not produce good workspace volume coverage [12], despite the fact that computational optimization produced designs that ultimately agreed with the basic premise of the heuristic. Furthermore, while the ideas of dominating stiffness and matched tube pairs are intuitively appealing and appropriate for some design problems (as shown by Bedell et al. in the context of cardiac surgery, for example [6]), they too will not always generalize. This is because in practice one cannot use arbitrarily many tubes, and each matched tube pair trades one degree of freedom (relative telescopic extension of one tube) in exchange for the intuition gained by the designer. This intuition gain must be weighed against the number of tubes available in diameters suitable for surgical objectives, given required tube tolerances and wall thicknesses. Most existing prototypes to date have used just 2 or 3 total tubes. The largest published number is 4 [26].

Open Topics in Modeling After a period of rapid advancement from 2006 to 2011, the past two years have seen slower modeling advancement, perhaps due to the models described in Sect. 2 having reached a sufficient level of detail to enable many applications and research on other topics. Additional effects that have been investigated include tube tolerances [39] and friction [41], but describing how both of these phenomena physically arise from tubes with finite clearances is still an open question. In terms of model evaluation, rigorous comparison and contrast of available approaches has yet to be attempted. It would be useful to compare methods like collocation, finite-element, finite-difference, and quasilinearization based on computational efficiency, accuracy, and numerical stability. Though some two-tube results exist [25, 66], general methods for model-based prediction of the presence multiple model equation solutions and detection of an impending snap have yet to be developed and would be particularly valuable to enable design and use of highly precurved tubes.

Open Topics in Control The existing literature in control addresses teleoperation (via several different methods) and stiffness control. Extensions to include advanced redundancy resolution methods and obstacle avoidance during teleoperation would be desirable. Similarly, user interfaces for continuum robots in general have not been well studied. To enhance steerable needle-type applications, a controller that causes the robot to approximately follow a planned deployment trajectory in the presence of perturbations would also be valuable. Also, concentric tube robots have not yet been used in applications that require high bandwidth such as the cardiac motion compensation studied by Kesner and Howe [36], but advancements are needed if concentric tube robots are eventually to be applied in such settings.

Open Topics in Sensing The major challenge in equipping concentric tube robots with diverse sensors is size. Even many MEMS force sensors remain too large once the sensor's housing is considered. One sensor that can be straightforwardly integrated is the fiber Bragg grating. The recent interest in the robotic needle community in these sensors appears likely to foreshadow their use in concentric tube robots in the near future. Beyond this, a wide variety of other sensors would be useful if/when they are sufficiently miniaturized to concentric tube robot size. The stiffness modulation approach in [48] also provides tip force values based on robot deflection and the desired stiffness behavior. Force sensing based on one dimensional beam bending of the last tube/guide wire was implemented with an elliptic integral interpolation map in [86], which showed that the interpolation method can speed up calculation and offer interactive computation rates for telemanipulation assistance. Future studies are needed to experimentally evaluate the influence of kinematic error on model-based force estimation methods.

7 Conclusion

As can be seen from the review of the state of the art in this paper, as well as the discussion of open questions, concentric tube robotics is a maturing field where foundational models now exist, yet there remain many opportunities further research and application in specific clinical interventions. Many such applications have been suggested, but few have been explored in depth. Interestingly, simple concentric tube devices were some of the earliest devices fabricated out of nitinol and were brought to market in 1985, the same year the very first robotic surgery was done with an industrial robot. With the development of robotic actuation, in the coming years it will likely be feasible to introduce concentric tube robot products with much greater capabilities, in both teleoperated settings and as steerable needles. Despite the fact that they have the potential to become commercial products in the relatively near term, concentric tube robots continue to be a rich source of design, modeling, control, and sensing challenges for the research community. If solved, each of these challenges has the potential to make the already good capabilities of these robots even better, and extend their reach into continually more complex surgical scenarios.

Acknowledgments The authors acknowledge support for this work from the National Science Foundation (IIS-1054331) and the National Institutes of Health (R01 EB017467 and R21 EB017952).

References

1. Abbott, J.J., Marayong, P., Okamura, A.M.: Haptic virtual fixtures for robot-assisted manipulation. Springer Tracts Adv. Robot. **28**, 49–64 (2007)
2. Abolhassani, N., Patel, R., Moallem, M.: Needle insertion into soft tissue: a survey. Med. Eng. Phys. **29**, 413–431 (2007)
3. Alterovitz, R., Patil, S., Derbakova, A.: Rapidly-exploring roadmaps: weighing exploration versus refinement in optimal motion planning. In: IEEE International Conference on Robotics and Automation, pp. 3706–3712 (2011)
4. Anor, T., Madsen, J.R., Dupont, P.: Algorithms for design of continuum robots using the concentric tubes approach: a neurosurgical example. In: IEEE International Conference on Robotics and Automation, pp. 667–673 (2011)
5. Arabagi, V., Gosline, A., Wood, R.J., Dupont, P.E.: Simultaneous soft sensing of tissue contact angle and force for millimeter-scale medical robots. In: IEEE International Conference on Robotics and Automation, pp. 4381–4387 (2013)
6. Bedell, C., Lock, J., Gosline, A., Dupont, P.E.: Design optimization of concentric tube robots based on task and anatomical constraints. In: IEEE International Conference on Robotics and Automation, pp. 398–403 (2011)
7. Berns, M.S., Tsai, E.Y., Austin-Breneman, J., Schulmeister, J.C., Sung, E., Ozaki, C.K., Walsh, C.J.: Single entry tunneler [SET] for hemodialysis graft procedures. In: Design of Medical Devices Conference, pp. 1–8 (2011)
8. Burdette, E.C., Rucker, D.C., Prakash, P., Diederich, C.J., Croom, J.M., Clarke, C., Stolka, P., Juang, T., Boctor, E.M., Webster III, R.J.: The ACUSITT ultrasonic ablator: The first steerable needle with an integrated interventional tool. In: SPIE 7629 (2010)
9. Burgner, J., Herrell, S.D., Webster III, R.J.: Toward flouroscopic shape reconstruction for control of steerable medical devices. ASME Dyn. Sys. Cont. Conf. **2**, 1–4 (2011)
10. Burgner, J., Swaney, P.J., Rucker, D.C., Gilbert, H.B., Nill, S.T., Russell, P.T., Weaver, K.D., Webster III, R.J.: A bimanual teleoperated system for endonasal skull base surgery. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2517–2523 (2011)
11. Burgner, J., Swaney, P.J., Bruns, T.L., Clark, M.S., Rucker, D.C., Webster III, R.J.: An autoclavable steerable cannula manual deployment device: Design and accuracy analysis. ASME J. Med. Dev. **6**(4), 041,007-1–041,007-7 (2012)
12. Burgner, J., Gilbert, H.B., Webster III, R.J.: On the computational design of concentric tube robots: Incorporating volume-based objectives. In: IEEE International Conference on Robotics and Automation, pp. 1185–1190 (2013)
13. Burgner, J., Rucker, D.C., Gilbert, H.B., Member, S., Swaney, P.J., Russell, P.T., Weaver, K.D., Webster, R.J.: A telerobotic system for transnasal surgery. In: IEEE/ASME Transactions on Mechatronics, pp. 1–11 (2013) (in press)
14. Burgner, J., Swaney, P.J., Lathrop, R.A., Weaver, K.D., Webster III, R.J.: Debulking from within: a robotic steerable cannula for intracerebral hemorrhage evacuation. IEEE Trans. Biomed. Eng. **60**(9), 2567–2575 (2013)
15. Butler, E.J., Folk, C., Cohen, A., Vasilyev, N.V., Chen, R., del Nido, P.J., Dupont, P.E.: Metal MEMS tools for beating-heart tissue approximation. In: International Conference on Robotics and Automation, pp. 411–416 (2011)
16. Butler, E.J., Hammond-Oakley, R., Chawarski, S., Gosline, A.H., Codd, P., Anor, T., Madsen, J.R., Dupont, P.E., Lock, J.:Robotic neuro-endoscope with concentric tube augmentation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2941–2946 (2012)
17. Comber, D.B., Cardona, D., Webster III, R.J., Barth, E.J.: Precision pneumatic robot for mri-guided neurosurgery. In: Design of Medical Devices Conference, vol. 35 (2012)
18. Comber, D.B., Barth, E.J., Webster III, R.J.: MR-compatible precision pneumatic active cannula robot. In: ASME J. Med. Dev. (2013) (in press)
19. Croom, J.M., Rucker, D.C., Romano, J.M., Webster III, R.J.: Visual sensing of continuum robot shape using self-organizing maps. In: IEEE International Conference on Robotics and Automation, pp. 4591–4596 (2010)

20. Cuschieri, A., Buess, G.: Future advances in endoscopic surgery. In: Cuschieri, A., Buess, G., Périssat, J. (eds.) Operative Manual of Endoscopic Surgery, pp. 339–347. Springer, Heidelberg (1992)
21. Daum, W.R.: Deflectable needle assembly. US Patent 6,572,593 (2003)
22. Dieci, L., Russell, R.D., Van Vleck, E.S.: Unitary integrators and applications to continuous orthonormalization techniques. SIAM J. Numer. Anal. **31**, 261–281 (1994)
23. Duerig, T., Pelton, A.: Nitinol: The book. http://www.nitinol.com/nitinon-the-book/a-historical-perspective
24. Dupont, P.E., Lock, J., Butler, E.: Torsional kinematic model for concentric tube robots. IEEE Int. Conf. Robot. Autom. **2009**, 2964–2971 (2009)
25. Dupont, P.E., Lock, J., Itkowitz, B., Butler, E.: Design and control of concentric-tube robots. IEEE Trans. Robot. **26**(2), 209–225 (2010)
26. Dupont, P.E., Gosline, A., Vasilyev, N., Lock, J., Butler, E., Folk, C., Cohen, A., Chen, R., Schmitz, G., Ren, H., del Nido, P.: Concentric tube robots for minimally invasive surgery. In: Hamlyn Symposium on Medical Robotics, pp. 3–5 (2012)
27. Furusho, J., Ono, T., Murai, R., Fujimoto, T., Chiba, Y., Horio, H.: Development of a curved multi-tube (CMT) catheter for percutaneous umbilical blood sampling and control methods of CMT catheters for solid organs. In: IEEE International Conference on Mechatronics and Automation, pp. 410–415 (2005)
28. Furusho, J., Katsuragi, T., Kikuchi, T., Suzuki, T., Tanaka, H., Chiba, Y., Horio, H.: Curved multi-tube systems for fetal blood sampling and treatments of organs like brain and breast. Int. J. Comput. Assist. Radiol. Surg. **1**(S1), 223–226 (2006)
29. Gilbert, H., Webster III, R.J.: Can concentric tube robots follow the leader? In: IEEE International Conference on Robotics and Automation, pp. 4866–4872 (2013)
30. Gosline, A.H., Vasilyev, N.V., Butler, E.J., Folk, C., Cohen, A., Chen, R., Lang, N., Del Nido, P.J., Dupont, P.E.: Percutaneous intracardiac beating-heart surgery using metal MEMS tissue approximation tools. Int. J. Robot. Res. **31**(9), 1081–1093 (2012)
31. Gosline, A.H., Vasilyev, N.V., Veeramani, A., Wu, M., Schmitz, G., Chen, R., Arabagi, V., del Nido, P.J., Dupont, P.E.: Metal MEMS tools for beating-heart tissue removal. In: IEEE International Conference on Robotics and Automation, pp. 1921–1926 (2012)
32. Gosline, A.H., Arabagi, V., Kassam, A., Dupont, P.E.: Achieving biocompatibility in soft sensors for surgical robots. In: Hamlyn Symposium on Medical robotics, pp. 5–6 (2013)
33. Graves, C.M., Slocum, A., Gupta, R., Walsh, C.J.:Towards a compact robotically steerable thermal ablation probe. In: IEEE International Conference on Robotics and Automation, pp. 709–714 (2012)
34. Greenblatt, E., Trovato, K., Popovic, A., Stanton, D.: Interlocking nested cannula. US Patent Application: 20110201887 (2011)
35. Iordachita, I., Sun, Z., Balicki, M., Kang, J.U., Phee, S.J., Handa, J., Gehlbach, P., Tayler, R.: A sub-millimetric, 0.25 mN resolution fully integrated fiber-optic force-sensing tool for retinal microsurgery. Int. J. Comput. Assist. Radiol. Surg. **4**, 383–390 (2009)
36. Kesner, S.B., Howe, R.D.: Position control of motion compensation cardiac catheters. IEEE Trans. Robot. **27**(6), 1045–1055 (2011)
37. Kratchman, L.B., Rahman, M.M., Saunders, J.R., Swaney, P.J., Webster III, R.J.: Toward robotic needle steering in lung biopsy: a tendon-actuated approach. In: SPIE (2011)
38. Kutzer, M.D., Segreti, S.M., Brown, C.Y., Taylor, R.H., Mears, S.C., Armand, M.: Design of a new cable-driven manipulator with a large open lumen: Preliminary applications in the minimally-invasive removal of osteolysis. In: IEEE International Conference on Robotics and Automation, pp. 2913–2920 (2011)
39. Lathrop, R.A., Rucker, D.C., Webster III, R.J.: Guidance of a steerable cannula robot in soft tissue using preoperative imaging and conoscopic surface contour sensing. In: IEEE International Conference on Robotics and Automation, pp. 5601–5606 (2010)
40. Lobaton, E.J., Fu, J., Torres, L.G., Alterovitz, R.: Continuous shape estimation of continuum robots using X-ray images. In: IEEE International Conference on Robotics and Automation, pp. 717–724 (2013)

41. Lock, J., Dupont, P.E.: Friction modeling in concentric tube robots. In: IEEE International Conference on Robotics and Automation, pp. 1139–1146 (2011)
42. Lock, J., Laing, G., Mahvash, M., Dupont, P.E.: Quasistatic modeling of concentric tube robots with external loads. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2325–2332 (2010)
43. Loser, M.: A new robotic system for visually controlled percutaneous interventions under X-ray or CT-fluoroscopy. Master's thesis, The Albert-Ludwig-University, Germany (2005)
44. Lyons, L.A., Webster III, R.J., Alterovitz, R.: Motion planning for active cannulas. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 801–806 (2009)
45. Lyons, L.A., Webster III, R.J., Alterovitz, R.: Planning active cannula configurations through tubular anatomy. IEEE Int. Conf. Robot. Autom. **1**, 2082–2087 (2010)
46. Mahvash, M., Dupont, P.E.: Bilateral teleoperation of flexible surgical robots. In: Proceedings of the New Vistas and Challenges in Telerobotics Workshop on IEEE International Conference on Intelligent Robots and Systems, pp. 58–64 (2008)
47. Mahvash, M., Dupont, P.E.: Stiffness control of a continuum manipulator in contact with a soft environment. IEEE/RSJ Int. Conf. Intell. Robot. Sys. **2010**, 863–870 (2010)
48. Mahvash, M., Dupont, P.E.: Stiffness control of surgical continuum manipulators. IEEE Trans. Robot. **27**(2), 334–345 (2011)
49. Melzer, A.: Instruments for endoscopic surgery. In: A. Cuschieri, G. Buess, J. Périssat (eds.) Operative Manual of Endoscopic Surgery, p. 35. Springer, Heidelberg (1992)
50. Melzer, A., Schurr, M.O., Lirici, M.M., Klemm, B., Stöckel, D., Buess, G.: Future trends in endoscopic suturing. Endosc. Surg. Allied Technol. **2**, 78–82 (1994)
51. Melzer, A., Schmidt, A., Kipfmüller, K., Grönmeyer, D., Seibel, R.: Technology and principles of tomographic image-guided interventions and surgery. Surg. Endosc. **11**, 946–956 (1997)
52. Okazawa, S., Ebrahimi, R., Chuang, J., Salcudean, S.E., Rohling, R.: Hand-held steerable needle device. IEEE/ASME Trans. Mechatron. **10**, 285–296 (2005)
53. Park, Y., Elayaperumal, S., Daniel, B., Ryu, S.C., Shin, M., Savall, J., Black, R.J., Moslehi, B., Cutkosky, M.R.: Real-time estimation of 3-d needle shape and deflection for MRI-guided interventions. IEEE/ASME Trans. Mechatron. **15**, 906–915 (2010)
54. Puangmali, P., Althoefer, K., Seneviratne, L.D., Murphy, D., Dasgupta, P.: State-of-the-art in force and tactile sensing for minimally invasive surgery. IEEE Sens. J. **8**, 371–381 (2008)
55. Reed, K.B., Majewicz, A., Kallem, V., Alterovitz, R., Goldberg, K., Cowan, N., Okamura, A.: Robot assisted needle steering. Robot. Autom. Mag. **18**, 35–46 (2011)
56. Ren, H., Dupont, P.E.: Tubular structure enhancement for surgical instrument detection in 3D ultrasound. Int. Conf. IEEE EMBS **2011**, 7203–7206 (2011)
57. Ren, H., Dupont, P.E.: Tubular enhanced geodesic active contours for continuum robot detection using 3D ultrasound. In: IEEE International Conference on Robotics and Automation, pp. 2907–2912 (2012)
58. Robinson, G., Davies, J.: Continuum robots—a state of the art. In: IEEE International Conference on Robotics and Automation, pp. 2849–2854 (1999)
59. Roesthuis, R.J., Kemp, M., van den Dobbelsteen, J.J., Misra, S.: Three-dimensional needle shape reconstruction using an array of fiber Bragg grating sensors. In: IEEE/ASME Transactions on Mechatronics (2013) (in press)
60. Rucker, D.C., Webster III, R.J.: Mechanics-based modeling of bending and torsion in active cannulas. In: IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechanics, pp. 704–709 (2008)
61. Rucker, D.C., Webster III, R.J.: Mechanics of bending, torsion, and variable precurvature in multi-tube active cannulas. In: IEEE International Conference on Robotics and Automation, pp. 2533–2537 (2009)
62. Rucker, D.C., Webster III, R.J.: Parsimonious evaluation of concentric-tube continuum robot equilibrium conformation. IEEE Trans. Biomed. Eng. **56**(9), 2308–2311 (2009)
63. Rucker, D.C., Jones, B.A., Webster, R.J.: A geometrically exact model for externally loaded concentric-tube continuum robots. IEEE Trans. Robot. **26**(5), 769–780 (2010)

64. Rucker, D.C., Webster III, R.J.: Computing Jacobians and compliance matrices for externally loaded continuum robots. IEEE Int. Conf. Robot. Autom. **3**, 945–950 (2011)
65. Rucker, D.C., Jones, B.A., Webster III, R.J.: A model for concentric tube continuum robots under applied wrenches. In: IEEE International Conference on Robotics and Automation, pp. 1047–1052 (2010)
66. Rucker, D.C., Webster III, R.J., Chirikjian, G.S., Cowan, N.J.: Equilibrium conformations of concentric-tube continuum robots. Int. J. Robot. Res. **29**(10), 1263–1280 (2010)
67. Schwartz, M.: New Materials, Processes, and Methods Technology. CRC Press, Boca Raton (2005)
68. Sears, P., Dupont, P.: A steerable needle technology using curved concentric tubes. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2850–2856 (2006)
69. Stoeckel, D., Melzer, A.: New developments in superelastic instruments for minimally invasive surgery. In: Presentation for "Changing Surgical Markets—Increasing Efficiency and Reducing Cost Through New Technology and Procedure Innovation" (1993)
70. Su, H., Cardona, D.C., Shang, W., Camilo, A., Cole, G.A., Rucker, D.C., Webster III, R.J., Fischer, G.S.: A MRI-guided concentric tube continuum robot with piezoelectric actuation: a feasibility study. In: IEEE International Conference on Robotics and Automation, pp. 1939–1945 (2012)
71. Swaney, P.J., Croom, J.M., Burgner, J., Gilbert, H.B., Rucker, D.C., Weaver, K.D., Russell III, P.T., Webster, R.J.: Design of a quadramanual robot for single-nostril skull base surgery. In: ASME Dynamic Systems and Control Conference (2012)
72. Terayama, M., Furusho, J., Monden, M.: Curved multi-tube device for path-error correction in a needle-insertion system. Int. J. Med. Robot. Comp. Assist. Surg. **3**(2), 125–134 (2007)
73. Torres, L.G., Alterovitz, R.: Motion planning for concentric tube robots using mechanics-based models. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5153–5159 (2011)
74. Torres, L.G., Webster III, R.J., Alterovitz, R.: Task-oriented design of concentric tube robots using mechanics-based models. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2012)
75. Trivedi, D., Rahn, C.D., Kier, W.M., Walker, I.D.: Soft robotics: biological inspiration, state of the art, and future research. Appl. Bionics Biomech. **5**, 99–117 (2008)
76. Vasilyev, N.V., Dupont, P.E., del Nido, P.J.: Robotics and imaging in congenital heart surgery. Future Cardiol. **8**(2), 285–296 (2012)
77. Walsh, C.J., Franklin, J., Slocum, A.H., Gupta, R.: Design of a robotic tool for percutaneous instrument distal tip repositioning. In: IEEE Engineering in Medicine and Biology Society, pp. 2097–2100 (2011)
78. Webster III, R.J.: Design and Mechanics of Continuum Robots for Surgery. Ph.D. thesis, The Johns Hopkins University (2007)
79. Webster III, R.J., Jones, B.A.: Design and kinematic modeling of constant curvature continuum robots: a review. Int. J. Robot. Res. **29**(13), 1661–1683 (2010)
80. Webster III, R.J., Kim, J.S., Cowan, N.J., Chirikjian, G.S., Okamura, A.M.: Nonholonomic modeling of needle steering. Int. J. Robot. Res. **25**, 509–525 (2006)
81. Webster III, R.J., Okamura, A., Cowan, N.J.: Toward active cannulas: miniature snake-like surgical robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2857–2863 (2006)
82. Webster III, R.J., Romano, J.M., Cowan, N.J.: Kinematics and calibration of active cannulas. In: IEEE International Conference on Robotics and Automation, pp. 3888–3895 (2008)
83. Webster III, R.J., Swensen, J.P., Romano, J.M., Cowan, N.J.: Closed-form differential kinematics for concentric-tube continuum robots with application to visual servoing. In: International Symposium on Experimental Robotics, pp. 485–494 (2008)
84. Webster III, R.J., Romano, J.M., Cowan, N.J.: Mechanics of precurved-tube continuum robots. IEEE Trans. Robot. **25**(1), 67–78 (2009)
85. Wei, W., Goldman, R.E., Fine, H.F., Chang, S., Simaan, N.: Performance evaluation for multi-arm manipulation of hollow suspended organs. IEEE Trans. Robot. **25**, 147–157 (2009)

86. Wei, W., Simaan, N.: Modeling, force sensing, and control of flexible cannulas for microstent delivery. ASME J. Dyn. Sys. Meas. Control **134**(4), 1–12 (2012)
87. Xu, R., Patel, R.V.: A fast torsionally compliant kinematic model of concentric-tube robots. Int. Conf. IEEE EMBS **2012**, 904–907 (2012)
88. Xu, R., Asadian, A., Naidu, A.S., Patel, R.V.: Position control of concentric-tube continuum robots using a modified jacobian-based approach. In: IEEE International Conference on Robotics and Automation, pp. 5793–5798 (2013)
89. Yu, H., Shen, J., Joos, K.M., Simaan, N.: Design, calibration and preliminary testing of a robotic telemanipulator for OCT guided retinal surgery. In: IEEE International Conference on Robotics and Automation, pp. 225–231 (2013)

A Framework for Real-Time Multi-Contact Multi-Body Dynamic Simulation

François Conti and Oussama Khatib

Abstract In this paper we propose a unified framework for the real-time dynamic simulation and contact resolution of rigid articulated bodies. This work builds on previous developments in the field of dynamic simulation, collision detection, contact resolution, and operational space control. However, the key to efficiency and real-time performance is a new parallel implementation of our collision detection and contact resolution algorithm which decomposes the problem into tasks that can be concurrently executed. Finally, the results and accuracy of our simulation models are compared for the first time against recorded motions of real articulated bodies colliding on a frictionless air floating table.

1 Introduction

Computer simulation is the use of a computer program or simulator to model the time varying behavior of a natural system using an abstract model. Computer simulations have become a useful part of mathematical modeling of many natural systems in physics, astrophysics, chemistry and biology, human systems in economics, psychology, social science, and engineering. The simulation of a system is represented as the running of the system's model; this can be used to explore and gain new insights into new technology and to estimate the performance of systems too complex for analytical solutions.

In the area of computer animation and dynamic simulation, Baraff [2] pioneered some of the early models for rigid bodies that relied on the representation of physical constraints. His work was later expanded to support more complex environments such as deformable objects, and clothing simulation [3–7] (Fig. 1).

To address the problem of computational complexity when modeling articulated objects with multiple degrees of freedom, Featherstone [15] proposed a new formu-

F. Conti (✉) · O. Khatib
Artificial Intelligence Laboratory, Stanford University, Stanford, CA 94305, USA
e-mail: conti@ai.stanford.edu

O. Khatib
e-mail: ok@ai.stanford.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts
in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_16

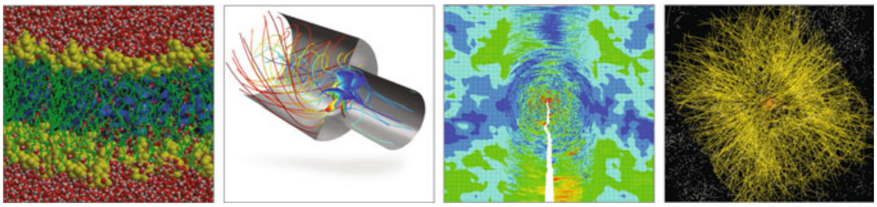


Fig. 1 Simulators. These tools are used to analyze the behavior of a system in areas such as physics, biology, chemistry, and engineering

lation that requires modeling only the true degrees of freedom of a system while avoiding the expensive task of solving the internal constraints. An optimized version of this approach was presented by Chang and Khatib [8]. Building on these foundations, Ruspini and Khatib [22, 23] proposed a generalized contact space representation to efficiently model and solve both contacts and collisions between complex articulated bodies in real-time. These new algorithms led to the development of an early simulation framework (SAI) capable of simulating complex articulated systems (e.g. humanoid robots) interacting in large virtual environments [18].

With the objective of further improving the computational speed of these simulations, we present in this paper a new collision detection and distance computation approach that greatly improves the speed and accuracy compared to previous works. The key to efficiency relies on a new parallel implementation of our collision detection algorithm which decomposes the problem into tasks that can be concurrently executed.

The rest of this paper is organized as follows. A review of the equations that describe the motion of unconstrained generalized articulated systems is presented in Sect. 2. Section 3 introduces our new collision detection approach, while the problem of contact resolution is addressed in Sect. 4. Finally, Sect. 5 evaluates the accuracy of our framework by comparing the motion of real colliding bodies with simulated models. These results are then compared with other simulation frameworks.

2 Dynamics

The configuration of a n degree-of-freedom articulated system can be expressed by q, a set of n independent generalized coordinates that uniquely define the configuration of the system relative to a reference frame. Although there may be many choices for defining a set of generalized coordinates, these parameters are usually selected in a way which makes the equations of motion easier to solve. If these parameters are independent of one another, then the number of independent generalized coordinates is defined by the number of degrees of freedom of the system (Fig. 2).

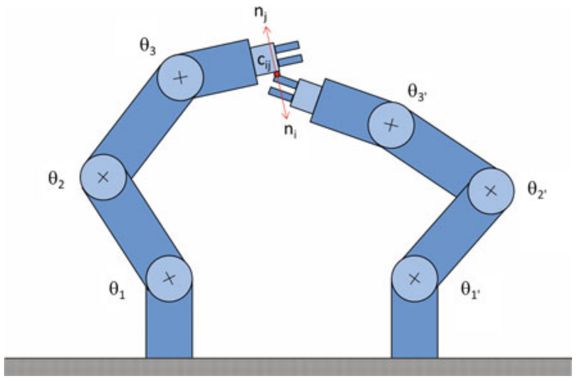


Fig. 2 Collision resolution. Contact interaction between two articulated mechanisms

The equation of motion for a generalized system can be expressed as

$$\Gamma = A(q)\ddot{q} + b(q, \dot{q}) + g(q), \tag{1}$$

where $A(q)$ represents the mass matrix, $b(q, \dot{q})$ the centrifugal and Coriolis vector, $g(q)$ the gravity-force vector, and Γ the generalized torque vector of the body.

Equation 1 is solved for $\ddot{q}$ to simulate the dynamic behavior of an object using its equation of motion.

$$\ddot{q} = A^{-1}(q)(\Gamma - b(q, \dot{q}) - g(q)). \tag{2}$$

In practice, the direct inversion of the mass matrix $A(q)$ is computationally expensive with a $O(n^3)$ asymptotic growth for a system composed of n degrees of freedom. Direct methods for computing the inverse dynamic equations have been proposed to reduce computational complexity. An example of such an approach is the algorithm proposed by Featherstone [15]. In this method a recursion on the links of the

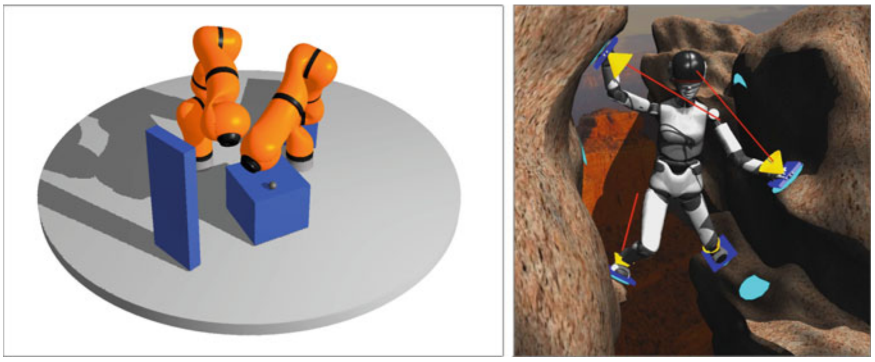


Fig. 3 Real-time simulation. Simulating in real-time the motion of articulated bodies in complex environments. A multi-frame operational space controller sends commands to the robots at 1000 Hz to control their posture and task

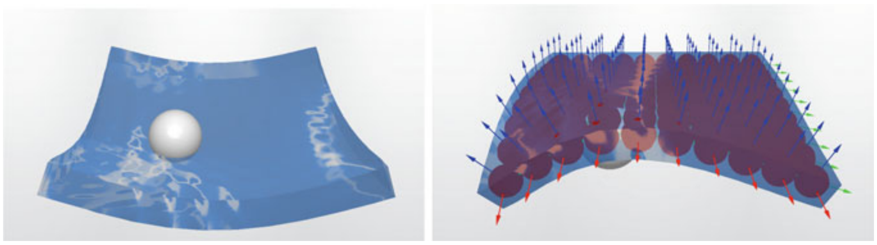


Fig. 4 Deformable models. Simulating the dynamic behavior of a deformable object. Six degree-of-freedom springs are used to model elongation, flexion, and torsion properties between the different mass nodes that compose the skeleton of the object [11]

articulated system is used to find the corresponding acceleration vector $\ddot{q}$ for a given generalized torque vector in $O(n)$ time.

Although the simulator uses a numerical solution without ever directly matrices A, A^{-1} or the link Jacobian J, it is understood that these terms can be written explicitly. It must be noted, however, that the final solution does not require explicit computation of these matrices. This computational simplification is of great significance, as the results of this analysis are applicable to model any type of system of the form shown in Eq. 1 (Figs. 3 and 4).

3 Collision Detection

In this section we present an improved approach for detecting collision and computing distances between rigid polyhedral models. The new framework is designed for parallel computing and has been implemented using the OpenMP and CUDA$^{\text{TM}}$ parallel computing platforms. By distributing the computation on multiple processors, this new collision detection framework has been shown to run significantly faster compared to earlier frameworks that exploited the use of only single CPU threads.

The processing pipeline of our collision detection and distance computation library is separated into three major phases: *broad*, *narrow*, and *resolution*. Each phase has a specific task to solve and requires a different view of the participating objects to be compared. In the following sections we review these different phases and describe their implementation.

3.1 Broad Phase

The broad phase performs a coarse and quick verification to determine which of the objects may potentially experience a collision at all, and produces a pair list of possible collision partners. For fast assessments, simple bounding sphere structures are used (See Fig. 5).

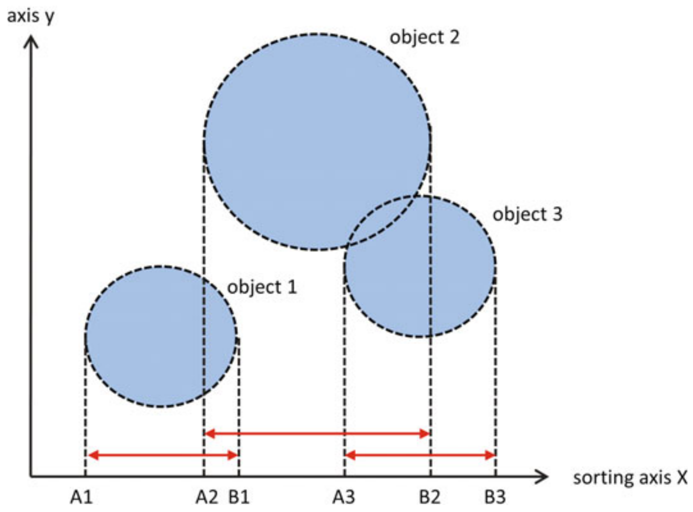


Fig. 5 Broad phase. Every object is enclosed in a sphere. A sort and sweep algorithm (CPU) is used to create a list of possible colliding candidates. The list is then forwarded to the narrow phase (GPU) for further evaluation

The verification is achieved by sorting the lower bounds and upper bounds of the bounding volume of each object along the axes x, y, and z. As the objects move, their boundary limits may overlap. If the bounding volumes overlap in all axes, they are marked to be tested in the narrow phase by a more precise and time consuming algorithm.

If the collision detection process is performed continuously at regular small time intervals, it is highly likely that the objects composing the environment do not move significantly between two simulation steps, and therefore the sorted lists of bounding volumes can be updated with a moderately small computational effort. Sorting algorithms, such as insertion sort, are particularly effective at organizing almost-sorted lists, and are therefore particularly effective for this task.

3.2 Narrow Phase

Upon receiving the collision pair list from the broad phase, the narrow phase searches whether the two objects in each pair are colliding, or for their nearest distance.

Before computing the distance between two objects, we use their underlying model to build a hierarchical bounding representation. The version of our algorithm assumes that each object is composed uniquely of triangles. This assumption is not fundamental and extending the approach to other representations is possible. The bounding representation is based on a convex hull decomposition of the entire object.

An extended GJK distance algorithm [16] is used to estimate the distance between pairs of convex hulls.

Similar to other collision detection algorithms [14, 19], the bounding representation consists of an approximately balanced binary tree. Each node of the tree contains the description of a single convex hull. The tree has the following two properties: the union of all the leaf nodes completely contains the surface of the object, and the convex hull at each node completely contains the convex hull representations of its descendant leaf nodes.

The idea behind the bounding representation is as follows. The leaf nodes represent the surface of the object. Interior nodes of the tree represent approximations of descendant leaf nodes. One can use the convex hull at an interior node to determine a lower bound for the distance to any of the descendant leaf nodes, and hence to the object's surface. Nodes that are close to the root of the tree represent many leaf nodes, but at a coarser resolution. Conversely, nodes near the bottom of the tree closely approximate the shape of the few leaf nodes below them. The tree represents a hierarchical description of the entire object.

The first step to building the tree is to perform a convex hull decomposition of the object. This problem has been extensively investigated [1, 9, 10, 20, 21] and is known to have an output of size $O(n^2)$, where n is the number of triangles composing the object.

These convex hulls compose the leaf nodes of the tree. To enable the search routine to determine which convex components to compare, we label each convex hull with child convex hulls for which it was created. A divide and conquer strategy is then used to build the interior nodes of the tree. The set of leaf nodes is divided into two approximately equal subsets. We build a tree for each of the subsets; these trees are combined into a single tree by creating a new node with each of the subtrees as children. The subtrees are built by recursively calling the same algorithm until the set consists of a single leaf node. Each node has a convex hull representation that contains all the convex hulls of the descendant leaf nodes and represents an approximation of these leaves. The two children of the node are intended to represent a slightly more accurate approximation of the same leaves.

To maximize the improvement of the approximation, we attempt to split a set of leaf nodes into two subsets so that the bounding volume for each subset becomes as small as possible. Unfortunately there are many ways to split a set of primitives into two groups, and selecting the optimal division remains an open research problem. In our implementation we compute the covariance matrix of the convex hull and search for the maximum spread direction, which becomes the splitting axis. From this point we divide the leaf nodes using the average value along this axis as the discriminant. Each of the resulting two subsets should be rather compact and contain approximately equal numbers of elements. After dividing the set into two subsets, we build trees for each subset by recursively invoking the algorithm, and then creating a new node with the two subtrees as children.

3.3 Determining Contact Points

To compute the exact distance between two objects, we need to identify a pair of points, one on each object, such that the distance between the points is less than or equal to the distance between any other pair.

We estimate the distance between the two objects by finding a pair of triangles such that the distance between the triangles is less than or equal to the distance between any other pair. The distance between triangles is computed using a convex distance algorithm. An overview of our algorithm to compute the distance, d, between objects operates as follows. We initially set d to infinity. The search routine attempts to show that the objects are within at least a distance d apart. Suppose the search finds two polygons from the underlying model that are less than d of each other; for the initial value of d this is not difficult. If the triangles intersect, then we know that the distance between the two objects is zero and we are done. Otherwise, we set d to the distance between the two triangles and continue the search with the new value of d. Eventually, the search shows either that the objects are a distance d apart or that the objects intersect.

The key to the algorithm is to be able to show the two objects are within a distance d from each other without examining all possible pairs of triangles. Since each triangle is part of a convex hull and enclosed by a set of leaf nodes in the bounding tree, we need only examine pairs of convex hulls for which a corresponding pair of leaf convex hulls are within less than a distance d from each other. Of course, if we had to examine all possible pairs of leaf nodes, then we would gain nothing, however, the hierarchical structure of the bounding tree enables us to avoid this situation.

The search routine finds pairs of leaf nodes that are less than a distance d apart. The search examines pairs of nodes in a depth-first manner starting with the root nodes of the two trees. If the distance between the nodes convex hulls is greater or equal to the current value of d, then, from the structure of the bounding trees, we know the distance between the two sets of descendant leaf hulls is greater than or equal to d and can thus be ignored. If the two nodes are less than d apart, we must further examine the children of the nodes. The distance between two convex hulls can be computed using one of the many available distance algorithms for convex objects. If the distance between the two lead nodes is less than d, then we have found a new minimum. If the distance is zero, i.e. the triangles intersect, then we know the distance between the objects is zero and the search need not continue. Otherwise, we set d to the new distance and continue the search (Fig. 6).

3.4 Implementation and Execution Time

The first stage of our collision detection algorithm occurs entirely at the CPU level and leads to a list of potential candidates that require further investigation. Each pair is then assigned to an individual GPU thread, which evaluates the distance. The

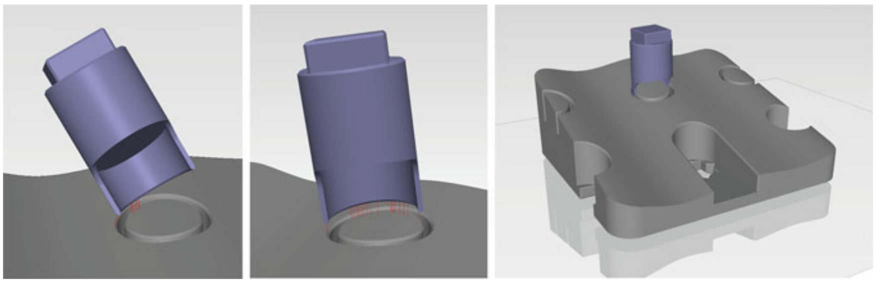


Fig. 6 Peg-in-hole assembly. The assembly task is manually performed using a six degree-of-freedom haptic device. The nearest points between the two objects are highlighted in *red* and are used to model the constraints between the parts

Table 1 Collision detection performance

Part description	Number of polygons	Average query time (ms) (CPU + GPU)	Average query time (ms) (CPU only)
Balance staff bearing	1,230	0.622	9.121
Click	4,420	1.823	22.112
Train bridge	5,221	2.928	33.276
Center wheel	11,220	1.853	31.163
Pallet bridge	13,441	4.121	65.201
Balance wheel	27,822	3.646	58.994
Ratchet wheel	55,420	3.435	55.103

This table compares the query times for single parts between the CPU and GPU versions of the collision detector. All times were recorded in milliseconds

distances between nodes is computed by traversing the collision tree as described in the previous section. The algorithm used to evaluate the distance between two convex hulls is derived from the algorithm developed by Gilbert et al. [16] and was implemented using the CUDA$^{\text{TM}}$ framework. The open source library QHULL was used to perform convex hull computations. Once the collision data is computed for each pair of objects, the information is merged on a single CPU thread and handed over to the contact solver.

For performance comparison, we created a scene containing 71 mechanical parts (see Table 1) each of which was composed of between 2,800 and 56,000 triangles. The experiment consisted of having an operator assemble a selection of parts using a six degree-of-freedom haptic device. During the experiment, all 71 parts were active in the simulation at all times. The collision detection process was shared between the GPU and CPU, while the trajectories of every part were recorded to disc during the simulation. After the experiment was completed, a CPU version of our collision detector was tested over the recorded trajectories for comparison. The evaluation was performed using a six core 3.9 GHz Intel Core i7-3960X using an Nvidia GeForce GTX680 graphic card. Timings were measured in milliseconds.

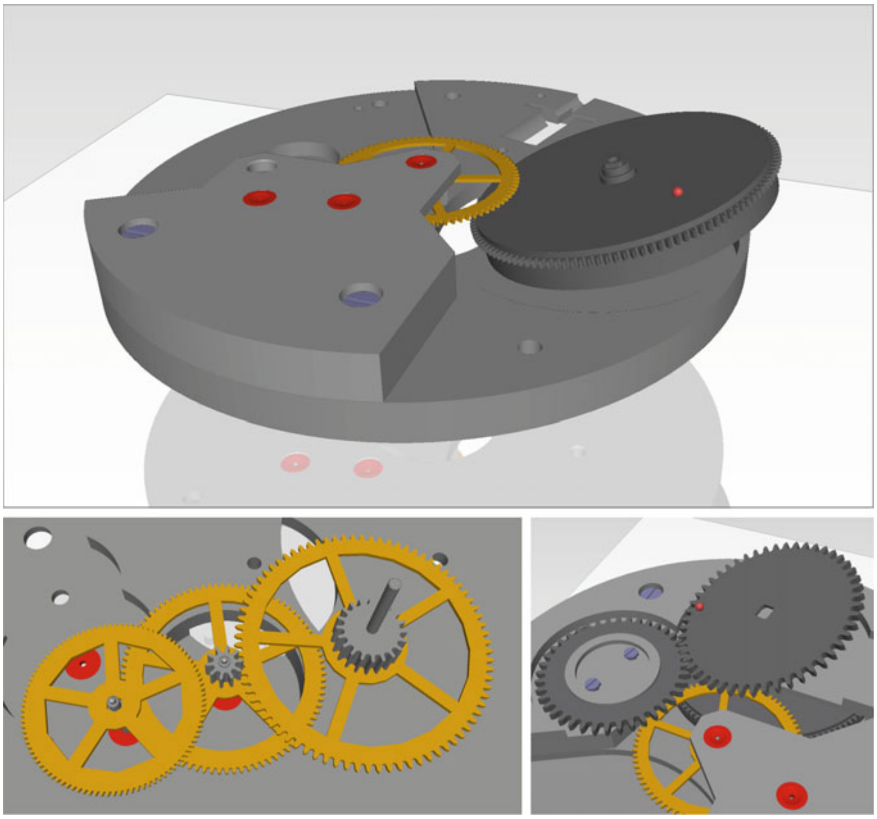


Fig. 7 Virtual assembly. Our framework was evaluated by haptically assembling a virtual model of a mechanical wrist watch. The model was composed of 71 mechanical parts with a total polygon count of 1.4 million. The user could manipulate in real-time any component of the mechanism using an active seven degree-of-freedom sigma.7 haptic device. Through the haptic device, the user felt the interaction forces and constraints between the different parts

Results obtained for this particular 3D model demonstrate that the GPU version of the collision detector can operate 10–15x faster compared to the CPU version. It must be noted that the same experiments performed with smaller number of parts (<5) would perform faster on a CPU version alone due to the reduced overhead (Fig. 7).

4 Contact Resolution

When a collision occurs between two bodies, a set of contact points is reported by the collision detector. The contact space parameters associated with these points will generally not be independent, e.g. four legged table resting on a flat surface. For

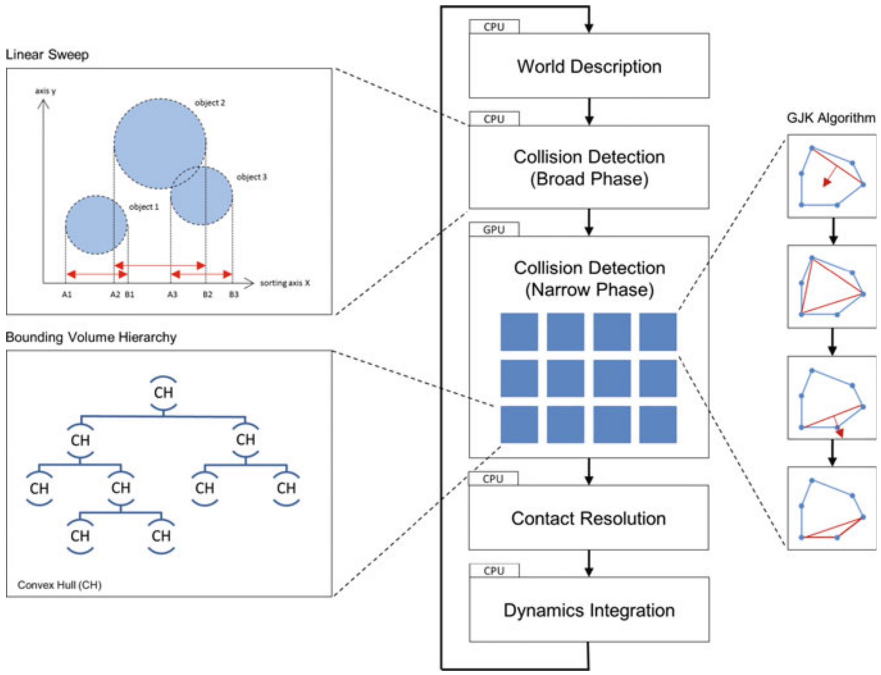


Fig. 8 System overview. The broad phase of the collision detection runs on the CPU and leads to a list of potential candidates that require further investigation. Upon receiving the collision pair list, the narrow phase operates on the GPU searching whether the two objects in each pair are colliding, or for their nearest distance. The collision results are returned to the CPU where contact resolution and dynamics are computed

this reason two sets of contact space parameters x and $x_\oplus$ are defined. Vector $x_\oplus$ consists of the full set of contact space parameters (one per contact point). A subset of the contact parameters $x \subseteq x_\oplus$ is defined and contains only the minimum active contact points of the entire contact space. A contact point is considered active if the force or impulse applied at the contact point is non-zero. It is important to note that this classification is unknown when the collision detector reports the list of contact points, but is identified later (Fig. 8).

From the algorithm proposed by Ruspini and Khatib [23], a selection matrix S such that $x = Sx_\oplus$ selects the members of $x_\oplus$ that belong to x. Given these spaces we can define other parameters for velocity $v_\oplus = \dot{x}_\oplus$, acceleration $a_\oplus = \ddot{x}_\oplus$, forces $f_\oplus = Sf$, and impulses $p_\oplus = Sp$. At the time of contact/collision t, $s_\oplus = 0$, and a Jacobian $v_\oplus = J_\oplus \dot{q}$ can be found. Furthermore, the augmented operational contact inertia matrix can be defined as follows:

$$\Lambda_\oplus{}^{-1} = J_\oplus A^{-1} J_\oplus{}^T. \tag{3}$$

This matrix is very similar to the operational space inertia matrix used in robotic control [17], and represents the effective mass perceived at all the contact points and characterizes the dynamic relationships between the contact points. Being composed from a set of independent contact parameters, the inverse active contact space inertia matrix

$$\Lambda^{-1} = S{\Lambda_{\oplus}}^{-1}S^T \tag{4}$$

is positive definite and therefore invertible.

If two or more bodies in the system are colliding, then some elements of $v_{\oplus}$ are negative. In a system of rigid bodies an impulse must be applied to prevent the objects from interpenetrating. Since the nature of the deformations that occurs during a real collision are quite complex, several analytical methods have been proposed to compute the needed impulse forces. Here we will examine one of the most common models for rigid body collision. This framework is sufficiently general to allow other contact models to be used.

A common empirical model is to require that for each active contact point, the velocity after the collision must be $-\varepsilon$ times the relative velocity of the contact point prior to the collision, where ε is a known coefficient of restitution. This constraint can be written as:

$$v^{+} = -\varepsilon v^{-}, \tag{5}$$

where v^{-} and v^{+} are the relative velocity vectors of the contacts before and after the collision. The above constraint describes only the behavior of the active contact points. At these points the impulse force must be greater than zero:

$$p > 0. \tag{6}$$

Lastly, an additional constraint is required to constrain the motion at all the contact points:

$$v^{+} \geqslant -\varepsilon v^{-}. \tag{7}$$

The active contact points satisfy this constraint by default. The constraint requires that if a contact force is inactive (contact impulse force is zero), then the relative velocity at the contact point must be at least as large as it would be if the point were active. The zero impulse force requirement on the non-active contact points can be achieved by defining

$$p_{\oplus} = S^T p \tag{8}$$

Deriving the impulse constraint equations is covered in detail in [23] and is summarized here:

$${p_{\oplus}}^T \Lambda_{\oplus}^{-1} p_{\oplus} + (1+\varepsilon) p_{\oplus}^T J_{\oplus} q^{-} = 0, p_{\oplus} \geqslant 0, \Lambda_{\oplus}^{-1} p_{\oplus} \geqslant -(1+\varepsilon) j_{\oplus} q^{-} \tag{9}$$

The resulting system of equations can be solved using a quadratic programming package. Once a solution of the augmented contact space impulse vector $P_{\oplus}$ has been found, it is easy to compute the vector of active contact parameters and the selection matrix S. The non-zero terms of $p_{\oplus}$ form the active set of contact points p. The post collision in configuration space velocities $\dot{q}^{+}$ created by a contact space impulse p is given by

$$\dot{q}^{+} = \dot{q}^{-} + \triangle\dot{q} = \dot{q}^{-} + A^{-1}J^{T}p, \tag{10}$$

and the integration of the equations of motion continue from this updated state.

5 Experimental Results

To evaluate the performance and correctness of this framework, we developed an air table to record the motion of articulated bodies moving and colliding freely on a frictionless plane. The table integrates a smooth surface perforated by miniature air holes placed along a grid pattern at 25 mm intervals with dimensions of 180 cm $\times$ 100 cm. A high air flow and high static pressure fan (San Ace 172 from Sanyo) was used to create a cushion of air sufficient to lift the objects.

To measure the location and speed of the objects floating on the table, we used a high resolution tracking camera (Optitrack Flex 13 from Naturalpoint) to track their position and orientation. This camera offers a resolution of 1280 $\times$ 1024 pixels with a frame rate of 120 Hz. The location of each object on the table was estimated by tracking the position of a set of reflective markers attached to each object. The software library provided with the tracking device offered the means to identify each marker location, its size, and roundness using proprietary imaging techniques running in real-time. The linear and angular velocities of each object were then derived by measuring the displacements of the different markers at regular time intervals.

The circular discs used for this experiment were constructed from acrylic. This material was selected for its higher coefficient of restitution ($\varepsilon_{measured} = 0.87$) and lower coefficient of friction. Each disc weighed 35 g with a diameter of 100 mm. The discs were either used alone (unjointed configuration) or connected in groups of three using a pair of articulated links (jointed configuration) (Fig. 9).

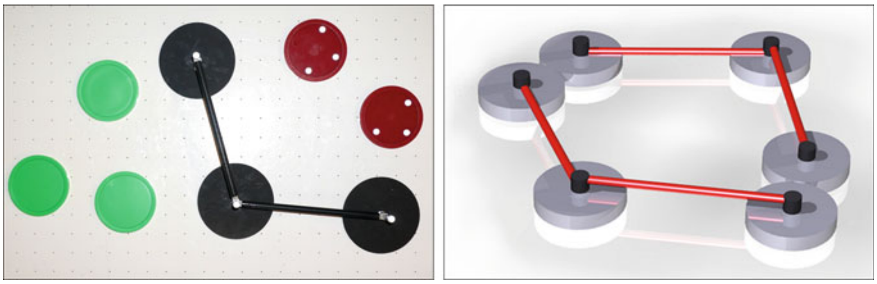


Fig. 9 Experimental air table. (*Left*) A selection of jointed and unjointed discs placed on our experimental air table. (*Right*) A virtual representation of a linked object used in our simulation framework

In the jointed configuration, the discs were connected using a rigid shaft made of carbon fiber. A second shaft would connect the third disk to the first link through a free revolute joint. A high precision ball bearing was used to reduce joint friction to negligible levels.

A series of experiments were conducted by having the objects collide against each other or against the edges of the air table. During each experiment the camera would track their motion and compute their linear and angular velocities.

The objective of the experiment was to measure the linear and angular velocities *before* and *after* impact, and to compare these results with simulated velocities computed *after* impact given some initial conditions measured *before* impact. To also compare the performances of our simulation framework with other simulation libraries, we performed the same experiments using three well-known and commonly used simulation frameworks namely, ODE [24], Bullet [12], and Moby [13].

For the purpose of our simulations, each disc was modeled using a polygonal representation composed of 720 triangles. To model the properties of the table, the virtual discs were constrained to a two-dimensional frictionless plane, while the (optional) articulated links were modeled as constraints between the discs. Collisions were computed between the different bodies and the edges of the air table, and the integration time step for all simulations was set to 0.1 ms.

The results of our physical experiments are presented in the following sections for *unjointed* and *jointed* configurations. At the beginning of each experiment, the objects were placed on the air-flow enabled table. The operator provided a short impulse by hand to drive the objects toward a target direction where the collisions would occur. During this time live images were acquired by the tracker in real time while the position of each marker was extracted from the imaging data and recorded to memory. From the recorded data, the velocities were estimated at every time step. The time of each impact was identified by evaluating the shape of the recorded trajectories and by searching for sudden velocity changes. Once these conditions were identified, the measured data was programmed into the simulator to configure the virtual models with the same conditions as in the real experiment shortly before impact. From that point on, the simulator integrated the scene over a short time interval until the first collision occurred; at that point the new velocities were estimated for each body. These new output velocities were then compared against the recorded values and the relative error estimated for each simulator. The physical properties of each object (mass, inertia, and center of mass) were estimated directly from the CAD models developed for the experiment.

5.1 Experiment: Collisions with Free Bodies

In our first experiment three discs were simultaneously propelled on the surface of our air table at velocities ranging from 0.2–0.5 m/s. The trajectories were recorded for the three discs and their first collision with an edge of the table or another disc identified. Their respective linear and angular velocities were measured shortly before and after impact. In the tables presented in Table 2 we compare the velocity values after impact and display the results for the measured and simulated systems. For unjointed bodies, we observed that the simulated linear velocities remained within a relative error range of 12 % compared to the measured values. These disparities are consistent given the limited accuracy of the tracking device, the error of the estimated coefficient of restitution, and the unaccounted for friction effects between the surfaces of the objects. Furthermore, additional experiments showed a high sensitivity in the estimated angular velocity when lower resolution polygonal models were used to estimate collisions. These disparities come from the discretization of a curved continuous surface into a finite set of polygons. The differences between the different simulators were also caused by the uneven accuracy of the collision algorithms reporting slightly different values for the time and location of the collision impact.

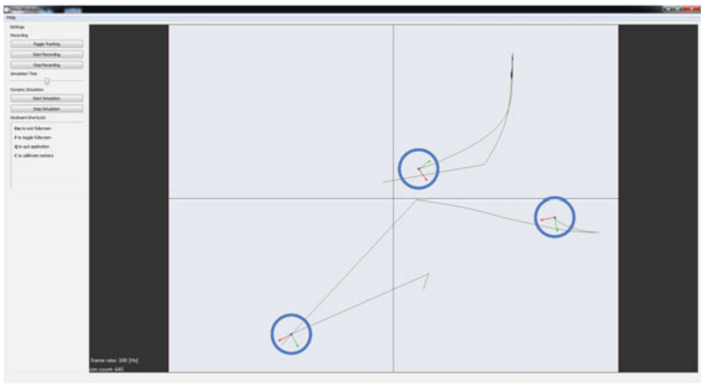


Table 2 Experiment using unjointed discs

System	Linear velocity (m/s)	Relative error (%)	Angular velocity (rad/s)	Relative error (%)
Measured	0.221	–	0.184	–
Stanford	0.231	4.5	0.176	4.3
ODE	0.243	9.9	0.168	8.7
Bullet	0.239	8.1	0.171	7.0
Moby	0.216	2.2	0.177	3.8
System	Linear velocity (m/s)	Relative error (%)	Angular velocity (rad/s)	Relative error (%)
Measured	0.388	–	4.41	–
Stanford	0.346	10.8	4.07	7.7
ODE	0.448	15.4	4.94	12.0
Bullet	0.432	11.3	4.85	9.9
Moby	0.362	6.7	4.02	8.8

(continued)

Table 2 (continued)

System	Linear velocity (m/s)	Relative error (%)	Angular velocity (rad/s)	Relative error (%)
Measured	0.442	–	0.72	–
Stanford	0.410	7.2	0.78	8.3
ODE	0.512	15.8	0.64	11.1
Bullet	0.488	10.4	0.68	5.5
Moby	0.421	4.7	0.77	6.9

Measured and simulated velocities for three different collisions immediately after impact

5.2 *Experiment: Collisions with Articulated Bodies*

In this second experiment we evaluated our contact model using the articulated system composed of three discs presented above. Similar to the first experiment, the articulated system is manually propelled on the surface of our air table with a linear velocity ranging from 0.2–0.5 m/s. The trajectories were recorded for each disc, and the first collision between one of the discs and an edge of the table was identified. In this experiment we observed how the collision impact affected the motion of the opposite link. For comparison purposes, we estimated the joint angle q (see Table 3) and derived the angular velocity $\dot{q}$ immediately before and after impact. For the jointed configuration we observed a much larger disparity between the different simulators. This greater disparity was likely caused by the uneven accuracy of the different collision algorithms as reported in the first experiment. However, the more accurate results observed with our proposed simulation framework lead us to conclude that the combination of Featherstone's approach [15] with Ruspini's contact resolution algorithm offers a more accurate way to model collisions between complex articulated body systems.

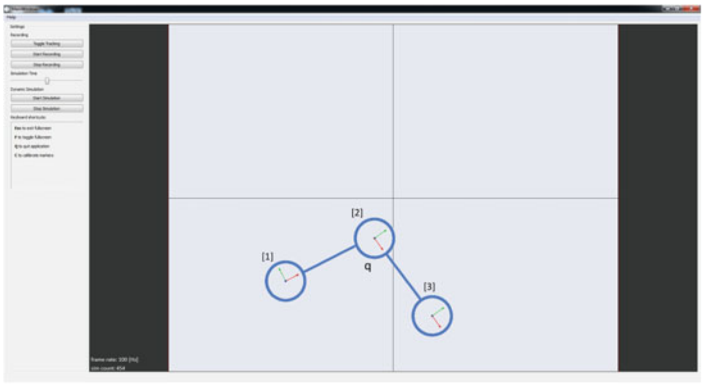


Table 3 Experiment using jointed discs

System	Joint velocity ($\dot{q}$) (rad/s)	Relative error (%)
Measured	0.721	–
Stanford	0.654	9.2
ODE	0.489	32.2
Bullet	0.520	27.8
Moby	0.636	11.8
System	Joint velocity ($\dot{q}$) (rad/s)	Relative error (%)
Measured	2.441	–
Stanford	2.136	12.5
ODE	1.743	28.6
Bullet	1.986	18.6
Moby	2.108	13.6

The object is composed of two links connected through a free revolute joint. The table presents the measured and simulated joint velocity after impact

6 Conclusion

In this work we introduced a unified framework that permits the dynamic computation and contact resolution of complex articulated multi-body systems with real-time performances. A parallel numerical method was proposed for efficient collision detection. The results obtained with the proposed method demonstrate that the new GPU version of the collision detector operates about 10–15 times faster than the earlier CPU version. Furthermore, to validate our dynamic models, we evaluated the proposed simulation framework by comparing simulated models against recorded motions of real articulated bodies colliding on a frictionless air-table. Initial experimental results showed that for either jointed or unjointed mechanical systems, the error rate between the simulated and actual measured velocities was contained within a 15 % error margin. Future work will include further understanding the degree of accuracy of our models when many collisions occur simultaneously. New collision models will also be developed to simulate objects of different kinematics, material stiffness, and friction properties. We also plan to further refine our experimental air table by using camera devices that offer higher resolution images with faster acquisition times. Finally, our research has mostly focused on rigid articulated bodies. The ability to model and interact with deformable models is also of great interest, in particular in the areas of surgery planning and simulation. To achieve these goals, new dynamic models and collision detection algorithms for deformable models will need to be developed to meet the real-time requirements of interactive dynamic simulation, but without introducing significant approximations or simplifications.

References

1. Bajaj, C.L., Dey, T.K.: Convex decomposition of polyhedra and robustness. SIAM J. Comput. **21**(2), 339–364 (1992)
2. Baraff, D.: Analytical methods for dynamic simulation of non-penetrating rigid bodies. In: Thomas, J.J. (ed.) SIGGRAPH, pp. 223–232. ACM (1989)
3. Baraff, D.: Curved surfaces and coherence for non-penetrating rigid body simulation. In: Baskett, F. (ed.) SIGGRAPH, pp. 19–28. ACM (1990). ISBN 0-201-50933-4
4. Baraff, D.: Coping with friction for non-penetrating rigid body simulation. In: Thomas, J.J. (ed.) SIGGRAPH, pp. 31–41. ACM (1991). ISBN 0-89791-436-8
5. Baraff, D.: Issues in computing contact forces for non-penetrating rigid bodies. Algorithmica **10**(2–4), 292–352 (1993)
6. Baraff, D.: Fast contact force computation for nonpenetrating rigid bodies. SIGGRAPH, pp. 23–34. ACM (1994). ISBN 0-89791-667-0
7. Baraff, D.: Linear-time dynamics using lagrange multipliers. SIGGRAPH, pp. 137–146 (1996)
8. Chang, K.S., Khatib, O.: Operational space dynamics: efficient algorithms for modeling and control of branching mechanisms. Proc. IEEE Int. Conf. Robot. Autom. **1**, 850–856 (2000)
9. Chazelle, B.: Convex partitions of polyhedra: a lower bound and worst-case optimal algorithm. SIAM J. Comput. **13**(3), 488–507 (1984)
10. Chazelle, B., Palios, L.: Triangulating a nonconvex polytope. Discret. Comput. Geom. **5**, 505–526 (1990)
11. Conti, F., Barbagli, F., Balaniuk, R., Halg, M., Lu, C., Morris, D., Sentis, L., Warren, J., Khatib, O., Salisbury, K.: The chai libraries. Proc. Eurohaptics **2003**, 496–500 (2003)
12. Coumans, E.: Bullet Physics Library. http://www.bulletphysics.org (2001)
13. Drumwright, E.: Moby Simulator. http://physsim.sourceforge.net (2008)
14. Ehmann, S.A., Lin, M.C.: Accurate and fast proximity queries between polyhedra using convex surface decomposition. Comput. Graph. Forum **20**(3), 500–511 (2001)
15. Featherstone, R.: Robot dynamics. Scholarpedia **2**(10), 3829 (1987)
16. Gilbert, E.G., Johnson, D.W., Keerthi, S.S.: A fast procedure for computing the distance between complex objects in three-dimensional space. IEEE J. Robot. Autom. **4**(2), 193–203 (1988)
17. Khatib, O.: A unified approach for motion and force control of robot manipulators: the operational space formulation. IEEE J. Robot. Autom. **3**(1), 43–53 (1987)
18. Khatib, O., Brock, O., Chang, K.S., Conti, F., Ruspini, D.C., Sentis, L.: Robotics and interactive simulation. Commun. ACM **45**(3), 46–51 (2002)
19. Quinlan, S.: Efficient distance computation between non-convex objects. ICRA, pp. 3324–3329. IEEE Computer Society (1994). ISBN 0-8186-5330-2
20. Rappoport, A.: The n-dimensional extended convex differences tree (ecdt) for representing polyhedra. In: Symposium on Solid Modeling and Applications, pp. 139–147 (1991)
21. Ruppert, J., Seidel, R.: On the difficulty of triangulating three-dimensional nonconvex polyhedra. Discret. Comput. Geom. **7**, 227–253 (1992)
22. Ruspini, D.C., Khatib, O.: Collision/contact models for the dynamic simulation of complex environments. In: 9th International Symposium of Robotics Research (ISRR'99), pp. 185–195. Snowbird (1997)
23. Ruspini, D.C., Khatib, O.: A framework for multi-contact multi-body dynamic simulation and haptic display. In: Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000), vol. 2, pp. 1322–1327 (2000). doi:10.1109/IROS.2000.893204
24. Smith, R.: Open Dynamics Engine. http://www.ode.org/ (2001)

Part III
Intelligence and Learning

Constructive Developmental Science: A Trans-Disciplinary Approach Toward the Fundamentals of Human Cognitive Development and Its Disorders, Centered Around Fetus Simulation

Yasuo Kuniyoshi

Abstract How does human mind develop? What causes developmental disorders? Recent studies suggest the importance of the fetal period in human development. However, study of human fetuses is strongly constrained by technical and ethical difficulties. This project aims at understanding the principles of human development by analyzing and modeling it from the fetal period. Integrating robotics, medicine, psychology, neuroscience, and Tohjisha-kenkyu (first-person view research of developmental disorders), we establish a new trans-disciplinary research field called Constructive Developmental Science. Its contributions include a new understanding of human development and its disorders, comprehensive diagnostic methodologies, and truly appropriate assistive technology.

1 Introduction

A quest for the constructive principles of intelligent systems for truly adaptive behavior in the real world has gone through piece-wise cognitive modeling and learning mechanism studies and has recently started to address the continuous developmental process of human cognition from the very beginning via the complex interactions between body, environment and nervous system [19].

In medical studies of developmental disorders, the main focus has been on identification of genetic correlates. But recently the importance of the environmental factors from the perinatal period has been recognized [8], and a growing number of reports [11, 15, 17] point out the risk factors of preterm infants who experience abbreviated gestational periods and correlation of abnormal neonatal motor patterns to later developmental disorders. Therefore understanding the fetal/neonatal development has become an urgent and important issue.

In 2012, a five-year project "Constructive Developmental Science", led by Yasuo Kuniyoshi, supported by MEXT Grant-in-Aid for Scientific Research on Innovative

Y. Kuniyoshi (✉)
The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
e-mail: kuniyosh@isi.imi.i.u-tokyo.ac.jp

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_17

Areas with total funding of 1.1 billion yen, was launched in Japan. It addresses the above described convergent focus of study, i.e. human development from the fetal period and its relationship to developmental disorders, by truly trans-disciplinary collaboration between roboticists, cognitive/neuro-scientists, medical scientists and Tojisha-Kenkyu (first person view study of developmental disorders).

2 Organization of Constructive Developmental Science

Figure 1 shows an overview of the organization and aims of Constructive Developmental Science.

Tojisha-Kenkyu (C) undertakes a phenomenological or internal observer investigation of developmental disorders and propose hypotheses to be shared by all project members. Already, they have proposed that the essential characteristic of autism spectrum disorders (ASD) may not be the impairment of social cognition, as commonly assumed so far, but the difficulty of information integration starting at the level of bodily sensations. They are also working on the advanced methodologies of Tojisha-Kenkyu as well as development of novel assistive technologies for people with developmental disorders.

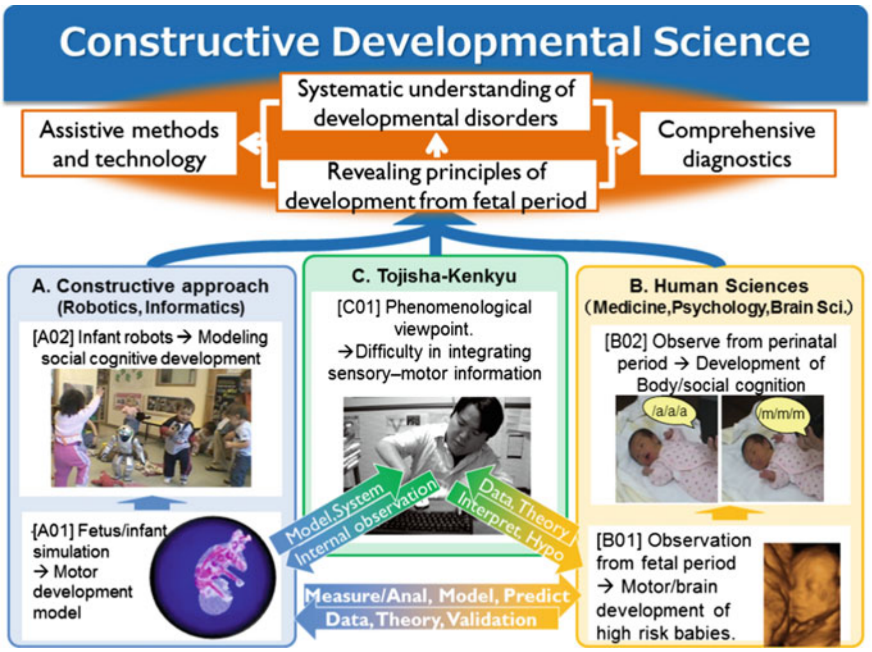

Constructive Developmental Science
Systematic understanding of developmental disorders
Assistive methods and technology
Comprehensive diagnostics
Revealing principles of development from fetal period
A. Constructive approach (Robotics, Informatics)
[A02] Infant robots → Modeling social cognitive development
[A01] Fetus/infant simulation → Motor development model
C. Tojisha-Kenkyu
[C01] Phenomenological viewpoint. → Difficulty in integrating sensory–motor information
Model, System
Internal observation
Data, Theory
Interpret, Hypo
Measure/Anal, Model, Predict
Data, Theory, Validation
B. Human Sciences (Medicine, Psychology, Brain Sci.)
[B02] Observe from perinatal period → Development of Body/social cognition
/a/a/a
/m/m/m
[B01] Observation from fetal period → Motor/brain development of high risk babies.

Fig. 1 Organization and aims of “Constructive Developmental Science”

One of the two groups in Human Science (B) is the medical group (B01) covering obstetrics/gynecology, pediatrics and psychiatrics. They undertake a systematic continuous observation and measurements of early human development starting from fetal period to 3-year-old until the diagnosis of developmental disorders can be made. The other group consists of cognitive scientists (B02). They will focus on how bodily sensation and social cognition are related in early human development.

Constructive approach (A), consisting of roboticists and neuroscientists etc. plays an integrative role, based on the shared hypothesis from Tojisha-Kenkyu and data from Human Sciences, construct simulation/robotic models of continuous human development and conduct experiments. Group A01 constructs a simulation model of fetal/neonatal development which plays a core role in establishing a working integration of data and theories from other groups. Group A02 models infant-caregiver interactions and develop novel assistive technologies based on the findings.

In the following sections, after reviewing the trends and the state of the art of developmental science, recent progress of simulation study of human fetal development is presented, with discussions about future steps and open problems.

3 Developmental Sciences Converging on Fetal Development

Human mind and behavior emerge from the complex interactions between the genes, body, nervous system and various environments from uterus to the real world with physical/social/cultural structures (Fig. 2), and continuously change over the lifetime [27].

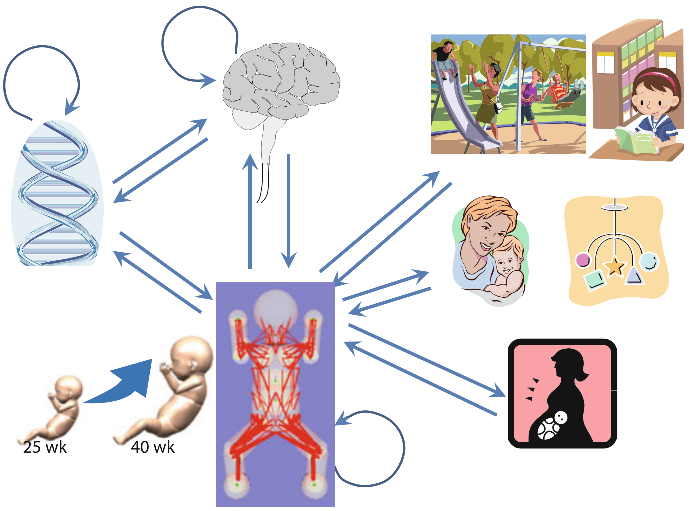


Fig. 2 Complex interaction between genes, body, nervous system and various environments

Thus a reductionist approach is insufficient for one to fully comprehend the nature of human development and its disorders. Rather, it is necessary to investigate the entire structure of total interactions and the principles of their change.

The changes of the interactions are caused by the interactions themselves. Thus it is a continuous boot-strap process. Understanding such a process requires starting from the genesis of the interactions and revealing the logic of their changes.

Individual developments follow different trajectories while converging from time to time on certain capabilities like crawling, walking, speaking at certain ages. Some trajectories deviate from the typical group, often regarded as "disorders"at certain timings exhibiting different capabilities, and some of such trajectories can later converge with the typical ones (Fig. 3). Understanding this global dynamical structure, being able to account for individual trajectories as well as the global convergence structures, is the ultimate goal of developmental science.

Constructive developmental approach [2, 18, 21] attempts to reveal the principles of development by re-constructing the total continuous development of the interactions from their genesis.

A constructive approach, first used in the field of complex systems science, consists of the following steps; (1) Abduction: set a hypothetical generating principle, (2) Simulate: embed the principle in its environment and start the dynamics, (3) Observe & Feedback: observe the resulting behavior, compare with the target of investigation, and modify the generating principle based on the error. Continue the cycle until the error minimizes (Fig. 4).

In the field of developmental cognitive robotics [3, 18, 20], although learning models for isolated cognitive functions have been proposed, a model which unifies them into a working principle of continuous development is yet to be proposed.

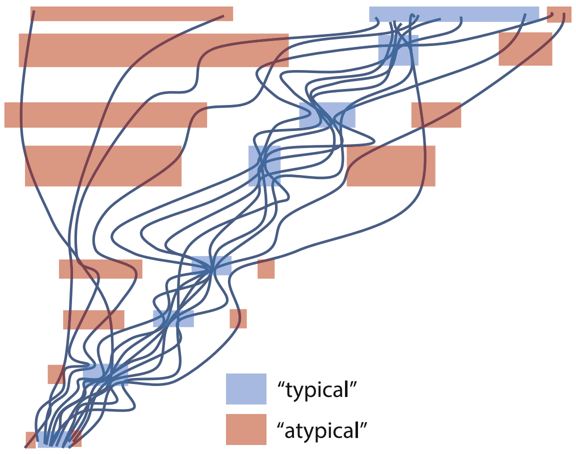


Fig. 3 Global structure of developmental trajectories. Each *curved line* represent individual developmental pathway. *Rectangles* represent typical functional convergences and atypical symptoms

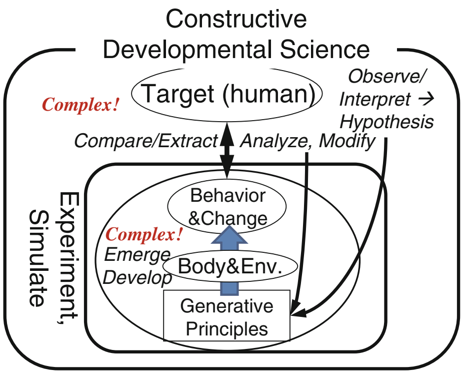


Fig. 4 Methodology of Constructive Developmental Science

Kuniyoshi et al. [19, 21] proposed the world's first computer simulation of a human fetus consisting of the body, nervous system and environment (Fig. 5 right), and demonstrated that a connection between the subcortical nervous system and the dynamics of the body and environment causes the emergence of a variety of natural motor behaviors. And the sensory-motor information generated by these motor behaviors drives self-organization of cortical maps. Thus, Kuniyoshi et al. substantiated the role of the physical body in the development of the human brain, the "Body Shapes Brain" principle, by the constructive approach.

In developmental psychology, neuroscience and medicine (i.e. obstetrics/ gynecology and pediatrics), recent advances in measurement and experimental methodologies evoked rapid growth in our knowledge of fetal and infant motor and

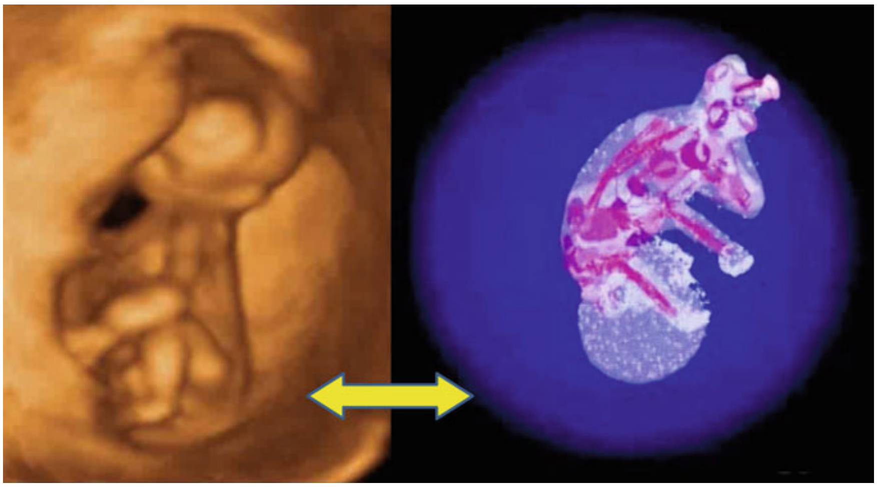


Fig. 5 "4D" image (*left*, courtesy of Prof. Konishi, Doshisha Univ.) and a simulation model (*right*) of a of a human fetus

brain development, sensory-motor responses and learning abilities. For example, the now highly popular real-time 3D ultra-sound imaging (also called 4D ultrasound) enables the detailed visualization of fetal movements and even changes in facial expressions (Fig. 5 left). And fetal brain imaging (MRI) is carried out at a scale of thousands.

Recent medical research has focused on the identification of genetic correlates of developmental disorders, especially Autism Spectrum Disorder (ASD), Attention Deficit Hyperactivity Disorder (ADHD) and Learning Disability (LD). However, recently it has been pointed out that identification of genes falls short of providing full explanations about human developmental disorders; rather, detailed investigation of environmental factors is necessary [8].

Recent cohort studies report that abnormalities in the perinatal environment as well as the abbreviation of gestational periods as experienced by preterm infants correlate with higher risks of developmental disorders [11, 15, 17]. Multiple research fields seek to understand the relationship between perinatal environmental interaction and human development.

In Japan, a unique systematic research methodology called Tojisha-kenkyu (first person view, peer-supported research) is evolving from mere autobiographic notes also seen in other countries. It provides novel scientific hypotheses based on internal observer view of developmental disorders as experienced by the researchers themselves [4].

One study states that ASD as experienced first-person is not simply “an impairment of social cognition” as conventionally defined, but is rather “peculiarities of somatic sensations” and “difficulties of integrating information”.

To summarize, there is a consensus between basic human sciences and the constructive approach that environmental interactions during the perinatal period are crucial in human development, and to our advantage, advanced technology for measurements and experiments in this key period is at our disposal. And Tojisha-kenkyu is also focusing on the foundation of cognition preceding sociality. Thus the time is ripe to redefine the field of developmental science via trans-disciplinary collaborations.

4 Computer Simulation of Early Human Development

Advances in imaging technology such as ultrasound has allowed us to confirm that prenatal human fetuses are in fact moving quite actively while developing inside the uterus, and even exhibit learning abilities.

By computer simulating this active developmental processes, we attempt to reveal the core principles of development and also how environmental factors affect the processes.

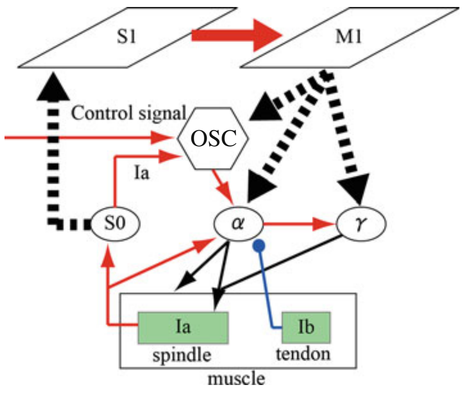


Fig. 6 Central/peripheral nervous system model. Broken arrows denote learning connections. S1:primary somatosensory area, M1: primary motor area, OSC: neural oscillator, S0: spinal sensory neuron, α: spinal α motor neuron, γ: spinal γ motor neuron, spindle: muscle spindle, tendon: Golgi tendon organ. In experiments using tactile sensation, tactile receptors are accommodated and connected to OSC and α with learning connections

As a platform for conducting the simulation, we developed a unified model of the human fetus/infant consisting of musculoskeletal body structure, sensory organs and (partial) nervous system (Fig. 6) placed inside a simulated uterine environment [19, 23].

4.1 Fetus and Infant Model

We conducted a thorough survey of anatomical literature and also collected physiological data from medical and biological research institutions to determine the structure and parameters of 198 main skeletal muscles as well as muscle spindles and Golgi tendon organs comprising the proprioception, along with the dimensions of each body part such as its length, mass and inertia in order to reproduce a realistic human fetus/infant body model (Fig. 5). The body parameters can be set according to a desired gestational age (in weeks), in order to simulate the continuous physical growth of a human fetus.

About 1500 tactile receptors are distributed over the surface of the body, enabling the simulation of self-contact, contact with the uterine wall and resistance due to amniotic fluid if inside the uterus and self-contact and contact with the wall and ground if outside the uterus.

In order to assess whether differences in perinatal sensory experience affect development, we distribute the tactile receptors in three different ways: (1) human-like (with regard to two-point discrimination thresholds), (2) uniformly over the entire body and (3) inverse human-like (Fig. 7).

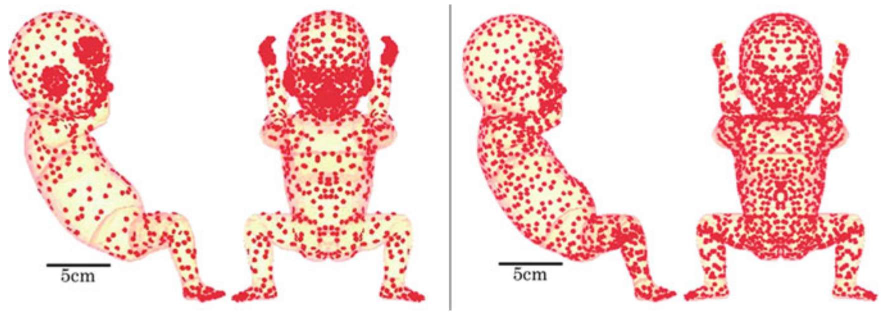


Fig. 7 Tactile receptor distribution. *Left*: human-like. *Right*: uniform

Vision is also incorporated to assess the role of multimodal sensory integration in development.

4.2 Simulating the Development of Human Fetus and Infant

Human fetal motor development is divided into 16 sequential temporal phases according to de Vries et al. [6]. Connection between the fetus' neocortex and the spinal cord has been confirmed at the earliest in the 16th gestational week in humans. Prior to that, spontaneous primitive motor behaviors known as General Movements (GMs) are observed.

GMs are driven by neural oscillators in spinal and brainstem circuitries and generate whole-body motor behavior. Our hypothesis is that they promote self-organization in neuronal connectivity resulting in developmental changes of motor behaviors. Therefore, sensory-motor experiences in the fetal period may have wide consequences in development.

Assuming intrauterine sensory experience is dominated by somatosensation, motor development was compared for the three different distributions of tactile receptors described above (Fig. 7).

In these experiments, tactile sensory inputs are connected to the neural oscillators and $alpha$ motor neurons in the spinal circuitry, in addition to the proprioceptive reflex circuits. The cortical models (S1, M1) are dissociated because the target of simulation is the period before the cortico-spinal connections start to function.

For each condition, the fetus simulation underwent a separate learning phase in the simulated intra-uterine environment, followed by an observation phase during which motor behavior was measured for analysis.

Our analysis shows that if and only if tactile receptors are distributed in human-like pattern (Fig. 8 left) do isolated limb movements (sudden arm or leg movements) and hand-to-face contacts persistently increase in qualitatively similar patterns as

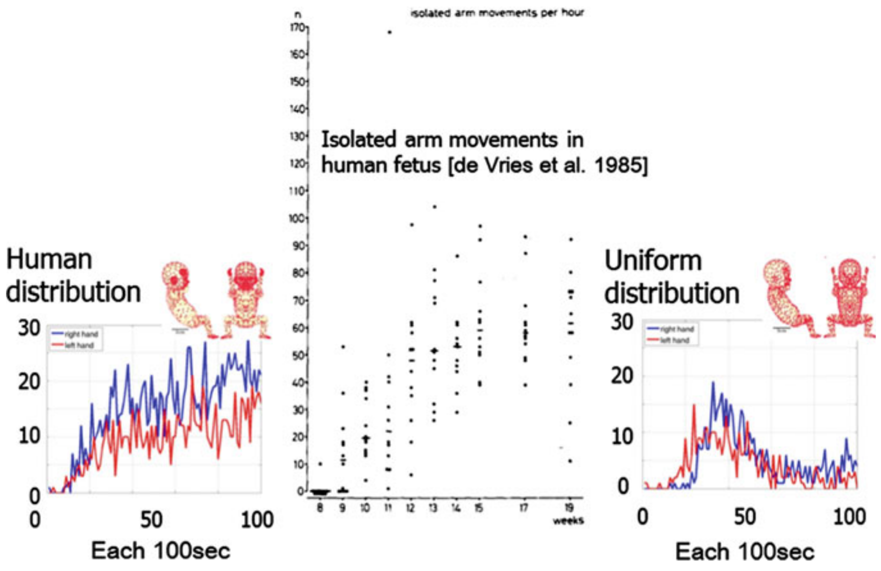


Fig. 8 Occurrence trend of independent (jerky) arm motion. *Left*: with human-like tactile distribution, comparable to, Center: human fetus data, but different from *Right*: uniform distribution. The center graph is adopted from Fig.11 of de Vries et al. 1985 [7]

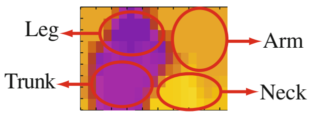


Fig. 9 Topographic body map of *right* M1$\Rightarrow \alpha$ at t = 12,000 s

observed in human fetal development [7]. Such patterns collapse if tactile receptors are not distributed according to human physiology (Fig. 8 right) [23].

In addition, we simulated fetal and infant development during the perinatal period using the full version of the nervous system incorporating the self organizing neural network models of the primary somatosensory area and primary motor area of the human cerebral cortex (Fig. 6).

Analysis of the post-learning cortical network confirmed that a topographic map representing the body parts was self-organized (Fig. 9). Such cortical representations may be precursors to body schema and motor representations.

The above results suggest the following developmental principle: The information structure inherent in embodiment is manifested via body movements driven by the subcortical nervous system, into the behavior patterns and sensory-motor information structures, which are reflected in the self-organization of the central nervous system.

Another important account is that, through experimentation, factors which may or may not cause macroscopic effects on development can be differentiated. We have also shown ([23]) the possibility that some *innate* reflexive motor behaviors may be acquired via prenatal motor learning.

4.3 Towards Body Schema and Social Cognition

Recently, we have made the following research progress:

1. Demonstration of the importance of amniotic fluid presence in cortical somatosensory map formation [26].
2. In a series of simulation experiments where parameters controlling the fetal environment, fetal motility, and nervous system are each altered, we found that abnormalities in the body representation in the cortical somatosensory area occurs. The abnormal conditions correspond to the following known characteristics of preterm infants; Early exposure to the extra-uterine environment, reduced complexity/variety of motor behavior, and abnormal excitation/inhibition balance of the cortical neural network [29].
3. With respect to multisensory integration, fetal experiences give rise to bi-modal neurons which integrate proprioception and vision of the hand [10].
4. Imitation of facial expressions by infants, which leads to social awareness, can be explained by development of *superior colliculus* during the fetal period [25].
5. G. Taga's team in our group A01 established correlations between GM features of preterm infants at term age and developmental delays at age 3 [16].
6. T. Inui in our group A01 proposed a unified framework of development and disorders of a cortical network related to social cognition [12]. Based on the cortical network abnormalities, an etiology of Williams syndrome and ASD is proposed, with accounts for their various symptoms.

All the above findings account for the relevance of very early development to later cognitive capabilities. Result 5 establishes that early motor behavior is relevant to developmental delays measured three years later. This suggests that early motor behavior and later cognitive capabilities share a common basis.

The findings 1–4 above suggest relevance of insufficient intra-uterine sensory-motor experiences to increased risks of developmental disorders with preterm infants. They also implicate that complications in the uterine environment such as due to fibroids may also affect early development. Such hypotheses can be substantiated by constructive approach altering environmental and temporal parameters in the fetal development simulation.

4.4 Open Problems

Although the current fetus and infant simulation model is drastically simplified compared to real human babies, the experiments using the model can capture some essential characteristics of early development, providing invaluable means for understanding the fundamental principles.

Our approach provides plausible causal explanations on how some fetal/perinatal environmental/bodily/neural factors may lead to developmental disorders. It has the potential to deepen our understanding of the principles of human development as well as to improve the diagnosis and selection of proper clinical care for patients with developmental disorders.

Several important open issues must be addressed if our approach is to truly contribute to the understanding of developmental disorders.

First, the temporal period which can be simulated must be largely extended to cover up to the three-year old threshold when medical diagnoses for developmental disorders are usually confirmed.

Although it is no simple task, our current goal is to construct a continuous developmental model which covers from prenatal to infant (one-year old) periods in order to understand which factors contribute to the formation of the first foundation of human cognition.

In order to connect the infant model to the three-year-old, existing (and evolving) developmental models of higher cognitive functions from fields such as developmental cognitive robotics and developmental cognitive neurosciences [14] can be integrated to account for how the infant cognition develops to acquire social cognition or fails to do so.

5 Conclusions

Our project is the world's first endeavor to construct a simulation of continuous human development from fetus to infant in terms of continuous complex interactions between body, environment and nervous system, and make comprehensive comparisons with clinical data, guided and evaluated by first-person observations and hypotheses by Tojisha-Kenkyu, attempting to establish a deep unified understanding of developmental principles and disorders.

There are many existing research on large-scale simulations of the nervous system such as the Blue Brain project [22], now evolved into Human Brain Project (HBP) [28], among others [1, 9, 13]. Modeling and simulation of human musculoskeletal system is also an established field ([24] and others). However, there are no other simulations in the world which allow continuous development while interactions taking place between the nervous system, whole body and environment.

There is also significant recent progress in modeling human cognitive development, but the majority of research has focused on postnatal acquisition of isolated cognitive functions. Our group is unique in that we model cognitive development beginning in the prenatal fetal period.

In medical sciences, while tremendous progress has been made in imaging technologies, there are severe limitations to physiological experimentation on human fetuses in terms of experimental techniques as well as ethical considerations.

A constructive approach to developmental science which combines clinical observation and computer simulation provides an entirely new approach to the investigation of fetal development and can be expected to benefit in the comprehensive clinical diagnosis and care of individuals with developmental disorders; the approach may be called "Simulation-Based Medicine".

Also, by having the models and findings evaluated under the scrutiny of medicine and by focusing our efforts in ensuring that data precision and reliability meet the standards of modern clinical research, we aim to solidify the rising potential of the emerging field of Constructive Developmental Science [5].

Acknowledgments This research is supported by MEXT Grant-in-Aid for Scientific Research on Innovative Areas 24119001, 24119002. Yukie Nagai, Yukuo Konishi, Masako Myowa-Yamakoshi and Shin'ichiro Kumagaya serve as the principal investigators of the research groups A02, B01, B02, C01 respectively. We thank the researchers who have contributed to the planning and execution of this research, in particular, Minoru Asada, Toshio Inui, Ichiro Tsuda, as well as the fellow colleagues who have engaged in our discussions. We thank researchers and students for their productive contribution to the progress of this research.

References

1. Ananthanarayanan, R., Modha, D.S.: Anatomy of a cortical simulator. In: Proceedings of the Supercomputing '07: the ACM/IEEE SC2007 Conference on High Performance Networking and Computing, pp. 1–12. Association for Computing Machinery Press, New York (2007)
2. Asada, M., Kuniyoshi, Y.: Robot Intelligence. Iwanami (2006) (In Japanese)
3. Asada, M., et al.: Cognitive developmental robotics as a new paradigm for the design of humanoid robots. Robot. Auton.Syst. **37**(2–3), 185–193 (2001)
4. Ayaya, S., Kumagaya, S.: Hattatsu-shogai Tojisha-kenkyu (First-Person Research of Developmental Disorders). Igaku-Shoin (2008). (In Japanese)
5. Constructive developmental science Web site. http://devsci.isi.imi.i.u-tokyo.ac.jp/ (2012)
6. de Vries, J., et al.: The emergence of fetal behavior. i. qualitative aspects. Early Hum. Dev. **7**, 301–322 (1982)
7. de Vries, J., et al.: The emergence of fetal behavior. ii. quantitative aspects. Early Hum. Dev. **12**, 99–120 (1985)
8. Editorial: The mind's tangled web. Nature **478**(3), 5 (2011)
9. Eliasmith, C., et al.: A large-scale model of the functioning brain. Science **338**, 1202–1205 (2012)
10. Fujii, K., et al.: Development of multisensory integration and prediction: Fetus simulation with cortex model. In: Association for the Scientific Study of Consciousness (2013)
11. Hallmayer, J., et al.: Genetic heritability and shared environmental factors among twin pairs with autism. Arch. Gen. Psychiatry **68**(11), 1095–1102 (2011)
12. Inui, T.: Toward a unified framework for understanding the various symptoms and etiology of autism and williams syndrome. Jpn. Psychol. Res. **55**(2), 99–117 (2013)
13. Izhikevich, E.M., Edelman, G.M.: Large-scale model of mammalian thalamocortical systems. Proc. Natl. Acad. Sci. **105**(9), 3593–3598 (2008)
14. Johnson, M.H.: Developmental Cognitive Neuroscience (3rd Ed.). Blackwell (2011)
15. Kanemaru, N., et al.: Specific characteristics of spontaneous movements in preterm infants at term age are associated with developmental delays at age 3 years. Dev. Med. Child Neurol. **55**(8), 713–721 (2013)
16. Kanemaru, N., et al.: Specific characteristics of spontaneous movements in preterm infants at term age are associated with developmental delays at age 3 years. Dev. Med. Child Neurol. **55**, 713–721 (2013)
17. Karmel, B.Z., et al.: Early medical and behavioral characteristics of NICU infants later classified with ASD. Pediatrics **126**, 1–11 (2010)

18. Kuniyoshi, Y.: The science of imitation - towards physically and socially grounded intelligence-. In: RWC Symposium, vol. TR-94001, pp. 123–124. RWC Technical Report (1994)
19. Kuniyoshi, Y., Sangawa, S.: Early motor development from partially ordered neural-body dynamics - experiments with a cortico-spinal-musculo-skeletal model. Biol. Cybern. **95**(6), 589–605 (2006)
20. Kuniyoshi, Y., et al.: From humanoid embodiment to theory of mind. In: F. Iida, et al. (eds.) Embodied Artificial Intelligence, International Seminar, Dagstuhl Castle, Germany, July 7–11, 2003, Revised Selected Papers, Lecture Notes in Artificial Intelligence, vol. 3139. Springer (2004). ISBN: 3-540-22484-X
21. Kuniyoshi, Y., et al.: Emergence and development of embodied cognition: a constructivist approach using robots. Prog. Brain Res. **164**, 425–445 (2007)
22. Markram, H.: The blue brain project. Nat. Rev. Neurosci. **7**, 153–160 (2006)
23. Mori, H., Kuniyoshi, Y.: A human fetus development simulation: Self-organization of behaviors through tactile sensation. In: IEEE 9th International Conference on Development and Learning (ICDL 2010), pp. 82–97 (2010)
24. Nakamura, Y., et al.: Dynamic computation of musculo-skeletal human model based on efficient algorithm for closed kinematic chains. In: Proceedings of International Symposium on Adaptive Motion of Animals and Machines, pp. SaP-I-2 (2003)
25. Pitti, A., et al.: Modeling the minimal newborn's intersubjective mind: the visuotopic-somatotopic alignment hypothesis in the superior colliculus. PLoS ONE **8**(7), 1–14 (2013)
26. Sasaki, R., et al.: Tactile stimuli from amniotic fluid guides the development of somatosensory cortex with hierarchical structure using human fetus simulation. In: IEEE ICDL-EPIROB (2013)
27. Smith, L., Thelen, E.: Development as a dynamic system. Trends Cognit. Sci. **7**(8), 343–348 (2003)
28. Waldrop, M.M.: Brain in a box. Nature **482**, 456–458 (2012)
29. Yamada, Y., et al.: Impacts of environment, nervous system and movements of preterms on body map development: Fetus simulation with spiking neural network. In: IEEE ICDL-EPIROB (2013)

Personalizing Intelligent Systems and Robots with Human Motion Data

Gentiane Venture, Ritta Baddoura, Yuta Kawashima, Noritaka Kawashima and Takumi Yabuki

Abstract According to Merhabian, more than 90 % of human-human communication is non-verbal when expressing affects and attitudes. Further studies have shown that a large proportion of non-verbal communication can be attributed to posture and to gesture. They communicate information about action: intent, meaning, as well as information about internal states such as affects. Emotional understanding is a key for satisfying and successful interaction between two or more humans, it must also be true for human-robot interaction. In this paper we explore the importance of non verbal information and communication, typically motion data, and how it can be used to develop and to personalize intelligent systems and robots. First, we present and discuss our findings on the strong correlation between what humans feel during an unannounced interaction with a humanoid robot and their movements and attitudes. Then, we propose a framework that uses not only the kinematics information of movements but also the dynamics. We use the direct measure of the dynamics when available. If not we propose to compute the dynamics from the kinematics, and use it to understand human motions. Finally, we discuss some developments and concrete applications in the field of health care and HRI.

G. Venture (✉)
Tokyo University of Agriculture and Technology, 2-24-16 Nakacho,
Koganei, Tokyo, Japan
e-mail: venture@cc.tuat.ac.jp

R. Baddoura
INSERM, Lyon, France

Y. Kawashima
Tokyo University of Agriculture and Technology, Tokyo, Japan

N. Kawashima
National Rehabilitation Center for People with Disabilities, Tokorozawa, Japan

T. Yabuki
Tokyo University of Agriculture and Technology, Tokyo, Japan

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts
in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_18

1 Background and Motivations

Robots and intelligent agents gradually appear in our society and examples of their presence in homes [20], schools [24], care centers [10], hospitals [5, 43], factories [23] and museums [7, 11] though increasing, are still new for most of us. The variety in robots: Roomba, SAYA, Paro, NAO, Baxter, Tinker, RHINO... (from the above cited articles to name a few), in humans and in possible encounters: with or without direct physical interaction, with or without verbal directives... make the study of human robot interaction (HRI) complex and always particular. At the same time HRI studies and HRI processes understanding are more expressly needed to understand what is or what will be the relationship between humans and robots [45]. It is of crucial importance to understand what can guaranty successful interactions between us and an intelligent agent. What can guaranty continuous engagement and satisfaction of users? How much information can we collect with non-verbal communication? We usually don't ask directly our friends or family if they are happy, we just can read it from their behavior. The same if they are sad, tired or scared. If we want to use intelligent systems and robots to support us in our daily activities, addressing these questions and understanding these processes is of major importance since a smooth communication between us and the system is a mandatory requirement for a successful interaction and long term satisfaction.

Social acceptance and social well-being are difficult to comprehend in a human-human interaction, knowing that what might be acceptable or satisfying for someone may be differently perceived by someone else, because of differences in culture, education, history and perception. This is the reason why personalizing and personalized systems are so much important. It is clear that few social first encounters happen without experiencing some ambiguity, some ambivalence or some strangeness. In HRI, the questions of social acceptance and "*successful*" interactions are more crucial since the difference between humans and robots is fundamental and ontological. The evaluation of such interactions is highly dependent on the robot: its appearance, its abilities, its features and its degree of autonomy. The evaluation of the interaction depends also on the perception of the robot: the appreciation, the readiness to adapt to it, the expectations [44]. Humans will have more and higher expectations from a robot that is highly anthropomorphic such as Asimo rather than from a robot like the vacuum cleaner Roomba. Moreover, the degree to which a human-like nature for a robot is needed is not yet sufficiently understood and studies that focus on such a human-like nature are still very rare [42, 44]. The concept of the *uncanny valley* introduced by Mori [38], and further studies by Mac Dorman [36], and by Bethel [6] showed that when humanoids are too similar to humans but also when robots have a high mechanical appearance [33, 35], they tend to be negatively perceived; whereas other recent studies brought solid and daring proofs that invalidate the hypothesis of the uncanny valley and the common reference to it as a general truth [4]. Most studies [4, 12, 50] agree on the fact that further research is needed to better understand and determine which aspects and degrees of similarity and likeability are required in order to enable more empathic and intuitive Human Robot Interaction (HRI).

Apart the design issues mentioned above, and the expectations of the robot abilities, the problem of the communication is also important. In our societies implicit social rules and communication play an important role and define social behavior, how we are expected and how we expect others to behave. Shall the robot comply with these social rules and good behaviors? Or should it just be task oriented, any type of social behavior being avoided and considered as irrelevant noise? Of course, if we want to create intelligent systems, such as companion robots, that are multi-task and that interact with us in different activities, that enable to maintain a high quality of life through years, the former is much desired rather than the latter. Thus our robots should understand us, at least sufficiently. The degree of understanding depends on the primary role of the robot and the frequency of interaction. A robot which assists us daily needs a more refined understanding (more personalization) than a robot we interact with only once in a while. Yet, the robot we interact seldom with still needs to be able to understand some of our basic behaviors and affects such as fear or anger to maintain a sufficient engagement and confidence level to fulfil the joint task.

A large corpus of research on verbal communication exists and is fundamental for explicit and targeted HRI; however, as Mehrabian pointed out in [37], the verbal communication (the meaning of words and sentences) accounts for only 7 % of the overall human-human communication when one is talking about their feelings or attitudes; and with 38 % attributed to voice, it leaves 55 % of communication as being purely non-verbal. Among the non-verbal communication attributes, though poorly studied as noted in [29], the whole body posture and gesture account for an important proportion [17, 37, 48]. By studying how we move and use our body during human-human and human-robot communication it is possible to learn about our internal state, because what we communicate with our bodies contains mainly internal information about our health [31], about our affects [16], and about the social rules that apply in the observed specific situation, more than any other way of communication. Thus, making it possible in the future to use this motion information to assess directly and “live” the human partner’s psychological state and mental and emotional experience, close to what Breazeal calls “mind reading” in [9], during the on-going interaction and if needed, to adjust the robot’s behavior to our inner experience and expectations. Making the robot emotionally intelligent. Our latest research results in that area, conducted in collaboration with a psychologist, are given in Sect. 2.

If the kinematics of the movements provides significant information as described above, the dynamics of the movement is also as much richer in information. A same movement or a same posture can be achieved using different level of energy and thus transfers different forces and moments, and requires different torques. It reveals different strategies of motor control and it can be an indicator of fatigue, injury or mental or motor disorders, or more simply of changes in our psychological state. There are extensive literature on motion recognition [30], motion understanding [8] using the kinematics information, yet very few addresses the problem of dynamics, forces and physical interaction with our environment, with others, and with a robot [32]. In order to make possible the physical interaction, a number of data regarding the human partner motion dynamics is necessary. We endeavor in providing reliable

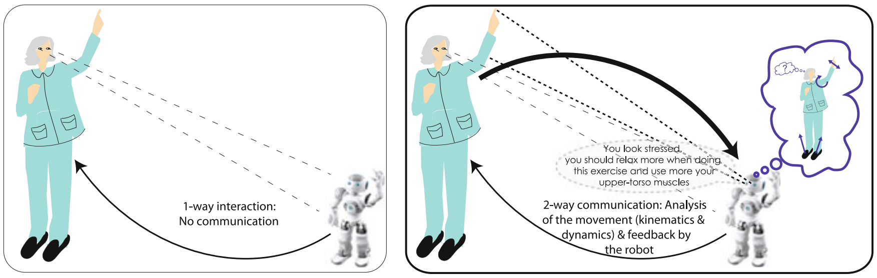

1-way interaction:
No communication
You look stressed, you should relax more when doing this exercise and use more your upper-torso muscles
2-way communication: Analysis of the movement (kinematics & dynamics) & feedback by the robot

Fig. 1 The global concept of what we aim at for a satisfactory and constructive HRI (not necessarily using a verbal feedback, here used for the clarity of the figure)

mathematical computations to provide these information. Our latest results in that area are given in Sect. 3

Finally, the understanding of human behavior from the kinematics and dynamics point of view is crucial when developing systems that will interact with us on the long term, or support us in repetitive and sometimes boring tasks such as rehabilitation, health care or education. In order to guaranty maximal efficiency, the engagement of the user is a key factor. An intelligent system that can overcome these issues of repetitiveness and lack of engagement, or even that can motivate and encourage is an important asset [13]. Personalizing the system with the user preferences and through time, and taking into account fatigue and affects in the loop will definitely be more attractive and efficient. This was already pointed out in [15] by Dautenhahn a decade ago, yet only a few studies have already started to tackle this issue concretely [22, 34]. In light of the works presented in Sects. 2 and 3, we propose and discuss in Sect. 4 possible extensions and applications to personalize intelligent systems and robots that will interact with us in the near future as an example is given in (Fig. 1).

2 The Importance of Movements in HRI and Feeling Familiar

2.1 Sociability, Feeling of Secure and the Concept of the familiar and their application in HRI

Most of the research conducted in HRI, particularly the ones interested in social and affective robotics, refer to a few generally-admitted expressions such as "socially adapted" interaction, "*social acceptance*", "*human-like presence*" and "*natural*" human-robot communication. Although frequently used by the scientific community, these expressions refer in reality to complex psychological and social concepts difficult to precisely define [33, 35, 46]. In addition, in [18] it was shown that the understanding of these concepts is inter-cultural, and in [19, 49] it was shown that the understanding of these concepts is also inter-personal. In [2], we proposed to use the “*familiar*” as a measure of the success of an interaction. As Hebb pointed out in [25], there is in the human attitude towards the new, a bias towards instant familiarity and unquestioning acceptance. Freud also in [21] theorized about the “Unheimlichkeit Gefühl”—which translations vary: “The uncanny”, “feeling of strangeness”, “incredible familiarity”-, which was later discussed abundantly in philosophy notably in relation to the theme of the double. The “Unheimlichkeit Gefühl” together with Jentsch’s elaboration of it [26], has inspired Mori’s concept of the “Uncanny valley” [38] cited previously. Beyond that, it is the subtle ambivalence it brings to what is known or unknown, new or acquainted that interests us: indeed this concept describes a *“bizarre feeling of strangeness”*, a feeling of *“being uncomfortably familiar”*, experienced when encountering a person or an object that seems familiar yet foreign and new at the same time. This ambivalent association of the strange and the familiar within a feeling that is triggered by a new situation but still draws from past experience is useful when working on the interaction between a human and a humanoid robot.

With our experiments we want to verify if there is a correlation between how the participants behave with the robot, what the participants feel when interacting with the robot, very likely for the first time, and how this correlate with their evaluation of the familiar, of the sociability and how it is transposed in their movements. More particularly, if we can use the familiar to qualify the interaction, and if their movements is affected by their inner state in a quantifiable manner. We also want to verify if the social character of the robot is important to conduct the interaction successfully and if the participants engage more with the robot for a hedonic behavior or a useful (task-oriented) joint behavior.

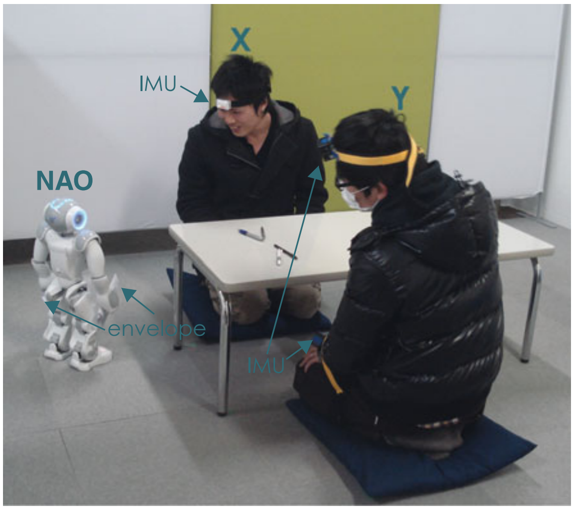


Fig. 2 Snapshot of one of the experiments. In a Japanese style room, candidates are seated on the floor which gives the appropriate height to interact with NAO. X is seated on the *left* hand of NAO, Y is seated on the *right* hand of NAO

2.2 A Simple Experiment Rich in Findings

Our experiment, as shown in Fig. 2, is designed with the robot NAO (Aldebaran), but can be achieved with any other robot. We deliberately chose a feed-forward control to insure perfect repeatability. 40 participants are recruited by pair of X and Y participants, thus 20 pairs (14 women, 26 men) volunteered to participate. NAO's behavior slightly differs from X to Y when handing the envelope: 'Smooth Handing' versus 'Keeping 4 sec the envelope'. The participants are only invited to wear 2 IMU to capture their movements, and answer a questionnaire about HRI and are informed that the set is filmed and IMU are used for hand and head motion capture. They do not know that they will actually interact with a robot. NAO punctuates the interaction's beginning and end with non-verbal greetings [40]. It has a real task to accomplish: bringing both envelopes with the questionnaire to be filled by the participants. The interaction: greetings and envelope exchange, is left up to the participants: it can happen or not, since the participants receive no particular instructions and are left to their own judgement. The questionnaire, in Japanese, consists of 3 different sections: 7-point Likert scale, Multiple Choice Questions, and open-ended questions, addressing different topics about their experience with robots, and in particular the present experience. The experiment is extensively described in [3].

The motion data are analyzed from the IMU and from the video. From the IMU data, the 3 components of the rotational velocity and the 3 components of the acceleration are post-processed separately to obtain three types of information. (1) Frequency analysis (F.): First a simple frequency analysis on the hand motion data (angular velocity) is performed during the grasping motions to take the envelope (F.TE) and when answering the robot goodbye (F.GB) by waving the hand, the frequency (*Hz*) of the first pick is used. (2) Motion smoothness (S.): There are several methods to assess the motion smoothness [41], we chose the jerk metrics ($1/s^2$). For that the acceleration are used to compute the jerk magnitude averaged over the overall motion and normalized with respect to the peak speed. The smaller the jerk metric is the smoother the movement is. The smoothness is computed during for the overall head and torso movements (S.OA), when taking the envelope (S.TE) and when waving goodbye (S.GB). (3) Time spent looking at NAO (T.): from the video recordings we estimated the percentage of time each participants spent during the overall interaction looking at the robot VS the time spent looking elsewhere. (3) Motion intensity (I.): it corresponds to an integration over the time of the interaction of the data. It is computed during the envelope exchange (I.TE) and when greeting back goodbye (I.GB). (4) From PCA of feature vectors we find cluster formation in the motion data [52]. We calculate the distance (D.) from the cluster center for each participant. In order to use the data in the statistical analysis, each of the above parameter is normalized by the obtained maximal value and evaluated in a 0–7 scale. We calculate the descriptive statistics for the participants' responses to the robot's engaging actions and to their answers to the questionnaire and the movement information described above. The results obtained for the sociability and the familiarity are summarized in Tables 1, 2, 3, 4.

The earlier exposure was low for participants X (M = 2.5, SD = 2.0, SEM = 0.4) and medium to low for participants Y (M = 3.8, SD = 2.4, SEM = 0.5). 40 % of X reported almost having never been exposed to robots, 45 % said to be familiar with robots from movies and literature. 40 % of Y reported to have been exposed at least once to a real robot, 55 % said to be familiar with robots from movies and literature.

Most participants (80 % X, 75 % Y) understood NAO's intention of giving them the envelope and found easy for them to react to its action, only 35 % of X and 25 % of Y found it easy to decide on how/whether to react to its greetings.

The sociable character of the robot is strongly related to its politeness, promoted through its greetings, with the comfort and the security felt during the interaction, and with the evaluation of the familiar. It is also clearly correlated with the participants' reaction and their movements during the interaction as the results in Table 2 suggest. The higher the evaluation of the robot sociable character is the more participants engaged in interacting with it, with higher movement frequencies, both when taking the envelope and greeting good bye, with more intense greeting movements and with smoother overall movements. On the other hand, when the interaction was evaluated has frightening, there is a strong correlation wit ha low frequency movement when greeting good bye ($Corr = -0.61, p < 0.1$). This clearly shows that the participants' movements are affected by their perception of the robot and the interaction, and thus their movement can be used as an indicator of their inner state, and for the good completion of the task.

The familiar felt during the interaction is also an excellent parameter to assess the interaction. As shown in Table 3, the familiar is strongly associated with the evaluation of comfort, security, with making sense of the interaction and understanding it, the easiness to complete the task and the sociable character of the robot. Basically the familiar is associated with a positive experience with the robot; it can even be experienced during a new situation; and that it can be experienced in relation to a machine. It is also associated with a feeling of absurdity and strangeness, which verifies that the hypothesis of Freud [21] and Jentsch [26] are also valid in the case of an interaction with a robot. The evaluation of the familiar also correlates with the participants' reactions. The higher the familiar is felt the more participants engage wit the robot in both taking the envelope and responding to the robot's greetings. Moreover, the higher the evaluation of the familiar is, the higher the frequency of waving good bye is, and the smoother are the motions. Finally, the higher the familiar is the more participants tend to react similarly when taking the envelope and their motions are alike. We also found that the more the robot was assessed as being hostile, the more participants's response when taking the envelope differ from the cluster

Table 1 Correlation between sociability of the robot and the different items in the questionnaire

	NAO polite	Interaction comfortable	Interaction secure	Interaction familiar
NAO sociable	0.75$^{*}**$	0.53**	0.63**	0.90***

* $p < 0.1$ ** $p < 0.05$ *** $p < 0.001$

Table 2 Correlation between Sociability of the robot and the movements observed

	F.TE	F.GB	S.OA	I.GB
NAO sociable	0.31*	0.88***	0.44**	0.37**

* $p < 0.1$ ** $p < 0.05$ *** $p < 0.001$

Table 3 Correlation between the familiar and the different items in the questionnaire

	Interaction absurd	Interaction comfortable	Interaction secure	Interaction meaningful	Interaction easy	NAO sociable
Familiar	0.47**	0.61**	0.76***	0.47**	0.57**	0.90***

* $p < 0.1$ ** $p < 0.05$ *** $p < 0.001$

Table 4 Correlation between the familiar during the interaction and the movements observed

	NAO handing envelope X	NAO handing envelope Y	Greeting GB X	Greeting GB Y
Familiar	0.71**	0.56**	0.49**	0.53**
	F.GB	S.OA	S.GB	D.
Familiar	0.57*	0.36**	0.65*	-0.48**

* $p < 0.1$ ** $p < 0.05$ *** $p < 0.001$

average. Again here, the participants' movements clearly reflect their inner feelings with respect to the robot and the interaction.

The sociability of the humanoid robot and the familiar of the interaction appear to be two essential parameters to successfully and adequately engage humans in an interaction with a robot (e.g. a joint-task such as exchanging an envelope). These results also show what we hypothesized: that the participants' motions, emotions and their perception of the robot and of the interaction with it are strongly associated during the interaction. The sole kinematics measurement of the motion can provide information that can be used to adjust the robot's behavior to its human partner.

3 Going Further: Tools for Dynamics Analysis

In the previous section we only used the motion kinematics. However the dynamics of the movements can also provide some information to quantify the interaction with the robot or the understand the human behavior and its interaction with the environment. With an increasing interest in compliant robots, which can interact safely and wisely with their environment, the force-moment information has appeared to be crucial [14, 39]. In this section we propose two approaches that can help in estimating the contact force with the environment, and use the contact force as information to classify movements. The first one requires the measurement of the motion itself, i.e. its kinematics, that can be measured using any kind of motion capture technology. The second supposes that we only measure the contact forces, which can be the case with in-sole force measurement devices, force plates, or floor equipped with force sensors. The first aspect is useful to understand how we use our environment and our body, it finds applications in HRI, in motor function rehabilitation. The second aspect can be more appropriate for applications in rehabilitation or daily life monitoring and support.

3.1 Contact Forces Computation

When the motion information is available (whatever the system used to obtained it: IMU, optical motion capture, encoders...), it is rather simple to compute the generalized external forces acting, i.e. the sum of the external forces, on the human or the robot. One need only to know the kinematic information and the inertial parameters of the considered system. Our work on dynamics identification of humanoids and humans solves both problems [1, 47]. From the generalized contact forces it is possible to estimate the contact forces at each contact point. It is trivial when there is only one contact point since all the force is uniquely defined. When there are more than one contact point, using the position of the Zero Moment Point (ZMP) or the Center of Pressure (CoP), and its distance from the contact points helps solving this problem. In particular the vertical force comes in a simple inverse proportional of this distance. The method is fully described in [28]. Our results with human data shown in Fig. 3, for a random motion alternating single and double support, and for gait are extremely encouraging and show that we can estimate the vertical force with an excellent accuracy, which of course depends on the accuracy of the measured kinematics information. For the lateral forces, the results are not as good, yet they can clearly show the profile expected. Their value is more affected by the contact detection, which here is done using external markers position here, where a foot switch would have given better results. These results are used in gait analysis of stroke patients, to evaluate rehabilitation strategies. They provide a totally personalized model and analysis, thus helping in personalized diagnosis and rehabilitation procedure design.

3.2 Motion Analysis from Contact Forces

Furthermore, the contact force-moment with the ground provide interesting information regarding the motion that is executed. Walking, exercising... have typical force-moment profiles. If it is mathematically possible that two different motions have the same contact force-moment signature, it is likely that in our daily life activities distinct motions have distinct signatures. This is more true when considering gymnastic exercises, or rehabilitation exercises. Using that assumption, we propose to classify and recognize whole body motions using solely the contact force-moment information. Using a simple classifier based on the computation of features vectors and PCA decomposition [27] we can highlight these contact force-moment signa-

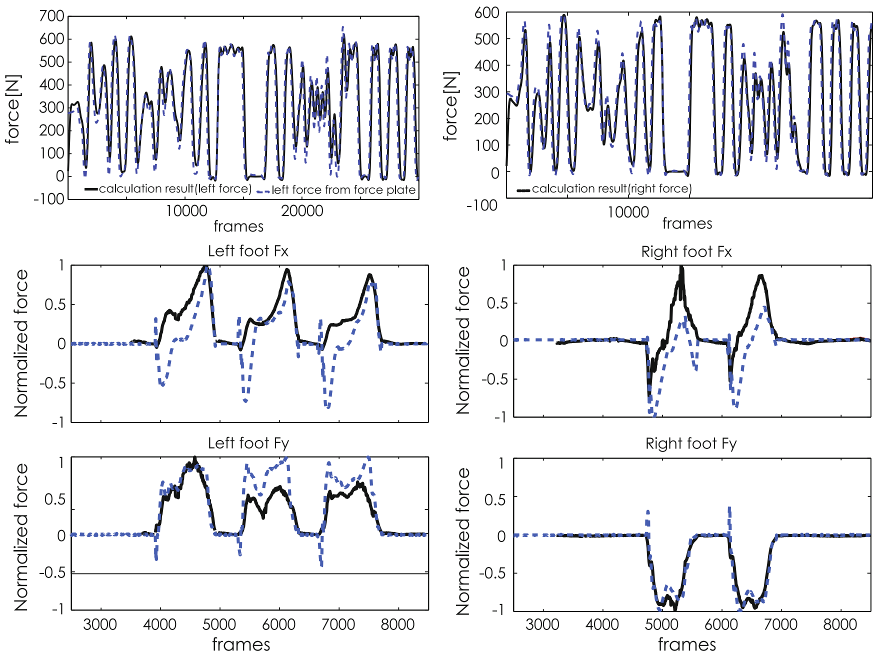


Fig. 3 *Top*: vertical force for a random movement; *Middle* and *bottom* floor force during gait. The *blue dotted line* is the ground truth measured by the force plates, the *black line* is the estimated one

tures. Using the algorithm proposed in [53], we can use the contact force-moment profiles for motion recognition [51].

Our tested data-set made of 2 participants repeating 3 times a series of 7 distinct exercises, thus a total of 42 exercises, taken form a exercises' television program. One exercise can contain the same motion repeated a few times. The data are manually segmented to extract each single motion. The visualization in 3D of the PCA results is given in Fig. 4.

Our algorithm performs with more than 85 % successful recognition regardless of the candidate. The confusion between movements mainly occurs with movements that are too similar and poorly executed: for example bending the torso forward and bending the torso sideward. Our results show that the contact force-moment can be used to classify and recognize movements. Applications for health monitoring and rehabilitation are already ongoing. Further applications during HRI are expected too.

4 Applications and Discussion

Our study, presented in Sect. 2 shows that an unexpected first encounter with a specific humanoid robot (NAO) is a rich, yet subjective and different, experience for each

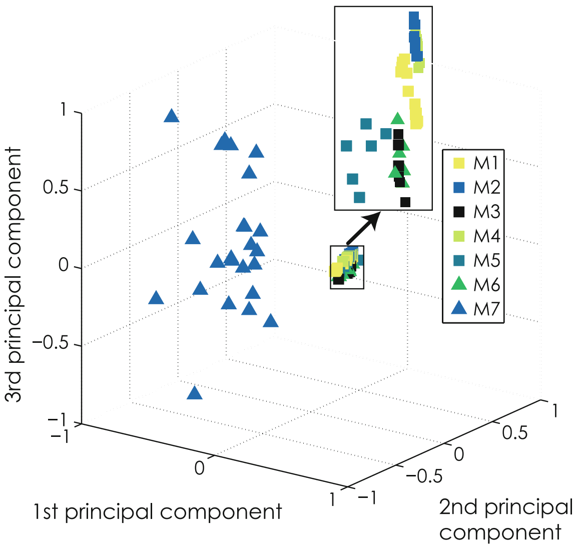


Fig. 4 Classification results for the seven different tested motions. Each maker correspond to a single iteration of one motion. The highlighted data in the *box* show a close up of the PCA results for motion M1 to M6

of the 40 participants. The analysis of their spontaneous arm and head movements, as well as the analysis of their evaluation of the robot and of interacting with it, clearly say that none of the participants was indifferent to the encounter, even though the whole interaction was non-verbal. It is worth considering these aspects before robots spread in our daily environment since robots, and in particular humanoids, are creatures that will affect our daily life, our behaviors and our emotions more than any other technology. They may affect our relation with others, and our perception of the self, creating deep psychological and social changes either for our benefit or for our detriment. To address these issues and to try to understand better why we are so found of and yet so scared of robots further research closely in relation with psychologists is tremendously needed.

To elaborate more complex experimental protocols to investigate HRI and to gain a deeper understanding of the interaction with robots novel tools are necessary to measure these interactions. Our results have shown that using motions as study parameters is not unjustified. The affective signature in our motions provide a very powerful tool to analyze HRI in natural settings and get rid of the usual questionnaires that oblige the participants for a-posteriori introspection, or when filled step-by-step remove the naturalness of the social interactions, which in a sense is failing in providing adequate information. Measuring the non-verbal components of the communication and understanding it as crucial as developing robots that can talk and hear since non-verbal communication is the main medium for affective communication. These robots and systems can effectively learn to understand our personalized movements and react adequately to maintain the necessary level of engagement.

With the democratization of whole-body motion capture technology for games and human machine interface, there is now a wide range of sensing technology available at low cost and with increasing specifications. Such developments have considerable ripple effects on robot technology developments since they provide plug-and-play hardware and even SDK. These equipment can now be used for analysis in real situation or close to real situations, rather than in lab environment mimicking poorly real social interactions. More generally, personalized systems will be even more demanded and for that studying also human-human interaction can provide a baseline of human behavior for HRI and help in personalizing these systems.

Moreover, our computational methods for the contact forces, and our method for motion classification and recognition from the contact forces, provide dynamics information that is also rich in information in regard with the interaction. Our results strongly underline the dependencies existing between human movement and inner experience, especially in relation to the robot's social character, the interaction's perception of safety, its meaning and the interest it represents for the participants, more globally their engagement in and their evaluation of the overall interaction. Adding the forces here will surely bring in some refined analysis of the interaction and some new insight to personalized intelligent systems and robots not only for health-care applications, but more generally for social interactions too.

Acknowledgments This research is partially supported by the "Women's Future Development Organization".

References

1. Ayusawa, K., Venture, G., Nakamura, Y.: Identifiability and identification of inertial parameters using the underactuated base-link dynamics for legged multibody systems. Int. J. Robot. Res. (2013) (in press)
2. Baddoura, R., Matuskata, R., Venture, G.: The familiar as a key-concept in regulating the social and affective dimensions of HRI. In: Proceedings of the IEEE/RAS International Conference on Humanoid Robots, pp. 234–241 (2012)
3. Baddoura, R., Venture, G.: Social versus useful HRI: experiencing the familiar, perceiving the robot as a sociable partner and responding to its actions. Int. J. Soc. Robot. (2013)
4. Bartneck, C., Kanda, T., Ishiguro, H., Hagita, N.: My robotic doppelganger—a critical look at the uncanny valley theory. In: 18th IEEE International Symposium on Robot and Human Interactive Communication, pp. 269–276 (2009)
5. Baxter, P., Belpaeme, T., Cañamero, L., Cosi, P., Demiris, Y., Enescu, V., Hiolle, A., Kruijff-Korbayova, I., Looije, R., Nalin, M. et al.: Long-term human-robot interaction with young users. In: IEEE/ACM Human-Robot Interaction Conference (Robots with Children Workshop) (2011)
6. Bethel, C., Murphy, R.: Affective expression in appearance—constrained robots. In: ACM SIGCHI/SIGART Conference on Human Robotics, Interaction (2006)
7. Bickmore, T., Vardoulakis, M.: L. and Pfeifer, and D. Schulman. Tinker: a relational agent museum guide. Autonom. Agents Multi-Agent Syst. **27**(2), 254–276 (2013)
8. Biswas, R., Thrun, S., Fujimura, K.: Recognizing activities with multiple cues. Hum. Motion Underst. Model. Capture Anim. Lect. Notes Comput. Sci. **4814**, 255–270 (2007)
9. Breazeal, C., Gray, J., Berin, M.: Mindreading as a foundational skill for socially intelligent robots. Robot. Res. Springer Tracts Adv. Robot. **66**, 383–394 (2011)

10. Broadbent, E., Tamagawa, R., Patience, A., Knock, B., Kerse, N., Day, K., MacDonald, B.: Attitudes towards health-care robots in a retirement village. Australasian J. Ageing **31**(2), 115–120 (2012)
11. Burgard, W., Cremers, A., Fox, D., Hänel, D., Lakemeyer, G., Schulz, D., Steiner, W., Thrun, S.: The museum tour-guide robot rhino. In: Autonome Mobile Systeme 1998. Informatik aktuell, pp. 245–254. Springer, Heidelberg (1999)
12. Canamero, L.: Playing the emotion game with feelix: What can a lego robot tell us about emotion? Socially intelligent agents: Creating relationships with computers and robots, pp. 69–76 (2002)
13. Castellano, G., Leite, I., Pereira, A., Martinho, C., Paiva, A., McOwan, P.: Detecting engagement in HRI: An exploration of social and task-based context. In: International Conference on Privacy, Security, Risk and Trust and International Conference on Social Computing, pp. 421–428 (2012)
14. Colome, A., Pardo, D., Alenya, G., Torras, C.: External force estimation during compliant robot manipulation. In: IEEE International Conference on Robotics and Automation, pp. 3535–3540 (2013)
15. Dautenhahn, K.: Robots we like to live with?!—a developmental perspective on a personalized, life-long robot companion. In IEEE International Workshop on Robot and Human Interactive, Communication (2004)
16. de Gelder, B.: Towards the neurobiology of emotional body language. Nat. Rev. Neurosci. **7**, 242–249 (2006)
17. Ekman, P., Friesen, W.: Head and body cues in the judgement of emotion—a reformulation. Percept. Motor Skills **24**, 711–724 (1967)
18. Fanaswala, I., Browning, B., Skar, M.: Interactional disparities in english and arabic native speakers with a bi-lingual robot receptionist. In: Proceedings of the International Conference on Human-Robot Interaction (2011)
19. Fischer, K.: Interpersonal variation in understanding robots as social actors. In: Proceedings of the International Conference on Human-Robot, Interaction, pp. 53–60 (2011)
20. Forlizzi, J., Di Salvo, C.: Service robots in the domestic environment: a study of the roomba vacuum in the home. In: Proceedings of the 1st ACM SIGCHI/SIGART Conference on Human-robot Interaction, HRI '06, pp. 258–265, ACM, New York (2006)
21. Freud, S.: The Uncanny (das unheimliche, 1919). Penguin Books Limited, UK (1919)
22. Glas, D.F., Wada, K., Shiomi, M., Kanda, T., Ishiguro, H., Hagita, N., Environment personal service: a robot that greets people individually based on observed behavior patterns. In: International Conference on Human Robot Intercation, pp. 129–130 (2013)
23. Guizzo, E., Ackerman, E.: The rise of the robot worker. IEEE Spect. **49**(10), 34–41 (2012)
24. Hashimoto, T., Verner, I., Kobayashi, H.: Human-like robot as teacher's representative in a science lesson: An elementary school experiment. In: Robot Intelligence Technology and Applications 2012. vol. 208 of Advances in Intelligent Systems and Computing, pp. 775–786. Springer, Heidelberg (2013)
25. Hebb, D.: A Textbook of Psychology (1958)
26. Jentsch, E.: On the psychology of the uncanny (zur psychologie des unheimlichen). Psychiatrisch-Neurologische Wochenschrift, pp. 195–198 (1906)
27. Kadone, H., Nakamura, Y.: Segmentation, memorization, recognition and abstraction of humanoid motions based on correlations and associative memory. In: IEEE/RAS Proceedings of the International Conference on Humanoid Robots, pp. 1–6 (2006)
28. Kawashima, Y., Venture, G., Kawashima, N.: Contact force computation from inverse dynamics: Application to gait analysis. In: Proceedings of the International Conference on Posture and Gait, pp. 332–333 (2012)
29. Kleinsmith, A., Bianchi-Berthouze, N.: Affective Body Expression Perception and Recognition: A Survey. IEEE Trans. Affect. Comput. **4**(1), 15–33 (2013)
30. Kulic, D., Takano, W., Nakamura, Y.: Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains. Int. J. Robot. Res. **27**, 761 (2008)

31. Kulic, D., Venture, G., Nakamura, Y.: Detecting changes in motion characteristics during sports training. In: Proceedings of the IEEE International Conference on Engineering in Medicine and Biology, pp. 4011–4014 (2009)
32. Lee, D., Ott, C., Nakamura, Y., Hirzinger, G.: Physical human robot interaction in imitation learning. In: IEEE International Conference on Robotics and Automation, pp. 3439–3440 (2011)
33. Lee, E.-J.: Flattery may get computers somewhere, sometimes: The moderating role of output modality, computer gender, and user gender. Int. J. Hum. Comput. Stud. **66**, 789–800 (2008)
34. Lee, M.K., Forlizzi, J., Kiesler, S., Rybski, P., Antanitis, J., Savetsila, S.: Personalization in HRI: A Longitudinal Field Experiment. In: International Conference on Human Robot Intercation, pp. 319–326 (2012)
35. Lee, N., Shin, H., Sundar, S.S.: Utilitarian vs. hedonic robots, role of parasocial tendency and anthropomorphism in shaping user attitudes. In: Proceedings of the International Conference on Human-Robot Interaction, pp. 183–184 (2011)
36. Mac Dorman, K., Ishiguro, H.: The uncanny advantage of using androids in cognitive and social science research. Interact. Stud. **7**(3), 297–337 (2006)
37. Mehrabian, A., Ferris, S.R.: Inference of attitudes from nonverbal communication in two channels. J. Consult. Psychol. **31**(3), 248 (1967)
38. Mori, M.: Bukimi no tani (the uncanny valley). Energy, pp. 33–35 (1970)
39. Moro, F., Tsagarakis, N., Caldwell, D.: Walking in the resonance with the coman robot with trajectories based on human kinematic motion primitives (KMPS). In: Autonomous Robots, pp. 1–17 (2013)
40. Reeves, B., Nass, C.: The Media Equation. CSLI Publications, Stanford (1996)
41. Rohrer, B., Fasoli, S., Krebs, H., Hughes, R., Volpe, B., Frontera, W., Stein, J., Hogan, N.: Movement smoothness changes during stroke recovery. J. Neurosci. **22**(18), 8297–8304 (2002)
42. Salvini, P., Laschi, C., Dario, P.: Design for acceptability: Improving robots' coexistence in human society. Int. J. Soc. Robot. **2**, 451–460 (2010)
43. Shibata, T.: Therapeutic seal robot as biofeedback medical device: Qualitative and quantitative evaluations of robot therapy in dementia care. Proc. IEEE **100**(8), 2527–2538 (2012)
44. E. Takano, T. Chikaraishi, Y. Matsumoto, Y. Nakamura, H. Ishiguro, and K. Sugamoto. Psychological effects on interpersonal communication by bystander android using motions based on human-like needs. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3721–3726 (2009)
45. Tisseron, S.: De l'animal numerique au robot de compagnie: quel avenir pour l'intersubjectivite? Revue francaise de psychanalyse **75**, 149–159 (2011)
46. Turkle, S.: A nascent robotics culture: New complicities for companionship. AAAI Technical report series (2006)
47. Venture, G., Ayusawa, K., Nakamura, Y.: Dynamics identification of humanoid systems. In: Proceedings of the CISM-IFToMM Symposium on Robot Design, Dynamics, and Control (ROMANSY), pp. 301–308 (2008)
48. Wallbott, H.: Bodily expression of emotion. Eur. J. Soc. Psychol. **28**, 879–896 (1998)
49. Walters, M., Syrdal, D., Koay, K., Dautenhahn, K., Boekhorst, R.: Human approach distances to a mechanical-looking robot with different robot voice styles. In: Proceedings of the IEEE International Symposium Robot and Human Interactive, Communication, pp. 707–712 (2008)
50. Weiss, A., Bernhaupt, R., Tscheligi, M., Wollherr, D., Kuhnlenz, K., Buss, M.: A methodological variation for acceptance evaluation of human-robot interaction in public places. In IEEE International Symposium on Robot and Human Interactive, Communication (2008)
51. Yabuki, T., Venture, G.: Motion recognition from contact force measurement. In: Proceedings of the IEEE International Conference Engineering in Medicine and Biology, pp. 7245–7248 (2013)
52. Zhang, T., et al.: Individual recognition from gait using feature value method. Cybern. Inform. Tech. **12**, 86–95 (2012)
53. Zhang, T., Venture, G.: Biometrics from gait using feature value method. Artif. Intell. Methodol. Syst. Appl. Lect. Notes Comput. Sci. **7557**, 325–333 (2012)

Beyond Geometric Path Planning: Learning Context-Driven Trajectory Preferences via Sub-optimal Feedback

Ashesh Jain, Shikhar Sharma and Ashutosh Saxena

Abstract We consider the problem of learning preferences over trajectories for mobile manipulators such as personal robots and assembly line robots. The preferences we learn are more intricate than those arising from simple geometric constraints on robot's trajectory, such as distance of the robot from human etc. Our preferences are rather governed by the surrounding context of various objects and human interactions in the environment. Such preferences makes the problem challenging because the criterion of defining a good trajectory now varies with the task, with the environment and across the users. Furthermore, demonstrating optimal trajectories (e.g., learning from expert's demonstrations) is often challenging and non-intuitive on high degrees of freedom manipulators. In this work, we propose an approach that requires a *non-expert* user to only incrementally improve the trajectory currently proposed by the robot. We implement our algorithm on two high degree-of-freedom robots, PR2 and Baxter, and present three intuitive mechanisms for providing such incremental feedback. In our experimental evaluation we consider two context rich settings—household chores and grocery store checkout—and show that users are able to train the robot with just a few feedbacks (taking only a few minutes). Despite receiving sub-optimal feedback from non-expert users, our algorithm enjoys theoretical bounds on regret that match the asymptotic rates of optimal trajectory algorithms.

This work was done when Sharma was an intern at the Cornell University.

A. Jain (✉) · A. Saxena
Cornell University, Department of Computer Science, Ithaca, NY, USA
e-mail: ashesh@cs.cornell.edu

A. Saxena
e-mail: asaxena@cs.cornell.edu

S. Sharma
Indian Institute of Technology, Kanpur, India
e-mail: ss2986@cs.cornell.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_19

1 Introduction

Recent advances in robotics have resulted in mobile manipulators with high degree of freedom (DoF) arms. However, the use of high DoF arms has so far been largely successful only in structured environments such as manufacturing scenarios where they perform same repetitive motions (e.g., recent deployment of Baxter on assembly lines). A major challenge in the deployment of these robots in unstructured environments (such as a grocery checkout counter or in our homes) is their lack of understanding of user preferences and thereby not producing desirable motions. In this work we address the problem of learning preferences over trajectories for high DoF robots such as Baxter or PR2. We consider a variety of household chores for PR2 and grocery store checkout tasks for Baxter.

A key problem for high DoF manipulators lies in identifying an appropriate trajectory for a task. An appropriate trajectory not only needs to be valid from a geometric point (i.e., feasible and obstacle-free, the criterion that most path planners focus on), but it also needs to satisfy the user's preferences. Such users' preferences over trajectories vary between users, between tasks, and between the environments the trajectory is performed in. For example, a household robot should move a glass of water in an upright position without jerks while maintaining a safe distance from nearby electronic devices. In another example, a robot checking out a santoku knife[1] at a grocery store should strictly move it at a safe distance from nearby humans. Furthermore, straight-line trajectories in Euclidean space may no longer be the preferred ones. For example, trajectories of heavy items should not pass over fragile items but rather move around them. These preferences are often hard to describe and anticipate without knowing where and how the robot is deployed. This makes it infeasible to manually encode (e.g., [26]) them in existing path planners (e.g., [10, 39, 44]) a priori.

We learn user preferences over trajectories via eliciting sub-optimal suggestions from the user for improving a trajectory. Unlike in other learning settings, where an expert first demonstrates optimal trajectories [4] for a task to the robot, our learning model does not rely on the user's ability to demonstrate optimal trajectories a priori. Instead, our learning algorithm explicitly guides the learning process and merely requires the user to incrementally improve the robots trajectories thereby learning user preferences and not that of expert's. This procedure of learning from sub-optimal suggestions is known as coactive learning and has been previously studied in information retrieval [41]. We contribute by introducing this new method of learning to the robotics community and highlight its advantages over learning from demonstration for high DoF robots. We build a system to realize this learning algorithm on PR2 and Baxter robots, and also leverage the robot specific design to allow users easily give preference feedback required by our algorithm.

Our experiments show that a robot trained using this approach can autonomously perform new tasks and if need be, only a small number of interactions are sufficient

[1] A kitchen knife originating in Japan.

to tune the robot to the new task. *Since the user does not have to demonstrate a (near) optimal trajectory to the robot*, the feedback is easier to provide and more widely applicable. Nevertheless, it leads to an online learning algorithm with provable regret bounds that decay at the same rate as for optimal demonstrations.

In our empirical evaluation, we learn preferences for a two high DoF robots, PR2 and Baxter, on a variety of household and grocery checkout tasks respectively. Using the expressive trajectory features from our previous work [19], we show how our algorithm learns preferences from online user feedback on a broad range of tasks for which object properties are of particular importance (e.g., manipulating sharp objects with humans in the vicinity). We extensively evaluate our approach on a set of 35 household tasks and 16 grocery checkout tasks, both in batch experiments as well as through robotic experiments wherein users provide their preferences on the robot. Our results show that our system not only quickly learns good trajectories on individual tasks, but also generalizes well to tasks that the algorithm has not seen before. We now describe the learning setting along with mechanisms for eliciting preference feedback and highlight attributes of Baxter robot that makes it well suited for our algorithm.

2 Coactive Learning with Incremental Feedback

We propose an online algorithm for learning preferences in trajectories from sub-optimal user feedback. At each step robot receives a task as input and outputs a trajectory that maximizes its current estimate of some score function. It then observes a user feedback—an improved trajectory—and updates the score function to better match the user preferences. This procedure of learning via iterative improvement is known as coactive learning.

Our goal is to even learn from the feedback given by non-expert users. We therefore require the feedback to only be *incrementally* better (as compared to being close to optimal) in expectation, and will show that such feedback is sufficient for the algorithm's convergence. It is in contrast to learning from demonstration (LfD) methods [1, 25, 36, 37] which require (near) optimal kinesthetic demonstrations of the complete trajectory. Such demonstrations can be extremely challenging and non-intuitive to provide for many high DoF manipulators [2]. Instead, we found that it is more intuitive for users to give an incremental feedback on certain high DoF arms such as Barrett WAM and Baxter. With a zero-force gravity-compensation (zero-G) mode, the robot arms becomes light and the users can effortlessly steer them to desired configuration. On Baxter, this zero-G mode is automatically activated when a user holds the robot's wrist (see Fig. 1, middle). We use this zero-G mode as a feedback method for incrementally improving the trajectory by correcting a waypoint. We now summarize three feedback mechanisms that enable the user to iteratively provide improved trajectories.

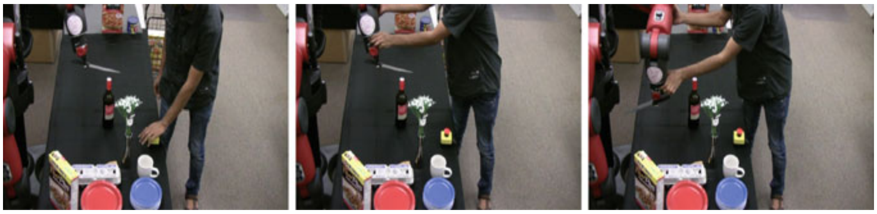


Fig. 1 **Zero-G feedback mechanism**: Robot checking out items in a grocery store moves knife close to human. *(Left)* User stops the trajectory and *(Middle)* activates the zero-G mode by holding Baxter's wrist. *(Right)* User then improves the trajectory by moving the knife away and rotating it

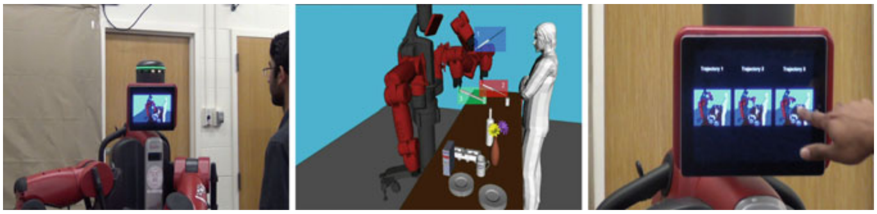


Fig. 2 **Re-rank feedback mechanism**: *(Left)* Robot ranks trajectories using the score function and *(Middle)* displays *top* three trajectories on a touch screen device (iPad here). *(Right)* As feedback, the user improves the ranking by selecting the third trajectory

(a) Re-ranking: We display the ranking of trajectories using OpenRAVE [12] on a touch screen device and ask the user to identify whether any of the lower-ranked trajectories is better than the top-ranked one. User sequentially observes the trajectories in order of their current predicted scores and clicks on the first trajectory which is better than the top ranked trajectory. Figure 2 shows three trajectories for moving knife. As feedback user moves the trajectory at rank 3 to the top position. Likewise, Fig. 3 shows three trajectories for moving an egg carton. Using the current estimate of score function robot ranks them as red (1st), green (2nd) and blue (3rd). Since eggs are fragile user moves green trajectory to the top position.

(b) Zero-G: This feedback allows the user to correct trajectory waypoints by physically changing robot's arm configuration as shown in Fig. 1. This feedback is useful (i) for bootstrapping the robot, (ii) for avoiding local maxima where the top trajectories in the ranked list are all bad but ordered correctly, and (iii) when the user is satisfied with the top ranked trajectory except for minor errors. A counterpart of this feedback is keyframe based LfD [2] where an expert demonstrates a sequence of optimal waypoints instead of the complete trajectory.

(c) Interactive: For the robots whose hardware does not permit zero-G feedback, such as PR2, we built an alternative interactive Rviz-ROS [16] interface for allowing the users to improve the trajectories by waypoint correction. Figure 4 shows a robot moving a bowl with one bad waypoint (in red), and the user provides a feedback by correcting it. This feedback serves the same purpose as zero-G but it's elicited via simulator.

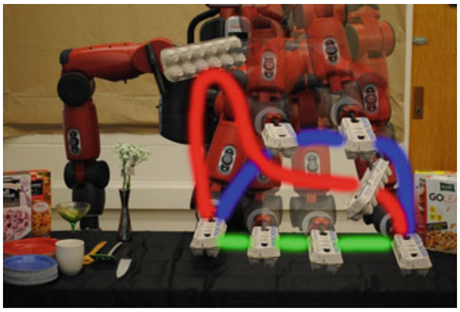


Fig. 3 **Re-ranking feedback**: Shows three trajectories for moving egg carton from *left* to *right*. Using the current estimate of score function robot ranks them as *red*, *green* and *blue*. As feedback user clicks the *green* trajectory. **Preference**: Eggs are fragile. They should be kept *upright* and near the supporting surface

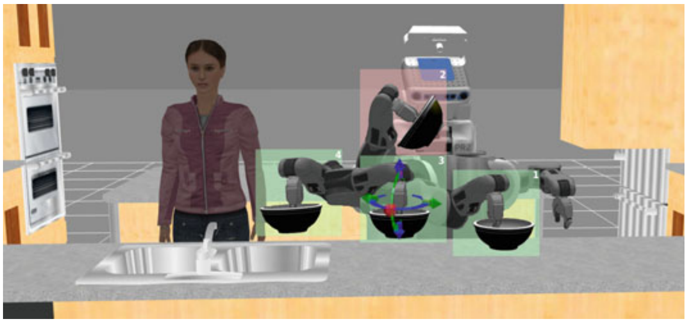


Fig. 4 **Interactive feedback**. Task here is to move a bowl filled with water. The robot presents a bad trajectory with waypoints 1–2–4 to the user. As feedback user moves waypoint 2 (*red*) to waypoint 3 (*green*) using Rviz interactive markers. The interactive markers guides the user to correct the waypoint

Note that in all three kinds of feedback, the user never reveals the optimal trajectory to the algorithm but just provides a slightly improved trajectory (in expectation).

3 Learning and Feedback Model

We model the learning problem in the following way. For a given task, the robot is given a context x that describes the environment, the objects, and any other input relevant to the problem. The robot has to figure out what is a good trajectory y for this context. Formally, we assume that the user has a scoring function $s^*(x, y)$ that reflects how much he values each trajectory y for context x. The higher the score, the better the trajectory. Note that this scoring function cannot be observed directly, nor do we assume that the user can actually provide cardinal valuations according to this function. Instead, we merely assume that the user can provide us with *preferences* that reflect this scoring function. The robot's goal is to learn a function $s(x, y; w)$ (where w are the parameters to be learned) that approximates the user's true scoring function $s^*(x, y)$ as closely as possible.

Interaction Model. The learning process proceeds through the following repeated cycle of interactions.

Step 1: The robot receives a context x and uses a planner to sample a set of trajectories, and ranks them according to its current approximate scoring function $s(x, y; w)$.

Step 2: The user either lets the robot execute the top-ranked trajectory, or corrects the robot by providing an improved trajectory $\bar{y}$. This provides feedback indicating that $s^*(x, \bar{y}) > s^*(x, y)$.

Step 3: The robot now updates the parameter w of $s(x, y; w)$ based on this preference feedback and returns to step 1.

Regret. The robot's performance will be measured in terms of regret, $REG_T = \frac{1}{T}\sum_{t=1}^{T}[s^*(x_t, y_t^*) - s^*(x_t, y_t)]$, which compares the robot's trajectory y_t at each time step t against the optimal trajectory y_t^* maximizing the user's unknown scoring function $s^*(x, y)$, $y_t^* = argmax_y s^*(x_t, y)$. Note that the regret is expressed in terms of the user's true scoring function s^*, even though this function is *never observed*. Regret characterizes the performance of the robot over its whole lifetime, therefore reflecting how well it performs *throughout* the learning process. We will employ learning algorithms with theoretical bounds on the regret for scoring functions that are linear in their parameters, making only minimal assumptions about the difference in score between $s^*(x, \bar{y})$ and $s^*(x, y)$ in Step 2 of the learning process.

4 Learning Algorithm

For each task, we model the user's scoring function $s^*(x, y)$ with the following parametrized family of functions.

$$s(x, y; w) = w \cdot \phi(x, y) \tag{1}$$

w is a weight vector that needs to be learned, and $\phi(\cdot)$ are features describing trajectory y for context x. We further decompose the score function in two parts, one only concerned with the objects the trajectory is interacting with, and the other with the object being manipulated and the environment

$$s(x, y; w_O, w_E) = s_O(x, y; w_O) + s_E(x, y; w_E) = w_O \cdot \phi_O(x, y) + w_E \cdot \phi_E(x, y) \tag{2}$$

For more details on the features $\phi_O(\cdot)$ and $\phi_E(\cdot)$, we refer the readers to our previous work Jain et al. [19].

4.1 Computing Trajectory Rankings

For obtaining the top trajectory (or a top few) for a given task with context x, we would like to maximize the current scoring function $s(x, y; w_O, w_E)$.

$$y^* = \arg\max_y s(x, y; w_O, w_E). \tag{3}$$

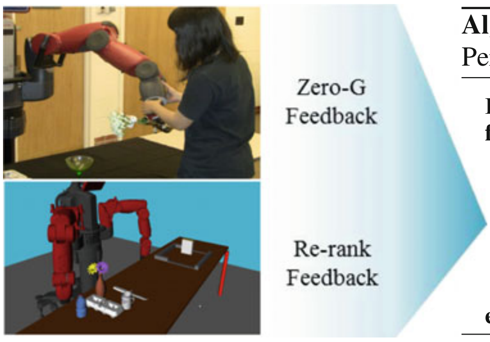
Second, for a given set $\{y^{(1)}, \dots, y^{(n)}\}$ of discrete trajectories, we need to compute (3). Fortunately, the latter problem is easy to solve and simply amounts to sorting the trajectories by their trajectory scores $s(x, y^{(i)}; w_O, w_E)$. Two effective ways of solving the former problem is either discretizing the state space or directly sampling trajectories from the continuous space. Previously both approaches [3, 6, 7, 11, 46] have been studied. However, for high DoF manipulators sampling based approaches [6, 11] maintain tractability of the problem, hence we take this approach. More precisely, similar to [6], we sample trajectories using rapidly-exploring random tree (RRT) [27].[2] Since our primary goal is to learn a score function on trajectories we now describe our learning algorithm and for more details on sampling trajectories we refer interested readers to [15, 17].

4.2 Learning the Scoring Function

The goal is to learn the parameters w_O and w_E of the scoring function $s(x, y; w_O, w_E)$ so that it can be used to rank trajectories according to the user's preferences. To do so, we adapt the Preference Perceptron algorithm [41] as detailed in Algorithm 1, and we call it the Trajectory Preference Perceptron (TPP). Given a context x_t, the top-ranked trajectory y_t under the current parameters w_O and w_E, and the user's feedback trajectory $\bar{y}_t$, the TPP updates the weights in the direction $\phi_O(x_t, \bar{y}_t) - \phi_O(x_t, y_t)$ and $\phi_E(x_t, \bar{y}_t) - \phi_E(x_t, y_t)$ respectively. Figure 5 shows an overview of our system design.

Despite its simplicity and even though the algorithm typically does not receive the optimal trajectory $y_t^* = \arg\max_y s^*(x_t, y)$ as feedback, the *TPP enjoys guarantees on the regret* [41]. We merely need to characterize by how much the feedback improves on the presented ranking using the following definition of expected α-informative feedback: $E_t[s^*(x_t, \bar{y}_t)] \geq s^*(x_t, y_t) + \alpha(s^*(x_t, y_t^*) - s^*(x_t, y_t)) - \xi_t$. This definition states that the user feedback should have a score of $\bar{y}_t$ that is—in expectation over the users choices—higher than that of y_t by a fraction $\alpha \in (0, 1]$ of the maximum possible range $s^*(x_t, \bar{y}_t) - s^*(x_t, y_t)$. If this condition is not fulfilled due to bias in the feedback, the slack variable ξ_t captures the amount of violation. In this way any feedback can be described by an appropriate combination of α and ξ_t. Using these

[2]When RRT becomes too slow, we switch to a more efficient bidirectional-RRT.The cost function (or its approximation) we learn can be fed to trajectory optimizers like CHOMP [39] or optimal planners like RRT* [23] to produce reasonably good trajectories.



Algorithm 1 Trajectory Preference Perceptron. (TPP)

```
Initialize $w_O^{(1)} \leftarrow 0$, $w_E^{(1)} \leftarrow 0$
**for** $t = 1$ to $T$ **do**
    Sample trajectories $\{y^{(1)}, ..., y^{(n)}\}$
    $y_t = argmax_y s(x_t, y; w_O^{(t)}, w_E^{(t)})$
    Obtain user feedback $\bar{y}_t$
    $w_O^{(t+1)} \leftarrow w_O^{(t)} + \phi_O(x_t, \bar{y}_t) - \phi_O(x_t, y_t)$
    $w_E^{(t+1)} \leftarrow w_E^{(t)} + \phi_E(x_t, \bar{y}_t) - \phi_E(x_t, y_t)$
**end for**
```

Fig. 5 Shows our system design, for grocery store settings, which provides users with three choices for iteratively improving trajectories. In one type of feedback (**zero-G** or **interative** feedback in case of PR2) user corrects a trajectory waypoint directly on the robot while in the second (**re-rank**) user chooses the top trajectory out of 5 shown on the simulator

two parameters, the proof by Shivaswamy and Joachims [41] can be adapted to show that average regret of TPP is upper bounded by $E[REG_T] \leq \mathscr{O}(\frac{1}{\alpha\sqrt{T}} + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t)$.

5 Related Work

Teaching a robot to produce desired motions has been a long standing goal and several approaches have been studied. Most of the past research has focussed on mimicking expert's demonstrations, for example, autonomous helicopter flights [1], ball-in-a-cup experiment [25], planning 2-D paths [36, 37], etc. Such settings (learning from demonstration, LfD) assume that kinesthetic demonstrations are intuitive to an end-user and it is clear to an expert what constitutes a good trajectory. In many scenarios, especially involving high DoF manipulators, this is extremely challenging to do [2].[3] This is because the users have to give not only the end-effector's location at each time-step, but also the full configuration of the arm in a spatially and temporally consistent manner. In our setting, the user never discloses the optimal trajectory (or provide optimal feedback), but instead, the robot learns preferences from sub-optimal suggestions for how the trajectory can be improved.

Some later works in LfD provided ways for handling noisy demonstrations, under the assumption that demonstrations are either near optimal [48] or locally optimal [29]. Providing noisy demonstrations is different from providing relative preferences, which are biased and can be far from optimal. We compare with an algorithm for noisy LfD learning in our experiments. A recent work [47] leverages user feedback to learn rewards of a Markov decision process. Our approach advances over [47] and

[3]Consider the following analogy. In search engine results, it is much harder for the user to provide the best web-pages for each query, but it is easier to provide relative ranking on the search results by clicking.

Calinon et al. [9] in that it models sub-optimality in user feedback and theoretically converges to user's hidden score function. We also capture the necessary contextual information for household and grocery store robots, while such context is absent in [9, 47]. Our application scenario of learning trajectories for high DoF manipulations performing tasks in presence of different objects and environmental constraints goes beyond the application scenarios that previous works have considered. We use appropriate features that consider robot configurations, object-object relations, and temporal behavior, and use them to learn a score function representing the preferences in trajectories.

In other related works, Berenson et al. [5] and Phillips et al. [34] consider the problem of trajectories for high-dimensional manipulators. They store prior trajectories for computational reasons for different tasks. These methods are complementary to ours, in that we could leverage their database of trajectories and train our system on samples drawn from it. Other recent works such as [13, 14, 45] consider generating human-like trajectories. These works are complementary to ours in that humans-robot interaction is an important aspect and such ideas could be incorporated in our approach.

In past, learning from demonstration [18, 31] and various interactive methods (e.g. human gestures) [8, 43] have been employed to teach assembly line robots. However, these methods either required the user to demonstrate an optimal trajectory or interactively show the complete sequence of actions which the robot remembered for future use. Recent works [32, 33] in human robot collaboration learn human preferences over a sequence of sub-tasks in assembly line manufacturing. However, these works are agnostic to the user preferences over robot's trajectories. Our algorithm can complement their's by learning preferences over the trajectories thereby achieving better human robot collaboration.

6 Experiments and Results

We first describe our experimental setup, then present quantitative results (Sect. 6.2), and then present robotic experiments on PR2 and Baxter (Sect. 6.3).

6.1 Experimental Setup

Task and Activity Set for Evaluation. We evaluate our approach on 35 robotic tasks in household setting and 16 pick-and-place tasks in a grocery store checkout setting. For household activities we use PR2, and use Baxter for the grocery store setting. To assess the generalizability of our approach, for each task we train and test on scenarios with different objects being manipulated, and/or with a different environment. We evaluate the quality of trajectories after the robot has grasped the items and while it moves them for checkout. Our work complements previous works on grasping items

[28, 40], pick and place tasks [20], and detecting bar code for grocery checkout [24]. We consider following three most commonly occurring activities in household and grocery stores:
(1) Manipulation centric: These activities primarily care for the object being manipulated. Hence the object's properties and the way robot moves it in the environment is more relevant. Examples of such household activities are pouring water into a cup or inserting pen inside a pen holder, Fig. 6 (Left). While in grocery store such activities could include moving flower vase, or moving fruits and vegetables, which can be damaged when dropped/pushed into other items. We consider *pick-and-place, pouring and inserting activities* with following objects: *cup, bowl, bottle, pen, cereal box, flower vase, tomatoes*. Further, in every environment we place many objects, alongwith the object to be manipulated, to restrict simple straight line trajectories.
(2) Environment centric: These activities also care for the interactions of the object being manipulated with the surrounding objects. Our object-object interaction features [19] allow the algorithm to learn preferences on trajectories for moving fragile objects like egg cartons or moving liquid near electronic devices, Fig. 6 (Middle). We consider moving *fragile items like egg carton, heavy metal boxes near a glass table, water near laptop and other electronic devices*.
(3) Human centric: Sudden movements by the robot put the human in a danger of

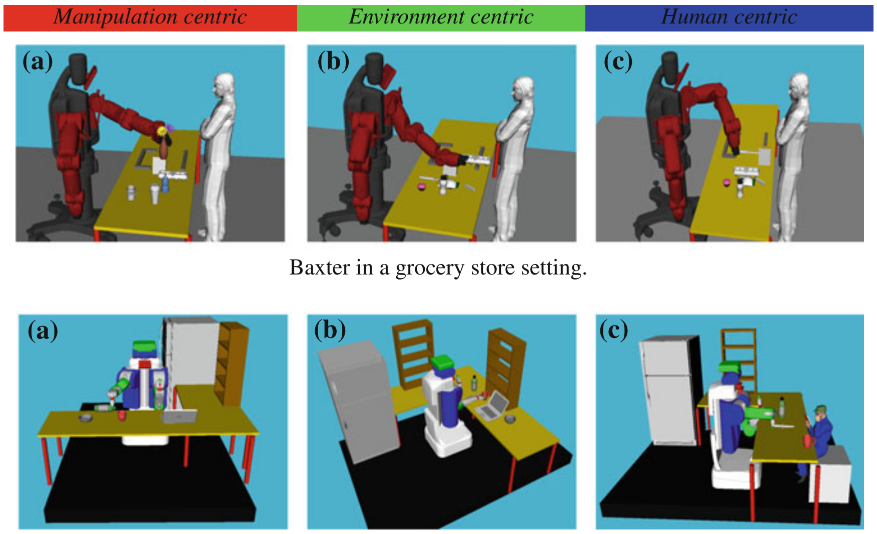


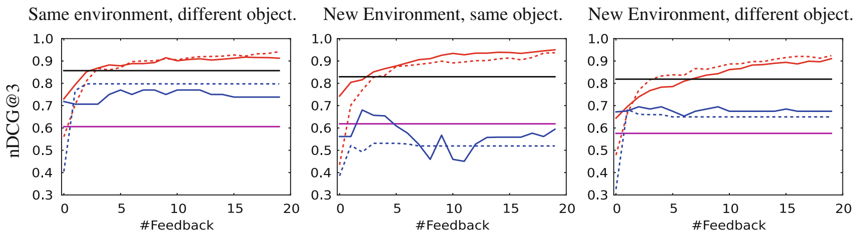
Fig. 6 Robot demonstrating different grocery store and household activities with various objects (*Left*) *Manipulation centric:* while pouring water the tilt angle of bottle must change in a particular manner, similarly a flower vase should be kept *upright*. (*Middle*) *Environment centric*: laptop is an electronic device so robot must carefully move water near it, similarly eggs are fragile and should not be lifted too high. (*Right*) *Human centric:* knife is sharp and interacts with nearby soft items and humans. It should strictly be kept at a safe distance from humans. (**Best viewed in color**)

getting hurt. We consider activities where a robot manipulates *sharp objects such as knife*, Fig. 6 (Right), *moves a hot coffee cup or a bowl of water with a human in vicinity*.

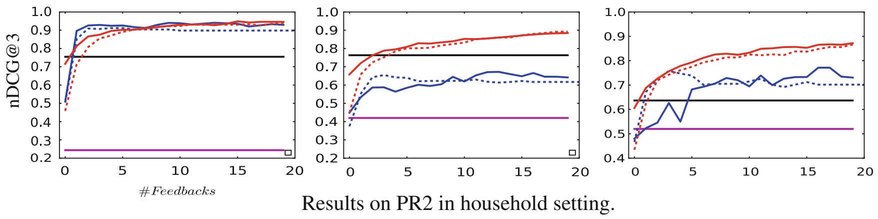
Baseline algorithms. We evaluate the algorithms that learn preferences from online feedback, under two settings: (a) *untrained*, where the algorithms learn preferences for the new task from scratch without observing any previous feedback; (b) *pre-trained*, where the algorithms are pre-trained on other similar tasks, and then adapt to the new task. We compare the following algorithms:

- *Geometric*: It plans a path, independent of the task, using a BiRRT [27] planner.
- *Manual*: It plans a path following certain manually coded preferences.
- *TPP*: Our algorithm, evaluated under both *untrained* and *pre-trained* settings.
- *Oracle-svm*: This algorithm leverages the expert's labels on trajectories (hence the name *Oracle*) and is trained using SVM-rank [21] in a batch manner. This algorithm is *not realizable in practice*, as it requires labeling on the large space of trajectories. We use this only in pre-trained setting and during prediction it just predicts once and does not learn further.
- *MMP-online*: This is an online implementation of Maximum margin planning (MMP) [37, 38] algorithm. MMP attempts to make an expert's trajectory better than any other trajectory by a margin, and can be interpreted as a special case of our algorithm with 1-informative feedback. However, adapting MMP to our experiments poses two challenges: (i) we do not have knowledge of optimal trajectory; and (ii) the state space of the manipulator we consider is too large, and discretizing makes learning via MMP intractable. We therefore train MMP from online user feedback observed on a set of trajectories. We further treat the observed feedback as optimal. At every iteration we train a structural support vector machine (SSVM) [22] using all previous feedback as training examples, and use the learned weights to predict trajectory scores for the next iteration. Since we learn on a set of trajectories, the argmax operation in SSVM remains tractable. We quantify closeness of trajectories by the l_2-norm of difference in their feature representations, and choose the regularization parameter C for training SSVM in hindsight, to give an unfair advantage to MMP-online.

Evaluation metrics. In addition to performing a user study (Sect. 6.3), we also designed two datasets to quantitatively evaluate the performance of our online algorithm. We obtained experts labels on 1300 trajectories in grocery setting and 2100 trajectories in household setting. Labels were on the basis of subjective human preferences on a Likert scale of 1–5 (where 5 is the best). Note that these absolute ratings are never provided to our algorithms and are only used for the quantitative evaluation of different algorithms. We quantify the quality of a ranked list of trajectories by its normalized discounted cumulative gain (nDCG) [30] at positions 1 and 3. While nDCG@1 is a suitable metric for autonomous robots that execute the top ranked trajectory (e.g., grocery checkout), nDCG@3 is suitable for scenarios where the robot is supervised by humans, (e.g., assembly lines). We also report average nDCG value over a given number of feedback iterations.



Results on Baxter in grocery store setting.



Results on PR2 in household setting.

Fig. 7 Study of generalization with change in object, environment and both. Manual, Oracle-SVM, Pre-trained MMP-online (—), Untrained MMP-online (– –), Pre-trained TPP (—), Untrained TPP (– –)

6.2 Results and Discussion

We now present the quantitative results where we compare TPP against the baseline algorithms on the data set of labeled trajectories.

How well does TPP generalize to new tasks? To study generalization of preference feedbacks we evaluate performance of TPP-pre-trained (i.e., *TPP* algorithm under *pre-trained* setting) on a set of tasks the algorithm has not seen before. We study generalization when: (a) only the object being manipulated changes, e.g., a bowl replaced by a cup or an egg carton replaced by tomatoes, (b) only the surrounding environment changes, e.g., rearranging objects in the environment or changing the start location of tasks, and (c) when both change. Figure 7 shows nDCG@3 plots averaged over tasks for all types of activities for both household and grocery store settings.[4] TPP-pre-trained starts-off with higher nDCG@3 values than TPP-untrained in all three cases. Further, as more feedback are provided, performance of both algorithms improves and they eventually give identical performance. We further observe, generalizing to tasks with both new environment and object is harder than when only one of them changes.

How does TPP compare to MMP-online? MMP-online proceeds by assuming every user feedback as optimal, and hence over the time it accumulates many contradictory/sub-optimal training examples. We empirically observe MMP-online generalizes better in grocery store setting than the household setting (Fig. 7), however

[4]Similar results were obtained with nDCG@1 metric, not included here due to space constraints.

under both settings its performance remains much below TPP. This also highlights the sensitivity of MMP to sub-optimal demonstrations.

How does TPP compare to Oracle-svm? Oracle-svm starts off with nDCG@3 values higher than any other algorithm (Fig. 7). The reason being, it is pre-trained using expert's labels on trajectories, and for the same reason it not realizable in practice. Furthermore, in less than 5 feedback on new task TPP improves over Oracle-svm, which is not updated since it requires expert's labels on test set.

How does TPP compare to Manual? We encode some preferences into the planners e.g., keep a glass of water upright. However, some preferences are difficult to specify, e.g., not to move heavy objects over fragile items. We empirically found (Fig. 7) the resultant manual algorithm produces poor trajectories—in comparison with TPP—with an average nDCG@3 of 0.44 over all types of household activities.
Table 1 reports nDCG values averaged over 20 feedback in untrained setting. For both household and grocery checkout activities TPP performs better than other baseline algorithms.

How does TPP perform with weaker feedback? To study the robustness of TPP to less informative feedback we consider the following variants of re-rank feedback:

1. Click-one-to-replace-top: User observes the trajectories sequentially in order of their current predicted scores and clicks on the first trajectory which is better than the top ranked trajectory.

2. Click-one-from-5: Top 5 trajectories are shown and user clicks on the one he thinks is the best after watching all 5 of them.

3. Approximate-argmax: This is a weaker feedback, here instead of presenting top ranked trajectories, five random trajectories are selected as candidate. The user selects the best trajectory among these 5 candidates. This simulates a situation when computing an argmax over trajectories is prohibitive and therefore approximated.

Figure 8 shows the performance of TPP-untrained receiving different kinds of feedback and averaged over three types of activities in grocery store setting. When feedback is more α-informative the algorithm requires fewer of those to learn preferences. In particular, click-one-to-replace-top and click-one-from-5 are more informative than approximate-argmax and therefore require fewer feedback to reach a given nDCG@1 value. Approximate-argmax being the least informative continues to show slow improvement. Since all the feedback are α-informative, for some $\alpha > 0$, eventually TPP-untrained is able to learn the preferences.

6.3 Robotic Experiment: User Study in Learning Trajectories

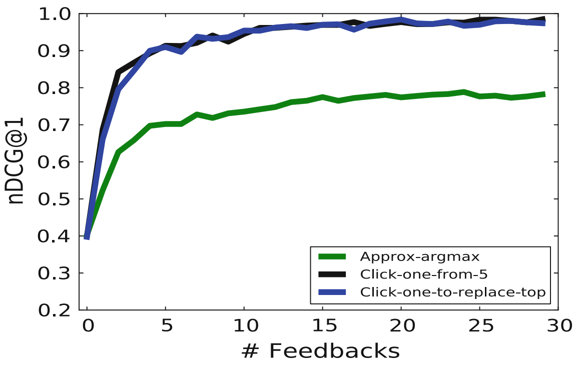
We perform a user study of our system on Baxter and PR2 on a variety of tasks of varying difficulties in grocery store and household settings respectively. Thereby we

Table 1 Comparison of different algorithms in untrained setting

	Grocery store setting on Baxter				Household setting on PR2			
Algorithms	Manipulation centric	Environment centric	Human centric	Mean	Manipulation centric	Environment centric	Human centric	Mean
Geometric	0.46 (0.48)	0.45 (0.39)	0.31 (0.30)	0.40 (0.39)	0.36 (0.54)	0.43 (0.38)	0.36 (0.27)	0.38 (0.40)
Manual	0.61 (0.62)	0.77 (0.77)	0.33 (0.31)	0.57 (0.57)	0.53 (0.55)	0.39 (0.53)	0.40 (0.37)	0.44 (0.48)
MMP-online	0.47 (0.50)	0.54 (0.56)	0.33 (0.30)	0.45 (0.46)	0.83 (0.82)	0.42 (0.51)	0.36 (0.33)	0.54 (0.55)
TPP	**0.88 (0.84)**	**0.90 (0.85)**	**0.90 (0.80)**	**0.89 (0.83)**	**0.93 (0.92)**	**0.85 (0.75)**	**0.78 (0.66)**	**0.85 (0.78)**

Table contains nDCG@1(nDCG@3) values averaged over 20 feedbackse

Fig. 8 Study of re-rank feedback



show our approach is practically realizable, and the combination of re-rank, zero-G/interactive feedback allows users to train the robot in few feedback.

Experiment setup: In this study, five users (not associated with this work) used our system to train Baxter on grocery checkout tasks, using zero-G and re-rank feedback. For training Baxter, the users provided zero-G feedback kinesthetically on the robot, while re-rank was elicited in a simulator. For PR2, in place of zero-G, two users provided interactive feedback on Rviz simulator. The two users were familiar with Rviz-ROS trained PR2 on household tasks.[5] A set of 10 tasks of varying difficulty level was presented to users one at a time, and they were instructed to provide feedback until they were satisfied with the top ranked trajectory. To quantify the quality of learning each user evaluated their own trajectories (self score), the trajectories learned by the other users (cross score), and those predicted by Oracle-svm, on a Likert scale of 1–5 (where 5 is the best). We also recorded the time a user took for each task—from start of training till the user was satisfied with the top ranked trajectory.

Is re-rank feedback easier to elicit from users than zero-G or interactive? In our user study, on average a user took 3 re-rank and 2 zero-G feedback per task to train a robot (Table 2). From this we conjecture, for high DoF manipulators re-rank feedback is easier to provide than zero-G—which requires modifying the manipulator joint angles. However, an increase in the number of zero-G (interactive) feedback with task difficulty suggests, Fig. 9 (Right), users rely more on zero-G feedback for difficult tasks since it allows precisely rectifying erroneous waypoints. Figures 10 and 11 show two example trajectories learned by a user.

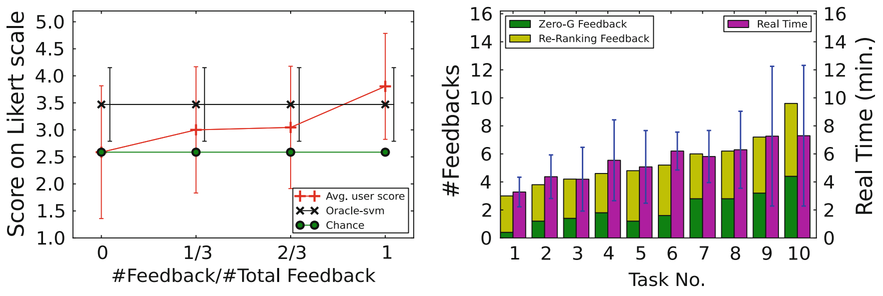
How many feedback a user takes to improve over Oracle-svm? On average, a user took 5 feedback to improve over Oracle-svm, Fig. 9 (Left), which is also consistent with our quantitative analysis. In grocery setting, user 4 and 5 were critical towards

[5]The smaller user size on PR2 is because it requires users with experience in Rviz-ROS. Further, we also observed users found it harder to correct trajectory waypoints in a simulator than providing zero-G feedback on the robot. For the same reason we report training time only on Baxter for grocery store setting.

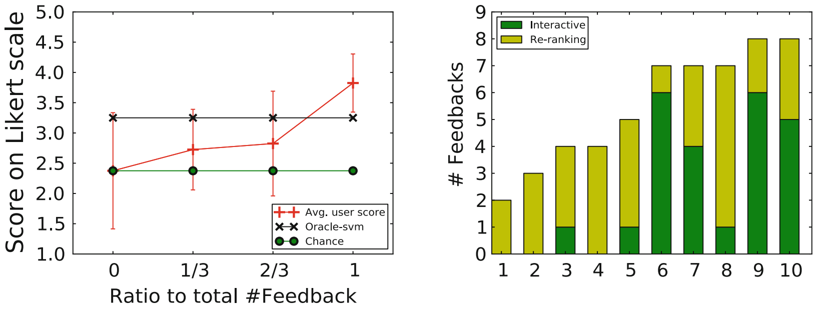
Table 2 Shows learning statistics for each user

User	# Re-ranking feedback	# Zero-G feedback	Average time (m)	Trajectory-quality	
				self	cross
1	5.4 (4.1)	3.3 (3.4)	7.8 (4.9)	3.8 (0.6)	4.0 (1.4)
2	1.8 (1.0)	1.7 (1.3)	4.6 (1.7)	4.3 (1.2)	3.6 (1.2)
3	2.9 (0.8)	2.0 (2.0)	5.0 (2.9)	4.4 (0.7)	3.2 (1.2)
4	3.2 (2.0)	1.5 (0.9)	5.3 (1.9)	3.0 (1.2)	3.7 (1.0)
5	3.6 (1.0)	1.9 (2.1)	5.0 (2.3)	3.5 (1.3)	3.3 (0.6)
User	# Re-ranking feedbacks	# Interactive feedbacks	Trajectory-quality		
			self	cross	
1	3.1 (1.3)	2.4 (2.4)	3.5 (1.1)	3.6 (0.8)	
2	2.3 (1.1)	1.8 (2.7)	4.1 (0.7)	4.1 (0.5)	

Self and cross scores of the final learned trajectories. The number inside bracket is standard deviation. **(Top)** Results for grocery store on Baxter. **(Bottom)** Household setting on PR2



Grocery store setting on Baxter.



Household setting on PR2.

Fig. 9 *(Left)* Average quality of the learned trajectory after every one-third of total feedback. *(Right)* Bar chart showing the average number of feedback (re-ranking and zero-G) and time required (only for grocery store setting) for each task. Task difficulty increases from 1 to 10

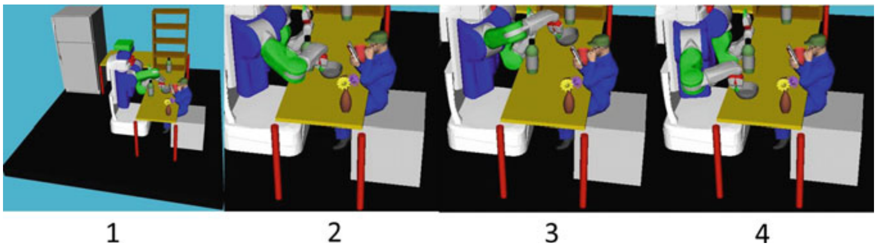


Fig. 10 Shows trajectories for moving a bowl of water in presence of human. Without learning robot plans an undesirable trajectory and moves bowl over the human (waypoints 1–3–4). After six user feedback robot learns the desirable trajectory (waypoints 1–2–4)

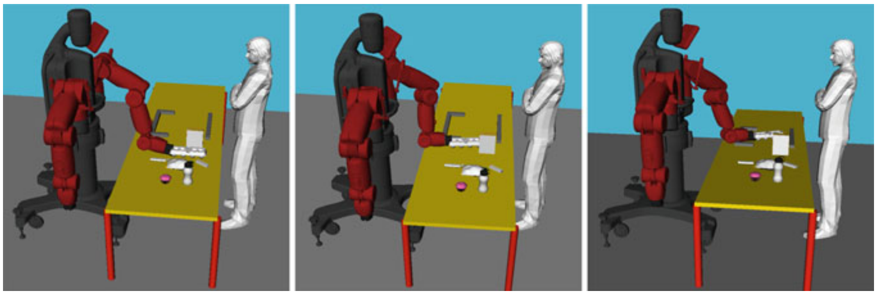


Fig. 11 Shows the learned trajectory for moving an egg carton. Since eggs are fragile robot moves the carton near the table surface. (*Left*) Start of trajectory. (*Middle*) Intermediate waypoint with egg close to the table surface. (*Right*) End of trajectory

trajectories learned by oracle-svm and gave them low scores. This indicate a possible mismatch in preferences between our expert (on whose labels oracle-svm trained) and user 4, 5.

How do users' unobserved score functions vary? An average difference of 0.6 between users' self and cross score (Table 2) in grocery checkout setting suggests preferences varied across users, but only marginally. In situations where this difference is significant and a system is desired for a user population, a future work might explore coactive learning for satisfying user population which has recently been applied to search engines [35]. For household setting the sample size is small to draw a such conclusion.

How long does it take for users to train a robot? We report training time only for grocery store setting, because the interactive feedback in household setting requires users with experience in Rviz-ROS. Further, we observed that users found it difficult to modify robot's joint angles in a simulator to their desired configuration. In grocery checkout setting, among all the users, user 1 had the strictest preferences and also experienced some early difficulties in using the system and therefore took longer than others. On an average, a user took 5.5 minutes per task which we believe is acceptable for most applications. Future research in human computer interaction, visualization and better user interface [42] could further reduce this time. For example, simultaneous visualization of top ranked trajectories instead of sequentially showing them to users (which we currently do) could bring down the time for re-rank feedback. Despite its limited size, through user study we show our algorithm is realizable in practice on high DoF manipulators. We hope this motivates researchers to build robotic systems capable of learning from *non-expert* users.

For more details, videos & code, visit: http://pr.cs.cornell.edu/coactive/

7 Conclusion

With manipulators in human environments, it is important for robots to plan motions that follow user's preferences. In this work, we considered preferences that go beyond simple geometric constraints and that considered surrounding context of various objects and humans in the environment. We presented a coactive learning approach for training robots these preferences through iterative improvements from non-expert users. Unlike in standard learning from demonstration approaches, our approach does not require the user to provide optimal trajectories as training data. We evaluated our approach on various household (with PR2) and grocery store checkout settings (with Baxter). Our experiments suggest that it is indeed possible to train robots within a few minutes with just a few incremental feedbacks from non-expert users.

Acknowledgments This research was supported by ARO, Microsoft Faculty fellowship and NSF Career award (to Saxena).

References

1. Abbeel, P., Coates, A., Ng, A.Y.: Autonomous helicopter aerobatics through apprenticeship learning. IJRR **29**(13) (2010)
2. Akgun, B., Cakmak, M., Jiang, K., Thomaz, A.L.: Keyframe-based learning from demonstration. IJSR **4**(4), 343–355 (2012)
3. Alterovitz, R., Siméon, T., Goldberg, K.: The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty. In: RSS (2007)
4. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. Robot. Autonom. Syst. **57**(5), 469–483 (2009)
5. Berenson, D., Abbeel, P., Goldberg, K.: A robot path planning framework that learns from experience. In: ICRA (2012)
6. Berg, J.V.D., Abbeel, P., Goldberg, K.: LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. In: RSS (2010)
7. Bhattacharya, S., Likhachev, M., Kumar, V.: Identification and representation of homotopy classes of trajectories for search-based path planning in 3d. In: RSS (2011)
8. Bischoff, R., Kazi, A., Seyfarth, M.: The morpha style guide for icon-based programming. In: Proceedings of the 11th IEEE International Workshop on RHIC (2002)
9. Calinon, S., Guenter, F., Billard, A.: On learning, representing, and generalizing a task in a humanoid robot. In: IEEE Transactions on Systems Man and Cybernetics (2007)

10. Cohen, B.J., Chitta, S., Likhachev, M.: Search-based planning for manipulation with motion primitives. In: ICRA (2010)
11. Dey, D., Liu, T.Y., Hebert, M., Bagnell, J.A.: Contextual sequence prediction with application to control library optimization. In: RSS (2012)
12. Diankov, R.: Automated Construction of Robotic Manipulation Programs. Ph.D. thesis, CMU, RI (2010)
13. Dragan, A., Srinivasa, S.: Generating legible motion. In: RSS (2013)
14. Dragan, A., Lee, K., Srinivasa, S.: Legibility and predictability of robot motion. In: HRI (2013)
15. Erickson, L.H., LaValle, S.M.: Survivability: Measuring and ensuring path diversity. In: ICRA (2009)
16. Gossow, D., Leeperand, A., Hershberger, D., Ciocarlie, M.: Interactive markers: 3-d user interfaces for ros applications [ros topics]. IEEE Robot. Autom. Mag. **18**(4), 14–15 (2011)
17. Green, C.J., Kelly, A.: Toward optimal sampling in the space of paths. In: ISRR (2007)
18. Hovland, G.E., Sikka, P., McCarragher, B.J.: Skill acquisition from human demonstration using a hidden markov model. In: ICRA (1996)
19. Jain, A., Wojcik, B., Joachims, T., Saxena, A.: Learning trajectory preferences for manipulators via iterative improvement. In: NIPS (2013)
20. Jiang, Y., Lim, M., Zheng, C., Saxena, A.: Learning to place new objects in a scene. IJRR, **31**(9) (2012)
21. Joachims, T.: Training linear svms in linear time. In: KDD (2006)
22. Joachims, T., Finley, T., Yu, C.: Cutting-plane training of structural SVMS. Mach Learn, **77**(1) (2009)
23. Karaman, S., Frazzoli, E.: Incremental sampling-based algorithms for optimal motion planning. In: RSS (2010)
24. Klingbeil, E., Rao, D., Carpenter, B., Ganapathi, V., Ng, A.Y., Khatib, O.: Grasping with application to an autonomous checkout robot. In: ICRA (2011)
25. Kober, J., Peters, J.: Policy search for motor primitives in robotics. Machine Learning, **84**(1) (2011)
26. Koppula, H.S., Saxena, A.: Anticipating human activities using object affordances for reactive robotic response. In: RSS (2013)
27. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. IJRR **20**(5), 378–400 (2001)
28. Lenz, I., Lee, H., Saxena, A.: Deep learning for detecting robotic grasps. In: RSS (2013)
29. Levine, S., Koltun, V.: Continuous inverse optimal control with locally optimal examples. In: ICML (2012)
30. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval, vol. 1, Cambridge University Press, Cambridge (2008)
31. Nicolescu, M.N., Mataric, M.J.: Natural methods for robot task learning: Instructive demonstrations, generalization and practice. In: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (2003)
32. Nikolaidis, S., Shah, J.: Human-robot teaming using shared mental models. In: HRI, Workshop on Human-Agent-Robot Teamwork (2012)
33. Nikolaidis, S., Shah, J.: Human-robot cross-training: Computational formulation, modeling and evaluation of a human team training strategy. In: IEEE/ACM ICHRI (2013)
34. Phillips, M., Cohen, B., Chitta, S., Likhachev, M.: E-graphs: Bootstrapping planning with experience graphs. In: RSS (2012)
35. Raman, K., Joachims, T.: Learning socially optimal information systems from egoistic users. In: Proceedings of the ECML (2013)
36. Ratliff, N.: Learning to search: structured prediction techniques for imitation learning. Ph.D. thesis, CMU, RI (2009)
37. Ratliff, N., Bagnell, J.A., Zinkevich, M.: Maximum margin planning. In: ICML (2006)
38. Ratliff, N., Silver, D., Bagnell, J.A.: Learning to search: Functional gradient techniques for imitation learning. Autonom. Robot. **27**(1), 25–53 (2009a)
39. Ratliff, N., Zucker, M., Bagnell, J.A., Srinivasa, S.: Chomp: Gradient optimization techniques for efficient motion planning. In: ICRA (2009b)

40. Saxena, A., Driemeyer, J., Ng, A.Y.: Robotic grasping of novel objects using vision. IJRR, **27**(2) (2008)
41. Shivaswamy, P., Joachims, T.: Online structured prediction via coactive learning. In: ICML (2012)
42. Shneiderman, B., Plaisant, C.: Designing The User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley Publication (2010)
43. Stopp, A., Horstmann, S., Kristensen, S., Lohnert, F.: Towards interactive learning for manufacturing assistants. In: Proceedings of the 10th IEEE International Workshop on RHIC (2001)
44. Sucan, I.A., Moll, M., Kavraki, L.E.: The Open Motion Planning Library. IEEE Robot. Autom. Mag. **19**(4):72–82 (2012). http://ompl.kavrakilab.org
45. Tamane, K., Revfi, M., Asfour, T.: Synthesizing object receiving motions of humanoid robots with human motion database. In: ICRA (2013)
46. Vernaza, P., Bagnell, J.A.: Efficient high dimensional maximum entropy modeling via symmetric partition functions. In: NIPS (2012)
47. Wilson, A., Fern, A., Tadepalli, P.: A bayesian approach for policy learning from trajectory preference queries. In: NIPS (2012)
48. Ziebart, B.D., Maas, A., Bagnell, J.A., Dey, A.K.: Maximum entropy inverse reinforcement learning. In: AAAI (2008)

Learning from Demonstrations Through the Use of Non-rigid Registration

John Schulman, Jonathan Ho, Cameron Lee and Pieter Abbeel

Abstract We consider the problem of teaching robots by demonstration how to perform manipulation tasks, in which the geometry (including size, shape, and pose) of the relevant objects varies from trial to trial. We present a method, which we call *trajectory transfer*, for adapting a demonstrated trajectory from the geometry at training time to the geometry at test time. Trajectory transfer is based on non-rigid registration, which computes a smooth transformation from the training scene onto the testing scene. We then show how to perform a multi-step task by repeatedly looking up the nearest demonstration and then applying trajectory transfer. As our main experimental validation, we enable a PR2 robot to autonomously tie five different types of knots in rope.

1 Introduction

This paper is concerned with teaching robots to perform manipulation tasks by demonstration. In other words, a human performs the task one or more times (by teleoperation or directly guiding the robot's end-effector), and a learning algorithm extracts the essence of these demonstrations so the robot can perform the task autonomously under different starting conditions.

Our main running example is tying knots in rope. Knot tying is required in several tasks of practical importance, including tying fasteners around wire bundles (common in aerospace applications) and surgical suturing. While it is our running example, our method is not specific to knot tying and we also experimentally illustrate its capabilities in folding clothing and tasks involving household objects.

The procedure for tying a given type of knot can be broken into several segments based on grab and release events. For example, one segment might be to grab the end of the rope, move it to form a loop, and then release it. For each of these segments, the end-effector trajectory ought to depend on the geometry of the rope in some way

J. Schulman (✉) · J. Ho · C. Lee · P. Abbeel
Department of Electrical Engineering and Computer Sciences at UC Berkeley, Berkeley, CA, USA
e-mail: john.d.schulman@gmail.com

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_20

that must be inferred by the algorithm. This paper presents a non-parametric way to learn from examples how the trajectory depends on the geometry of the initial state.

Our main contribution is to show how non-rigid registration can be used to adapt a demonstrated trajectory to a new situation. The problem of generalizing a single demonstration can be described as follows. Suppose that at training time, $\textsc{State}_{\text{train}}$ is the initial state of the rope (represented as an unorganized point cloud) and $\textsc{Traj}_{\text{train}}$ is the trajectory applied to that state by the demonstrator. At test time, the robot is presented with a new state $\textsc{State}_{\text{test}}$ and must generate a new trajectory $\textsc{Traj}_{\text{test}}$ based on the triple $(\textsc{State}_{\text{train}}, \textsc{Traj}_{\text{train}}, \textsc{State}_{\text{test}})$.

Our procedure, which we call *trajectory transfer*, is summarized as follows. We register $\textsc{State}_{\text{train}}$ to $\textsc{State}_{\text{test}}$, obtaining a nonrigid transformation $\mathbf{f} : \mathbb{R}^3 \to \mathbb{R}^3$, and then we apply $\mathbf{f}$ to $\textsc{Traj}_{\text{train}}$ to obtain the new trajectory $\textsc{Traj}_{\text{test}}$. See Fig. 1 for illustration. Trajectory transfer enables the robot to generalize from a single demonstration to a much larger part of the state space. If multiple demonstrations are available, one can look up the demonstration with the closest initial state and then use trajectory transfer.

Our second contribution is a method for completing multi-step tasks robustly using a large number of demonstrations of the task. The method is simple: we repeatedly look up the nearest neighbor, using the registration cost as the measure of distance, and then apply trajectory transfer. This simple procedure can be used as a complete policy that enables the robot to complete a multi-step task, eliminating the need to program a custom state machine. This policy recovers from failures, assuming that

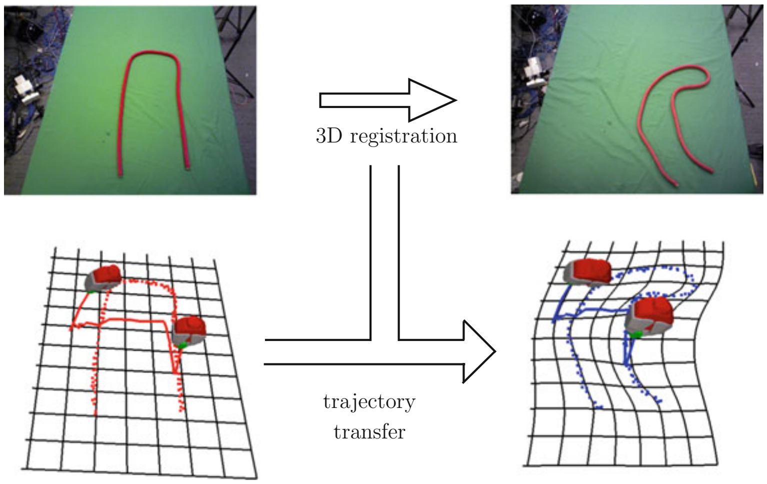


Fig. 1 Trajectory transfer as applied to the first stage (segment) of the overhand knot procedure. *Top left*: robot's first view of rope at training time. *Top right*: robot's first view of rope at testing time. *Bottom left*: point cloud of rope with demonstrated trajectory overlaid, along with x-y coordinate grid. *Bottom right*: point cloud of rope with warped trajectory generated by our algorithm, along with warped coordinate grid. Note that this trajectory is the first step of a multi-step procedure, and the warping will be performed at least two more times for this knot

the demonstrator has provided examples of the failure states along with the corrective motions that get out of them.

The main contributions of this work can be summarized as follows:

- This is the first work to use non-rigid registration to adapt end-effector trajectories to a new scene; we use a non-rigid registration formulation that is well-suited to this task.
- We describe how to use optimization-based motion planning to optimally execute the transferred gripper trajectories.
- We present experiments on knot tying and other multi-step tasks, where we repeatedly use the registration cost to choose the nearest demonstration.

2 Related Work

Nonrigid registration has been used in other fields to transfer information from one geometric entity to another. In medical image analysis, a patient's image is commonly registered to an atlas to locate anatomical structures [1]. In 3D modeling, it has been used to fill in missing parts of scans [2]. In computer vision, nonrigid registration has been used for object recognition and handwriting recognition [3]. Also in vision, using nearest-neighbor based techniques (not involving registration) to transfer various sorts of metadata from one image onto another has had some recent success [4–6].

Learning from demonstrations, also known as programming by demonstration, has always been a topic of interest in robotics; see [7] for a review. Calinon et al. [8, 9] have advanced one line of work that is similar in motivation to ours, in that they develop a learning method to perform manipulation tasks under varying initial conditions. Their approach is based on learning a mixture of Gaussians to encode the joint distribution of the robot trajectory and the environment state variables, so that by conditioning on the environment state, they can infer the appropriate trajectory. Ye and Alterovitz have augmented this approach with a repair stage that does motion planning around the learned trajectory to avoid obstacles [10].

The approach of Calinon and Billard assumes access to a featurization of the environment, since regression requires an input vector of fixed dimensionality. Their approach is most applicable in situations when the learned trajectory depends on a few landmarks in the environment that can be reliably detected. Their approach is not applicable to knot tying, where there is no fixed-length vector of landmarks that can be extracted from every rope configuration. In contrast, our approach operates directly on point clouds—the outputs of our sensor hardware—so it is suitable for this task, and it can be applied to a new task without developing a new vision system.

Rope manipulation and knot tying have been a subject of robotics research for almost three decades [11]. Researchers have addressed the problem with motion planning [12, 13], learning from observation (using knot theoretic representations) [14], robotic hands with tactile feedback [15], and fixtures that enable robust open-loop execution [16]. Note that our work is targeted at general manipulation tasks and is not specialized for the knot tying problem.

3 Generalizing from One Example: Trajectory Transfer

This section describes our method for generalizing from a single demonstrated trajectory. The problem is follows: at training time, the initial state is $\textsc{State}_{\text{train}}$, and the demonstrator provides a trajectory $\textsc{Traj}_{\text{train}}$. At testing time, the robot is presented with a new state $\textsc{State}_{\text{test}}$ and must generate an appropriate trajectory $\textsc{Traj}_{\text{test}}$.

First we'll review non-rigid registration, which lies at the core of our approach. Non-rigid registration finds a mapping from a "source" geometry to a "target" geometry. In our application, the source geometry is $\textsc{State}_{\text{train}}$, and the target geometry is $\textsc{State}_{\text{test}}$.

3.1 Non-rigid Registration

3.1.1 Registration with Known Correspondences

First let us suppose that the source geometry consists of K landmark points $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_K$ and the target geometry consists of K corresponding landmark points $\mathbf{p}'_1, \mathbf{p}'_2, \ldots, \mathbf{p}'_K$. Then the registration problem is to compute a function $\mathbf{f} : \mathbb{R}^3 \to \mathbb{R}^3$ that maps each source point to its corresponding target point, which can be quantified as the optimization problem

$$\underset{\mathbf{f}}{\text{minimize}} \left\{ \textsc{Regularizer}(\mathbf{f}) + \sum_k \left\| \mathbf{f}(\mathbf{p}_k) - \mathbf{p}'_k \right\|^2 \right\}. \tag{1}$$

The regularizer encourages the function to be smooth, at the expense of increasing the norm of the residuals $\left\| \mathbf{p}'_k - \mathbf{f}(\mathbf{p}_k) \right\|$.

We'll use a particular regularizer: the thin plate spline functional,

$$\textsc{Regularizer}(\mathbf{f}) = \int d\mathbf{x} \sum_{i \in \{x,y,z\}} \left\| D^2 f_i(\mathbf{x}) \right\|^2_{\text{Frob}} \tag{2}$$

where $\|\cdot\|_{\text{Frob}}$ refers to the Frobenius norm, and i indexes over the spatial dimensions of the range of $\mathbf{f}$. The thin plate spline regularizer (2) encourages $\mathbf{f}$ to be globally smooth and assigns zero cost to affine functions. It has been observed in other fields (e.g., [17]) that thin plate splines extrapolate well on spatial data; extrapolation is important for our application. As shown in [18], Eq. (1) can be analytically solved efficiently; details are provided in Appendix "Thin plate splines".

3.1.2 Registration Without Known Correspondences

In many problems of interest we are not given corresponding landmarks, rather we have two unorganized point clouds $\mathbf{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_M\}$ and $\mathbf{P}' = \left\{\mathbf{p}'_1, \ldots \mathbf{p}'_N\right\}$ and we want to find a transformation $\mathbf{f}$ so that $f(\mathbf{P}) = \{\mathbf{f}(\mathbf{p}_1), \ldots, \mathbf{f}(\mathbf{p}_M)\}$ is similar to $\mathbf{P}'$ in some sense. We use a modification of the TPS-RPM algorithm of Chui and Ragnarajan [19], which alternates between the following steps (1) estimate correspondences between source and target point clouds, and (2) fit a thin plate spline transformation based on these correspondences. Details are provided in Appendix "Iterative Registration Algorithm".

3.2 Trajectory Transfer Procedure

Our trajectory transfer procedure consists of three steps:

Step 1: Find a transformation f from the training scene to the test scene. Suppose $\mathbf{P}_{\text{train}}$ and $\mathbf{P}_{\text{test}}$ are the point clouds of the manipulated object in the training and test scene, respectively. We perform non-rigid registration as described in Sect. 3.1.2 to obtain the transformation $\mathbf{f}$ that maps $\mathbf{P}_{\text{train}}$ onto $\mathbf{P}_{\text{test}}$.

Step 2: Apply transformation f to the demonstrated end-effector trajectory. Suppose the demonstrated end-effector trajectory is given by a series of poses $\mathbf{T}_1, \mathbf{T}_2, \ldots, \mathbf{T}_T$, where each pose $\mathbf{T}_t$ consists of a position $\mathbf{p}_t$ and an orientation $\mathbf{R}_t$.

We transform the positions and orientations as follows, to adapt the trajectory to the test situation:

$$\mathbf{p}_t \to \mathbf{f}(\mathbf{p}_t) \tag{3}$$

$$\mathbf{R}_t \to \text{orth}\left(\mathbf{J}_\mathbf{f}(\mathbf{p}_t)\mathbf{R}_t\right). \tag{4}$$

Here, $\mathbf{J}_\mathbf{f}(\mathbf{p})$ is the 3×3 Jacobian matrix of $\mathbf{f}$ evaluated at $\mathbf{p}$,

$$\mathbf{J}_\mathbf{f} = \begin{pmatrix} \partial f_x/\partial x & \partial f_x/\partial y & \partial f_x/\partial z \\ \partial f_y/\partial x & \partial f_y/\partial y & \partial f_y/\partial z \\ \partial f_z/\partial x & \partial f_z/\partial y & \partial f_z/\partial z \end{pmatrix}, \tag{5}$$

and orth$(\cdot)$ is a function that orthonormalizes a 3×3 matrix (e.g. using the SVD).

Equation (3) says that we apply the warping function $\mathbf{f}$ to all of the positions. As for rotations, the natural way to transform a vector $\mathbf{v}$ at a point $\mathbf{p}$ through a function $\mathbf{f}$ is to multiply it by $\mathbf{J}_\mathbf{f}(\mathbf{p})$, the Jacobian.[1] Equation (4) applies this transformation to

[1]In differential geometry, given a mapping $\mathbf{f}$ between two manifolds $\mathscr{M}$ and $\mathscr{N}$, the so-called pushforward maps the tangent space $T_\mathbf{p}$ at $\mathbf{p} \in \mathscr{M}$ to the tangent space $T_{\mathbf{f}(\mathbf{p})}$ at $\mathbf{f}(\mathbf{p}) \in \mathscr{N}$. In terms of coordinates, it multiplies a vector $\mathbf{v} \in T_\mathbf{p}$ by the Jacobian of the transformation [20].

the x, y, and z axes of the gripper, and then orthogonalizes the resulting basis so it corresponds to a gripper pose.

Note that if $\mathbf{f}$ happens to be a rigid transformation $\mathbf{T}_\mathbf{f}$, then our method simply left-multiplies each end-effector pose $\mathbf{T}_t$ by it

$$\mathbf{T}_t \rightarrow \mathbf{T}_\mathbf{f}\mathbf{T}_t. \tag{6}$$

Step 3: Convert the end-effector trajectory into a joint trajectory. The simplest approach would be to use inverse kinematics. We found that this approach was not sufficient because there is often no continuous and collision-free trajectory that achieves the desired end-effector trajectory, so we need to compromise. To enable the robot to follow the trajectory as closely as possible while satisfying constraints, we formulate the following optimization problem on the joint trajectory $\boldsymbol{\theta}_{1:T}$:

$$\begin{aligned}
&\underset{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_T}{\text{minimize}} \left[\sum_{t=1}^{T-1} \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t\|^2 + \mu \sum_{t=1}^{T} \left\| \text{err}\left(\tilde{\mathbf{T}}_t^{-1} \cdot \text{fk}(\boldsymbol{\theta}_t)\right)\right\|_{\ell_1}\right] \\
&\text{subject to} \\
&\quad \text{No collisions, with safety margin } d_{\text{safe}} \\
&\quad \boldsymbol{\theta}_{\min} \leq \boldsymbol{\theta}_{1:T} \leq \boldsymbol{\theta}_{\max} \qquad \text{(Joint limits)}
\end{aligned}$$

Here, $\tilde{\mathbf{T}}_t$ is the desired end-effector pose at time t, $\text{fk}(\cdot)$ indicates the robot's forward kinematics function applied to $\boldsymbol{\theta}_t$, and μ is a scalar parameter. $\text{err}(\cdot)$ is an error function that maps a pose in $SE(3)$ to an error vector in $\mathbb{R}^6$. In particular, after decomposing a pose $\mathbf{T}$ into translation $\mathbf{p}$ and quaternion rotation $\mathbf{q}$, the error vector is simply given by $(p_x, p_y, p_z, q_x, q_y, q_z)$.

We solve this problem using the trajectory optimization method from [21]. We initialize with the joint trajectory from the demonstration, which is in roughly the right part of configuration space. In our experiments, this initialization strategy led to finding good locally-optimal trajectories.

We will illustrate the trajectory transfer procedure with a two-dimensional toy example, where the task is to draw a two-dimensional curve through four guide-points. Note that this example merely illustrates the trajectory of positions, not orientations. The left image of Fig. 2 shows the training situation: environment shown in solid lines, gripper tip trajectory shown as a dotted line, coordinate grid lines shown as thin solid lines. The right image shows the test situation for which we want to predict a good gripper trajectory. The registered points are the four corners. First, we use the method of thin plate splines to find a function that maps the four corners of the square in the training situation to the four vertices of the new quadrilateral. Then we apply the nonlinear transformation $\mathbf{f}$ to the demonstrated path to obtain a new path (dotted line), which has the same topological characteristics. The warped grid lines are shown.

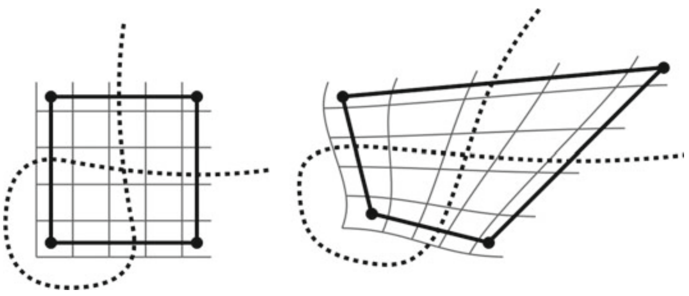


Fig. 2 Illustration of trajectory transfer procedure on a cartoon 2-D example. *Left*: training situation. *Right*: testing situation

3.3 Some Intuition on Conditions Under Which Trajectory Transfer Is Likely to Succeed

3.3.1 Cost Function Invariance

Our trajectory transfer procedure can be justified by an invariance assumption, which relates it to cost function learning. In inverse optimal control, one learns a cost function $L(\textsc{State}, \textsc{Traj})$ assuming that $\textsc{Traj}$ is generated according to $\textsc{Traj} = \arg\min_{\textsc{Traj}} L(\textsc{State}, \textsc{Traj})$. Probability density function estimation is closely related—here, L is the negative log-likelihood. Our trajectory transfer procedure can be justified by assuming that there is a class of smooth transformations $\mathbf{f}$ with the following property:

$$L(\textsc{State}, \textsc{Traj}) = L(\mathbf{f}(\textsc{State}), \mathbf{f}(\textsc{Traj})) \tag{7}$$

Here, $\mathbf{f}(\textsc{Traj})$ means transforming the trajectory as described in Sect. 3.2.

Given that the demonstration trajectory has low cost, and $\mathbf{f}$ transforms $\textsc{State}_{\text{train}}$ into $\textsc{State}_{\text{test}}$, it follows that trajectory transfer produces a low-cost trajectory:

$$\begin{aligned} L(\textsc{State}_{\text{test}}, \mathbf{f}(\textsc{Traj}_{\text{train}})) &\approx L(\mathbf{f}(\textsc{State}_{\text{train}}), \mathbf{f}(\textsc{Traj}_{\text{train}})) \\ &= L(\textsc{State}_{\text{train}}, \textsc{Traj}_{\text{train}}). \end{aligned} \tag{8}$$

Equation (7) is a strong assumption and defines L on a large part of the state space using a single $(\textsc{State}, \textsc{Traj})$ pair. That said, one can imagine situations where Eq. (7) does not hold, even for a rigid transformation $\mathbf{f}$—for example, when absolute orientation of the robot's end-effector matters.

3.3.2 Dynamics Invariance

An alternative perspective is that trajectory transfer works in scenarios where the dynamics of the system are approximately invariant—more properly, *covariant*—

under sufficiently smooth coordinate transformations. Suppose that the effect of applying the trajectory $\textsc{Traj}$ is given by the propagator $\Pi_{\textsc{Traj}}$, so that

$$\Pi_{\textsc{Traj}}(\textsc{State}^t) = \textsc{State}^{t+1} \tag{9}$$

The dynamics are defined to be covariant under the transformation $\mathbf{f}$ if and only if

$$\mathbf{f}(\Pi_{\textsc{Traj}}(\textsc{State}^t)) = \Pi_{\mathbf{f}(\textsc{Traj})}(f(\textsc{State}^t)) \tag{10}$$

which is illustrated by the following commutative diagram

$$\begin{tikzcd}
\left\{\textsc{State}^t_{\text{train}}\right\} \arrow[r, "\Pi_{\textsc{Traj}}"] \arrow[d, "\mathbf{f}"'] & \left\{\textsc{State}^{t+1}_{\text{train}}\right\} \arrow[d, "\mathbf{f}"] \\
\left\{\textsc{State}^t_{\text{test}}\right\} \arrow[r, "\Pi_{\mathbf{f}(\textsc{Traj})}"] & \left\{\textsc{State}^{t+1}_{\text{test}}\right\}
\end{tikzcd} \tag{11}$$

Let G denote a goal set, i.e., a set of desirable final states. Suppose that the assumptions hold:

1. $\mathbf{f}(\textsc{State}^t_{\text{train}}) = \textsc{State}^t_{\text{test}}$, i.e., $\mathbf{f}$ transforms $\textsc{State}_{\text{train}}$ into $\textsc{State}_{\text{test}}$.
2. $\textsc{State}^{t+1}_{\text{train}} \in G$, i.e., the demonstration ends up in the goal state.
3. $\mathbf{f}(G) = G$, i.e., the goal set is preserved by $\mathbf{f}$
4. Equation (10) holds, i.e., the dynamics are covariant under $\mathbf{f}$.

Then by applying the trajectory $\mathbf{f}(\textsc{Traj})$ to $\textsc{State}^t_{\text{test}}$, we obtain $\mathbf{f}(\textsc{State}^{t+1}_{\text{train}}) \in G$, i.e., the final state is in the goal set.

In the knot tying domain, the goal set is defined topologically, and a transformation $\mathbf{f}$ will preserve it, provided that it is a homeomorphism (i.e., a continuous function whose inverse is continuous), justifying assumption 3. Assumption 4 holds approximately in this domain; we have checked this qualitatively by comparing the final states of the rope at training and test time.

4 Generalizing from Many Examples: Nearest Neighbor Method

Intuitively, if $\textsc{State}_{\text{train}}$ is very close to $\textsc{State}_{\text{test}}$, then trajectory transfer is likely to work. Given a collection of demonstrations, with initial states $\textsc{State}_1, \textsc{State}_2, \ldots, \textsc{State}_K$, we can select the one that is closest to $\textsc{State}_{\text{test}}$, and use that one for trajectory transfer:

$$\textsc{State}_{i_*} = \arg \min_{i \in 1,2,\ldots,K} \textsc{Distance}(\textsc{State}_i, \textsc{State}_{\text{test}}). \tag{12}$$

One crucial question is how to measure distance. One natural distance measure of geometric similarity is the bidirectional fitting cost used in our registration procedure.

$$\begin{aligned} &\textsc{Regularizer}(\mathbf{f}) + \sum_k \left\|\mathbf{f}(\mathbf{p}_k) - \mathbf{p}'_k\right\|^2 \\ &+\textsc{Regularizer}(\mathbf{g}) + \sum_k \left\|\mathbf{g}(\mathbf{p}'_k) - \mathbf{p}_k\right\|^2 \end{aligned}$$

We found that this distance function agreed very well with our intuition about which states are closer and appropriate for trajectory transfer.

We can use this distance function to define a simple policy for completing multi-step tasks. The policy loops through the following steps until task completion:

1. Acquire a new observation $\textsc{State}_{\text{test}}$.
2. Choose the nearest demonstration using the registration cost, as in Eq. (12).
3. Apply trajectory transfer to obtain a new trajectory $\textsc{Traj}_{\text{test}}$.
4. Execute $\textsc{Traj}_{\text{test}}$ on the robot.

This policy enables the robot to recover from errors, provided that the demonstrator has provided a demonstration starting from the failure state.

5 Experiments with Rope

We enabled the PR2 to autonomously tie five different types of knot: overhand, figure-eight, double-overhand, square knot, and clove hitch (around a pole). The latter four knots are shown at several stages of execution in Fig. 3. Videos are available at http://rll.berkeley.edu/isrr2013lfd.

Teaching was performed kinesthetically, by guiding the robot's arms through the motion with its controllers off. The demonstrator noted *look*, *start*, and *stop* times, which indicated when to acquire a point cloud for the initial configuration and when the motion begins and ends. Point clouds were acquired from an RGBD camera mounted on the PR2 head. We extracted the rope points using color filtering (the rope was red or white, and the tablecloth was green).

We first performed an exploratory experiment with one demonstration per knot tie, where we measured how robust the procedure was under perturbations of the rope state of the demonstrations. We randomly perturbed the rope as follows. First the rope was laid out in the initial configuration from the demonstration, Then each of five points, uniformly spaced along the length of the rope, was manually dragged by a distance d_{pert} in a random direction. Then, we executed the knot-tying procedure and judged its success in tying the knot.

We performed five trials for each type of knot. The rates of success of these knot ties are shown in Table 1. As expected, the failure rate increases at higher d_{pert}, where the test situation is further away from the training situation. Two of the common failure modes were (1) the rope would end up in a previously unseen crossing state,

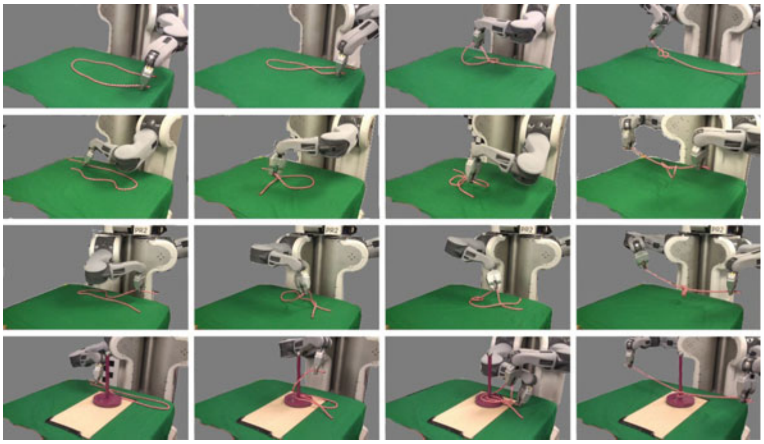


Fig. 3 Robot executing knot ties. *Top* row to *bottom* row: figure-eight knot, double-overhand knot, square knot, and clove hitch. Figure 1 already illustrated the overhand knot.

Table 1 Results for generalizing from a single demonstration: knot-tie success versus size of random perturbations

Knot type	Segments	Success ($d_{pert} = 3$cm)	($d_{pert} = 10$cm)
Overhand	3	5/5	4/5
Figure-eight	4	5/5	1/5
Dbl-overhand	5	3/5	3/5
Square	6	5/5	3/5
Clove hitch	4	1/5	0/5

often due to a small difference in its position; (2) the robot would grab the wrong piece of rope, or two pieces instead of one.

The above results suggest that when starting from a state that is very close to a demonstration, the success rate is high, at least for the easier knots. We hypothesized that if we collected enough demonstrations, every rope state of interested would be near some demonstration's initial state, so we would achieve that high level of performance over the entire state space by doing nearest neighbor lookups.

To test this hypothesis and explore how our algorithm performs in the many-demonstrations regime, we collected a large number of demonstrations of the overhand knot. The dataset contains a total of 148 trajectory segments, which include 36 demonstrations of the standard 3-step sequence and 40 additional segments starting from failure states—states that prevent the standard knot tying procedure from proceeding and require corrective actions.

Our initial (qualitative) results are promising: we find that the nearest-neighbor policy from Sect. 4 is able to successfully tie knots from a much larger set of initial configurations than was possible with a single demonstration, and it is also able to choose corrective movements to recover from the failure states. Two executions are shown in Fig. 4.

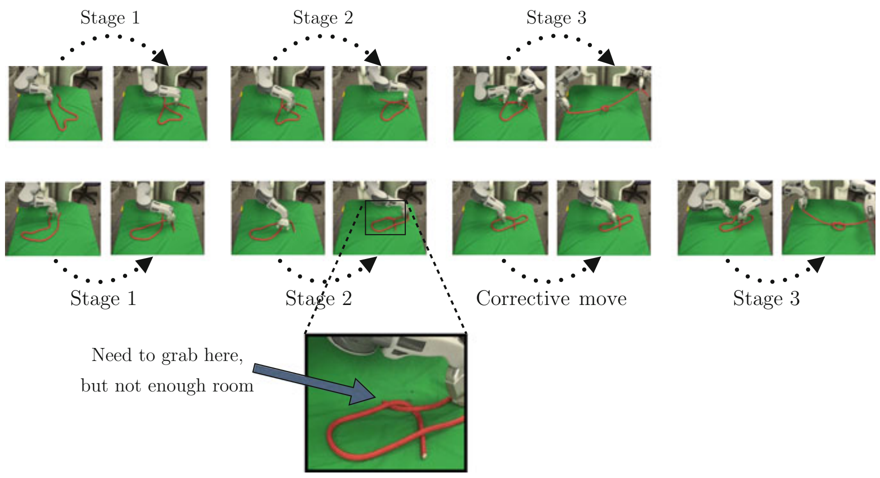


Fig. 4 Two successful executions of an overhand knot, based on our dataset of 148 demonstration segments. The *top* row shows the usual three-stage procedure. The *bottom* row shows a situation where an extra step was needed. After stage two, the rope ended up in a difficult state—the end that it needs to grab was too short. However, the nearest neighbor lookup found a demonstration with a very similar starting state and a corrective movement, which (after trajectory transfer) made the end graspable.

6 Experiments on Other Manipulation Tasks

We performed some other experiments to validate that the proposed method can be applied to manipulation tasks other than knot tying. The three tasks considered were folding a T-shirt, picking up a plate using a non-trivial two-arm motion, and opening up a bottle. The former task was executed using three segments, whereas the latter two consisted of one open-loop trajectory segment. We performed these tasks with the same algorithm and code that was used for the knot tying. See Fig. 5.

7 Conclusion

We have presented a method for adapting trajectories to new situations, based on non-rigid registration between the geometry of the training scene and the testing scene. This enables us to successfully perform a task under various initial conditions based on a single demonstration. Our method is justified by invariance assumptions as discussed in Sect. 3.3.

The non-rigid registration metric can also be used to find the nearest demonstration, when multiple demonstrations have been provided. This simple scheme enables our algorithm to successfully perform challenging multi-step manipulation tasks.

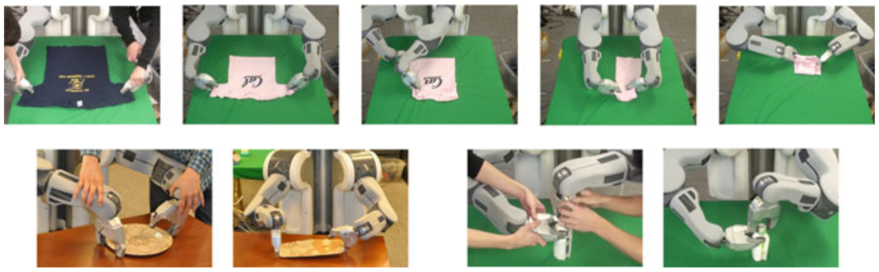


Fig. 5 *Top* row, *left* image: recording demonstration on a large T-shirt. *Top* row, *right* images: executing shirt folding procedure autonomously on small T-shirt, based on single demonstration on large T-shirt. *Bottom* row, *left*: demonstration and autonomous execution of plate pickup. Round plate from training was registered to rectangular plate at test time. *Bottom* row, *right*: demonstration and autonomous execution of bottle opening. The bottle used at test time had a different size and shape from the one used for training

8 Source Code

Complete source code and tutorials for our software are available at http://rll.berkeley.edu/rapprentice. Other supplementary material for this paper is available at http://rll.berkeley.edu/isrr2013lfd.

Acknowledgments This research has been funded in part by the Intel Science and Technology Center on Embedded Computing, by an AFOSR YIP grant, by a Sloan Fellowship, by a SURF Rose Hills Fellowship, and by NSF under award 1227536. We thank Sachin Patil and Ken Goldberg for their inspiring ideas and valuable advice. We thank Henry Lu and Robbie Gleichman for substantial contributions to the experiments. We would like to acknowledge the following open-source software that we used: OpenRAVE [22], ROS [23], PCL [24], OpenCV [25].

Appendix

This appendix describes our registration procedure, which is based of the TPS-RPM algorithm of Chui and Ragnarajan [19].

Thin plate splines

The classic method of smoothed thin plate splines [18] minimizes the following cost functional on $f : \mathbb{R}^d \to \mathbb{R}$:

$$J(f) = \sum_i (y_i - f(\mathbf{x}_i))^2 + \lambda \, \|f\|_{\text{tps}}^2 \tag{13}$$

Here, $\|f\|_{\text{tps}}$ is a measure of curvature or distortion, and is defined as

$$\|f\|^2_{\text{tps}} = \int d\mathbf{x} \left\|D^2 f(\mathbf{x})\right\|^2_{\text{Frob}} \tag{14}$$

where $D^2 f$ is the matrix of second partial derivatives of f, and $\|\cdot\|^2_{\text{Frob}}$ indicates its Frobenius norm, i.e., the sum of squares of its entries. λ is a parameter that controls the tradeoff between smoothness and goodness-of-fit.

Remarkably, the minimizer to the functional in Eq. (13) is a finite-dimensional expansion in terms of basis functions, centered around the data points $\mathbf{x}_i$, plus an affine part:

$$f(\mathbf{x}) = \sum_i a_i K(\mathbf{x}_i, \mathbf{x}) + \mathbf{b}^T\mathbf{x} + c \tag{15}$$

where K is the kernel function, and in 3D, $K(r) = -r$ (after dropping the irrelevant constant factor.)

In non-rigid registration, one needs to solve for a function $\mathbf{f} : \mathbb{R}^3 \to \mathbb{R}^3$, rather than a scalar-valued function. We can build a vector-valued function $\mathbf{f}$ by combining three scalar-valued components of the form (15), i.e., $\mathbf{f}(\mathbf{x}) = (f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}))^T$. Thus $\mathbf{f}$ has the form

$$\mathbf{f}(\mathbf{x}) = \sum_i \mathbf{a}_i K(\mathbf{x}_i, \mathbf{x}) + \mathbf{B}\mathbf{x} + \mathbf{c} \tag{16}$$

for $\mathbf{a}_i \in \mathbb{R}^3$, $\mathbf{B} \in \mathbb{R}^{3\times3}$, $\mathbf{c} \in \mathbb{R}^3$. Adding an additional regularization term $r(\mathbf{B})$ on the linear part of the transformation, we obtain the following optimization problem

$$\begin{aligned} \min_{\mathbf{A},\mathbf{B},\mathbf{c}} \left\|\mathbf{Y} - \mathbf{KA} - \mathbf{XB} - \mathbf{1}\mathbf{c}^T\right\|^2_{\text{Frob}} + \operatorname{trace}(\mathbf{A}^T\mathbf{KA}) + r(\mathbf{B}) \\ \text{subject to } \mathbf{X}^T\mathbf{A} = \mathbf{0}_{3\times3} \text{ and } \mathbf{1}^T\mathbf{A} = \mathbf{0}_{1\times3} \end{aligned} \tag{17}$$

where $\mathbf{X} = \left(\mathbf{x}_1\ \mathbf{x}_2 \cdots\right)^T$, $\mathbf{Y} = \left(\mathbf{y}_1\ \mathbf{y}_2 \cdots\right)^T$, and $\mathbf{A} = \left(\mathbf{a}_1\ \mathbf{a}_2 \cdots\right)^T$. For certain choices of $r(\mathbf{B})$ This problem completely decouples the three components of $\mathbf{f}$, i.e., we are separately fitting three functions for the separate output dimensions.

We found that the regularization $r(\mathbf{B})$ was necessary because sometimes the transformation is underdetermined in certain dimensions. For our experiments, we used $r(\mathbf{B}) = \operatorname{trace}\left((\mathbf{B} - \mathbf{I})^T\mathbf{D}(\mathbf{B} - \mathbf{I})\right)$, where $\mathbf{D}$ is a diagonal matrix. With this quadratic regularization term, the optimization problem still can be solved analytically as a least squares problem.

Iterative Registration Algorithm

This section considers the problem raised in Sect. 3.1.2 of registering two point clouds where we are not given correspondences between their points. We use a modification of the TPS-RPM algorithm of Chui and Ragnarajan [19], where the main modification is to jointly fit a forward transformation **f** and inverse transformation **g**. The algorithm iterates between two steps: soft assignment between the points, and fitting the pair of thin plate spline transformations **f**, **g**. First, let us assume that we have N source points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ and M target points $\mathbf{y}_1, \ldots, \mathbf{y}_M$.

The full optimization problem that we solve is the following

$$\begin{aligned}\underset{\mathbf{f},\mathbf{g}}{\text{minimize}} \sum_{n=1}^{N}\sum_{m=1}^{M} C_{nm}\left(\|\mathbf{f}(\mathbf{x}_n)-\mathbf{y}_m\|^2+\|\mathbf{g}(\mathbf{y}_m)-\mathbf{x}_n\|^2\right)+ \\ \textsc{Regularizer}(\mathbf{f})+\textsc{Regularizer}(\mathbf{g}) \qquad (18)\end{aligned}$$

such that $\mathbf{C} \in \mathbb{R}^{N\times M}$ is is the unique solution to the following constraints:

$$\begin{aligned} C_{nm} = s_n t_m \exp\left(-(\|\mathbf{y}_m-\mathbf{f}(\mathbf{x}_n)\|^2+\|\mathbf{x}_n-\mathbf{g}(\mathbf{y}_m)\|^2)/2\sigma^2\right), \\ n=1,\ldots,N,\ m=1,\ldots,M \\ \sum_{m=1}^{M} C_{nm} = 1, \quad n=1,\ldots,N \qquad (19) \\ \sum_{n=1}^{N} C_{nm} = N/M \quad m=1,\ldots,M \qquad (20)\end{aligned}$$

where $\mathbf{s}\in\mathbb{R}_+^N$ and $\mathbf{t}\in\mathbb{R}_+^M$ are scaling factors (uniquely determined by constraints (19) and (20)), and σ is a parameter that controls the correspondence difference, which will by systematically varied in the optimization procedure below.

Alternating optimization procedure. Following the Chui and Ragnarajan [19], we alternate between fitting (solving for **f**, **g**) and soft assignment (solving for **C**). Meanwhile, we are exponentially decreasing two parameters: a scale parameter σ that controls the correspondence distance, and the regularization parameter λ for the thin plate spline fitting. These parameters will be decreased exponentially in the series $\sigma_1, \sigma_2, \ldots, \sigma_{\textsc{NumIterations}}$ and $\lambda_1, \lambda_2, \ldots, \lambda_{\textsc{NumIterations}}$. The algorithm is given below:

```
Initialize $\mathbf{f}, \mathbf{g}$ to the identity
 For i = 1 to NumIterations
   Compute correspondence matrix $\mathbf{C}$
   (using $\mathbf{X}, \mathbf{Y}, \mathbf{f}, \mathbf{g}, \sigma_i$)
   Fit forward and inverse transformations $\mathbf{f}, \mathbf{g}$
   (using $\mathbf{X}, \mathbf{Y}, \mathbf{C}, \lambda_i$)
```

References

1. Hajnal, J.V., Hill, D.L.: Medical Image Registration. CRC Press, Boca Raton (2010)
2. Pauly, M., Mitra, N.J., Giesen, J., Gross, M.H., Guibas, L.J.: Example-based 3D scan completion. In: Symposium on Geometry Processing, pp. 23–32 (2005)
3. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. IEEE Trans. Pattern Anal. Mach. Intell. **24**(4), 509–522 (2002)
4. Efros, A.A., Freeman, W.T.: Image quilting for texture synthesis and transfer. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 341–346, ACM (2001)
5. Malisiewicz, T., Gupta, A., Efros, A.A.: Ensemble of exemplar-SVMS for object detection and beyond. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp. 89–96, IEEE (2011)
6. Liu, C., Yuen, J., Torralba, A.: Nonparametric scene parsing: Label transfer via dense scene alignment, In: IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009, pp. 1972–1979, IEEE (2009)
7. Calinon, S.: Robot programming by demonstration. In: Springer Handbook of Robotics, pp. 1371–1394, Springer, New York (2008)
8. Calinon, S., Guenter, F., Billard, A.: On learning, representing, and generalizing a task in a humanoid robot. IEEE Trans. Syst. Man Cybern. Part B: Cybern. **37**(2), 286–298 (2007)
9. Calinon, S., D'halluin, F., Caldwell, D., Billard, A.: Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework. In: 9th IEEE-RAS International Conference on Humanoid Robots, 2009. Humanoids 2009, pp. 582–588, IEEE, New Jersey (2009)
10. Ye, G., Alterovitz, R.: Demonstration-guided motion planning. In: International Symposium on Robotics Research (2011)
11. Inoue, H., Inaba, M.: Hand-eye coordination in rope handling. Robot. Res. First Int. Symp. **1**, 163–174 (1985)
12. Wakamatsu, H., Arai, E., Hirai, S.: Knotting/unknotting manipulation of deformable linear objects. Int. J. Robot. Res. **25**(4), 371–395 (2006)
13. Saha, M., Isto, P., Latombe, J.: Motion planning for robotic manipulation of deformable linear objects. In: International Symposium on Experimental Robotics (ISER) (2006)
14. Morita, T., Takamatsu, J., Ogawara, K., Kimura, H., Ikeuchi, K.: Knot planning from observation. In: Proceedings of the IEEE International Conference on Robotics and Automation, 2003 ICRA'03, vol. 3, pp. 3887–3892. IEEE, New York (2003)
15. Yamakawa, Y., Namiki, Y., Ishikawa, M., Shimojo, M.: One-handed knotting of a flexible rope with a high-speed multifingered hand having tactile sensors. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007, pp. 703–708. IEEE, New York (2007)
16. Bell, M.: Flexible object manipulation. Ph.D. thesis, Dartmouth College, Hanover, New Hampshire (2010)
17. Carr, J.C., Beatson, R.K., Cherrie, J.B., Mitchell, T.J., Fright, W.R., McCallum, B.C., Evans, T.R.: Reconstruction and representation of 3D objects with radial basis functions. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 67–76. ACM, New York (2001)
18. Wahba, G.: Spline models for observational data, Society for Industrial Mathematics, vol. 59 (1990)
19. Chui, H., Rangarajan, A.: A new point matching algorithm for non-rigid registration. Comput. Vis. Image Underst. **89**(2), 114–141 (2003)
20. Abraham, R., Marsden, J., Ratiu, T., Cushman, R.: Foundations of mechanics. Benjamin/Cummings Publishing Company, San Francisco (1978)
21. Schulman, J., Ho, J., Lee, A., Awwal, I., Bradlow, H., Abbeel, P.: Finding locally optimal, collision-free trajectories with sequential convex optimization. In: Proceedings of the Robotics: Science and Systems (2013)

22. Diankov, R., Kuffner, J.: Openrave: A planning architecture for autonomous robotics, Robotics Institute, Pittsburgh, PA, Technical report. CMU-RI-TR-08-34, p. 79 (2008)
23. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: Ros: an open-source robot operating system. In: ICRA workshop on open source software, vol. 3 (2009)
24. Rusu, R.B., Cousins, S.: 3D is here: Point cloud library (PCL). In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–4. IEEE, New York (2011)
25. Bradski, G.: The OpenCV library. Dr. Dobb's J. Softw. Tools (2000)

Part IV
Manipulation

Robust Contact Generation for Robot Simulation with Unstructured Meshes

Kris Hauser

Abstract This paper presents a numerically stable method for rigid body simulation of unstructured meshes undergoing forceful contact, such as in robot locomotion and manipulation. The key contribution is a new contact generation method that treats the geometry as having a thin virtual boundary layer around the underlying meshes. Unlike existing methods, it produces contact estimates that are stable with respect to small displacements, which helps avoid jitter or divergence in the simulator caused by oscillatory discontinuities. Its advantages are particularly apparent on non-watertight meshes and can easily simulate interaction with partially-sensed and noisy objects, such as those that emerge from low-cost 3D scanners. The simulator is tested on a variety of robot locomotion and manipulation examples, and results closely match theoretical predictions and experimental data.

1 Introduction

Physics simulators are useful tools for designing robust and safe robot behaviors because robot hardware is expensive and laborious to maintain. Control algorithms can be rapidly prototyped in simulation without the risk of damage to the robot or objects in the robots environment, and simulations provide developers with easy access to objective common benchmarks. For example, the recent DARPA Virtual Robotics Challenge asks teams to control a disaster relief robot capable of human skills in a physics simulator, with the assumption that high-quality virtual behaviors will be duplicable on a physical robot. But so far it has proven challenging to achieve high-fidelity simulations with low setup times. Existing simulators rarely yield "out of the box" performance for complex contact phenomena, and often take prohibitively long to tune. For stability, most existing simulators require that the environment and/or robot be equipped with collision hulls that approximate the true geometry with simple spheres, cylinders, boxes, or convex polyhedra. This is due to poor

K. Hauser (✉)
Indiana University, Bloomington, IN, USA
e-mail: hauserk@indiana.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_21

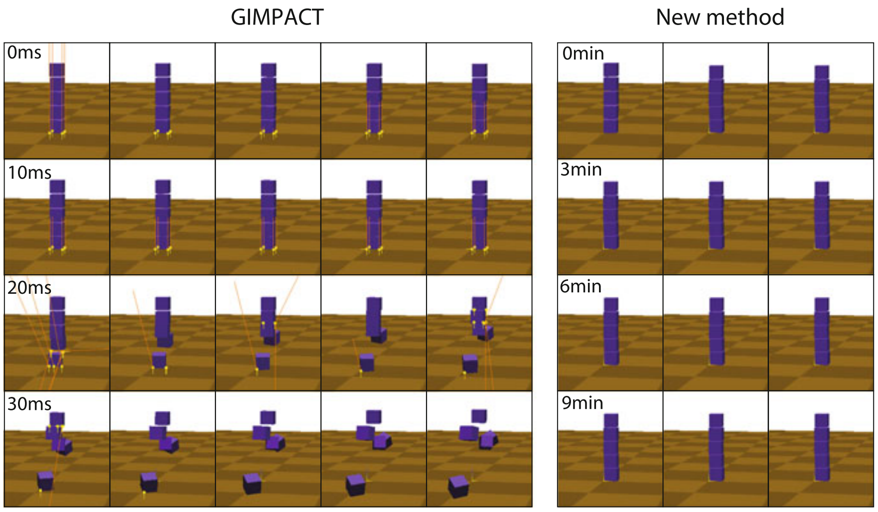


Fig. 1 A stack of triangulated cubes spaced 1 cm apart is dropped. *Left* using GIMPACT, the built-in mesh-mesh collision detection in the Open Dynamics Engine and Bullet libraries "blows up" nearly instantly after the second block touches the first (20 ms). Contact points and forces are displayed. *Right* the new method exhibits no perceptible jitter over several minutes of simulation time

contact handling between unstructured meshes, which induces simulation artifacts like jitter, phantom impulses, or worse, divergence ("blowing up") (Fig. 1).

This paper presents a simulator that achieves physically plausible simulations with contact between unstructured triangle meshes (so-called "polygon soup"). The heart of the method is a new contact generation scheme that yields stable contact handling (Fig. 1), and is invariant to changes of mesh topology and mesh resolution. It easily accommodates uncleaned CAD models, or noisy, partial meshes captured from 3D sensors, e.g., laser scanners or the Microsoft Kinect. This capability opens up new possibilities for rapid prototyping of robots interacting with complex environments and objects.

Particularly for objects in resting or sliding contact, discontinuities in contact generation reduce simulation accuracy and noticeably detract from realism. The technique presented in this paper is designed expressly to avoid such discontinuities. It introduces a new collision geometry representation that ensures a smooth change of contacts under small displacements for arbitrary meshes. Inspired by prior work on cloth simulation [1], the boundary layer expanded mesh (BLEM) represents object geometry as a triangular mesh fattened by a thin boundary layer. The expanded geometry is not computed explicitly, but rather only the width of the boundary layer is stored. The advantage of this approach is that as long as penetration is not too deep, penetration depth computation on a BLEM is equivalent to a distance computation on the underlying mesh. Distance computation is much more tractable than mesh-mesh penetration depths, and is exact as long as the boundary layer is not penetrated.

BLEMs are incorporated in an off-the-shelf rigid body simulator along with functionality to simulate robot sensors and motors. Experiments demonstrate that boundary layers only a few millimeters thick significantly improve stability above existing state-of-the-art robot simulators. Results match closely with theoretical predictions and experimental testing on a humanoid robot. The method successfully simulates many-DOF robots picking up sensed household objects and climbing on hands and legs on sensed terrains containing many holes and occlusions (Fig. 2).

The simulator is publicly available at http://klampt.org/ as part of the Klamp't (Kris' Locomotion and Manipulation Planning Toolkit) software package.

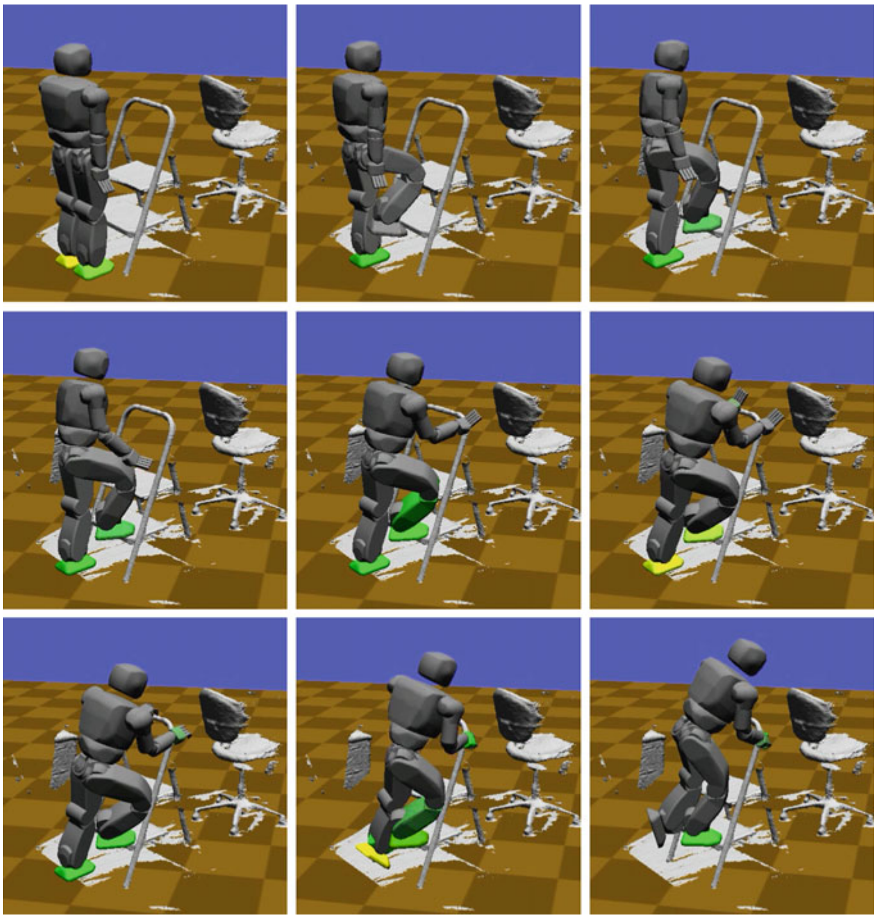


Fig. 2 Simulating the Hubo-II+ robot stepping up a stepladder sensed by a Microsoft Kinect camera. The environment model consists of 1 million triangles has many holes due to occlusions

2 Related Work

Many free and commercial robot simulation packages are available, the most popular ones being Gazebo [6, 7], Webots [5], USARSim [3, 23], and V-REP [4]. Like this work, all of them are built upon rigid body simulation engines meant for gaming and computer graphics (Open Dynamics Engine [17], Bullet [2], Nvidia's PhysX [16]). Each software package strongly recommends the use of geometric primitives or convex polyhedra for collision hulls. Non-convex, watertight meshes can be automatically decomposed into convex polyhedra [13], but complex geometries with small features and/or noise may require a huge number of convex pieces. The primary contribution of this work is a new contact generation scheme for unstructured meshes that need not be convex nor watertight.

Contact generation must be differentiated from other related problems in the simulation pipeline:

- Collision detection determines whether two geometries touch, but does not make an attempt to determine the objects motion after touching. Continuous collision detection is the problem of finding the first time of contact between moving bodies, and is often useful for preventing interpenetration of dynamically colliding bodies [19, 25]. However, it does not help with persistent contacts.
- Contact generation determines a representation for the local region of contact that will be resolved by the contact response stage [15]. In most systems this representation takes the form of a finite set of contact points and normal, possibly annotated with penetration depth.
- Contact response determines the future motion of objects in contact in order to prevent future interpenetration by generating physically-plausible contact forces and impulses. Solution techniques include complementarity problems [20], impulse-based methods [14], and penalty methods [21].

The problems of contact generation and contact response are highly interlinked, and the numerical stability of a simulation rests critically on the feedback loop between the two stages.

Interpenetration is impractical to prevent entirely, so simulators must anticipate small penetrations and resolve them after the fact. Hence, it is important for contact generation to compute accurate penetration depth estimates. Although doing so is exact and fast for geometric primitives and convex polyhedra, penetration depth is computationally hard to compute between nonconvex bodies. Minkowski sum methods lead to a complexity of $O(n^6)$ when only considering translational penetration depth, while considering translation and rotation leads to a complexity of $O(n^{12})$ [24]. Penetration depth approximation is an active field of study in computational geometry, but existing algorithms are still not fast enough for handling very large models with hundreds of thousands or millions of triangles [12]. Another approach [9] approximates penetration depth by precomputing a signed distance transform of the object on a volumetric grid, which leads to $O(1)$ time lookup of a point's penetration depth. Object-object contacts are detected by checking mesh vertices of one object with the signed distance table of another object, and vice versa. Other work, primarily designed for deformable objects, estimates penetration depths of finite-element meshes by propagating a wavefront from the colliding surface [10]. Unlike these methods, the current work does not require watertight meshes. However, it is less accurate for larger penetrations, so it is best suited for use with complementarity-based or impulse-based solvers that attempt to keep penetration close to zero throughout simulation.

Boundary layers around primitives have been explored in several contexts. The Blender Game Engine [22] modifies the Bullet simulator to support boundary layers around geometric primitives and convex polyhedra and is used to improve stability and model rounded objects. Most similarly to the current work, cloth simulators often fatten triangles of the mesh and apply penalty forces, which resolve most collisions [1]. The current method applies a similar approach to rigid-body meshes, with the primary effects of producing stabler contact estimates for mesh-mesh collisions and better tolerating numerical errors an off-the-shelf complementarity-based solver.

3 Contact Generation Method

3.1 Boundary-Layer Expanded Meshes

The boundary-layer expanded mesh (BLEM) is used as collision geometry throughout the method. A BLEM consists of a triangular mesh M along with a boundary layer width parameter $r \geq 0$ (Fig. 3), and treats collisions with the Minkowski sum of M and the sphere centered at the origin with radius r, $B(0, r)$:

$$G = M \oplus B(0, r) \tag{1}$$

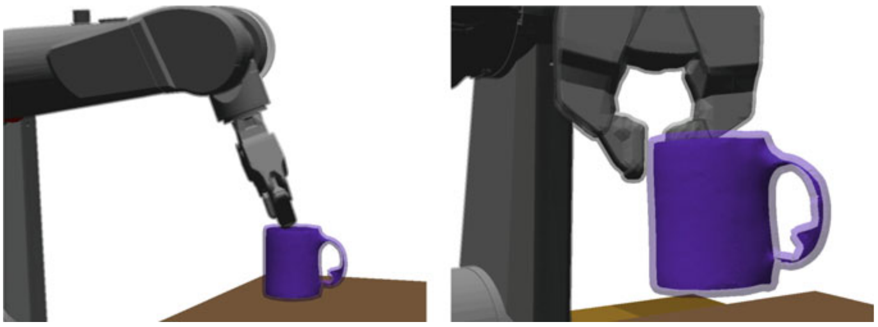


Fig. 3 Virtual boundary layers 2.5 mm thick allows a simulated robot to reliably pick up an object represented by a thin-shell triangulated mesh

Collision between objects A and B, represented as (M^A, r^A) and (M^B, r^B), is considered to occur when

$$(M^A \oplus B(0, r^A)) \cap (M^B \oplus B(0, r^B)) \neq \emptyset \tag{2}$$

or equivalently,

$$(M^A \oplus B(0, r^A + r^B)) \cap M^B \neq \emptyset. \tag{3}$$

3.2 Contact Generation

In addition to detecting collision, contact generation must produce a list of contact points, normals, and penetration depths so that an instantaneous motion of each point on object A in the direction of the normal reduces penetration depth. The method presented here uses a proximity checking approach on the underlying meshes, which is accelerated by a boundary volume hierarchy (Fig. 4).

Consider the set of n triangles in a mesh: $M = \bigcup_{i=1}^{n} t_i$. The Minkowski sum of the mesh and a sphere is simply the union of the fattened triangles:

$$M \oplus B(0, r) = \bigcup_{i=1}^{n} t_i \oplus B(0, r). \tag{4}$$

For objects (M^A, r^A) and (M^B, r^B), a pair of triangles t_i^A in M^A and t_j^B in M^B are considered to be in contact iff

$$\left(t_i^A \oplus B(0, r^A + r^B)\right) \cap t_j^B \neq \emptyset. \tag{5}$$

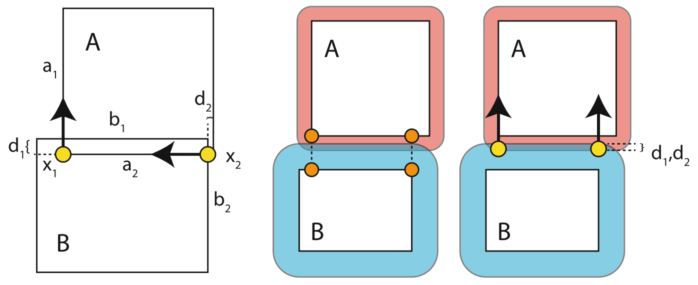


Fig. 4 *Left* Even for simple 2D cases, computing contacts with local mesh features produces poor results. The minimum penetration depths from the two pairs of colliding features give rise to one vertical and one horizontal contact. If simulated, this contact formation would lead to a rightward rotation of object A, which would cause increased penetration on the next timestep. *Right* the boundary-layer approach computes physically plausible contact points from the closest points on nearby features on the underlying meshes M_A and M_B

This is equivalently expressed using the distance operator $d(t, t')$ that computes the exact minimum distance between triangles.

$$d(t_i^A, t_j^B) \leq r^A + r^B. \tag{6}$$

Each evaluation of d is an $O(1)$ operation and uses basic mathematical operators. In other words, the contact generator outputs a single contact point for all pairwise triangles within distance $r^A + r^B$ (Fig. 5).

In the boundary-layer regime the pair of closest points (x^A, x^B) is unique and can also be computed in $O(1)$ time (Fig. 6). The penetration depth p is $r^A + r^B - \|x^A - x^B\|$, with contact normal $n = \frac{x^A - x^B}{\|x^A - x^B\|}$. The outputted contact point is $(x^A + x^B)/2 + n \cdot (r^B - r^A)/2$, which is placed in the middle of the overlap region. If the triangles intersect, the method finds the shortest retraction amongst all candidates as illustrated in Fig. 6. However, it is important to note that once triangles penetrate, all guarantees of globally consistency are lost and the simulator loses robustness.

To efficiently detect pairs of nearby triangles in sub-quadratic time, the simulator uses using a boundary volume hierarchy (BVH) method heavily based on the Proximity Query Package (PQP) [8]. In precomputation, a BVH consisting of progressively finer bounding volumes is computed for each mesh. Oriented bounding boxes are used in the current implementation. To detect contacts between two BVHs, a depth-first-search is performed on pairs of bounding volumes and branches are pruned if the bounding boxes are more than distance $r^A + r^B$ apart. Each box-box distance computation is an $O(1)$ operation. Finally, at the leaves of the search, primitive triangles are tested for proximity using the distance operator $d(\cdot, \cdot)$. The computational complexity of this procedure is highest when many pairs of triangles are in close proximity, and is negligible when objects are distant.

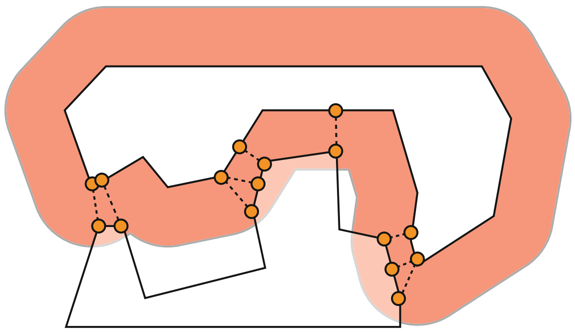


Fig. 5 The contact generator produces a contact point for all pairs of simplices whose distances are less than the sum of the boundary layer widths

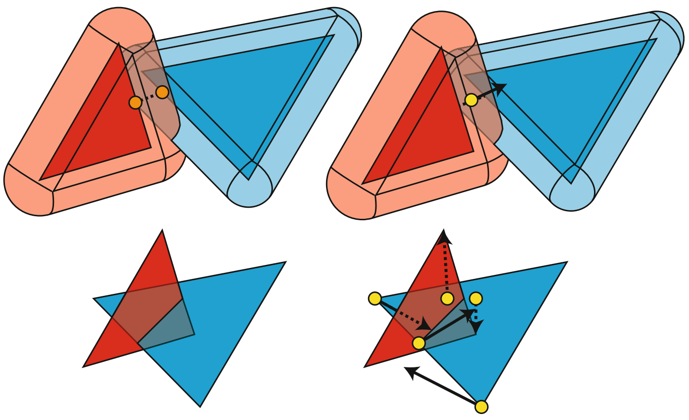


Fig. 6 *Top* In the boundary layer regime, the closest points on the underlying triangles correspond unambiguously to the deepest penetrating points of the expanded triangles. It is unnecessary to consider the topology of neighbors in the mesh. *Bottom* when triangles intersect, the direction of shortest retraction is not necessarily consistent with that of the neighbors, and the simulator loses robustness

3.3 Choosing Boundary Thickness

Contact generation is robust in the boundary layer regime, i.e., when BLEMs overlap but the underlying meshes do not interpenetrate. If meshes penetrate, the simulator falls back to less reliable contact generation methods. Hence, thicker boundary layers lead to more stable simulation. But, they lead to poorer approximations of object geometry. It may be possible to shrink meshes by the boundary thickness before simulation in order to better preserve the volume and dimensions of the object. However, this approach eliminates small features like corners, blades, and filaments. So, boundary thickness must be chosen carefully to balance the objectives of robustness and geometric fidelity.

To prevent mesh overlap, two colliding objects should not penetrate the sum of their boundary layer widths $r^A + r^B$ in a single time step. So, the boundary layer must be at least $v_{rel} \cdot \Delta t$ where v_{rel} is the relative velocity between the objects and Δt is the simulation time step. Boundary layers must also be thicker if the collision response is "soft", such as with penalty-based methods, or if very high forces are generated on light objects. Because simulation robustness depends on the sum of boundary widths, it is still possible to simulate sharp or very thin objects with zero boundary as long as they only make contact with objects with relatively thick boundaries.

3.4 Contact Clustering

In its basic form, the method can produce an excessive number of contact points because a point is generated for all pairs of nearby triangles. For example, edge-edge contacts will have four replicated contact points, one for each combination of adjoining faces. Too many contact points slows down the contact response stage, and also has the potential to reduce numerical stability of complementarity problem solvers. To address this problem, the contact generator uses a clustering method that limits the number of computed contacts to a user-defined maximum k.

Each contact on a single body (x, n, p) is considered a vector in a 7-D space and k clusters are selected from this set using an axis-weighted distance metric. Experiments have tested k-means and various hierarchical clustering methods. Simulation stability does not appear to be sensitive to the choice of clustering method, although if k-means is initialized with random clusters, a small amount of noise is injected into the simulation. Hierarchical clustering tends to produce marginally stabler results, but can be more expensive when N is large. We use an $O(N^2 \log N)$ implementation. So, when N is large, the simulator switches to k-means for a lower cost of $O(kNs)$, where N is the number of contacts and s is the iteration count (our implementation uses $s = k$).

4 Implementation and Experimental Results

Although BLEMs can hypothetically be used with any existing physics engine that accepts point contacts, the current implementation in Klamp't uses Open Dynamics Engine (ODE) v0.12 for low-level physics simulation and collision response. Experimental results are obtained on a single core of a 2.67 GHz Intel Core i7 laptop. Except for timing results, all times refer to simulation time rather than wall clock time. All simulations use a 1 ms time step, a 2.5 mm boundary on the robot and objects, and a 0 mm boundary on static environment geometry. The maximum number of contacts on the robot is 50 and the maximum number on each rigid object is 20. Supplemental videos for all examples in this paper can be viewed at http://www.iu.edu/~motion/simulation/.

4.1 Implementation

A robot is modeled as a linkage of rigid bodies connected by pin joints. The simulator can emulate both torque controlled and PID controlled motors. All examples in this paper use standard PID control with trajectory tracking, so that the torque applied at each motor at time t is:

$$k_P(\theta_D(t) - \theta(t)) + k_D(\dot{\theta}_D(t) - \dot{\theta}(t)) + k_I \int_{s=0}^{t} (\theta_D(s) - \theta(s))ds \tag{7}$$

where θ and θ_D are the actual and desired joint positions and $\dot{\theta}$ and $\dot{\theta}_D$ are the actual and desired joint velocities. Torques are then capped to actuator limits before being applied to the rigid bodies. The gains k_P, k_I, and k_D are specified at each joint.

Stick-slip friction is also modeled at the joints, with a simple friction model that includes dry and viscous coefficients μ_D and μ_V, respectively. In sticking mode, the friction torque is equal and opposite to the torque applied to the motor, with a breaking force of μ_D. In slipping mode, the friction torque is $-sign(\dot{\theta})(\mu_D + \mu_V|\dot{\theta}|)$.

For each motor in a robot, k_P, k_I, k_D, μ_D and μ_V must be set to reasonable values. Due to explicit time stepping in ODE, large values, especially of k_D, may cause motor instability. These parameters must be either hand-tuned or calibrated from experimental data before simulating.

The simulator also provides functionality to emulate force/torque sensors, actuator current sensors, contact sensors, accelerometers, and tilt sensors. Sensing feedback can be incorporated into the control strategy using a straightforward plugin mechanism.

4.2 Timing

Figure 7 shows timing statistics collected on the example of Fig. 2. On a few frames, contact generation is the limiting factor, with occasional spikes of up to 83 ms where a large number of triangles of the robot come near the environment. But on average, contact response is the dominant cost, taking 25 ms compared to 10 ms for contact generation. Clustering comprises approximately 2 ms of the average time. These plots also demonstrate that contact response becomes a dominant cost at a relatively small number of contacts (approximately 50) while the original contact detection stage routinely produces thousands of contacts for a similar computational cost. This highlights the fact that contact clustering is a critical step for achieving interactive simulation times.

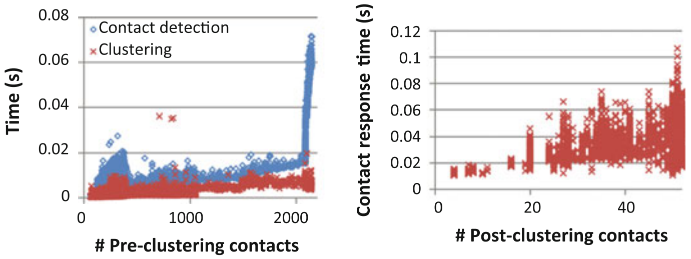


Fig. 7 Timing results for contact detection and contact response as a function of the number of contacts

4.3 *Humanoid Robot Force Sensor Simulation*

The Hubo-II+ is a 42 kg, 130 cm tall humanoid robot built by HuboLab at the Korean Advanced Institute of Science and Technology (KAIST). It consists of 57 rigid links including individual finger links, and 38 actuators (one actuator per finger). Numerical stability is a potential issue for simulating this robot, particularly when significant supporting forces are placed on the hands, because the mass ratio of the robot's torso to a finger link is over 150:1. The simulator has been used to simulate the robot's flat ground walking controller, and to verify the torque requirements and dynamic feasibility of a trajectory produced by a ladder climbing motion planner (Fig. 8).

Moreover, the simulator works sufficiently stably to generate physically plausible force/torque sensor readings that could be used for force control tasks. Figure 9 compares the results with experimental data from the force/torque sensors in the physical Hubo's ankles. Surprisingly, simulation came closer to ideal theoretical predictions! It appears that the discrepancies between the simulation and experimental data are caused by calibration issues in the physical robot: the physical robot appears to put 75 % of its weight on its right foot while standing on two feet, and moreover the sum of the support forces is not constant at rest.

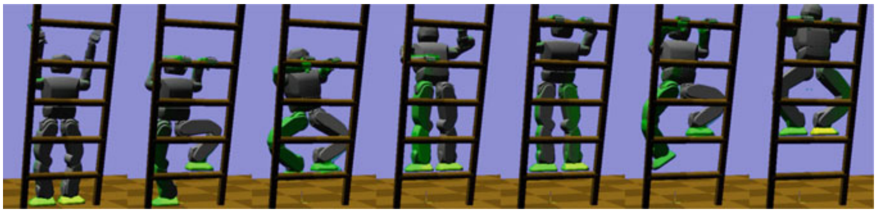


Fig. 8 A simulation of the Hubo-II+ dynamically climbing up a ladder to verify the feasibility of a quasi-statically stable motion calculated by a motion planner

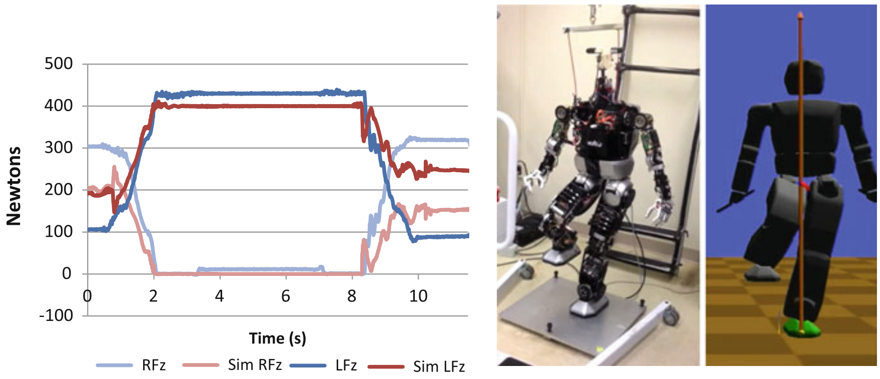


Fig. 9 Experimental comparison with force sensors at the feet of the Hubo-II+ humanoid robot as it shifts from two feet to balancing on its left foot, and then back to two feet

4.4 Comparison to GIMPACT

Because the simulator is built on top of Open Dynamics Engine, the BLEM contact generation method can be directly compared against the built-in contact detectors OPCODE and GIMPACT. This represents a fair comparison to the state-of-the-art in robot simulation: Gazebo and V-REP allow a choice between ODE and Bullet engines, both of which use GIMPACT; USARSim uses OPCODE; and Webots uses ODE. Our experiments immediately determined OPCODE to be unsuitable for realistic simulation because it would quickly "blow up" even on simple examples. GIMPACT is significantly better, and obtains realistic results for many scenarios: coarse meshes that are not-too-concave, and no edgewise contact between triangles.

Figure 1 illustrates that GIMPACT "blows up" almost instantaneously on a simple cube stacking example where each of the cubes is given by a triangulated mesh. At the instant that cube-to-cube contact is made, GIMPACT produces a problematic set of contacts that causes catastrophic failure soon after. By contrast, the new method

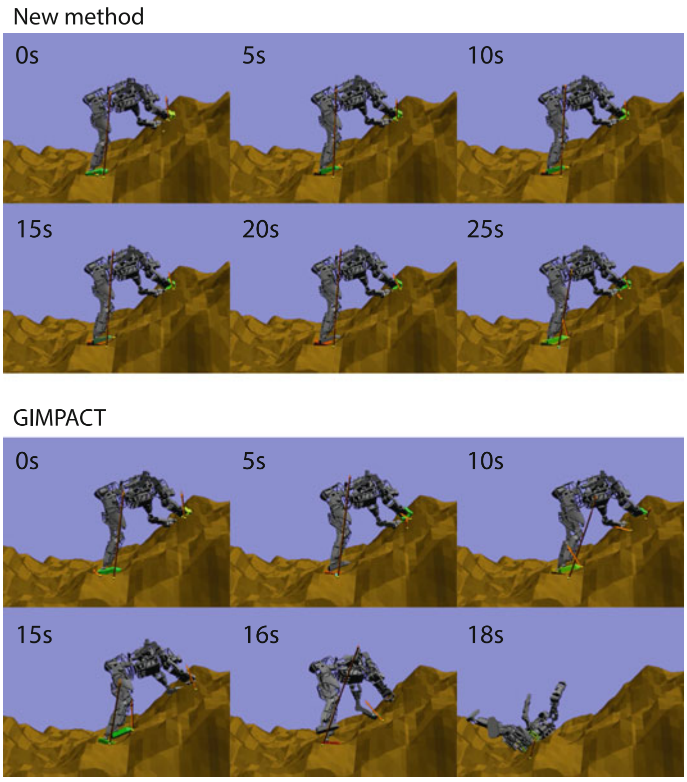


Fig. 10 A model of the Boston Dynamics Atlas humanoid balancing on hands and feet on rough, fractal-generated terrain. Contact forces are depicted as *arrows*. Using GIMPACT, the robot jitters until it begins an unrecoverable fall at 15 s

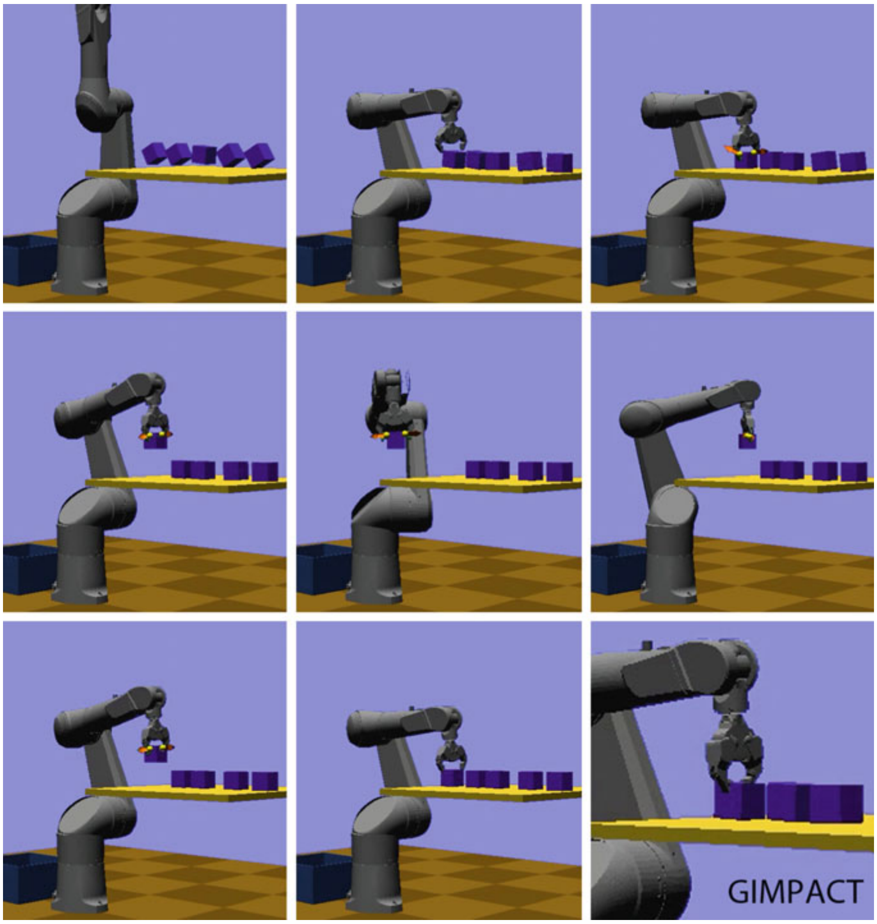


Fig. 11 A Staubli TX90L robot affixed with the gripper from the Willow Garage PR2 picks up a block, shakes it, and places it down using the new simulator. Contact forces are visualized as arrows. The last frame shows that using GIMPACT, the gripper fingers immediately slip through the block and fails to lift the block entirely

stays stable for tens of minutes, and appears to be stable in perpetuity. It has also been tested on a 10 cube stack for 10 min with less than a tenth of a millimeter of deviation from the theoretical prediction.[1]

The example of Fig. 10 shows that even when GIMPACT appears to work for a short while, subtle jitter can accumulate to yield implausible results. The Boston Dynamics Atlas robot model from DRCSim v2.2.0 is placed in a stable stance on its hands and feet on a rough terrain. The robot attempts to maintain its posture.

[1]ODE provides an option to disable simulation of rigid bodies that are not moving, which improves the stability of stacks. Such functionality was not used in these tests.

Using the new method, the robot flexes somewhat due to PID control but ultimately stays stably in place. After 3 min of simulation the robot maintained the same pose. Using GIMPACT, the robot stabilizes, but contact handling artifacts periodically apply impulses that cause the robot to jitter. After approximately 15 s these impulses grow sufficiently strong to push the robot off-balance.

The example of Fig. 11 demonstrates that the BLEM approach can handle contact between meshes of fine and coarse resolutions, while GIMPACT has problems with fine-resolution meshes. Here each fingertip of the Willow Garage PR2 gripper is modeled with approximately 1,000 triangles 1 mm^2 in size, while the cubes are coarse, 12-triangle meshes. With BLEMs, the robot lifts the cube, shakes it left and right, and places it back on the table as would be expected. With GIMPACT, the triangles of the finger mesh simply pass through the cube and the grasp fails.

4.5 Simulating Interaction with Sensed Objects

The next set of examples evaluate the ability to simulate robots interacting with objects or environments that are partially-modeled by 3D sensors.

Figure 12 shows an industrial robot interacting with several mugs sensed by stereo vision moving around the mug. The resulting point cloud was segmented from the table and meshed into a thin shell. The model has no volume and no bottom. The robot reaches into a tray of several objects and stirs them around. No interpenetration between the mugs, robot, or tray occurred during this simulation. The robot can also lift mugs from the handle or from the sides using friction (Fig. 3).

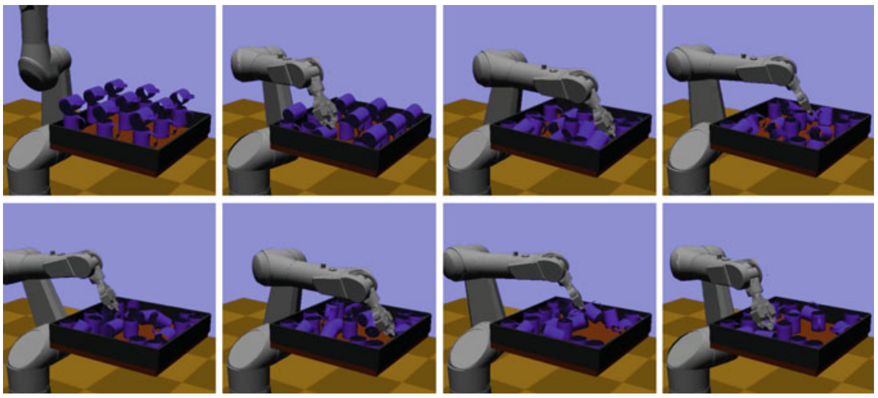


Fig. 12 A Staubli TX90L robot reaches into a tray of 18 mugs and stirs it around. Each mug was sensed using stereo vision. The mug model consists of 22,300 triangles, located only on the outward-facing sides of the cylinder and the outward-facing side of the handle (i.e., a thin shell). Throughout simulation no interpenetrations occur

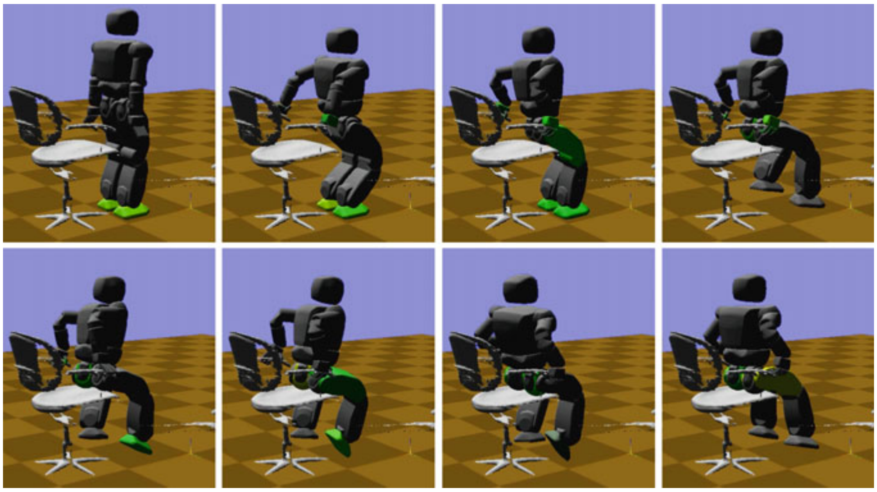


Fig. 13 The Hubo-II+ sits on a chair sensed by a Kinect. The environment model consists of 150,000 triangles and has missing data on the underside of the seats and armrests and on the back of the seat. The robot is somewhat too short to sit directly onto the chair, and instead it grasps and regrasps the armrests multiple times to readjust its posture and shift onto the chair

The final examples demonstrate locomotion on static environment meshes captured using the Microsoft Kinect sensor. Figures 2 and 13 demonstrate the Hubo-II+ humanoid climbing with its hands and legs on meshes obtained through the Kinect-Fusion algorithm [11]. These meshes are generated under indoor lighting conditions with a hand-held Kinect, with no attempt to build a complete model. As a result, the back and lower faces of the ladder and the chair are missing. Although the real chair is a swivel chair, the simulation model is fixed. Because these meshes are extremely large—1 million and 150,000 triangles, respectively—simulation is relatively slow. For 1 second of simulation, both examples take approximately 1 min of wall clock time, with computation dominated by the contact response step.

5 Conclusion

This paper described a contact generation method that achieves state-of-the-art stability for "out of the box" robot simulation with unstructured triangle meshes. Experimental results demonstrate physically plausible simulations on a variety of manipulation and locomotion examples, including interactions with noisy, partial 3D scans of real-world objects.

Interesting future directions might include studying the accuracy/speed tradeoffs of clustering, and developing better clustering methods. For example, contact points scattered on a plane can safely be reduced to their convex hull without any loss of accuracy in contact response. Another possibility is to incorporate methods for accelerating BLEM contact detection with large meshes or point cloud data sets [18], possibly using GPUs. Finally, it may be possible to develop methods that adapt boundary thickness and time stepping dynamically to support collisions between very thin objects, or to employ fallback methods that retract meshes out of collision once boundary layers are penetrated.

Acknowledgments The author thanks Anna Eilering for capturing the Kinect meshes and H. Andy Park for testing the Hubo robot's force sensors. This work is partially supported by Defense Advanced Research Projects Agency (DARPA) award # N65236-12-1-1005 for the DARPA Robotics Challenge. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of DARPA.

References

1. Bridson, R., Fedkiw, R., Anderson, J.: Robust treatment of collisions, contact and friction for cloth animation. In: ACM Transactions on Graphics (Proc. ACM SIGGRAPH) (2002)
2. Bullet physics engine. http://bulletphysics.org/ (2013). Accessed 31 Oct 2013
3. Carpin, S., Lewis, M., Wang, J., Balakirsky, S., Scrapper, C.: Usarsim: a robot simulator for research and education. In: IEEE International Conference on Robotics and Automation, pp. 1400–1405 (2007). doi:10.1109/ROBOT.2007.363180
4. Coppelia Robotics: virtual robot experimentation platform (v-rep). http://www.coppeliarobotics.com/ (2013). Accessed 31 Oct 2013
5. Cyberbotics: Webots: the mobile robot simulation software. http://www.cyberbotics.com/ (2013). Accessed 31 Oct 2013
6. Gazebo. http://gazebosim.org/ (2013). Accessed 31 Oct 2013
7. Gerkey, B.P., Vaughan, R.T., Howard, A.: The player/stage project: tools for multi-robot and distributed sensor systems. In: Proceedings of 11th International Conference on Advanced Robotics, pp. 317–323 (2003)
8. Gottschalk, S., Lin, M., Manocha, D.: OBB-tree: a hierarchical structure for rapid interference detection. In: ACM SIGGRAPH, pp. 171–180 (1996)
9. Guendelman, E., Bridson, R., Fedkiw, R.: Nonconvex rigid bodies with stacking. In: ACM Transactions on Graphics (Proc ACM SIGGRAPH) (2013)
10. Heidelberger, B., Teschner, M., Keiser, R., Müller, M., Gross, M.: Consistent penetration depth estimation for deformable collision response. In: Proceedings of Vision, Modeling, Visualization, pp. 339–346 (2004)
11. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A.: Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In: ACM Symposium on User Interface Software and Technology (2011)
12. Kim, Y., Otaduy, M., Lin, M., Manocha, D.: Fast penetration depth computation for physically-based animation. In: Symposium on Computer Animation (2002)
13. Lien, J.M., Amato, N.M.: Approximate convex decomposition of polyhedra. In: Proceedings of the ACM Symposium on Solid and Physical Modeling (SPM). Beijing, China (2007)
14. Mirtich, B.: Impulse-based dynamic simulation of rigid body systems. Ph.D. thesis, University of California, Berkeley (1996)
15. Nguyen, B., Trinkle, J.: dvc3d: a three dimensional physical simulation tool for rigid bodies with contacts and coulomb friction. In: The 1st Joint International Conference on Multibody System Dynamics (2010)

16. Nvidia Corporation: Physx. http://www.nvidia.com/object/nvidia_physx.html (2013). Accessed 31 Oct 2013
17. Open dynamics engine. http://www.ode.org/ (2013). Accessed 31 Oct 2013
18. Pan, J., Sucan, I., Chitta, S., Manocha, D.: Real-time collision detection and distance computation on point cloud sensor data. In: IEEE International Conference on Robotics and Automation (2013)
19. Redon, S., Kheddar, A., Coquillart, S.: Fast continuous collision detection between rigid bodies. Comput. Graph. Forum **21**(3), 279–287 (2002)
20. Stewart, D., Trinkle, J.: An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction.International Journal of Numerical Methods in. Engineering **39**, 2673–2691 (1996)
21. Tang, M., Manocha, D., Otaduy, M.A., Tong, R.F.: Continuous penalty forces. In: ACM Transactions on Graphics (Proc ACM SIGGRAPH) (2012)
22. The blender project. http://www.blender.org/ (2013). Accessed 31 Oct 2013
23. Usarsim. http://usarsim.sourceforge.net/ (2013). Accessed 31 Oct 2013
24. Zhang, L., Kim, Y., Varadhan, G., Manocha, D.: Generalized penetration depth computation. Comput. Aided Des. **39**(8), 625–638 (2007)
25. Zhang, X., Lee, M., Kim, Y.J.: Interactive continuous collision detection for non-convex polyhedra. Vis. Comput. **22**(9), 749–760 (2006). doi:10.1007/s00371-006-0060-0

Manifold Representations for State Estimation in Contact Manipulation

Michael C. Koval, Nancy S. Pollard and Siddhartha S. Srinivasa

Abstract We investigate the problem of using contact sensors to estimate the configuration of an object during manipulation. Contact sensing is very discriminative by nature and, therefore, the set of object configurations that activate a sensor constitutes a lower-dimensional manifold in the configuration space of the object. This causes conventional state estimation methods, such as particle filters, to perform poorly during periods of contact. The *manifold particle filter* addresses this problem by sampling particles directly from the contact manifold. When it exists, we can sample these particles from an *analytic representation* of the contact manifold. We present two alternative sample-based contact manifold representations that make no assumptions about the object-hand geometry: *rejection sampling* and *trajectory rollouts*. We discuss theoretical considerations behind these three representations and compare their performance in a suite of simulation experiments. We show that all three representations enable the manifold particle filter to outperform the conventional particle filter. Additionally, we show that the trajectory rollout representation performs similarly to the analytic method despite the rollout method's relative simplicity.

1 Introduction

Humans effortlessly use their sense of touch to manipulate objects. Imagine groping around on a nightstand for a glass of water, or feeling around a cluttered kitchen cabinet while searching for the salt shaker. Each of these tasks involves *contact manipulation* during which we make *persistent contact* with the environment. Observing contact is critical during these tasks to localize objects during manipulation.

M.C. Koval (✉) · N.S. Pollard · S.S. Srinivasa
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: mkoval@cs.cmu.edu

N.S. Pollard
e-mail: nsp@cs.cmu.edu

S.S. Srinivasa
e-mail: siddh@cs.cmu.edu

Armed with real-time observations from tactile sensors [4, 11, 15], manipulators should also be able to estimate the *state* of the manipulated object—a problem we formalize as the *state estimation for contact manipulation* problem (Sect. 2).

Early work attempted to solve this problem by deriving analytical state estimators and controllers to track and control the pose of an object from contact positions based on simple models of physics [5]. However, these models fail to accurately capture the reality of manipulation because there is a large amount of uncertainty in both the object's motion and the robot's observations.

Other work has employed a Bayesian approach by using a particle filter to estimate the pose [19] and physical properties [20] of an object during manipulation. However, our prior work [6] revealed that the conventional particle filter (CPF) performs poorly at real-time update rates and suffers from a startling problem: *the CPF systematically performs worse as sensor resolution increases* (Sect. 3).

This problem arises because contact sensing accurately *discriminates* between contact and no-contact. Topologically, the set of states that are consistent with a contact observation lies in the lower dimensional *observable contact manifold* embedded in the configuration space of the object (Sect. 2). Particles sampled from the state space during contact have low probability of being on the observable contact manifold and, as a result, there is *particle starvation* in the vicinity of the true state. The *manifold particle filter* (MPF) provides a principled way of solving this problem by sampling particles directly from the observable contact manifold (Sect. 4).

Applying the MPF to contact manipulation requires sampling particles from the observable contact manifold [6]. When it exists, an analytic representation (AM) of the manifold provides an exact and computationally efficient way of sampling from the dual proposal distribution (Sect. 5). However, computing an analytic representation of the contact manifold is not always possible.

We present (Sect. 6) two alternative sample-based contact manifold representations that make no assumptions about the object-hand geometry: rejection sampling (RS) and trajectory rollouts (TR). The RS representation distributes samples uniformly in the space surrounding the manifold, while the TR representation concentrates many samples on the regions of the manifold that we are most likely to encounter during execution.

Our results (Sect. 7) reveal the trade-offs between these representations. RS performs the worst. By distributing samples uniformly everywhere, even in unlikely regions, RS sparsely covers the observable contact manifold. Surprisingly, TR performs nearly as well as AM. By focusing samples on likely regions, TR saturated those regions at a resolution that was indistinguishable from the AM representation. A key reason is that likely regions occupy only a small portion of the observable contact manifold in our experiments, where the hand pushes straight towards the object.

Our key takeaway is to exploit structure. By exploiting the manifold structure of the contact state estimation problem, we are able to outperform the CPF. Furthermore, by exploiting the geometry of the hand-object interaction with trajectory rollouts, we are able to perform as well as the analytical method.

We are excited by our future directions. First, in generalizing state to include material properties and shape, which would enable us to simultaneously estimate shape and pose from contact. Second, in closing the loop between state estimation and control to develop robust closed-loop policies for contact manipulation.

2 State Estimation for Contact Manipulation

Let $x \in X$ be the state of a dynamical system which evolves over time under actions $u \in U$ and produces observations $z \in Z$. The *state estimation problem* addresses the computation of the *belief state*, a probability distribution over the current state x_t given the past history of actions $u_{1:t}$ and observations $z_{1:t}$ [17]:

$$b(x_t) = p(x_t | z_{1:t}, u_{1:t}). \tag{1}$$

We focus on the *state estimation for contact manipulation problem*, where the state is the pose $x \in X = SE(n)$ (Fig. 1-Left) of the manipulated object and an action $u \in U$ (Fig. 1-Middle) is a relative motion of the hand. During contact, the object moves according to the stochastic *transition model* $p(x_t | x_{t-1}, u_t)$ that encodes the physics of the object's motion in response to pushing action u_t. The stochasticity of the transition model may be due to unknown physical properties of the object (e.g. friction coefficients), imperfections in the physics simulation, or error in executing u_t [2].

Contact sensors attached to the surface of the hand provide observations $z_t \in Z$ (Fig. 1-Right) that indicate whether the object is touching the sensor. This is equivalent to testing whether $x_t \in X_o$, where $X_o \subseteq X$ is the set of states where the object is in contact with one or more sensors. While $x \in X_o$, z_t may provide additional information about the configuration of the object through noisy measurement of its contact with the hand. Both of these properties are combined into the stochastic

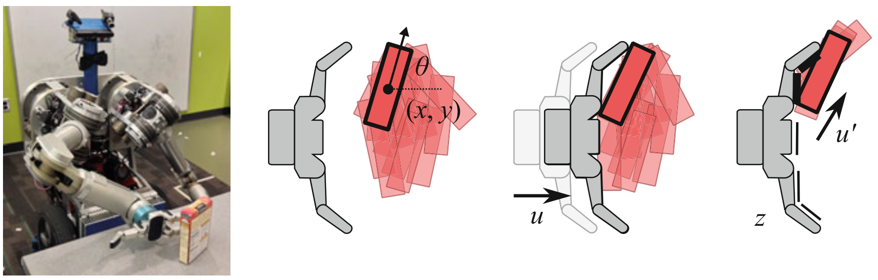


Fig. 1 HERB pushing a *rectangular box* across the table. The state $x \in X$ is the pose of the box relative to the hand. An action $u \in U$ is a relative motion of the hand. After taking action u, HERB receives an observation $z \in Z$ indicating where the object touched the hand

observation model $p(z_t | x_t, u_t)$ as the probability of state x_t generating observation z_t after executing action u_t.

The Contact Manifold. Contact manipulation poses a unique state estimation challenge because the state evolves on a lower-dimensional manifold embedded in X. We can partition X into three parts depending upon the type of contact occurring between the hand and the object: (1) penetrating contact X_{pen}, (2) non-penetrating contact X_c, and (3) no contact X_{free}. These three sets are defined by the interplay between the geometry of the object and the geometry of the hand.

Let $P_h \subseteq \mathbb{R}^n$ be the geometry of the hand and $P_o(x) \subseteq \mathbb{R}^n$ be the geometry of the object at configuration $x \in X$. The set of all object poses that are in collision with the hand form the *configuration space obstacle* [8]

$$X_{obs} = \mathrm{CO}_o(P_h) = \{x \in X : P_h \cap P_o(x) \neq \emptyset\}$$

of the hand in the object's configuration space.

Any configuration in $X_{pen} = \mathrm{int}(X_{obs})$ is invalid because the object penetrates the hand. Conversely, any configuration in $X_{free} = X \backslash X_{obs}$ is in *free space* where the object is out of contact with the hand. Therefore, any valid object configuration that is in contact with the hand must lie on the *contact manifold* $X_c = X_{obs} \backslash \mathrm{int}(X_{obs})$.

Figure 2 shows the contact manifold for the BarrettHand pushing an elongated rectangular box in $X = SE(2)$. Topologically, the contact manifold is a torus in $SE(2)$ with the top and bottom edges of the θ-dimension identified. The manifold is repeated twice along the θ-axis because the box exhibits rotational symmetry.

The Observable Contact Manifold. We know that $x \in X_c$ during periods of contact. However, our contact sensors may not be able to sense contact over the entire surface

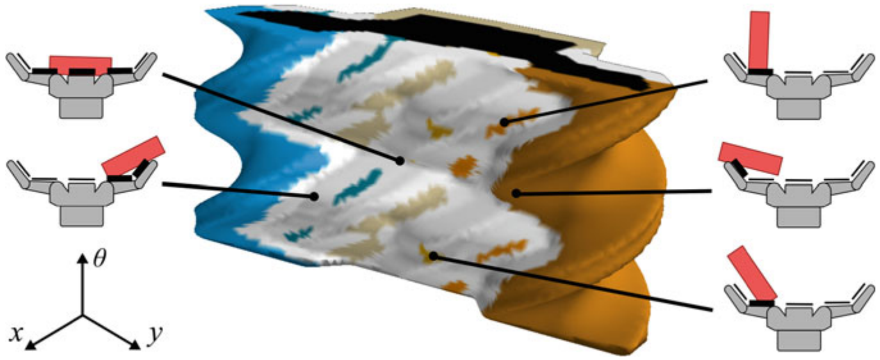


Fig. 2 Contact manifold X_c for a two-dimensional BarrettHand pushing a rectangular box. Each point $x \in X_c$ corresponds to a configuration of the object $x \in X_c$ that is in non-penetrating contact with the hand and is uniquely colored by the active contact sensors. Configurations that are in contact with multiple sensors are *white*

of the hand. We will define the *observable contact manifold* $X_o \subseteq X_c$ as the set of object poses that are capable of generating contact observations $z \in Z_c$.

Let $P_s \subseteq P_o \setminus \text{int}(P_o)$ denote the surface of the hand that is instrumented with contact sensors. The set of observable states X_s that could generate a contact observation is given by the configuration space obstacle

$$X_s = \text{CO}_o(P_s) = \{x \in X : P_s \cap P_o(x) \neq \emptyset\}$$

of the sensors in the object's configuration space. The *observable contact manifold* $X_o = X_s \cap X_c$ consists of the set of valid object configurations that have high probability of generating a contact observation $z \in Z_c$.

Figure 2 shows the contact manifold colored by which sensors are active at each point. Any state in the large, dark orange region of the manifold are in contact with—and, thus, are likely to activate—the left distal contact sensor. States in the large white region of the manifold are simultaneously in contact with multiple sensors.

Discriminative Observation Model. Contact sensors accurately discriminate between contact and no-contact. We call an observation model *discriminative* if we can partition the set of observations Z into sets of contact $Z_c \subseteq Z$ and no-contact $Z_{nc} = Z \setminus Z_c$ observations such that there are few false-positive and false-negative indications of contact. Therefore, a discriminative observation model satisfies $\Pr(z \in Z_c | x \in X_o, u) > 1 - \varepsilon$ during periods of contact and $\Pr(z \in Z_{nc} | x \notin X_o, u) > 1 - \varepsilon$ during no-contact. We otherwise make no assumptions about the ability of an observation to localize the object.

3 Conventional Particle Filter

The *Bayes filter* is the most general algorithm for filtering a belief state given a sequence of actions and observations by recursively constructing $b(x_t)$ from $b(x_{t-1})$ using the update rule

$$b(x_t) = \eta\, p(z_t | x_t, u_t) \int_X p(x_t | x_{t-1}, u_t) b(x_{t-1}) dx_{t-1} \tag{2}$$

where η is a normalization factor. The terms $p(z_t | x_t, u_t)$ and $p(x_t | x_{t-1}, u_t)$ are, respectively, the observation and transition models. The recursion is initialized with a prior belief $b(x_0)$ provided by task-specific knowledge or other sensors (e.g. an object recognition system).

The *particle filter* [17] is a non-parametric formulation of the Bayes filter that represents the belief state $b(x_t)$ with a discrete set of samples. The samples $X_t = \{x_t^{[i]}\}_{i=1}^n$ are called *particles* and are distributed according to the belief state $x_t^{[i]} \sim b(x_t)$. The particle filter implements the Bayesian update (Eq. 2) by recursively constructing X_t from X_{t-1} using a technique called *importance sampling*.

Algorithm 1 CPF

```
Require: $X_{t-1}$, particles from time $t-1$
Ensure: $X_t$, particles sampled from $b(x_t)$
1: $X_t \leftarrow \emptyset$
2: for all $x_{t-1}^{[i]} \in X_{t-1}$ do
3:     $x_t^{[i]} \sim p(x_t|x_{t-1}^{[i]}, u_t)$
4:     $w_t^{[i]} \leftarrow p(z_t|x_t^{[i]}, u_t)$
5:     $X_t \leftarrow \{x_t^{[i]}\} \cup X_t$
6: $X_t \leftarrow \text{Resample}(X_t)$
```

Algorithm 2 MPF

```
Require: $X_{t-1}$, particles from time $t-1$
Require: $k$, number of dual particles to sample
Ensure: $\bar{X}_t$, particles sampled from $b(x_t)$
1: $\bar{X}_{M_i} \leftarrow \emptyset$ for $i = 1, \ldots, m$
2: for $1, \ldots, k$ do
3:     $M_i \sim \Pr(x_t \in M_i)$
4:     if $M_i \neq M_m$ then
5:         $\bar{x}_t^{[i]} \sim p(z_t|M_i, x_t, u_t)/p(z_t|M_i, u_t)$
6:         $\bar{w}_t^{[i]} \leftarrow \text{EstimateDensity}(X_{t-1}, u_t, \bar{x}_t^{[i]})$
7:         $\bar{X}_{M_i} \leftarrow \{\bar{x}_t^{[i]}\} \cup \bar{X}_{M_i}$
8:     end if
9: $\bar{X}_{M_m} \leftarrow \text{CPF}(X_{t-1}, u_t, z_t) \cap M_m$
10: $\bar{X}_t \leftarrow \text{Resample}(\sum_{i=1}^{m} \Pr(x_t \in M_i)\bar{X}_{M_i})$
```

The conventional particle filter (CPF) is summarized in Algorithm 1. The key insight behind this realization is that it is difficult to directly sample from the *target distribution* (Eq. 2), but it is relatively easy to sample from the transition model. Therefore, we sample $x_t^{[i]}$ from the *proposal distribution* $\int_X p(x_t|x_{t-1}, u_t)b(x_{t-1})\, dx_{t-1}$ (line 3) by forward-simulating X_{t-1} to X_t using the motion model. Next, we compute an *importance weight* $w_t^{[i]} = p(z_t|x_t, u_t)$ for each forward-simulated particle (line 4).

The importance weights result from dividing the target distribution by the proposal distribution. As a result, the samples $x_t^{[i]}$, along with their importance weights $w_t^{[i]}$, are distributed according to the target distribution $b(x_t)$ [17]. Intuitively, the weighting step incorporates the observation model into the update by assigning higher weight to particles that are consistent with z_t.

The particle filter periodically *resamples* the set of weighted particles (line 6) with replacement to distribute X_t according to the desired posterior $b(x_t)$. Frequent resampling is necessary to prevent the weights from growing unbounded and degenerating over time [17].

Particle Starvation During Contact. The particle filter, as described above, is agnostic to the observation model and has been applied to a variety of application domains [10, 19]. However, contact sensors are unique because they operate in two discrete states: contact and no contact. When $z \in Z_c$, the belief state has a singular component that is concentrated on the lower-dimensional observable contact manifold. Conversely, when $z \in Z_{nc}$, $p(z_t|x_t, u_t)$ is uniform over free space and provides little useful information. This property makes contact sensors fundamentally different than cameras and depth sensors, which have relatively smooth observation models.

In practice, particle filters are updated in discrete steps. The execution of an action concentrates any states that penetrate the hand onto the contact manifold. As a result, the hand's contact sensors gain full dimensionality and the CPF is not completely ineffective at estimating the state. However, the CPF requires a large number of particles to increase the probability that some fall into the small swept volume of each sensor [6]. As a result, the CPF suffers from *particle starvation* during periods of contact: there are often no particles in the vicinity of the true state.

Figure 3-Top shows an effect that particle starvation has on the post-contact performance of the CPF. The conventional particle filter correctly filters the belief state before contact in (a–b). However, after contact occurs, $b(x_t)$ becomes singular and importance sampling fails. As a result, the CPF converges to an erroneous belief that the box rolling off of the finger tip instead of settling into the palm.

Surprisingly, this effect causes the particle filter to *perform worse as sensor resolution or the update rate increases* [6]. As sensor resolution increases, the swept volume of each sensor becomes narrower. As the update rate increases, the distance traveled by the hand between updates decreases, and the swept volume becomes shorter. As a result, the CPF requires a large number of particles to successfully track the state.

We have shown that the conventional particle filter is poorly suited for the contact manipulation problem because the state evolves on a lower-dimensional manifold.

4 Manifold Particle Filter

Suppose the state space X is partitioned into m disjoint components $M = \{M_i\}_{i=1}^{m}$, where $M_1, \dots, M_{m-1} \subseteq X$ are manifolds and $M_m = X \setminus \cup_{i=1}^{m-1} M_i$ is the remaining free space. The belief state $b(x)$ may have a singular component with non-zero probability concentrated on the lower-dimensional manifolds $\{M_i\}_{i=1}^{m-1}$.

We redefine the belief state as the weighted sum

$$b(x_t) = \sum_{M_i \in M} b(x_t|M_i) \Pr(x_t \in M_i) \tag{3}$$

over manifolds, where $b(x_t|M_i)$ is the belief over M_i given that $x_t \in M_i$.[1]

The manifold particle filter (MPF), summarized in Algorithm 2, also represents the belief using particles. For each particle, we first choose which manifold to sample from according to $M_i \sim \Pr(x_t \in M_i)$. Then, we sample the particle $\bar{x}_t^{[i]} \sim b(x_t|M_i)$ from the corresponding conditional belief using a sampling technique that is appropriate for the structure of M_i.

Ideally, we would compute $\Pr(x_t \in M_i)$ by marginalizing over M_i. Unfortunately, this is fundamentally impossible for two reasons. First, marginalizing requires knowledge of $b(x_t)$, precisely the distribution that we are trying to estimate. Second, $\int_{M_i} b(x_t)\, dx_t = 0$ because M_i is a measure zero set.

Instead, we approximate $\Pr(x_t \in M_i)$ using only the most recent observation

$$\Pr(x_t \in M_i) \approx \frac{p(z_t|M_i, u_t)}{p(z_t|u_t)} \tag{4}$$

[1]From this point forward we will use $b(x_t)$ as shorthand for the weighted sum in Eq. 3.

where $p(z_t|M_i, u_t)$ is the probability that z_t was generated by a an $x_i \in M_i$ and $p(z_t|u_t) = \int_X p(z_t|x_t, u_t)\,dx_t$ is the prior probability of receiving observation z_t. Equation (4) is a good approximation in the case where $p(z_t|x_t, u_t)$ accurately discriminates between the manifolds.

Finally, we sample a particle $\bar{x}_t^{[i]}$ according to the belief distribution over the chosen manifold $b(x_t|M_i)$. Our key insight is that we can apply a different sampling technique for each M_i that is specifically designed to take advantage of the structure of the manifold. For the manifolds $\{M_i\}_{i=1}^{m-1}$, we sample from the dual proposal distribution [16] as described below. In the case of the free space M_m, we sample $\bar{x}_t^{[i]}$ with the conventional technique and reject any $\bar{x}_t^{[i]} \in \cup_{i=1}^{m-1} M_i$. This rejection sampling step is necessary to avoid biasing the estimate of $b(x_t)$ towards the manifolds.

Dual Proposal Distribution. Importance sampling from the conventional proposal distribution fails on M_i for $i < m$ because they are lower-dimensional manifolds. In this case, we will sample from the *dual proposal distribution* [16]

$$\bar{x}_t^{[i]} \sim \eta \frac{p(z_t|M_i, x_t, u_t)}{p(z_t|M_i, u_t)}, \tag{5}$$

where η is a normalization constant. We can find the corresponding importance weights

$$\bar{w}_t^{[i]} = \int_{M_i} p(\bar{x}_t^{[i]}|x_{t-1}, u_t) b(x_{t-1}|M_i) dx_{t-1}. \tag{6}$$

by dividing the target distribution (Eq. 2) by the proposal distribution (Eq. 5).

The conventional proposal distribution forward-predicts using the motion model and computes importance weights using the observation model. Conversely, the dual proposal distribution samples particles from the observation model and weights them by how well they agree with the motion model [16].

Mixture Proposal Distribution. Just as how the conventional proposal distribution performs poorly with accurate sensors, the dual proposal distribution performs poorly when there is observation noise [16]. The MPF uses the dual proposal distribution to sample from the manifolds and, as a result, shares the same weakness.

We use a *mixture proposal distribution* [16] to mitigate this effect by combining both sampling techniques. Instead of sampling all of the particles from the MPF, we sample n particles from the CPF and d particles from the MPF. We then combine the two sets of particles with the weighted sum $(1 - \phi)X_t + \phi\bar{X}_t$ before resampling. The *mixing rate* $0 \leq \phi \leq 1$ is a parameter that allows the algorithm to smoothly transition from the CPF ($\phi = 0$) to the MPF ($\phi = 1$).

Intuitively, $d = |\bar{X}_t|$ is the number of particles necessary to simultaneously cover all of the manifolds and $n = |X_t|$ is the number of additional particles necessary to represent $b(x_t)$ in free space.

5 Manifold Particle Filter for Contact Manipulation

In this section, we will apply the MPF to the state estimation for contact manipulation problem. To do so, we will define the observable contact manifold X_o and free space X_{free} as the relevant subsets of X. We also describe a technique for computing the importance weights $w_i^{[i]}$ using kernel density estimation [13].

Figure 3 shows the performance of the MPF relative to the CPF. Before contact (a–b), $\Pr(x_t \in X_o) \approx 0$ and both filters update using the conventional proposal distribution. After contact (c–d), $\Pr(x_t \in X_o) \approx 1$ and the manifold particle filter begins sampling from X_o. Sampling from the observable contact manifold allows the MPF to accurately track the object's pose during persistent contact.

Importance Sampling from the Contact Manifold. We must weight the samples drawn from the dual proposal distribution with their corresponding importance weights $\bar{w}^{[i]} = \int_X p(x_t|x_{t-1}, u_t)b(x_{t-1})dx_{t-1}$. Intuitively, this integrates our belief state $b(x_{t-1})$ prior to taking action u_t into $b(x_t)$ [16].

We evaluate $\bar{w}^{[i]}$ by forward-simulating the previous set of particles X_{t-1} to time t by sampling from $p(x_t|x_{t-1}, u_t)$, then evaluating the density of the distribution at $\bar{x}_t^{[i]}$ using a density estimation technique [13]. Ideally, we would compute a density estimate over the manifold X_o. Unfortunately, while there has been some work on density estimation on Riemannian manifolds [12], it is difficult to apply these algo-

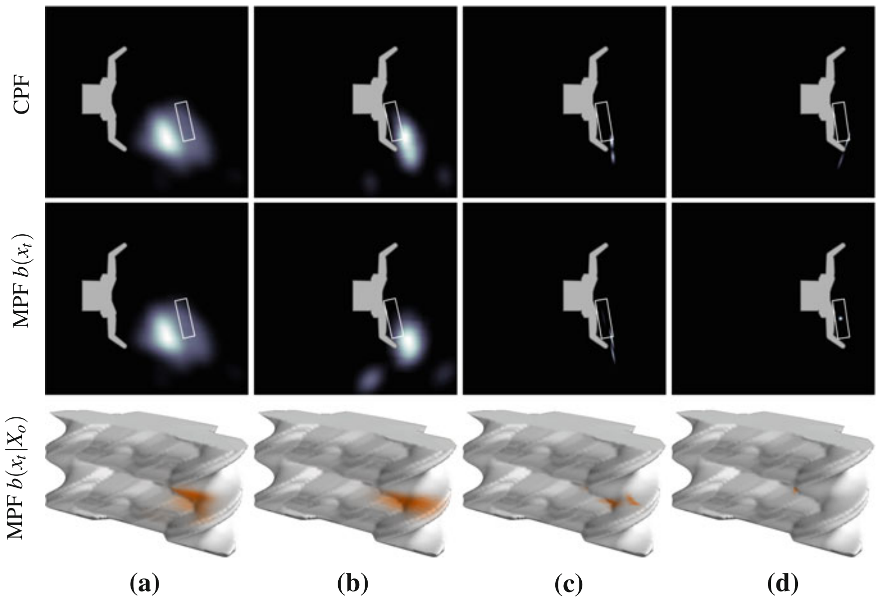


Fig. 3 Snapshots of the CPF and MPF during execution. Unlike the CPF, the MPF avoids particle starvation by explicitly tracking the probability distribution on the observable contact manifold X_o. **a** Prior belief. **b** Pre-contact. **c** Post-contact. **d** Final belief

rithms to the approximate and sample-based representations of X_o described below. This is exacerbated by the fact that many of our forward-simulated particles will not lie on X_o.

Instead, we use kernel density estimation [13] to approximate the probability density over X, then restrict the estimate to $X_o \subset X$. Figure 3 shows an example of the resulting density estimate over X_{free} (Fig. 3-Middle) and X_o (Fig. 3-Bottom) computed using Gaussian kernels with bandwidths selected by Silverman's rule of thumb [14].

6 Representing the Contact Manifold

Implementing the manifold particle filter requires sampling particles from X_o with probability proportional to their observation likelihood $x_t^{[i]} \sim \eta p(z_t|x_t, u_t)/p(z_t|u_t)$. Sampling from this distribution requires maintaining a representation of the observable contact manifold X_o.

We will discuss three possible representations of the contact manifold. Two of these, the rejection sampling (Fig. 4a) and trajectory rollout (Fig. 4b) representations, approximate the continuous manifold X_o with large set of discrete samples. The third technique (Fig. 4c) takes advantage of additional structure in geometry of the problem to solve for an analytic representation of X_o.

Rejection Sampling. The most straightforward way of sampling from $X_o \subset X$ is through rejection sampling in the ambient space X. *Rejection sampling* iteratively samples candidate states $x^{[i]} \sim \text{uniform}(X)$ until it finds a sample $x^{[i]} \in X_o$ in the desired set. Using this technique, we can generate a large set of samples $\tilde{X}_o = \{x^{[i]}\}_{i=1}^n \subset X_o$ that densely cover X_o in a pre-computation step. At runtime, we importance sample from the discrete set $\tilde{X}_o$ with $w^{[i]} = p(z_t|x_t, u_t)/p(z_t|u_t)$ as importance weights.

Unfortunately, rejection sampling fails for the same reason as the conventional particle filter: X_o is a measure-zero set and there is zero probability of successfully sampling an $x^{[i]} \in X_o$ [6]. Instead, we rejection sample from the set

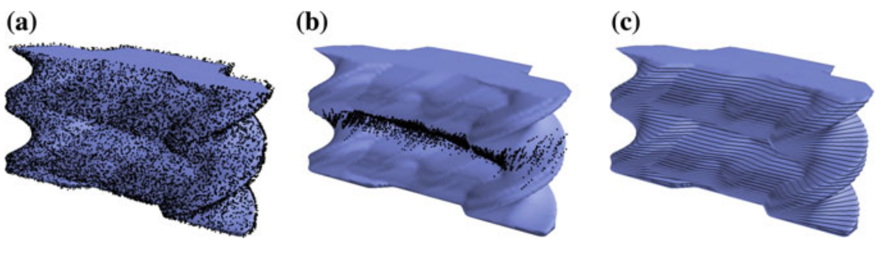


Fig. 4 Several approximate representations of the contact manifold. Representations **a** and **b** approximate X_o with discrete sets of samples. Representation **c** computes an approximate, analytic representation X_o

$$\tilde{X} = \left\{x \in X : \min_{p_s \in P_s, p_o \in P_o(x)} ||p_s - p_o|| \leq \varepsilon\right\}$$

of object configurations that are within distance $\varepsilon \in \mathbb{R}^+$ of the hand. The set $\tilde{X}_o$ is a reasonable approximation for X_o when ε is on the same order of magnitude as the numerical inaccuracies of the motion and observation models (e.g. simulation step size).

Figure 4a shows X_o covered by a set of 10,000 rejection-sampled configurations $\tilde{X}_o$ of the BarrettHand in contact with the rectangular box shown in Fig. 2. The samples $\tilde{X}_o$ are not exactly on X_o and are distributed uniformly over the ambient space X. This is, in most cases, an acceptable approximation for a true uniform distribution over X_o.

Trajectory Rollouts. Rejection sampling attempts to densely cover all of X_o with samples $\tilde{X}_o$ that are independent of the prior belief $b(x_0)$. As a result, many of the samples generated by rejection sampling will be found in regions of X_o that remain low probability during the entire duration of execution. We can exploit this structure by concentrating more samples in the regions of X_o that we are likely to encounter during execution.

We can generate samples $\tilde{X}_o$ that are biased towards these regions by performing trajectory rollouts for a set of sampled beliefs. We begin by sampling a particle from the prior $x_0^{[i]} \sim b(x_0)$. Next, we forward-simulate the particle for T steps using the motion model $x_t^{[i]} \sim p(x_t|x_{t-1}, u_t)$ with u_t chosen according to our policy.[2] Finally, we add any $x_t^{[i]} \in X_o$ to $\tilde{X}_o$. This process repeats until $|\tilde{X}_o|$ reaches the desired size.

Figure 4b shows 10,000 samples taken from 3000 trajectory rollouts with a fixed "move straight" action and $b(x_0)$ roughly centered in front of the hand. The trajectory rollout technique achieves dense coverage of the reachable area of the state space—which consists of the front of the hand with orientations consistent with $b(x_0)$—at the cost of sparse coverage of the rest of the manifold.

Unfortunately, the non-uniformity of our samples means that $\tilde{X}_o$ is biased towards absorbing regions of the state space. We compensate for this bias through importance sampling: we assign each $x^{[i]} \in \tilde{X}_o$ an importance weight $w^{[i]} = p(z|x, u)/[p(z|u)\, \tilde{p}(x)]$ where $\tilde{p}(x)$ is the density of $\tilde{X}_o$ at x. We estimate $\tilde{p}(x)$ using a standard kernel density estimation technique [13] of $\tilde{X}_o$ and, thus, produce samples that are uniformly distributed over the ambient space X.

Analytic Representation. In some special cases of hand-object geometry we can compute an analytic representation of X_o. This is possible, for example, in the common case where P_h and P_o are polygons in $\mathbb{R}^2$ [8] or polyhedra in $\mathbb{R}^3$ [7].

Without loss of generality, we will consider polygonal objects in $SE(2)$. In this case, we can geometrically compute the C-obstacle $X_{obs}(\theta)$ for a fixed orientation θ of the object as

$$X_{obs}(\theta) = P_h \oplus -P_o\left([0, 0, \theta]\right)$$

[2]If the policy is not known, we sample $u \sim \text{uniform}(U)$.

where $A \oplus B = \{a + b : a \in A, b \in B\}$ denotes the Minkowski sum of sets A and B.

Since P_h and $P_o(\theta)$ are polygonal, $X_{obs}(\theta)$ is also polygonal and can be computed via a convolution of P_h and $P_o(\theta)$ [18]. The contact manifold $X_c(\theta)$ at orientation θ simply consists of the perimeter of the polygon $X_{obs}(\theta)$. Figure 4c shows several θ-isocontours of X_c superimposed over a high-resolution polyhedral approximation of the contact manifold. The same process can be repeated with P_h and P_s to construct an analytic representation of $X_o(\theta)$.

Finally, we approximate the observable contact manifold as a union $\tilde{X}_o = \cup_{\theta \in \Theta} X_o(\theta)$ over a large, discrete set of orientations Θ.[3] Discretizing θ approximates X_o with a polyhedron $\tilde{X}_o$ that shares the same polygonal iso-contours at all $\theta \in \Theta$.

Sampling an $x^{[i]} \sim \tilde{X}_o$ is possible by first sampling a $\theta \in \Theta$, then uniformly sampling an $x^{[i]}$ from our analytical representation of $X_o(\theta)$. Alternatively, one could sample from an approximate, polyhedral representation of $\tilde{X}_o$ by interpolating between iso-contours. In both cases, the samples are correctly drawn uniformly with respect to a measure defined over the lower-dimensional X_o.

7 Experiments and Results

We designed a set of simulation experiments to compare the MPF with the CPF for the state estimation for contact manipulation problem, and to explore the differences between the three representations of the contact manifold.

Based on the particle starvation problem, we hypothesize that

H1. *The MPF will outperform the CPF after contact.*

Among the three representations of the contact manifold, we expect the rejection sampling (RS) representation to perform the worst due to its relatively sparse distribution of samples. The trajectory rollout (TR) representation solves this problem by concentrating samples on the regions of the contact manifold that we are most likely to encounter.

Therefore, we hypothesize:

H2. *Trajectory rollouts will outperform rejection sampling.*
H3. *The analytic contact manifold will outperform rejection sampling.*

However, we hypothesize that the analytic representation will outperform both the RS and TR representations because it exactly represents the contact manifold:

H4. *The analytic contact manifold will perform best.*

[3]Uniformly discretizing θ may miss critical events where the object first comes into or leaves contact with the hand. If these events are important, it is possible to analytically solve for the critical values of θ through careful analysis of the geometry [3].

Experimental Design. We implemented CPF, MPF-AM, MPF-RS, and MPF-TR in a custom two-dimensional kinematic simulation environment with polygonal geometry. Each experiment consisted of a simulated BarrettHand pushing a rectangular box in a straight line at a speed of 1 cm/s for 50 cm. The initial belief state was set to $b(x_0) = \mathscr{N}(0, \Sigma)$ with $\Sigma^{1/2} = \text{diag}[5\,\text{cm}, 5\ \text{cm}, 20°]$.

Motion Model. We simulated the motion of the object using a penetration-based quasistatic physics simulator [9] with a 1 mm step size. During each update, the finger-object coefficient of friction μ and the radius of the object's pressure distribution c were sampled from the Gaussian distributions $\mu \sim N(0.5, 0.2^2)$ and $c \sim N(0.05, 0.01^2)$ truncated at $\mu, c > 0$.

Observation Model. Binary observations were simulated for each of the hand's sensors by computing the intersection of the contact sensor with the object's geometry. Ground-truth observations were simulated by applying the same observation model to a special "ground truth" particle sampled from $b(x_0)$.

Dependent Measure. We measure performance of the estimators by tracking the root mean square error (RMSE) of the object's position (Fig. 5a-Top) and orientation (Fig. 5a-Bottom) over a large number of experiments.

Conventional versus Manifold Particle Filter (H1). Both the CPF and the MPF used 100 particles. The MPF used an analytic representation of the contact manifold and a mixing rate of $\phi = 0.1$.

Figure 5a shows that—as expected—both filters behave similarly before contact ($t \leq 0$) and there not a significant difference in RMSE. After contact ($t > 0$), the MPF quickly achieves 4.4 cm less RMSE than the CPF. These results support hypothesis H1: the MPF achieves lower post-contact error than the CPF.

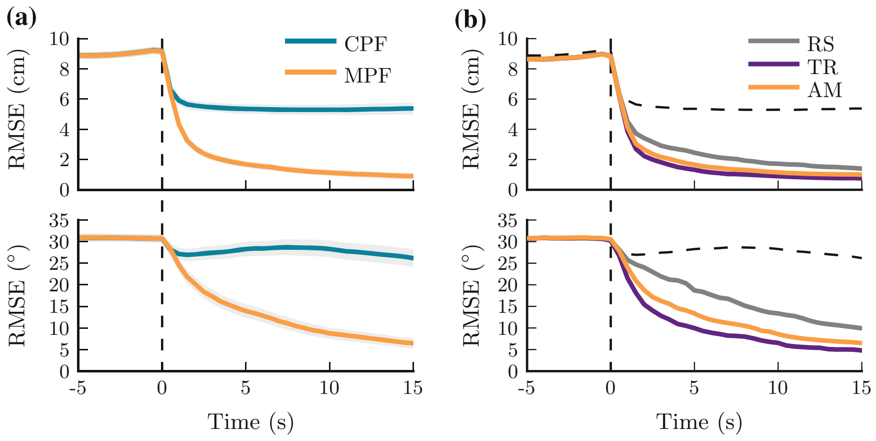


Fig. 5 **a** Estimation error of the CPF and MPF. **b** Performance of MPF using the rejection-sampled (RS), trajectory-rollout (TR), and analytical (AM) manifold representations. In both cases, the data is aligned such that contact occurs at $t = 0$

Contact Manifold Representation (H2–H4). We also compared the RMSE error of the MPF using the rejection sampling (RS), trajectory rollouts (TR), and an analytic (AM) representations of the contact manifold. The RS representation consisted of 10,000 samples that were held constant throughout all of the experiments. The TR representation generated a different set 10,000 samples for each experiment by collecting five samples each from 2000 trajectory rollouts. Finally, the AM representation was implemented by sampling from polygonal iso-contours of X_o spaced every 3° of angular resolution.

Figure 5b shows that all three implementations of the MPF outperformed the CPF. As expected, the AM and TR representations both outperformed the RS representation, supporting hypotheses H2 and H3. This occurs because the RS representation attempts to sparsely cover the entire surface X_o with a relatively small number of samples, while the TR representation densely covers the states that we are most likely to reach.

Surprisingly, hypothesis H4 was not supported by the data: the AM representation did not achieve lower error than the TR representation. This occurred because the TR representation was able to saturate the regions of X_o that we are likely to encounter during execution. By doing so, the TR representation achieves such dense coverage of the relevant parts of that it is unlikely to fail at sampling from the dual proposal distribution.

Sampling Failures. Figure 6 supports our intuition that the relatively poor performance of the RS representation is a result of it frequently failing to sample from the dual proposal distribution. The TR and AM representations fail to sample from the dual proposal distribution for only $<30\,\%$ of updates. Conversely, the RS representation fails to sample $>70\,\%$ of the time. When sampling fails, the MPF behaves identically to the CPF and suffers from the same problem of particle starvation. As a result, the RS representation performs relatively poorly compared to the RS and TR representations in Fig. 5b.

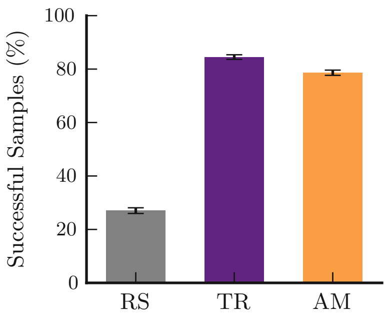


Fig. 6 Percent of the time that the MPF succeeded at sampling from the dual proposal distribution during contact. Sampling fails when all particles sampled from the contact manifold have a low probability $p(z_t|x_t, u_t)$ of generating z_t

8 Discussion and Future Work

In this section, we discuss how partial sensor coverage and different contact manifold representations effect the manifold particle filter. Additionally, we discuss several possible ways of addressing the limitations of the MPF in future work.

Contact Manifold Representations. We discussed several possible implementations of the contact manifold that can be used to sample from the dual proposal distribution: rejection sampling (RS), trajectory rollouts (TR), and an analytic representation (AM). Each of these representations makes different assumptions about the structure of the problem.

The RS and TR representations approximate X_o with a discrete set of pre-computed samples $\tilde{X}_o$. These techniques make no assumptions about the geometry of the problem and widely applicable. Both techniques outperform the CPF, but MPF-TR outperforms MPF-RS. This occurs because the TR representation concentrates $\tilde{X}_o$ in the regions of the state space that we are most likely to see during execution. As a result, sampling from the dual proposal distribution is less likely to fail with TR than RS.

Unlike RS, the set of samples $\tilde{X}_o$ generated by TR are specific to $b(x_0)$ and cannot be generalized between problem instances. Even worse, pre-computing $\tilde{X}_o$ requires rolling out a large number of trajectories using the computationally expensive motion model. In summary, TR trades more pre-computation time for better online performance.

When it exists, an analytic representation of the contact manifold provides an exact representation of X_o. For polygonal geometry in $SE(2)$, the analytic representation requires minimal pre-processing and could possibly be updated in real-time as the geometry of the hand changes. Additionally, it is efficient to uniformly sample states from X_o at runtime. Unlike the sample-based representations, these samples will be distributed uniformly with respect to the measure over X_o instead of the underlying space X. Finally, there is no chance of failing to sample from the dual proposal distribution due to a sparsity of samples.

The Observability of Contact. Contact sensors frequently do not cover the entire surface of a hand. For example, the proximal links of the BarrettHand are not covered with tactile sensors and the SynTouch BioTac [4] sensor only provides tactile sensing on the interior of the fingertip. Even the iHY hand [11], which tightly integrates TakkTile sensors [15] into its mechanical design, does not cover the outside surface of the hand with sensors. As a result, it is important to consider the effect that observability of contact has on our state estimation ability.

The difference between "contact" and "observed contact" is captured in our definitions of the contact manifold X_c and the observable contact manifold $X_o \subseteq X_c$. The geometry of the non-observable region of the contact manifold $X_{no} = X_c \backslash X_o$ impacts the difficulty of the state estimation for contact manipulation problem. Ideally, the transition model will quickly move states out of X_{no} into X_o by pushing them into contact with a sensor. Any stable states in X_{no}, e.g. those that come to rest against a flat surface, will accumulate belief during execution.

Contact with Multiple Objects. We implicitly assume that the hand can only contact the object that we are manipulating. This may not be possible in highly cluttered environments where we must contact multiple objects to achieve the desired task [1]. In future work, we hope to explore methods of generalizing the MPF to environments with multiple—both static and movable—objects. We believe it is possible to do so through limited factoring of the belief state (e.g. through Rao-Blackwellization) to avoid requiring exponentially more particles.

Shape Uncertainty. We assume that the hand and object both have known geometry. This is often not true when using compliant/under-actuated hands (e.g. the iHY hand [11]) or manipulating un-modeled objects. Small variations of the object-hand geometry can cause large changes in the shape and topology of the contact manifold. We hope to address this additional source of uncertainty in future work by considering distributions over object and hand geometry. This would in effect, create a "fuzzy" contact manifold that consists of the union of several hypothesized contact manifolds.

Acknowledgments This work was supported by a NASA Space Technology Research Fellowship and the DARPA Autonomous Robotic Manipulation Software Track (ARM-S) program. We would also like to thank Mehmet Dogar, Anca Dragan, and the members of the Personal Robotics Lab for their helpful input.

References

1. Dogar, M., Hsiao, K., Ciocarlie, M., Srinivasa, S.S.: Physics-based Grasp planning through clutter. In: RSS (2012)
2. Dogar, M.R., Srinivasa, S.S.: Push-grasping with dexterous hands: mechanics and a method. In: IEEE/RSJ IROS (2010)
3. Farahat, A.O., Stiller, P.F., Trinkle, J.C.: On the geometry of contact formation cells for systems of polygons. In: IEEE T-RO (1995)
4. Fishel, J.A., Loeb, G.E.: Sensing tactile microvibrations with the biotac comparison with human sensitivity. In: IEEE/RAS-EMBS BioRob (2012)
5. Jia, Y., Erdmann, M.: Pose and motion from contact. In: IJRR (1999)
6. Koval, M.C., Dogar, M.R., Pollard, N.S., Srinivasa, S.S.: Pose estimation for contact manipulation with manifold particle filters. In: IEEE/RSJ IROS (2013)
7. LaValle, V.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
8. Lozano-Pèrez, T.: Spatial Planning: a configuration space approach. In: IEEE T-C (1983)
9. Lynch, K.M.,Maekawa, H., Tanie, K.: Manipulation and active sensing by pushing using tactile feedback. In: IEEE/RSJ IROS (1992)
10. Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: IJCAI (2003)
11. Odhner, L., Jentoft, L.P., Claffee, M.R., Corson, N., Tenzer, Y., Ma, R.R., Buehler, M., Kohout, R., Howe, R.D., Dollar, A.M.: A compliant, underactuated hand for robust manipulation. In: CoRR (2013)
12. Pelletier, B.: Kernel density estimation on Riemannian manifolds. Stat. Prob. Lett. (2005)
13. Rosenblatt, M., et al.: Remarks on some nonparametric estimates of a density function. Ann. Math. Stat. **27**, 832–837 (1956)
14. Silverman, B.W.: Using kernel density estimates to investigate multimodality. J. R. Stat. Soc. **43**, 97–99 (1981)

15. Tenzer, Y., Jentoft, L.P., Howe, R.D.: Inexpensive and Easily customized tactile array sensors using MEMS barometers chips. In: IEEE (2012)
16. Thrun, S., Fox, D., Burgard, W.: Monte Carlo localization with mixture proposal distribution. In: AAAI (2000)
17. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press, Cambridge (2005)
18. Wein, R.: 2D Minkowski Sums. In: CGAL User and Reference Manual. CGAL Editorial Board, 4.3 edition (2013)
19. Zhang, L., Trinkle, J.C.: The application of particle filtering to grasping acquisition with visual occlusion and tactile sensing. In: IEEE ICRA (2012)
20. Zhang, L., Lyu, S., Trinkle, J.: A dynamic bayesian approach to simultaneous estimation and filtering in grasp acquisition. In: IEEE ICRA (2013)

Exploitation of Environmental Constraints in Human and Robotic Grasping

Raphael Deimel, Clemens Eppner, José Álvarez-Ruiz, Marianne Maertens and Oliver Brock

Abstract We investigate the premise that robust grasping performance is enabled by exploiting constraints present in the environment. These constraints, leveraged through motion in contact, counteract uncertainty in state variables relevant to grasp success. Given this premise, grasping becomes a process of successive exploitation of environmental constraints, until a successful grasp has been established. We present support for this view by analyzing human grasp behavior and by showing robust robotic grasping based on constraint-exploiting grasp strategies. Furthermore, we show that it is possible to design robotic hands with inherent capabilities for the exploitation of environmental constraints.

1 Introduction

Humans are excellent graspers. Despite decades of research on robotic grasping, we have yet to establish the same level of competency in robotic systems. What lets humans grasp so well? There are many answers to this question, most are associated with active research areas in robotics. In this paper, we explore the hypothesis that human grasp performance is to a significant extent the result of carefully orchestrated interactions between the hand and the environment. We investigate how this hypothesis may impact the development of versatile robotic grasping systems.

R. Deimel (✉) · C. Eppner · J. Álvarez-Ruiz · O. Brock
Robotics and Biology Laboratory, Technische Universität Berlin, Berlin, Germany
e-mail: raphael.deimel@tu-berlin.de

C. Eppner
e-mail: clemens.eppner@tu-berlin.de

J. Álvarez-Ruiz
e-mail: jose.a.alvarezruiz@tu-berlin.de

O. Brock
e-mail: oliver.brock@tu-berlin.de

M. Maertens
Modelling of Cognitive Processes Group, Technische Universität Berlin,
Berlin, Germany
e-mail: marianne.maertens@tu-berlin.de

The premise of the work reported in this paper is the following: *A competent grasper must exploit constraints present in the environment by employing physical contact so as to counteract uncertainty in state variables most relevant to grasp success.* If this premise is true, robust and versatile grasping is the process of determining sequences of motions to leverage these constraints in the most effective manner.

Video recordings of human grasping provide immediate support for our premise. Human grasps routinely involve contact with the environment. However, we are not aware of systematic studies on the use and purpose of such contacts in the psychology literature. In this paper, we begin to close this gap by reporting on novel human grasping experiments. We establish a set of parameters to characterize contact with the support surface during grasping. And we present human grasping experiments to show that the use of environmental constraints increases with the human's uncertainty about the environment.

Ongoing research on robotic grasping provides further support for our premise. Novel gripper and hand designs often include compliant materials or actuators. In our view, this does not only lead to more robust interactions between hand and the grasped object, it also facilitates the exploitation of environmental constraints. There are several studies of novel hands, reviewed in the next section, that deliberately exploit environmental constraints in specific application scenarios or for specific grasps. Research in grasp planning also has begun to leverage environmental constraints, however, either to a limited extent or in specifically tailored approaches. Beyond these instances, to the best of our knowledge, there is no comprehensive approach for the generic, orchestrated use of environmental constraints in robotic grasping.

In the remainder of this paper, we outline the beginnings of an integrated research agenda towards robotic grasping by leveraging environmental constraints. This agenda spans the study of human grasping, the development of appropriate grasp strategies, the required perceptual strategies to determine when each of the strategies is most appropriate, and the design of robotic hands tailored for the exploitation of environmental constraints.

2 Background

To support our claim, we divide related work into three categories based on the types of interactions they consider. The first category, which marks the beginnings of grasping research in robotics, analyzes static grasps. The second category exploits interactions between hand and object. The final and most recent category exploits interactions between hand, object, and environment, enabling the consideration of environmental constraints for robust grasping.

2.1 Force and Form Closure

Early grasping research emphasizes the concepts of force and form closure, reflecting a static grasping relationship between hand and object. A grasp is commonly expressed as a set of disembodied point contacts. Physical interactions occurring during the grasp—and sometimes even the limitations that result from the kinematics of the hand—are often not accounted for during grasp planning. These approaches require detailed models of both the environment and the hand.

This line of research continues to be active and successful, as evidenced by a large number of sophisticated and capable grasp planners and simulators [21]. In our experience, however, the grasps determined by these approaches do not reliably transfer to the real world. To provide adequate hardware for the precise placement of specific contact points on objects, researchers designed mechanically complex, rigid hands with many degrees of freedom [12].

Early studies of human grasping also followed this static view. This is reflected in grasp taxonomies, classifying grasp according to the final hand posture attained after the grasp process is completed [5, 11]. Even the early work on postural synergies, which has had a profound impact on robotics, initially only considered synergies of static grasp postures [24]. These studies do not capture the dynamic processes and the exploitation of environmental constraints we believe to be crucial for robust grasping.

2.2 Interactions Between Hand and Object

During grasp execution, mechanical compliance in the hand leads to an adaptation of the hand's configuration to the object's shape. This shape adaptation aids grasp success by compensating for uncertainties in actuation and world model, and by attaining a large number of contact points. Thereby shape adaptation significantly increases the chances of achieving force closure with a grasp. Much of the recent work in robotic grasping attempts to leverage this effect explicitly.

Rodriguez et al. [23] optimized the shape of non-compliant fingers to yield the same contact point configuration irrespective of object size. Shape adaptability can be enhanced by adding compliant parts [14]. The positive pressure gripper [1] represents an extreme case in this regard. It uses granular material to achieve compliance of the entire gripper to large parts of the object's geometry.

An effective way of achieving shape adaptability is underactuation. The SDM hand [8], the Velo gripper [4], the i-HY hand [22], and hands by Grioli et al. [13] couple the actuation of degrees of freedom using tendon-pulley systems, adapting the shape of the hand to the object while equalizing contact forces.

The nature of hand-object interaction has also been studied in humans. The effect of shape adaptability is well known for human hands and studies have elucidated the degree to which humans vary their behavior to take advantage of it. Christopoulos et al. [3] showed that humans react to pose uncertainty of a cylinder and orient their hand to align it, presumably to be able to maximize the benefits of shape adaptability. However, other experiments may point at the fact that humans also rely on more complex interactions with the environment for grasping under difficult conditions. When the vision of humans is impaired, they fail more often at first grasp attempts of isolated (environmental-constraint free) objects [20]. The degree of the demonstrated effect seems surprising. We believe that in this specific experiment, it is due to the lack of environmental constraints exploitable for grasping.

2.3 Interactions Between Hand, Object, and Environment

Features in the environment may constrain the motion of hand and object. This is most evident for support surfaces, such as tables and floors. These constraints, when used properly, can aid grasping. Furthermore, we postulate that the necessary perceptual information for leveraging such constraints is often easier to obtain than the information required for the successful execution of an unconstrained grasp.

Recent research in robotic grasping leverages environmental constraints in the suggested manner, e.g. to position the hand relative to the object [6], to cage objects [6, 17], or to fixate an object during planar sliding [6, 7]. Some pre-grasp manipulation relies on environmental constraints to improve grasp success. For example, Chang et al. [2] rotate pan handles into a specific orientation prior to grasping by exploiting the pan's friction and remote center of mass.

All of the aforementioned grasp strategies rely on multiple interactions prior to establishing force closure for the final grasp. These phases often are designed to reduce uncertainties in specific variables relevant to grasp success.

The idea of environmental constraints appears in early work by Lozano-Pérez, Mason, and Taylor [18, 19]. Here, the intrinsic mechanics of the task environment are exploited to eliminate uncertainty and to achieve robustness.

The study of environmental constraint exploitation by humans so far has been limited to replicating instances of observed behavior on robots [2, 15]. For example, Kaneko et al. [15] extracted a set of grasping strategies from observations of a human subject. These strategies include interactions with environmental constraints.

We believe that this recent trend towards the exploitation of environmental constraints represents an important opportunity to improve robotic grasping capabilities. To take full advantage of this opportunity, we should understand the strategies humans employ, transfer them to robotic systems, and develop robotic hands tailored to this exploitation.

3 Environmental Constraints in Human Grasping

In this paper we argue that competent grasping exploits constraints in the environment. In this section, we describe our work towards the identification of successful strategies for the exploitation of environmental constraints in human grasping. In a first step, we define operational measures to quantitatively characterize the exploitation of a specific environmental constraint, namely the support surface of a grasped object. We also show that the interaction with the support surface becomes more pronounced when grasping is made more difficult by impairing human vision. We view this finding as support for our main premise.

3.1 Quantifying Contact Interactions with Support Surfaces

We choose the following parameters to quantify the contact interaction with the support surface during a grasping trial: the number of distinct support contacts, N, the mean travel distance of all support contacts, $\overline{d}$ (spatial extent of contact interaction), the mean duration of all support contacts, $\overline{\Delta t_c}$ (temporal extent), and the maximum force exerted orthogonal to the support surface, f_{max} (energetic extent). Additionally, we measure the grasp duration $\overline{\Delta t_g}$, i.e. the time elapsed between the first contact with either the object or the support surface and object lift. We will show that these parameters serve as a meaningful characterization of the interaction with the support surface.

3.2 Experiment

Five right handed adults (aged 20–25 years, two females) participated in the experiment. They were naive to the rationale behind the experimental design. All participants reported normal or corrected-to-normal vision.

A grasp trial began with the participant's hand extended and resting at a start position, see Fig. 1. An object was placed at a fixed location on top of a tablet computer located behind an occlusion panel blocking the participant's view. Then, the occlusion panel was removed and the participant was able to observe the scene. After a delay of 3 s, the participant received an auditory signal to grasp the object. During the grasp, the tablet computer recorded N, $\overline{\Delta t_c}$, and $\overline{d}$, see Fig. 2. The trial ended with another auditory signal, in response to which the participant released the object and returned the hand to the start position. A force/torque sensor was attached to the tablet to determine contact with the object or the support surface, and to measure f_{max}. Finally, a camera was used to detect object lift.

The experiment was performed under two conditions: control and impaired. In the control condition, human vision was not altered. In the impaired condition, partic-

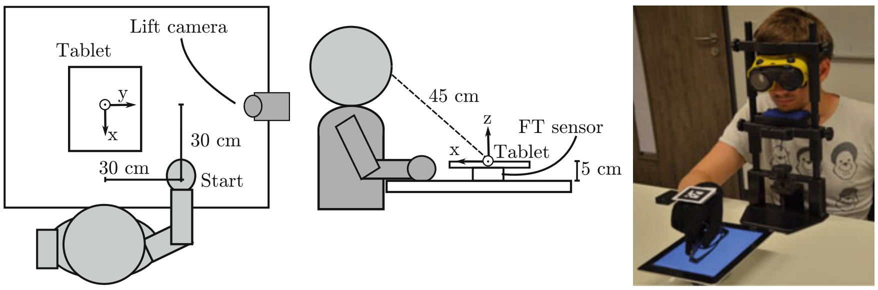


Fig. 1 Experimental setup

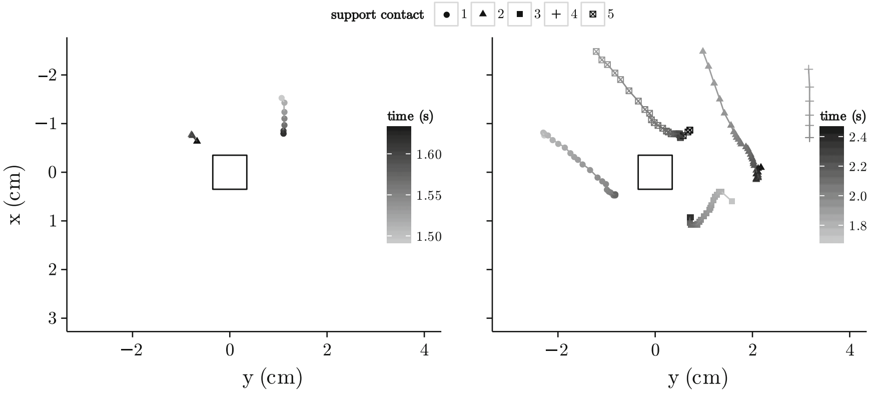


Fig. 2 Examples of the measured support contacts when grasping a screw (participant FG). *Left* control condition, *right* impaired condition. The participants were located in the positive direction of the *x* axis, see Fig. 1. The *black rectangle* indicates the object's initial position and size. Support contacts occurred within a time interval of ≈0.13 s and ≈0.78 s in the control and impaired conditions, respectively

ipants wore custom goggles that blurred details of the objects' shape and degraded depth perception, see Fig. 3. It is difficult to quantify the effects of the goggles, but they allowed to induce a consistent and severe reduction in human vision. The impaired condition trials preceded the control trials to prevent participants from observing the details of the object shapes. Each participant performed 100 trials: ten objects, five repetitions per object, each under two conditions.

We used the following objects: a button, a salt shaker, a roll of adhesive tape, a match box, a marker pen, sunglasses, a comb, a plastic screw, a toy, and a chestnut. All objects were painted black to remove color cues for distinguishing them and to homogenize the contrast with the surroundings (see Fig. 3). The participants wore a conductive glove to improve the reliability of the touchscreen measurements. Participants were seated as shown in Fig. 1, with their head supported by a chin and forehead rest. The viewing distance to the center of the tablet was 45 cm.

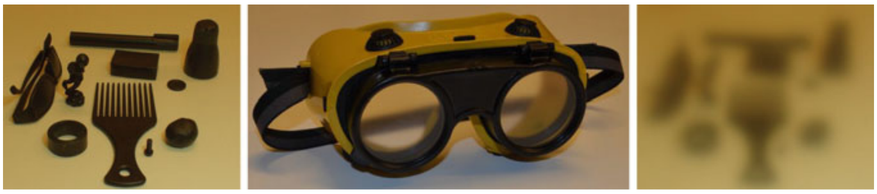


Fig. 3 From *left* to *right* the target objects, the blurring goggles, and the target objects as seen through the goggles

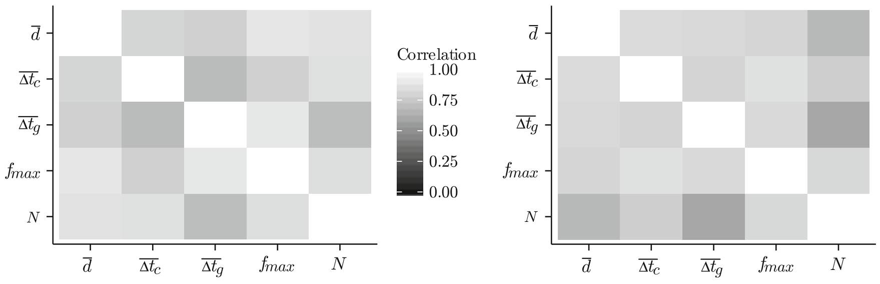


Fig. 4 Pearson's correlation coefficients averaged over all subjects; *left* control condition; *right* impaired condition

3.3 Results

The parameters N, $\overline{\Delta t_c}$, $\overline{d}$, f_{max}, and $\overline{\Delta t_g}$ measured in the five trials for an object were averaged for each participant. We performed two kinds of analyses: a correlation analysis and a series of tests for possible effects induced by the impairment.

The correlation analysis was performed based on the following reasoning: given that the chosen parameters are supposed to quantify different aspects of the same phenomenon, their inter-correlations should be high. Figure 4 depicts the Pearson correlation coefficients in a cross correlation matrix for all five parameters in the control and the impaired condition. There were differences between the two conditions. Further analysis suggests that those differences were due to different variances of the parameters between both conditions.

In the second part of our analysis, we tested if the magnitude of the chosen parameters was significantly larger in the impaired condition. A one-tailed paired t-test (Holm-Bonferroni corrected, and global $\alpha = 0.05$) revealed significant effects for all parameters and for all participants.

3.4 Discussion

The results of our study support two conclusions. First, the proposed parameters are meaningful for the characterization of the interaction with the support surface, as they exhibited high inter-correlations. Second, humans increase the interaction with the support surface when their vision is experimentally impaired as indicated by significant differences in the measured parameters between the two conditions. This is consistent with the main premise of this paper, i.e. that robust grasping should exploit environmental constraints to compensate for uncertainty. In our experiments, the visual impairment results in an increase in the number of support contacts, and an increase in the duration of support contacts and in their travel distance. Furthermore, the participants apply higher contact forces in the impaired condition. The grasping time in the impaired condition also increases significantly. Traditionally, in the study of human grasping, increased grasping time is interpreted as increased use of tactile feedback [10, 20]. We indeed noticed that in some trials, in particular under the impaired condition, the participants established and maintained contact with the support surface while still reaching for the object. This use of a environmental constraints, while not part of this study, can also be viewed as support for our premise.

This initial study is a first step towards a more detailed analysis of human strategies for exploiting environmental constraints during grasping. Our goal in this line of research will be to identify successful exploitation strategies and to characterize the conditions for which they are successful. We hope to transfer these insights to robots so as to endow them with improved grasping capabilities.

4 Exploitation of Environmental Constraints by Robots

In the previous section, we concluded that humans increase their use of an environmental constraint in response to perceptual uncertainty. In this section, we investigate how robots can exploit such constraints. Exploitation can be performed by devising appropriate grasping strategies or by designing hands tailored for this purpose. We report on our work in both of these areas.

4.1 Grasp Strategies

Our goal is to design grasp strategies that exploit environmental constraints to increase grasp success and to show that there are a variety of environmental constraints that can be leveraged by those strategies.

4.1.1 Exploiting Environmental Constraints Increases Grasp Success

We compare two grasp strategies that leverage the same environmental constraint to a different degree. The environmental constraint in this experiment is provided by the supporting table surface. As the height of objects decreases, grasping becomes more difficult. We expect grasp success to be higher if the constraint provided by the table surface to guide finger placement on the object is exploited to a higher degree.

Constant wrist pose: The first strategy was introduced in our prior work [9]. Grasp poses are generated by fitting geometric primitives like cylinders, spheres, and boxes to depth measurements of the scene. To increase the likelihood of grasp success, pre-grasp poses are refined in response to environmental constraints. For this strategy, the palm of the hand is aligned with the support surface. The hand is then positioned as low as possible above the support surface so that the fingers do not contact the surface during closing. This strategy uses the environmental constraint provided by the support surface to position the hand but does not exploit contact interactions.

Force-compliant closing: The second strategy uses force control to establish contact of the fingertips with the support surface and proceeds to slide the fingers along the surface during closing, maintaining constant contact force by compliantly repositioning the wrist (see Fig. 5). Kazemi et al. [17] present a similar strategy; while they control orientation based on force feedback, we employ visual feedback.

The main difference between the two compared strategies is that the first only attempts to come as close as possible to the surface using RGB-D information about the scene, whereas the second maintains physical contact with the surface throughout the whole grasp. The same environmental constraint—the table surface—is exploited visually in one and haptically in the other.

To evaluate the strategies we placed different sized cylinders (see Fig. 6a) on a table in front of a 7-DOF WAM equipped with a force-torque sensor and a Barrett Hand BH-262. All experiments reported in this section are averaged over five trials.

Figure 8 shows grasp success as a function of cylinder diameter. While big cylinders could be grasped reliably with both strategies, the grasp of smaller cylinders only succeeded with force-based exploitation of the environmental constraint. Strategy 1 causes the finger tips to hover slightly above the surface when contact with the object is made, due to the circular trajectory during hand closure. This insufficient

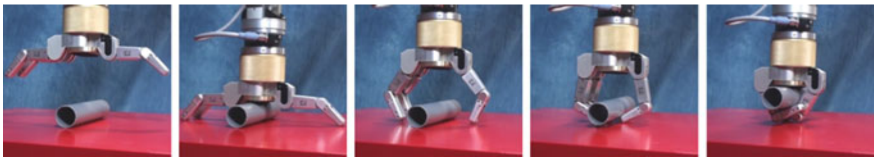


Fig. 5 Force-compliant closing strategy, from *left* to *right* positioning using visual feedback, contacting table surface using force feedback, closing fingers using position feedback while wrist is force-compliant in z-direction (last three images)

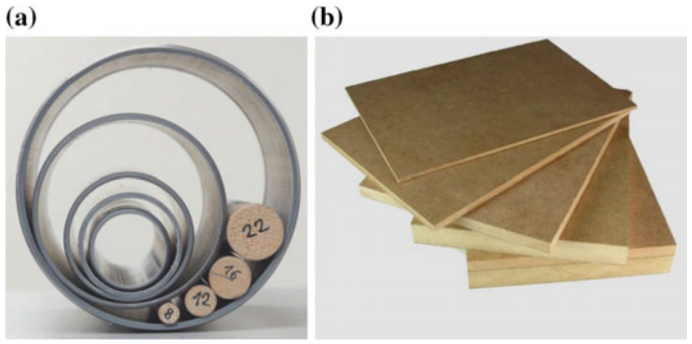


Fig. 6 Objects used in grasping experiments. **a** Cylinders with 8, 12, 16, 22, 32, 40, 50, 75 and 110 mm. **b** Blocks with height 3, 6, 10, 19 and 29 mm and weight 79, 158, 233, 451, and 684 g

exploitation of the surface constraint leads to a reduced success rate for small-sized objects. In contrast, the force-compliant finger closing uses the surface constraint at all times to position fingertips as close to the table as possible. Grasp success is not perfect though, as the cylinders can easily roll off the fingertips. An example of this failure mode is shown in Fig. 9a.

This experiment shows that exploiting a surface constraint to a higher degree leads to more robust grasping.

4.1.2 Exploiting Different Environmental Constraints

We want to show that there are multiple environmental constraints that can be exploited. To achieve good grasping performance in a variety of settings and for diverse objects, it is then necessary to employ the most appropriate strategy. The multitude of available constraints also necessitates perceptual capabilities to distinguish situations in which one strategy should be preferred over the other. To make this point, we implemented the slide-to-edge strategy and compared it to the previously presented force-compliant finger closing.

Slide-to-edge: The slide-to-edge strategy exploits a surface and an edge feature in the environment. It contacts the object using the surface, slides it towards an edge, and wraps the thumb around the protruding part of the object to establish a grasp. The different phases of our slide-to-edge strategy are illustrated in Fig. 7. This strategy can also be seen as a distinct pre-grasp interaction which reconfigures the object enabling contact on parts of it that were previously inaccessible. A similar strategy was presented in [16], focusing on the planning of feasible motions.

We evaluated the slide-to-edge strategy by comparing it to the force-compliant closing strategy for different sized blocks (see Fig. 6b) placed on a table as before. For all blocks, the slide-to-edge strategy achieves reliable performance (see Fig. 8), whereas the force-compliant strategy is only successful for flat blocks.

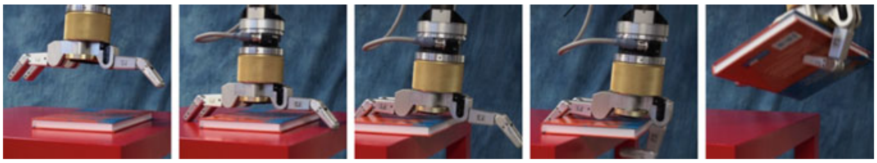


Fig. 7 Slide-to-edge grasp strategy, from *left* to *right* positioning using visual feedback, contact surface using force feedback, moving towards the edge while being force-compliant in the z-direction of the wrist, closing the thumb when the hand is above the edge (detected via visual feedback), and lifting

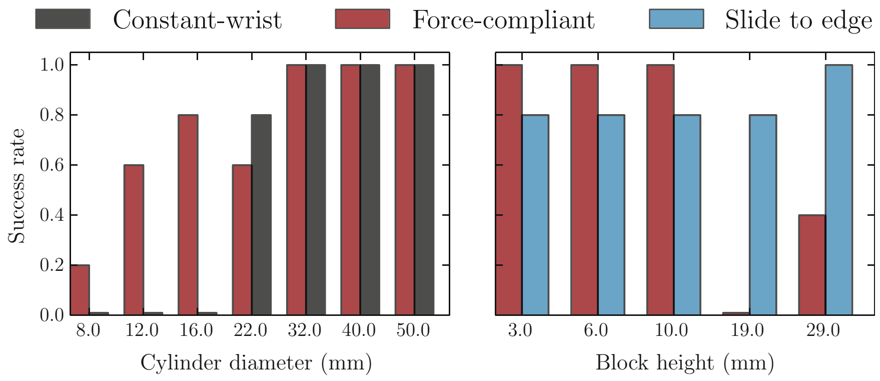


Fig. 8 Comparison of the three grasping strategies

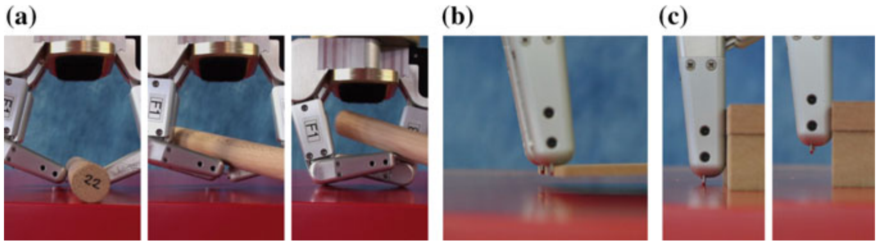


Fig. 9 Exemplary failure and success cases for the force-compliant closing strategy. **a** Failure due to object's inertia. **b** Chance success. **c** Failure due to slip

The slide-to-edge strategy is less sensitive to variation in the size and weight of the blocks. The flat and wide shape of the blocks enables the robot to move parts of them over the edge, creating the opportunity to perform a more reliable grasp on the shorter side of the block. Failure cases for the slide-to-edge strategy included wrong tracking during the visual servoing positioning, missing object contact during sliding, and premature thumb closing.

The force-compliant strategy succeeds when the fingernails jam against one of the block's sharp edges, as can be seen in Fig. 9b. This is achieved consistently for the smaller blocks. For taller blocks, the fingernails do not contact the object, leading to slip and grasp failure, as seen in Fig. 9c. In a few cases, however, the nails caught the object just before slipping out of the hand. While these cases are counted as grasp success in our experiments, one should note that the intended grasp was not achieved. Success must be attributed to coincidence and the design of the finger nails.

The experiment demonstrates that different ways of exploiting environmental constraints succeed under different conditions. It also shows that the success of exploiting environmental constraints depends object characteristics in non-trivial ways. It is therefore desirable to employ a variety of grasp strategies for which the conditions of success have been characterized. Perceptual skills then must classify environments according to which of the strategies' conditions of success are met best.

4.2 Hand Design

In this section we present our initial efforts to design a hand specifically for the exploitation of environmental constraint during grasping. If indeed exploitation of environmental constraints enables robust grasping, such hands should lead to improved grasping performance.

Environmental constraints can be exploited most effectively through contact. We therefore design hands so as to attain and maintain contact without the need for sophisticated sensing and control. We achieve this through the extensive use of under-actuation and passive compliance.

Our previously presented RBO Hand is shown in Fig. 10 [6]. It employs pneumatic continuum actuators in three fingers and has two deformable pads that form the palm. The hand is highly robust (does not fail after thousands of grasps), can withstand blunt collisions, is inherently safe, and easy and cheap to manufacture. And it achieves very robust grasping performance on objects with widely varying geometries, without sensing or control, simply by inflating the continuum actuators (see Fig. 10, a more detailed experimental evaluation for these objects can be found in reference [6]). We obtain these desirable properties at the expense of precise position or force control.

4.2.1 Surface-Constrained Grasp with the RBO Hand

Grasp strategies should take advantage of the hand's ability to compliantly attain and maintain contact with the environment. We presented such a strategy, the surface-constrained grasp, in prior work [6]. This strategy, illustrated in Fig. 11 and for a particularly difficult object in Fig. 12, makes extensive use of the environmental constraints provided by the support surface. It uses contact between the palm and the support to level the hand with the object. The fingers slide along the support to establish reliable contact with the object. Finally, the fingers adapt to the shape

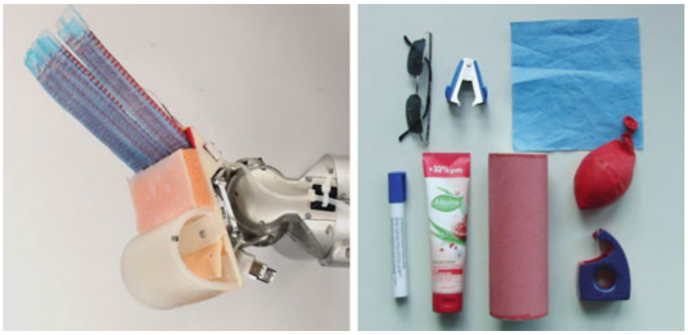


Fig. 10 RBO hand and a selection of objects it can grasp

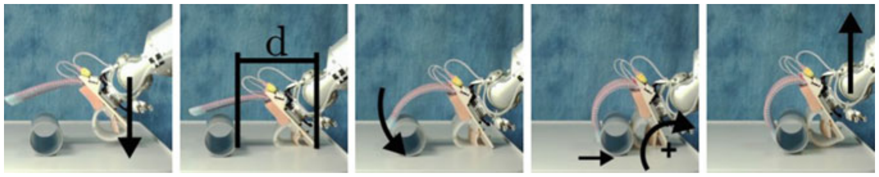


Fig. 11 Execution of a surface-constrained grasp, from *left* to *right* approach phase, exploitation of table to position and orient the hand, the fingers pull the object across the constraint surface, compliance achieves large contact area, and object lift

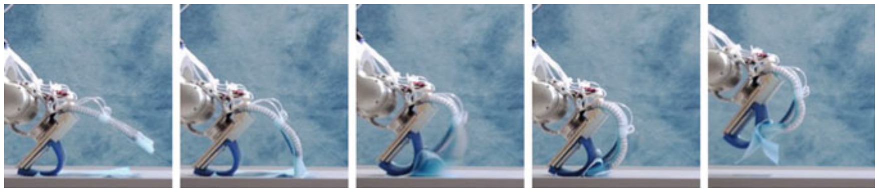


Fig. 12 Surface constrained grasp picking up a piece of tissue, from *left* to *right* exploitation of table to position and orient the hand, exploitation of table to ensure contact with the flat object, compliant grasping, established grasp, and object lift

of the object to establish a robust grasp. These ways of exploiting environmental constraints are facilitated by the hand's design and do not require sensing or control.

4.2.2 Slide-to-edge Grasp with RBO Hand

We implemented the slide-to-edge grasp from Sect. 4.1.2 using the RBO hand. An execution of this strategy is illustrated in Fig. 13. In the first phase, the hand's palm establishes contact with the edge, eliminating position uncertainty. Subsequently, the fingers are flexed and the fingertips establish contact with the table, achieving caging.

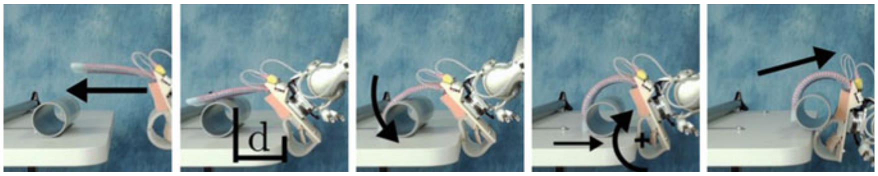


Fig. 13 Edge grasp with the RBO hand, from *left* to *right* approach, compliant contact with the edge, sliding object along surface, sliding fingers along surface, and lift

The hand rotates about the edge/palm contact to ensure contact between the fingers and the support surface, while the compliant fingers slide along the support surface until a grasp is established. Finally, the hand retracts from the edge at an angle of 15°, lifting the fingertips from the surface and detaching the palm form the edge at the same time.

4.2.3 Experiments

In previous work, we demonstrated the ability of the RBO hand to grasp a diverse set of objects of comparable size (see Fig. 10) [6]. We now complement these experiments by evaluating grasp performance with objects of widely varying sizes. We compare the previously published surface-constrained grasp with the novel slide-to-edge grasp. In all experiments, we measure the hand's ability to exploit environmental constraints during grasping by characterizing the region of success under systematic displacements of the object relative to the hand. We consider tolerance to significant displacement as a sign of good constraint exploitation.

In our grasping experiments, we used the set of cylinders shown in Fig. 6a. The set of blocks from Fig. 6b cannot be grasped by the current hand design due to limitations in actuation and hand aperture. Objects were displaced along one axis in twelve (ten) 200 mm increments, using nine different cylinder sizes, for a total of 108 (90) trials for the surface-constrained (slide-to-edge) grasp. To create a dense spatial coverage with a feasible number of experiments, every position was sampled only once for each object size.

The results of these experiments are shown in Fig. 14a. Both strategies achieved grasp success in large and contiguous areas of the explored parameter space. Consistent grasp success under significant variations in object size and placements is a strong indication for the robustness of constraint exploitation provided by the hand design. Note that the hand does not use sensing or control to achieve this grasping performance.

The results in Fig. 14a also show that the grasps are successful under different conditions. The surface-constrained grasp requires cylinders to be at least 22 mm in diameter, whereas the edge grasp requires the presence of an edge within about 100 mm of the object. This confirms the results from Sect. 4.1.2 and further empha-

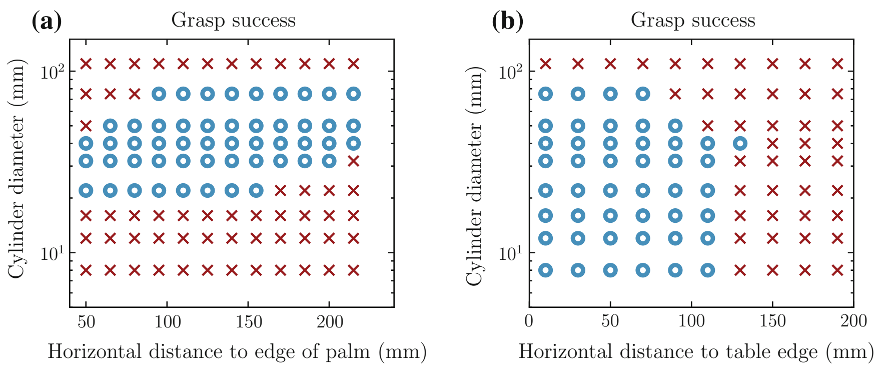


Fig. 14 Grasping results for cylinders of varying sizes and varying horizontal displacements: *circles* indicate grasp success, *crosses* indicate failure. **a** Surface-constrained grasp. Distance measure as indicated in Fig. 11. **b** Slide-to-edge grasp. Distance measure as indicated in Fig. 13

sizes the necessity of employing multiple strategies in response to the specific grasp problem.

Our experiments show that hand design targeted to exploit environmental constraints can lead to robust grasping performance for a variety of object shapes and sizes without the need for sensing and complex control. This advantage, however, comes at the cost of dexterity. The low number of actuators renders the precise control of finger forces, as required for in-hand manipulation for example, very difficult. We will explore in future research if it is possible to strike a good balance between passive compliance for constraint exploitation and detailed control for dexterous manipulation.

5 Conclusion

The work presented in this paper describes the early stages of an integrated research agenda in robotic grasping. This agenda combines the study of human grasping to identify strategies and principles leading to their competencies with the transfer of these principles to robotic grasp planners as well as to robotic hand design.

Informed by a growing body of research in robotic grasping, we formulated the premise that robust and reliable grasping must exploit environmental constraints during the grasping process. In support of this, we presented experiments to show that humans respond to increased difficulty in the grasping problem by increasingly exploiting environmental constraints. We believe that the study of human exploitation strategies will provide important insights into how robotic grasping algorithms can achieve robust grasping performance. Following this motivation, we presented several such strategies on two different robot platforms. Each of the strategies was tailored to exploit constraints commonly present in real-world grasping problems. We demonstrated the success of constraint exploitation in real-world grasping experiments. Finally, we demonstrated the utility of designing hands to facilitate the exploitation of environmental constraints by presenting a mechanically compliant and highly deformable hand. This hand robustly grasps objects of varying sizes and shapes, without the need for explicit force sensing or feedback control.

Acknowledgments We gratefully acknowledge the funding provided by the Alexander-von-Humboldt foundation, and the Federal Ministry of Education and Research (BMBF). We are equally grateful for funding provided by the First-MM project (European Commission, FP7-ICT-248258). We thank SimLab for their support.

References

1. Amend, J., Brown, E., Rodenberg, N., Jaeger, H., Lipson, H.: A positive pressure universal gripper based on the jamming of granular material. IEEE Trans. Robot. **28**(2), 341–350 (2012)
2. Chang, L., Zeglin, G., Pollard, N.: Preparatory object rotation as a human-inspired grasping strategy. In: IEEE-RAS International Conference on Humanoids, pp. 527–534 (2008)
3. Christopoulos, V.N., Schrater, P.R.: Grasping objects with environmentally induced position uncertainty. PLoS Comput Biol 5(10) (2009)
4. Ciocarlie, M., Mier Hicks, F., Stanford, S.: Kinetic and dimensional optimization for a tendon-driven gripper. In: IEEE International Conference on Robotics and Automation, pp. 217–224 (2013)
5. Cutkosky, M.R.: On grasp choice, grasp models, and the design of hands for manufacturing tasks. IEEE Trans. Robot. Autom. **5**(3), 269–279 (1989)
6. Deimel, R., Brock, O.: A compliant hand based on a novel pneumatic actuator. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), pp. 472–480. Karlsruhe, Germany (2013)
7. Dogar, M., Srinivasa, S.: Push-grasping with dexterous hands: Mechanics and a method. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2123–2130 (2010)
8. Dollar, A.M., Howe, R.D.: The highly adaptive SDM hand: design and performance evaluation. Int. J. Robot. Res. **29**(5), 585–597 (2010)
9. Eppner, C., Brock, O.: Grasping unknown objects by exploiting shape adaptability and environmental constraints. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2013)
10. Ernst, M.O., Banks, M.: Humans integrate visual and haptic information in a statistically optimal fashion. Nature **415**(6870), 429–433 (2002)
11. Feix, T., Pawlik, R., Schmiedmayer, H., Romero, J., Kragic, D.: A comprehensive grasp taxonomy. In: Robotics, Science and Systems: Workshop on Understanding the Human Hand for Advancing Robotic Manipulation (2009)
12. Grebenstein, M., Chalon, M., Friedl, W., Haddadin, S., Wimböck, T., Hirzinger, G., Siegwart, R.: The hand of the DLR hand arm system: designed for interaction. Int. J. Robot. Res. **31**(13), 1531–1555 (2012)
13. Grioli, G., Catalano, M., Silvestro, E., Tono, S., Bicchi, A.: Adaptive synergies: an approach to the design of under-actuated robotic hands. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 356–364 (2012)
14. Hirose, S., Umetani, Y.: The development of soft gripper for the versatile robot hand. Mech. Mach.Theory **13**(3), 351–359 (1978)
15. Kaneko, M., Shirai, T., Tsuji, T.: Scale-dependent grasp. IEEE Trans. Syst. Man Cybern. Part A: Syst. Hum. **30**(6), 806–816 (2000)

16. Kappler, D., Chang, L.Y., Pollard, N.S., Asfour, T., Dillmann, R.: Templates for pre-grasp sliding interactions. Robot. Auton. Syst. **60**(3), 411–423 (2012)
17. Kazemi, M., Valois, J.S., Bagnell, J.A.D., Pollard, N.: Robust object grasping using force compliant motion primitives. Technical Report CMU-RI-TR-12-04, Carnegie Mellon University, Robotics Institute (2012)
18. Lozano-Pérez, T., Mason, M.T., Taylor, R.H.: Automatic synthesis of fine-motion strategies for robots. Int. J. Robot. Res. **3**(1), 3–24 (1984)
19. Mason, M.T.: The mechanics of manipulation. In: IEEE International Conference on Robotics and Automation, pp. 544–548 (1985)
20. Melmoth, D.R., Finlay, A.L., Morgan, M.J., Grant, S.: Grasping deficits and adaptations in adults with stereo vision losses. Investig. Ophthalmol. Vis. Sci. **50**(8), 3711–3720 (2009)
21. Miller, A., Allen, P.: Graspit! a versatile simulator for robotic grasping. IEEE Robot. Autom. Mag. **11**(4), 110–122 (2004)
22. Odhner, L., Jentoft, L.P., Claffee, M.R., Corson, N., Tenzer, Y., Ma, R.R., Buehler, M., Kohout, R., Howe, R.D., Dollar, A.M.: A compliant, underactuated hand for robust manipulation. CoRR (2013) arXiv:1301.4394
23. Rodriguez, A., Mason, M.T.: Grasp invariance. Int.J. Robot. Res. **31**(2), 236–248 (2012)
24. Santello, M., Flanders, M., Soechting, J.: Patterns of hand motion during grasping and the influence of sensory guidance. J. Neurosci. **22**(4), 1426–1435 (2002)

Restraining Objects with Curved Effectors and Its Application to Whole-Arm Grasping

Jungwon Seo, Mark Yim and Vijay Kumar

Abstract This paper develops the theory and algorithms for immobilizing/caging polyhedral objects using curved (for example, planar, cylindrical, or spherical) effectors, in contrast to customary point effectors. We show that it is possible to immobilize all polyhedral objects with three effectors with possibly nonzero curvature, with finite extent. We further discuss how to cage the objects and obtain a stable grasp from such a cage. The theory can also be applied to immobilize/cage polygonal objects on the plane. As one application of the theory, we address the problem of whole-arm grasping with robot arms.

1 Introduction

Our main interest is in immobilizing and caging objects. If an object is immobilized, it can neither translate nor rotate. Caging seeks to establish obstacles around an object such that it cannot escape arbitrarily far away. In contrast to previous work, we do not limit ourselves to point effectors (fingertip contacts): it has been customary to consider point contacts and induced wrenches related to the contact normals in studying the mobility of a grasped object [6, 8]. In addition, a mobility theory based on the relative curvature of contacting bodies was established in [12, 13]. Parallel-jaw grasping [2] is one way to obtain stable grasps, although the grasps do not immobilize objects. Recently, it has been shown that grasping can be facilitated by forming cages first [11, 14]. There have also been efforts to compute cages [16].

In this work, we constructively show that all polyhedral objects can be immobilized with three curved (for example, planar, cylindrical, or spherical) effectors providing only frictionless, rigid, unilateral contacts. In [12], the authors proposed a conjecture that asks if general n-dimensional objects can be immobilized by n frictionless, suitably concave effectors; our result thus confirms the conjecture for polyhedral objects. We also discuss how to establish cages exploiting the concavity of the curved effectors and obtain a stable grasp from the cages by simple motion

J. Seo (✉) · M. Yim · V. Kumar
University of Pennsylvania, Philadelphia, PA, USA
e-mail: juse@seas.upenn.edu

planning. We thus propose to integrate effector design into the synthesis of grasps and cages. Although not as simple as point effectors, the curved effectors are simple enough to be easily manufactured, for example, by 3D printing, or emulated in many ways, for example, by cupping the fingers and palm of a multi-fingered hand.

As one application of our theory, we address whole-arm grasping, where robot arms grasp an object not only using their end-effectors but also possibly exploiting other contacts with the arms and torso. Even contact planning for whole-arm grasping becomes intractable as the number of point contacts to be established increases. Whole-arm grasping has been addressed by data-driven approaches [4]. In contrast, we shall introduce a model- and rule-based approach employing energy-based linkage reconfiguration [3, 5], which can be verified analytically. This work builds on our previous paper [15] that addressed the synthesis of spatial grasping; in this paper, we incorporate more complete algorithms and analysis for immobility and caging conditions.

The paper is organized as follows. We begin by reviewing some preliminary concepts in Sect. 2. We discuss our theory and algorithms for immobilizing/caging polyhedral/polygonal objects with curved effectors in Sect. 3. We then address the problem of whole-arm grasping in Sect. 4 as one application of the theory. We conclude in Sect. 5 with suggestions for future work.

2 Preliminaries on Grasping and Caging

We are concerned with an object in contact with effectors restricting its motion. A *grasp* refers to such a configuration with additional information on *contact wrenches* [6, 8], force/moment pairs exerted at the involved contacts. We only consider the contact wrenches of frictionless, rigid, unilateral contacts, which can be represented as the positive linear combination of unit wrenches along the contact normals (Fig. 1a, b). A grasp can be in *equilibrium* if the resultant wrench can be made zero in such a way that not all contact wrenches are equal to zero.

If there is no object *twist* consistent with the contact wrenches of an equilibrium grasp, the object is said to be *immobilized to the first order* (*form-closure*). Even

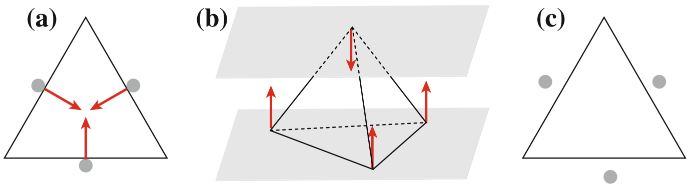


Fig. 1 **a** Immobilizing the regular *triangle* with the three point effectors located at the center of each edge. **b** Clamping the tetrahedron with the two plane effectors contacting the vertex-face pair. **c** Caging the *triangle* with the three point effectors. The *red arrows* are involved unit wrenches in **a**, **b**, and all upcoming figures

if such a twist exists, any finite motion may be restricted by considering *curvature effects*. For example, in Fig. 1a, the object can instantaneously rotate about its centroid (so a first-order kinematic analysis does not predict immobility), but any finite rotation will result in penetrating the effectors. This idea is formalized using the concept of *second-order immobility* [12, 13]. Seven (four) point effectors are required to immobilize general three-dimensional objects to the first (second) order. Such immobility conditions are purely geometric; information on contact geometry is thus sufficient to investigate first- or second-order immobility.

Although not immobilizing, *clamping* [2], also known as parallel-jaw grasping, is one way to realize a *stable* equilibrium grasp with two planar "jaws". Moreover, the grasp can be *force-closed* [6] by considering friction. Even if not, the clamped object can only move on the plane of the jaws. Consider the *antipodal pair* of a convex, polyhedral object, i.e., the intersection of the object with a pair of parallel support planes, which can be a vertex-vertex, vertex-edge, vertex-face, edge-edge, edge-face, or face-face pair. According to [2], the last four types of antipodal pair can provide a clamp as shown in Fig. 1b. Note that they can determine the *width* of the object, the minimum distance between two parallel supporting planes.

Without contacting an object, effectors may just surround the object such that it cannot escape from their *cage* (Fig. 1c). In [14], the concept of an *F-cage* was formalized. Let F be a scalar function defined on effector configurations. Then an F-cage is a configuration of the effectors that cages an object even if they have freedom to move while maintaining the value of F. An F-cage is an *F-squeezing (stretching)* cage if it still cages the object even if the effectors have freedom to move while decreasing (increasing) the value of F. For two point effectors, F can simply be the distance between them, and the prefix "F-" can be ignored as in [16].

3 Restraining Objects with Curved Effectors

In this section, we discuss how to immobilize and cage all polyhedral objects (or just *polyhedra* for short) with at most three curved effectors providing frictionless, rigid, unilateral contacts. It is sufficient to consider only *planar*, *cylindrical*, and *spherical effectors*, shown in Fig. 2, which will be used only for a planar, a line, and a point contact at a face, a convex edge, and a pointed vertex of a polyhedron, respectively, as can be previewed in Fig. 3.

3.1 Immobilizing Polyhedral Objects with Curved Effectors

We first discuss how to synthesize immobilizing grasps using at most three curved effectors shown in Fig. 2. We also show that our grasps are complete, that is, they can immobilize all polyhedra.

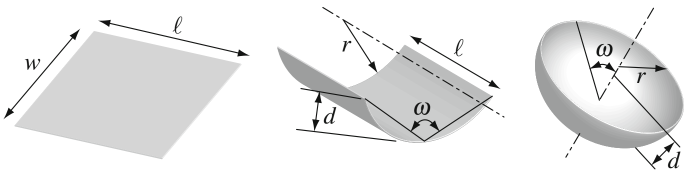


Fig. 2 Examples of a planar, a cylindrical, and a spherical effector. The planar effector has a *rectangular* shape. The latter two are cut parallel to the axis of revolution of a *cylinder* and a *sphere*, respectively. Their dimensions are shown as w (width), r (radius), ℓ (length), d (depth), and ω (aperture)

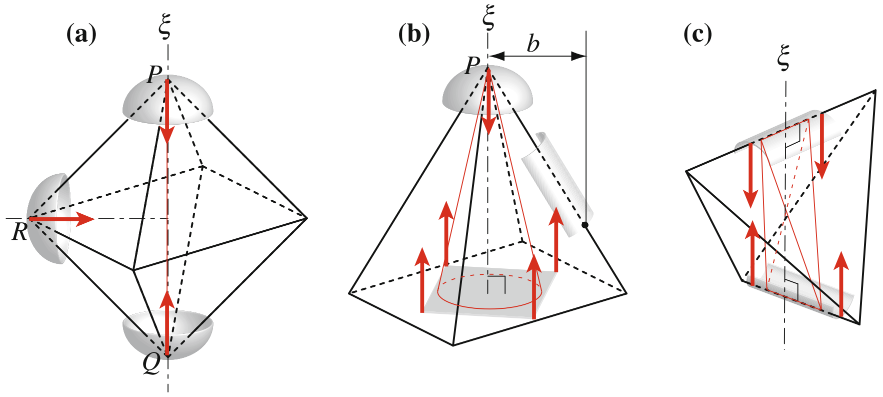


Fig. 3 Immobilizing grasps using (**a**) a vertex-vertex antipodal pair, (**b**) a vertex-face antipodal pair, and (**c**) an edge-edge antipodal pair. The inscribed *red line* segment, *right circular cone*, and *tetrahedron* respectively in (**a**), (**b**), and (**c**) will be used in Fig. 6

None of the effectors can individually immobilize polyhedra. However, two or three effectors can provide immobility by exploiting the antipodal pairs of the convex hull of a given polyhedron, as will be explained below. The nonzero curvature of a cylindrical or spherical effector and the multiple unit wrenches of a cylindrical or planar effector contact play an important role in the immobility. Note that if a *virtual* edge or face [10], that is, an edge or face belonging to the convex hull but not to the original polyhedron, needs to be contacted by a cylindrical or planar effector, w and ℓ (Fig. 2) should be large enough to entirely cover the virtual element.

Immobility using a vertex-vertex antipodal pair: Let (P, Q) denote the pair; see Fig. 3a. Suppose that the two support planes can be made perpendicular to $\overline{PQ}$ and contact only P and Q, respectively. We can have two spherical effectors contact P and Q, respectively, such that their contact normals are collinear to ξ (the line collinear to $\overline{PQ}$) because spherical effectors are locally flat and the vertices are pointed. Now the polyhedron can only rotate about ξ as long as the effectors have a radius less than $\frac{1}{2}d(P, Q)$, where $d(\cdot, \cdot)$ is the Euclidean distance between the two elements, because any finite displacement of $\overline{PQ}$ will result in penetrating the effectors. An additional contact by a spherical effector at R can further restrict any finite rotation about ξ if the radius of the effector is less than $d(R, \xi)$ and its contact normal intersects ξ.

Immobility using a vertex-face antipodal pair: Suppose that the pair determines the width. Consider a planar effector contacting the face and a spherical effector, whose radius can be arbitrary, contacting the vertex P such that its contact normal, collinear to ξ, orthogonally intersects the planar effector; see Fig. 3b. Now the object can only rotate about ξ: the object is at least clamped by the two effectors (only planar motion on the planar effector possible); moreover, any finite translation on the plane results in penetrating the spherical effector. An additional contact by a cylindrical effector at one of the edges incident to P can further restrict any finite rotation about ξ if its contact normal intersects ξ: the edge with the least slope with respect to the face satisfies the condition; the radius of the cylindrical effector should be less than b as shown in the figure.

Immobility using an edge-edge antipodal pair: Suppose that the pair determines the width. Consider two cylindrical effectors, whose radius can be arbitrary, respectively contacting the two edges such that their contact normals are all parallel to ξ, the common perpendicular of the two edges; see Fig. 3c. If ξ intersects both effectors, the object is immobilized: it is at least clamped by the two effectors; moreover, one of the cylindrical effectors only allows the object to translate along its axis of revolution, but such translation is not allowed by the other effector.

The elements that the effectors contact, except for the edge in Fig. 3b, can easily be reached because they belong to the convex hull; the minimum required aperture of a cylindrical or spherical effector can be determined by the object geometry. The following theorems verify the completeness of the grasps: all polyhedra can be immobilized by applying the grasps. We first verify that even only the first type of grasp can be sufficient.

Theorem 1 *Every polyhedron can be immobilized with three spherical effectors of appropriately chosen dimensions.*

Proof We first show that every polyhedron has a vertex-vertex antipodal pair (P, Q) that allows two support planes perpendicular to $\overline{PQ}$ and contacting only P and Q, respectively. Consider the collection of the vertices of the polyhedron. Let (P, Q) be a pair of vertices determining the maximum distance between two vertices of the collection. Consider two planes Π_P and Π_Q perpendicular to $\overline{PQ}$ and contacting the polyhedron respectively at P and Q. No other vertex of the polyhedron can be located on Π_P and Π_Q because (P, Q) determines the maximum distance. Therefore, Π_P (Π_Q) is supporting the polyhedron only at P (Q); (P, Q) is the desired vertex pair.

Next, we show that an additional vertex, denoted as R, can be found such that a contact normal at R by a spherical effector intersects the line of $\overline{PQ}$. Let R be the vertex that is the most distant from the line of $\overline{PQ}$. Consider a plane Π_R parallel to $\overline{PQ}$, perpendicular to the plane of $\triangle PQR$, and contacting the polyhedron at R. No other vertex of the polyhedron can be located on Π_R except on ξ_R, a line passing through R and parallel to $\overline{PQ}$ on Π_R, because R determines the maximum distance. However, R is not a mid-edge vertex. Therefore, we can make the contact normal of a spherical effector intersect the line of $\overline{PQ}$; R is the desired vertex.

We finally get immobility with three spherical effectors respectively contacting P, Q, and R as explained in **immobility using a vertex-vertex antipodal pair**. □

The next theorem shows completeness in terms of general polyhedra, i.e., polyhedra that do not have parallel elements that can be edges or faces.

Theorem 2 *Every general polyhedron can be immobilized with either (1) two cylindrical effectors or (2) a planar, a cylindrical, and a spherical effector of appropriately chosen dimensions.*

Proof Every general polyhedron can be clamped with either a vertex-face or an edge-edge antipodal pair. The polyhedron can then be immobilized using the antipodal pair by applying **immobility using a vertex-face antipodal pair** or **immobility using an edge-edge antipodal pair**. □

Our approach can be generalized in a number of ways. On the one hand, more types of grasps can be considered by employing other types of antipodal pairs. For example, Fig. 4 shows an immobilizing grasp with two spherical effectors and one planar effector on an edge-face antipodal pair; it can be proved that if the involved contact wrenches can be in equilibrium, then we actually get immobility. On the other hand, the effector shapes can also be generalized because we only need the curvature of contact points.

We now proceed to designing an algorithm to obtain the grasps. First, it takes $O(n \log n)$ expected time to compute the convex hull of a given polyhedron, where n is the number of the vertices. Then all of the antipodal pairs of the convex hull can be found in $O(n^2)$ time by applying a technique introduced in [2]. For a vertex-vertex or vertex-face antipodal pair, it additionally takes $O(n)$ time to find the third contact location. Note that, however, the overall complexity will also depend on collision detection algorithms if the global geometry of the effectors is to be checked. Figure 5 shows some example grasps found by the algorithm; more than 100 grasps could be found in less than 5 s with a simple C++ implementation.

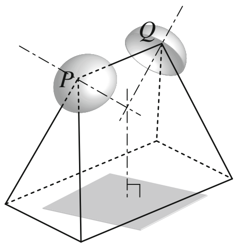


Fig. 4 An immobilizing grasp using the edge-face antipodal pair that determines the width

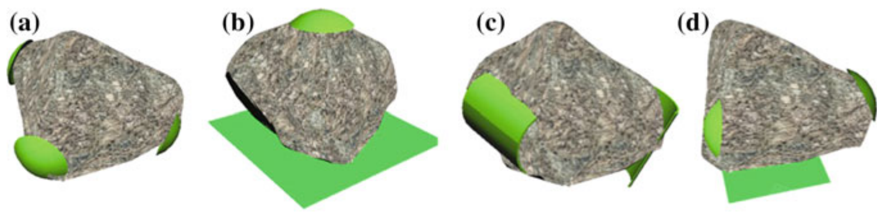


Fig. 5 The two or three curved effectors colored in *green* are grasping the rock model with 1,000 faces (*courtesy* Malcolm Lambert, Intresto Pty Ltd.). The rock is immobilized using **a** a vertex-vertex pair, **b** a vertex-face pair (a cylindrical effector is not shown), **c** an edge-edge pair, and **d** an edge-face pair

3.2 Caging Polyhedral Objects with Curved Effectors

We now discuss the synthesis of cages using the global geometry of the effector models in Fig. 2. The cages are based on the immobilizing grasps in Fig. 3; it is sufficient to use only the two effectors at the antipodal pairs. The inscribed shapes in Fig. 3, reproduced in Fig. 6, allow us to establish sufficient conditions for caging.

Cage using two spherical effectors: Given a vertex-vertex antipodal pair, (P, Q), we consider how to cage $\overline{PQ}$ with two spherical effectors; see Fig. 6a. Suppose that the effectors are only allowed to move such that their axes of revolution are always collinear; the axis is denoted as η. Then $\overline{PQ}$ is caged with the effectors if δ is small enough to guarantee (1) P and Q are respectively in the "pockets", i.e., the interior of the convex hull, of the effectors and (2) $c < d(P, Q)$, where c is the maximum opening between the two effectors.

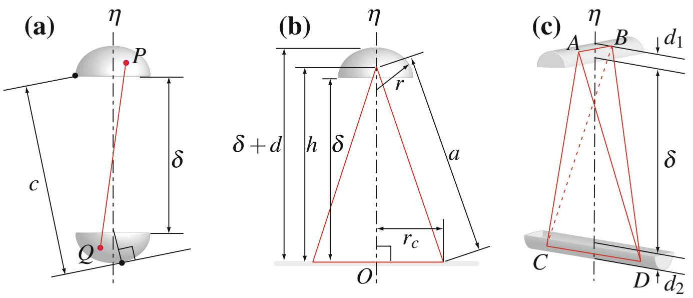


Fig. 6 Cages using two curved effectors. The *red* inscribed shapes are reproduced from Fig. 3. **a** A cage using two spherical effectors. $c = c(\delta)$ denotes the maximum distance between a point on the rim of one effector and a point on the surface of the other effector; thus $\overline{PQ}$ (and thus the polyhedron) cannot escape from the effectors if $c < d(P, Q)$. **b** A cage using a spherical and a planar effector. If $\delta + d < a$, the vertex of the cone cannot be located below (closer to the planar effector than) its current position. Therefore if additionally $\delta < h$, the cone (and thus the polyhedron) cannot escape from the two effectors. **c** A cage using two closed-ended cylindrical effectors. Note that δ denotes the vertical distance between the two effectors in all the three cases

Cage using a spherical and a planar effector: Given a vertex-face antipodal pair determining the width, we consider how to cage the cone in Fig. 6b with a spherical and a planar effector. First suppose that two infinitely large planar effectors are clamping the cone at the apex and base. We now allow the effectors to move in such a way that they remain parallel to each other. If their distance is less than a, the side length of the cone, the clamp can stably be recovered by just making the effectors approach each other; moreover, the distance between the apex and the effector at the base is always larger than h, the height of the cone. Instead of the planar effector at the apex, now consider a spherical effector only allowed to relatively translate along its axis η perpendicular to the other planar effector (Fig. 6b). Then the cone is caged with the effectors if δ is small enough to guarantee (1) the apex is in the pocket of the spherical effector, (2) $\delta + d < a$, and (3) $\delta < h$: (2) and (3) guarantee that the apex cannot escape from the pocket by the analysis above. Note that the planar effector at the base only has to be large enough to contain a disk of radius $r + r_c$ centered at O, where η intersects the planar effector.

Cage using two cylindrical effectors: Given an edge-edge antipodal pair determining the width, we consider how to cage the tetrahedron in Fig. 6c with two cylindrical effectors. Consider again the two infinitely large planar effectors clamping the tetrahedron at the edge pair $(\overline{AB}, \overline{CD})$. If their distance is less than a, the smallest value among $d(\overleftrightarrow{AB}, C)$, $d(\overleftrightarrow{AB}, D)$, $d(\overleftrightarrow{CD}, A)$, and $d(\overleftrightarrow{CD}, B)$, where $\overleftrightarrow{AB}$ is the line of $\overline{AB}$, and so on, the clamp can stably be recovered by just making the effectors approach each other. Moreover, the lowest (closest to the bottom effector) possible positions of A and B can be found by rotating the tetrahedron about $\overline{CD}$ lying on the bottom effector. The highest possible positions of C and D can also be found similarly. Instead of the planar effectors, now consider two cylindrical effectors facing each other and only allowed to relatively translate on their common perpendicular η. To simplify analysis, assume that the sides of the effectors are closed as can be seen in Fig. 6c. Then the tetrahedron is caged with the effectors if δ is small enough to guarantee (1) $\overline{AB}$ and $\overline{CD}$ are respectively in the pockets of the cylindrical effectors, (2) $\delta + d_1 + d_2 < a$, and (3) d_1 (d_2) is large enough to contain the lowest (highest) positions of A and B (C and D): (2) and (3) guarantee that the edges cannot escape from the pocket by the analysis above.

As a corollary of Theorems 1 and 2, the three types of cages are complete. Moreover, other effector geometry can also be considered; using cylindrical or spherical surfaces is just one way to satisfy the caging conditions.

The following theorem states what happens if the two caging effectors get closer.

Theorem 3 *For the three types of cages discussed above, an equilibrium grasp is obtained if the two effectors are controlled such that the relative velocity is along η and δ monotonically decreases until contact is established.*

Proof We first show that δ is a *grasping function* [14] for the cages. In each type, the configuration space of the two effectors can be represented as $\mathscr{M} = SE(3) \times SE(3)$; δ is a semi-algebraic scalar function $\delta : \mathscr{M} \to \mathbb{R}$ invariant with respect to the rigid transformations of the effectors as a whole in that it is the distance between the effectors (Fig. 6). Furthermore, the preimages of δ do not cage the object below (above) a certain value m (M) such that $m < M$, for example, $m = 0$ and $M = h$ in Fig. 6b. Then δ is a grasping function according to [14].

In addition, the cages are δ-*squeezing cages* [14] in that the object remains caged even if δ decreases, and then there exists a path in $\mathscr{M}$ that leads the effectors into a configuration that can realize an equilibrium grasp. Furthermore, in terms of the one-dimensional set representing the relative configuration space of the two effectors, δ can be considered as a convex, i.e., linear, function. Then we get to a configuration to realize an equilibrium grasp only by moving the effectors such that δ monotonically decreases by the result of [14]. $\square$

A translation that monotonically decreases δ will be referred to as a *squeezing motion* in the remaining discussion. Note that, however, the resultant grasp might not be the one we have expected; for example, the circular rim of a spherical effector may unexpectedly contact the object. Still, it is guaranteed to be a configuration to realize equilibrium. We may then add more contacts to further secure the grasp.

3.3 Immobilizing/Caging Polygonal Objects with Curved Effectors

In this subsection, the results of Sects. 3.1 and 3.2 are applied to polygonal objects (or just *polygons* for short) with curved effectors on the plane.

Corollary 1 *Every polygon can be immobilized with two circular effectors of appropriately chosen dimensions.*

Corollary 2 *Every general polygon, whose convex hull does not have parallel edges, can be immobilized with a linear and a circular effector of appropriately chosen dimensions.*

Here is a sketch of proof. For Corollary 1, consider a pair of vertices determining the diameter. Two circular effectors contacting the vertices can then provide planar immobility. For Corollary 2, we use the fact that a general polygon allows clamping only at an edge-vertex pair. Then a linear and a circular effector at the pair can provide planar immobility. To illustrate, see Fig. 7a, b and imagine the circular (linear) effectors are actually contacting the vertices (edge).

We can also construct planar cages of two curved effectors inspired by the geometry of the grasps; see Fig. 7a, b. As implied by the red line segment and cone in the figures, the caging conditions can be established similarly to Fig. 6a, b. We can also add more grasp/cage pairs. As shown in Fig. 7c, two point effectors suffice to immobilize/cage some concave polygons; the related caging condition is discussed in [16]. Finally, the following corollary can be proved similarly to Theorem 3.

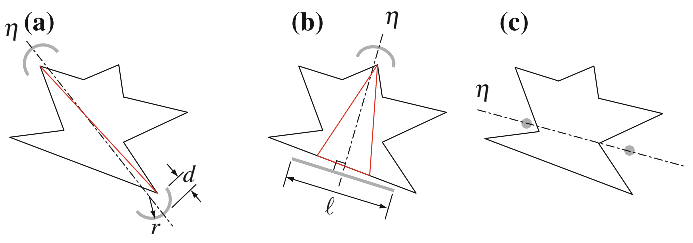


Fig. 7 The polygon can be caged by **a** the two circular effectors, **b** the linear and circular effectors, and **c** the two point effectors. Effector dimensions are shown as r (radius), ℓ (length), and d (depth)

Corollary 3 *For the cages shown in Fig. 7, an equilibrium grasp is obtained if the two effectors are controlled such that the relative velocity is along η and the distance monotonically decreases until contact is established.*

4 An Application to Whole-Arm Grasping

Our approach in Sect. 3 seeks to minimize the number of restraining effectors without losing stability. We show how this approach can be applied to whole-arm grasping with robot arms, which can be effective for grasping large, bulky objects with relatively small end-effectors and the armchain between them.

4.1 Approach

Whole-arm grasping is naturally related to an open kinematic chain. Here, we particularly consider a *planar* open kinematic chain where revolute joints are connecting rigid links moving on the plane; the platform will be just referred to as a *manipulator*. The planar architecture suffices to realize even our spatial grasps in that the three effective contact wrenches respectively from the three contacts should be from a *planar pencil* and thus coplanar. We again assume that the manipulator only provides frictionless, rigid, unilateral contacts. But, our strategy based on such conservative assumptions can also be effective for physical environments with nonzero friction and compliance. At least two of the manipulator links should be shaped to provide the curved effectors in order to apply our theory; we call them *end-effectors*. They may be made exchangeable to suit the size and shape of an arbitrary object (Fig. 8a, b). Without making dedicated end-effectors, the curved shapes may be emulated in some ways, e.g., the flexion of a joint for a circular effector (Fig. 8c) for planar grasping (Sect. 3.3).

Our approach to whole-arm grasping is composed of two phases: *preshaping* and *squeezing*. In the preshaping phase, we move the manipulator such that its two

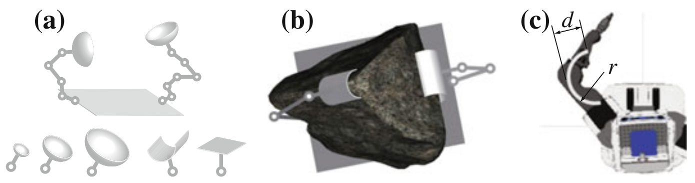


Fig. 8 **a** A manipulator for whole-arm grasping. The base and the two end-effectors of the planar open kinematic chain provide a planar and spherical surfaces, respectively. We can also consider a collection of exchangeable effectors with various sizes and shapes. **b** An example of an immobilizing whole-arm grasp on the rock model with the manipulator. **c** The flexion of the PR2's elbow can emulate a circular (or cylindrical) effector shown in the figure

end-effectors can cage the object. The fact that we can aim at any of our cages, whose collection is not a set of measure zero, can facilitate involved motion planning. In the squeezing phase, the two end-effectors perform a squeezing motion. Meanwhile, we can have other links contact the object without losing stability, which can be justified by the following corollary:

Corollary 4 *Suppose that the end-effectors of a manipulator are caging an object as shown in Figs. 6, 7. An equilibrium grasp is obtained if the manipulator moves such that the end-effectors are performing a squeezing motion until contact is established.*

Proof Recall Theorem 3. Here, the configuration space $\mathscr{M}$ of the manipulator can be thought of as $SE(3) \times \mathbb{S}^m$, where the first (second) term addresses the configuration of the base (m joints), but the same argument can also be applied by regarding δ, the distance between the two effectors, as a grasping function again. □

Although other links, besides the end-effectors, contact the object, the corollary shows that the grasp can still be in equilibrium as long as the end-effectors are squeezing. The squeezing phase can be performed in a blind manner without direct feedback of the object pose. In fact, the approach is also popular in multi-fingered grasping [7], where data-driven approaches have mainly been used for preshaping and squeezing, i.e., closing the hand. In contrast, we use a model-based approach possibly for a hyper-redundant arm that does not have an obvious closing motion.

4.2 Planning and Executing Whole-Arm Grasping

4.2.1 Planning Whole-Arm Grasping

The central algorithm is Algorithm 1; the key idea is as follows. Let $\mathscr{C}$ denote the configuration space of a given object-manipulator system, isomorphic to $SE(3) \times SE(3) \times \mathbb{S}^m$ for two independent rigid bodies (the object and the base of the manipulator) and m manipulator joints. It is assumed that the kinematic model of the system is known. The algorithm takes as input an initial configuration of the system $\mathbf{c}_i \in \mathscr{C}$, a

$6 + 6 + m$-dimensional vector; it returns a reference trajectory for the manipulator, $\gamma(s) : [0, 1] \rightarrow SE(3) \times \mathbb{S}^m$, where s is a non-dimensional parameter increasing with time. The following paragraphs elaborate each line of the algorithm.

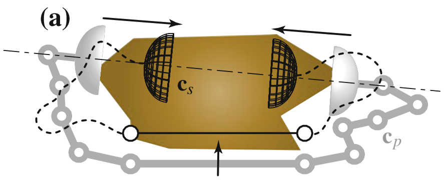
Algorithm 1 Motion planning for whole-arm grasping

```
Input: An initial configuration of an object-manipulator system, $\mathbf{c}_i \in \mathscr{C}$
Output: A reference trajectory for whole-arm grasping, $\gamma$
1: Construct two configurations: $\mathbf{c}_p \in \mathscr{C}$ for preshaping, $\mathbf{c}_s \in \mathscr{C}$ for squeezing.
2: Plan a trajectory $\gamma_{ip}$ for the preshaping: from $\mathbf{c}_i$ to $\mathbf{c}_p$.
3: Plan a trajectory $\gamma_{ps}$ for the squeezing: from $\mathbf{c}_p$ to $\mathbf{c}_s$ (possibly in parallel with Line 2).
4: Concatenate $\gamma_{ip}$ and $\gamma_{ps}$ into $\gamma$, the resultant trajectory from $\mathbf{c}_i$ to $\mathbf{c}_s$ via $\mathbf{c}_p$.
```

Line 1: We first construct $\mathbf{c}_p, \mathbf{c}_s \in \mathscr{C}$ that are supposed to describe configurations at which preshaping and squeezing should aim, respectively (Fig. 9a). They can thus be interpreted as desirable waypoints. $\mathbf{c}_p$ is constructed such that the two end-effectors cage the object in a kinematically feasible manner. $\mathbf{c}_s$ is constructed such that the manipulator deliberately intersects the object; one simple strategy is to make just the two end-effectors approach and intersect the object, but other links can also be considered as shown in Fig. 9a. At $\mathbf{c}_p$ and $\mathbf{c}_s$, the manipulator should be described as simple polygons (Fig. 9a) to facilitate the squeezing as will be explained. In planar grasping for polygons, $\mathbf{c}_p$ may be constructed simply by enclosing the object with the manipulator such that the opening is less than the width of the object.

Line 2: We plan for a trajectory from $\mathbf{c}_i$ to $\mathbf{c}_p$. During the motion, we do not want the manipulator to interact with the object; thus any collisions should be avoided. This can be considered as an ordinary path planning problem. Ultimately, some manipulations such as toppling or tilting might be needed to reach $\mathbf{c}_p$; but it is outside the scope of this paper.

Line 3: We now plan for a trajectory from $\mathbf{c}_p$ to $\mathbf{c}_s$ while ignoring the object geometry. In order to realize a squeezing motion, the manipulator should be regarded as a closed kinematic chain. Furthermore, the length of the virtual link between the



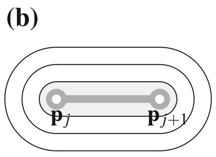


Fig. 9 **a** At the configuration $\mathbf{c}_p$ (in *grey*), the spherical end-effectors are caging the object. The wireframe shows the configurations of the end-effectors at $\mathbf{c}_s$ after the squeezing motion (see the *arrows*), where the longest link is also intersecting the object as shown. If we additionally consider the virtual link connecting the end-effectors, the manipulator configurations can be described as simple polygons. **b** The level set of $d(\cdot, \cdot)$ appeared in Eq. (1) allows us to address the planar shape of the link $\overline{\mathbf{p}_j\mathbf{p}_{j+1}}$ connecting the joints $\mathbf{p}_j$ and $\mathbf{p}_{j+1}$ for collision avoidance. Some of the level sets are shown as solid boundaries

two squeezing end-effectors must monotonically decrease. This is a hard problem in general, but can be efficiently solved by Iben et al.'s algorithm [5], for interpolating between two planar, simple polygons without any self-intersections. In the algorithm, link lengths can be fixed or monotonically changed, which allows us to implement squeezing. The resultant motion basically reconfigures the two polygons "towards each other" according to a metric defined between a pair of polygons, e.g., the ℓ^2-norm on the vector of vertex coordinates. Whenever the direct reconfiguration increases the value of an energy function such as

$$\mathscr{E} = \sum \frac{1}{d(\mathbf{p}_i, \overline{\mathbf{p}_j\mathbf{p}_{j+1}})^2}, \text{ each term is for joint } \mathbf{p}_i \text{ and link } \overline{\mathbf{p}_j\mathbf{p}_{j+1}}\ (i \neq j, j+1) \tag{1}$$

we follow the downward gradient of $\mathscr{E}$ to avoid self-collisions.

Iben et al.'s algorithm employed in Line 3 is basically for line segment links without joint limits, i.e., $\theta_i \in (-\pi, \pi)$ where θ_i is the angle of joint i ($\theta_i = 0$ when the two links are collinear). We further discuss how to adapt the algorithm so as to address link shapes that are not line segments and joint ranges narrower than $(-\pi, \pi)$, i.e., $\theta_i \in [-\ell_i, u_i]$ where $0 < \ell_i, u_i < \pi$. First, in order to address nonzero link volume, we propose to use $(d(\mathbf{p}_i, \overline{\mathbf{p}_j\mathbf{p}_{j+1}}) - \delta_j)^2$ as the denominator of each term of $\mathscr{E}$ where δ_j is determined for each link $\overline{\mathbf{p}_j\mathbf{p}_{j+1}}$ such that the collection of $\mathbf{x}$'s on the level set $d(\mathbf{x}, \overline{\mathbf{p}_j\mathbf{p}_{j+1}}) - \delta_j = 0$ can address the collision hull of the link (Fig. 9b). Note that two adjacent links overlap each other. Second, If (1) any θ_i is close to its limit, i.e., $\theta_i \in [-\ell_i, -\ell_i + \varepsilon]$ or $\theta_i \in [u_i - \varepsilon, u_i]$ for some $\varepsilon > 0$ and (2) the manipulator is described as a concave polygon, we propose to apply an *expansive motion* [3] to straighten all joints such that θ_i can return to the "safe" range beyond the margin of ε. During an expansive motion for a closed chain, every joint is monotonically unfolded until the chain is convexified. Such a motion always exists as long as the chain is described as a simple, concave polygon and is computed by convex optimization [3]. In case the manipulator is described as a convex polygon, there also exists such an angle-monotone motion to address the joint limit issues [1]. It can be shown that the adaptations do not affect the convergence of the algorithm.

Algorithm 1 can be performed efficiently. Line 1 can be considered as inverse kinematics problems; any collision-aware inverse kinematics algorithm can be applied. Line 2 can be implemented by efficient path planning algorithms such as RRT-based algorithms. Line 3 terminates in a finite number of steps computing the integral curve of a vector field [5].

4.2.2 Executing Whole-Arm Grasping

We make the manipulator follow the resultant trajectory in a quasi-static manner because we are basically concerned with the relative configuration of the object and manipulator. In fact, the motion will necessarily be interrupted on the way because it is designed to collide with the object. If there is neither friction nor compliance, the

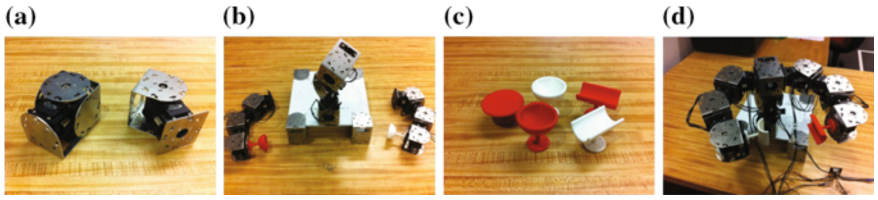


Fig. 10 **a** CKbot modules providing one rotational degree of freedom. **b** Two 3-d.o.f. arms and a 3-d.o.f. spine between them. **c** One planar effector, two spherical effectors, and two cylindrical effectors (*left to right*). **d** A two-armed modular manipulator

configuration where the manipulator stops moving can realize a caged, equilibrium grasp by Corollary 4; the configuration can thus be an acceptable grasp.

In many cases where we cannot ignore friction, the manipulator might get stuck on the way. In other words, it might not be able to reach the configuration of the ideal case because nonzero friction can cause *jamming* and *wedging* [6]. However, both phenomena imply *force-closure* [6], which in turn implies involved wrenches are in equilibrium. Thus the jammed or wedged configuration can also be an acceptable grasp; we then have more candidates for acceptable grasps due to nonzero friction.

Even if we cannot ignore compliance, the local stability of the resultant grasp is guaranteed. If the manipulator happens to immobilize the object, the object remains locally dynamically stable under a common stiffness model [12]; nonzero friction can further enhance the stability. Even if not immobilizing, the force-closure grasp by the caging effectors can be made stable [9]. The guaranteed stability allows us to exert internal forces, further securing the resultant grasp, only by position control, i.e., simply by letting the manipulator "move" as planned. A terminating condition can then be stated as follows: *terminate the motion of the manipulator after it stops moving and its joint torque values exceed appropriate threshold values.* The threshold values should take the safety of the system into account. More refined terminating conditions can be considered if visual or tactile sensing is available.

4.3 Implementing Whole-Arm Grasping

We implemented whole-arm grasping with CKbot,[1] our chain style modular robot system. In terms of kinematics, each module can be used as an 1-d.o.f. swivel or elbow joint (Fig. 10a). Figure 10b shows the subassemblies of two 3-d.o.f. arms and a 3-d.o.f. spine. The arms are planar chains, composed of three elbow joints, such that the energy-based squeezing can be applied; the spine, composed of two swivel joints and one elbow joint, provides all the three rotational degrees of freedom by realizing *z*-*y*-*z* Euler angles. Figure 10c shows 3D printed curved end-effectors compatible

[1]http://www.modlabupenn.org/ckbot.

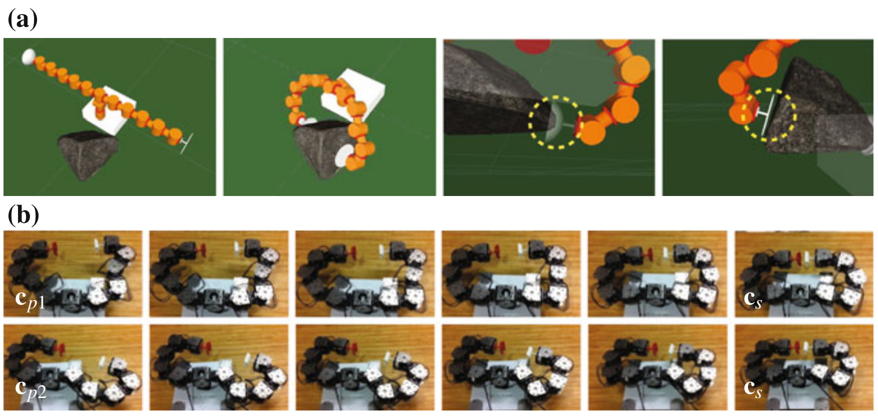


Fig. 11 **a** Preshaping: in the *first two panels*, the simulated modular manipulator with the 10-d.o.f. armchain is moving to the preshape where the end-effectors are caging the rock. In the *last two panels*, the end-effectors were zoomed in. Uncertainty in sensing and control can be accommodated in the cages. **b** Squeezing: in each row, the real, 10-d.o.f. armchain is reconfiguring from the preshape, $\mathbf{c}_{pi}$, to the common squeezed configuration, $\mathbf{c}_s$ (*left* to *right*); the distance between the end-effectors is monotonically decreasing without collisions

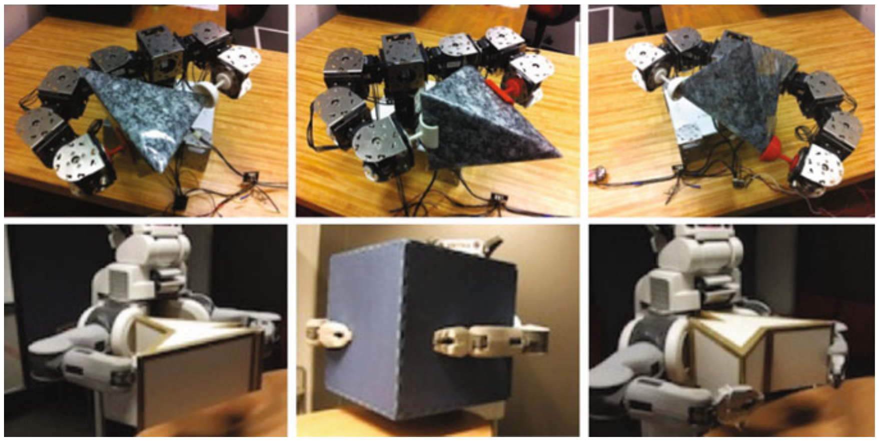


Fig. 12 Some example grasps by the modular manipulator (*top*) and the conventional manipulator, PR2 (*bottom*). The PR2 is applying planar grasping (Sect. 3.3) that can be effective for the prismatic objects

with CKbot; the arms in Fig. 10b actually have the two spherical end-effectors. In Fig. 10d, the finished two-armed modular manipulator is shown.

Our software implementing Algorithm 1 is organized as ROS[2] packages. They provide methods for grasp synthesis, preshaping, and squeezing. Figure 11a shows

[2]http://www.ros.org.

the simulated model of the modular manipulator, with two 5-d.o.f. arms, is preshaping for a cage. Figure 11b shows two examples of the energy-based squeezing for the 10-d.o.f. armchain. Finally, in the top row of Fig. 12, the tetrahedral carton, with the texture of marble, is grasped by the chain of 4-d.o.f. and 2-d.o.f. arms in the first panel, the chain of two 3-d.o.f. arms in the second panel, and the chain of 2-d.o.f. and 4-d.o.f. arms in the last panel. The grasps are respectively derived from the cages of one planar and one spherical, two cylindrical, and two spherical effectors as can be noticed by the end-effectors used in each grasp.

5 Concluding Remarks

We have presented the theory and algorithms for immobilizing and caging objects using at most three simple, curved effectors with only frictionless, rigid, unilateral contacts. Based on these results, we addressed the problem of whole-arm grasping for grasping objects that are large compared to the size of end-effectors. Our future work addresses optimizing the geometry of the curved effectors and incorporating sensing for feedback during grasping.

Acknowledgments We gratefully acknowledge the support of NSF 1328805, 1138847, and ARL Grant W911NF-10-2-0016.

References

1. Aichholzer, O., Demaine, E.D., Erickson, J., Hurtado, F., Overmars, M., Soss, M.A., Toussaint, G.T.: Reconfiguring convex polygons. Comput. Geom. Theory Appl. **20**, 1–2 (2001)
2. Bose, P., Bremner, D., Toussaint, G.T.: All convex polyhedra can be clamped with parallel jaw grippers. Comput. Geom. Theory Appl. **6**, 291–302 (1996)
3. Connelly, R., Demaine, E., Rote, G.: Straightening polygonal arcs and convexifying polygonal cycles. Discret. Comput. Geom. **30**, 205–239 (2003)
4. Hsiao, K., Lozano-Perez, T.: Imitation learning of whole-body grasps. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2006)
5. Iben, H., O'Brien, J., Demaine, E.: Refolding planar polygons. Discret. Comput. Geom. **41**(3), 444–460 (2009)
6. Mason, M.T.: Mechanics of Robotic Manipulation. MIT press, Cambridge (2001)
7. Miller, A.T., Knoop, S., Christensen, H.I., Allen, P.K.: Automatic grasp planning using shape primitives. In: Proceedings. IEEE International Conference on Robotics and Automation, 2003, vol. 2, pp. 1824–1829 (2003)
8. Murray, R.M., Li, Z., Sastry, S.S.: A Mathematical Introduction to Robotic Manipulation. CRC, Boca Raton (1994)
9. Nguyen, V.D.: Constructing stable grasps. Int. J. Robot. Res. **8**(1), 26–37 (1989)
10. Peshkin, M., Sanderson, A.: Reachable grasps on a polygon: the convex rope algorithm. IEEE J. Robot. Autom. **2**(1), 53–58 (1986)
11. Rimon, E., Blake, A.: Caging planar bodies by one-parameter two-fingered gripping systems. Int. J. Robot. Res. **18**(3), 299–318 (1999)

12. Rimon, E., Burdick, J.W.: Mobility of bodies in contact—part I: A 2nd-order mobility index for multiple-finger grasps. IEEE Trans. Robot. Autom. **14**, 696–708 (1998)
13. Rimon, E., Burdick, J.W.: Mobility of bodies in contact—Part II: how forces are generated by curvature effects. IEEE Trans. Robot. Autom. **14**, 709–717 (1998)
14. Rodriguez, A., Mason, M., Ferry, S.: From caging to grasping. Int. J. Robot. Res. **31**(7), 886–900 (2012)
15. Seo, J., Kumar, V.: Spatial, bimanual, whole-arm grasping. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2012)
16. Vahedi, M., van der Stappen, A.: Caging polygons with two and three fingers. Int. J. Robot. Res. **27**(11–12), 1308–1324 (2008)

Part V
Perception

Data Association for Semantic World Modeling from Partial Views

Lawson L.S. Wong, Leslie Pack Kaelbling and Tomás Lozano-Pérez

Abstract Autonomous mobile-manipulation robots need to sense and interact with objects to accomplish high-level tasks such as preparing meals and searching for objects. To achieve such tasks, robots need semantic world models, defined as object-based representations of the world involving task-level attributes. In this work, we address the problem of estimating world models from semantic perception modules that provide noisy observations of attributes. Because attribute detections are sparse, ambiguous, and are aggregated across different viewpoints, it is unclear which attribute measurements are produced by the same object, so *data association* issues are prevalent. We present novel clustering-based approaches to this problem, which are more efficient and require less severe approximations compared to existing tracking-based approaches. These approaches are applied to data containing object type-and-pose detections from multiple viewpoints, and demonstrate comparable quality to the existing approach using a fraction of the computation time.

1 Introduction

Much of the everyday human physical environment is made up of coherent physical objects. Environmental dynamics are well described in terms of the effects of actions on those objects. Perceptual systems are able to report detections of objects with type, location, color, and other properties. Humans naturally designate both goals and prior information in terms of objects. Thus, it is appropriate for robots to construct ‘mental models’ of their environment that are structured around objects, their properties, and their relations to one another.

L.L.S. Wong (✉) · L.P. Kaelbling · T. Lozano-Pérez
CSAIL, MIT, Cambridge, MA 02139, USA
e-mail: lsw@csail.mit.edu

L.P. Kaelbling
e-mail: lpk@csail.mit.edu

T. Lozano-Pérez
e-mail: tlp@csail.mit.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_25

In this work, we define it a semantic world model to be a set of objects with associated attributes and relations. For concreteness, consider the following tasks and their potentially relevant objects and attributes:

- Cooking steaks on a pan: Objects—Steaks, pan, stove, etc.
  Attributes—*CookedTime*, *Thickness*, *SteakPositionRelativeToPan*
- Finding chairs for guests: Objects—Furniture, people
  Attributes—*IsChair*, *Sittable* (if ¬*IsChair*), *Movable*, *Location*, *SittingOn*
- Rearranging objects on a table: Objects—Items on table
  Attributes—*Shape*, *Type*, *RelativePositionAndOrientation*, *GraspPoints*

A common theme underlying these tasks, and many others, is that successful planning and execution hinges on good world-state estimation and monitoring. Dynamic attributes listed above also highlight why object-based representations are uniquely suitable for dynamic tasks: transition dynamics tends to operate on the level of objects. For example, it is much more natural to express and reason about a piece of steak that is being cooked, as opposed to points in a point cloud or cells in an occupancy grid that are 'cooked'. Although we focus on the static case in this paper, our ultimate goal is to provide a framework for estimating and monitoring large semantic world models involving objects and attributes that change over time as a result of physical processes as well as actions by the robot and other agents.

In this work, we address the problem of constructing world models from semantic perception modules that provide noisy observations of attributes. Due to noise, occlusion, and sensors' limited field of view, observations from multiple viewpoints will typically be necessary to produce a confident world model. Because attribute detections are sparse, noisy, and inherently ambiguous, where it is unclear which attribute measurements were produced by the same object across different views, *data association* issues become critical. This is the greatest challenge; if the measurement-object correspondences were known, the resulting object-attribute posterior distributions would be efficiently computable.

We begin by stating a formal model for a simplified 1-D version of the world-model estimation problem in Sect. 2, and then review a classic solution approach based on tracking in Sect. 3. The main contribution of this work is the development of several novel clustering-based data association approaches, described in Sects. 4 and 5. Application of the semantic world-modeling framework to object type-and-pose estimation is then briefly discussed in Sect. 6, followed in Sect. 7 by experimental results using data collected with a Kinect sensor on a mobile robot.

2 The 1-D Colored-Lights Domain

The approaches described in this paper apply to domains of arbitrary complexity. For clarity of explanation we begin by introducing a model of minimal complexity and then use it for an initial demonstration of the methods.

The world consists of an unknown number (K) of stationary lights. Each light is characterized by its color c_k and its location $l_k \in \mathbb{R}$. A finite universe of colors of size T is assumed. A robot moves along this 1-D world, occasionally gathering partial views of the world with known field of views $[a_v, b_v] \subset \mathbb{R}$. Within each view, M_v lights of various colors and locations are observed, denoted by $o^v_m \in [T] \triangleq \{1, \ldots, T\}$ and $x^v_m \in \mathbb{R}$ respectively. These (o^v_m, x^v_m) pairs may be noisy (in both color and location) or spurious (false positive—FP) measurements of the true lights. Also, a light may sometimes fail to be perceived (false negative—FN). Given these measurements, the goal is to determine the posterior distribution over configurations (number, colors, and locations) of lights in the explored region of the world.

We assume the following form of noise models. For color observations, for each color t, there is a known discrete distribution $\phi^t \in \Delta^T$ (estimable from perception apparatus statistics) specifying the probability of color observations:

$$\phi^t_i = \begin{cases} \mathbb{P}\,(\text{no observation for color } t \text{ light}), & i = 0 \\ \mathbb{P}\,(\text{color } i \text{observed for color } t \text{ light}), & i \in [T] \end{cases}. \tag{1}$$

A similar distribution ϕ^0 specifies the probability of observing each color given that the observation was a false positive. False positives are assumed to occur in a proportion p_{FP} of object detections. For location observations, if the observation corresponds to an actual light, then the observed location is assumed to be Gaussian-distributed, centered on the actual location. The variance is *not* assumed known and will be estimated for each light from measurement data. For false positives, the location is assumed to be uniformly distributed over the field of view (Unif$[a_v, b_v]$).

Next, we present the core problem of this domain. Given sets of color-location detections from a sequence of views, $\left\{\left\{\left(o^v_m, x^v_m\right)\right\}_{m=1}^{M_v}\right\}_{v=1}^{V}$, we want to infer the posterior distribution on the configuration of lights $\{(c_k, l_k)\}_{k=1}^{K}$, where K is unknown as well. If we knew, for each light, which subset of the measurements were generated from that light, then we would get K decoupled estimation problems (assuming lights are independent from each other). With suitable priors, these single-light estimation problems admit efficient solutions; details can be found in the appendix.

The issue is that these associations are unknown. Therefore, we must reason over the *space* of possible data associations. For each observation, let z^v_m be the index of the light that the observation corresponds to (ranging in $[K]$ for a configuration with K lights), or 0 if the observation is a false positive. z^v_m is the latent association for measurement $\left(o^v_m, x^v_m\right)$. Let $\mathbf{z^v}$ be the concatenated length-M_v vector of all z^v_m variables in view v, and let $\{\mathbf{z^v}\}$ be the collection of all correspondence vectors from the V views. We then aggregate estimates over all latent associations[1]:

$$\mathbb{P}\left(\{(c,l)\} \mid \{\{(o,x)\}\}\right) = \sum_{\{\mathbf{z^v}\}} \mathbb{P}\left(\{(c,l)\} \mid \{\mathbf{z}\}, \{\{(o,x)\}\}\right) \mathbb{P}\left(\{\mathbf{z}\} \mid \{\{(o,x)\}\}\right). \tag{2}$$

[1]Indices have been dropped to reduce clutter; please refer to two paragraphs above for indices.

The first term is given by the decoupled estimation problems mentioned above, and results in a closed-form posterior distribution given in the appendix. The desired posterior distribution on the left is therefore, in exact form, a mixture over the closed-form posteriors. The problem is that the number of mixture components is exponential in M_v and V, one for each full association $\{\mathbf{z}^{\mathbf{v}}\}$, so maintaining the full posterior distribution is intractable. Finding tractable approximations to this light configuration posterior distribution is the subject of Sects. 3–5.

3 A Tracking-Based Approach

If we consider the lights to be stationary targets and the views to be a temporal sequence, a tracking filter approach can be used. Tracking simultaneously solves the data association (measurement correspondence) and target parameter estimation (light colors and locations) problems. Of the wide variety of existing approaches for this classic problem [4], the multiple hypothesis tracking (MHT) filter [18] is most appropriate because it allows for an unknown number of targets. In fact, Elfring et al. [11] recently adopted this approach to the semantic world-modeling problem, and have provided extensive rationale for using MHTs over other tracking approaches.

We provide a gist of the MHT approach and discuss a problematic issue below; readers are referred to Elfring et al. [11] for details. The MHT algorithm maintains, at every timestep (view) v, a distribution over all possible associations of measurements to targets up to v. At each view, MHT therefore needs to propagate *each* previous hypothesis forward with *each* possible association in view v. One way to consider this is as a tree, where nodes of depth v are associations up to view v, and a distribution is maintained on the leaves. Each view introduces a new layer of nodes, where the branching factor is the number of valid associations in that view.

Estimating this branching factor highlights the intractability of the MHT. Assume we know which of the existing targets are within the current field of view based on the hypothesis on previous views (this can be found by gating). Denote the indices of these targets as the size-K_v set $\{k\}^v$. Another common assumption used in the tracking literature is that in a single view, each target can generate at most one non-spurious measurement. We will refer to this as the one-measurement-per-object (OMPO) assumption. We now define validity of correspondence vectors $\mathbf{z}^v$. First, by the OMPO assumption, no entry may be repeated in $\mathbf{z}^v$, apart from 0 for false positives. Second, an entry must either be 0, and target index in $\{k\}^v$, or be a new (non-existing) index; otherwise, it corresponds to an out-of-range target. A correspondence $\mathbf{z}^v$ is valid if and only if it satisfies both conditions.

The following quantities can be found directly from $\mathbf{z}^v$:

$$n_0 \triangleq \text{Number of false positives (0 entries);} \tag{3}$$

$$n_\infty \triangleq \text{Number of new targets (non-existing indices);}$$

$$\delta_k \triangleq \mathbb{I}\left\{\text{Target } k \text{ is detected } (\exists m.\, z_m^v = k)\right\}, k \in \{k\}^v;$$
$$n_1 \triangleq \text{Number of matched targets} = M_v - n_0 - n_\infty = \sum_k \delta_k,$$

where $\mathbb{I}\{\cdot\}$ denotes the indicator function and M_v is the number of measurements in view v. The number of valid associations is given by the following expression:

$$\sum_{n_0=0}^{M_v} \sum_{n_\infty=0}^{M_v-n_0} \binom{M_v}{n_0, n_1, n_\infty} \times \binom{K_v}{n_1} \times n_1! = \sum_{n_0=0}^{M_v} \sum_{n_\infty=0}^{M_v-n_0} \frac{M_v!}{n_0! n_\infty!} \times \frac{K_v!}{n_1!(K_v - n_1)!}. \tag{4}$$

Even with 4 measurements and 3 within-range targets, the branching factor is 304, so considering all hypotheses is clearly intractable. Many hypothesis-pruning strategies have been devised (e.g., [6, 14]), the simplest of which include keeping the best hypotheses or hypotheses with probability above a certain threshold. More complex strategies to combine similar tracks and reduce the branching factor have also been considered. In the experiments of Sect. 7 we simply keep hypotheses with probability above a threshold of 0.01. As we will demonstrate in the experiments, an MHT filter using this aggressive pruning strategy can potentially cause irreversible association errors and make overconfident conclusions.

4 A Clustering-Based Approach

If we consider all the measurements together and disregard their temporal relationship, we expect the measurements to form clusters in the product space of colors and locations ($[T] \times \mathbb{R}$), allowing us to derive estimates of the number of lights and their parameters. In probabilistic terms, the measurements are generated by a mixture model, where each mixture component is parameterized by the unknown parameters of a light. Since the number of lights in the world is unknown, we also do not want to limit the number of mixture components a priori.

A useful model for performing clustering with an unbounded number of clusters is the Dirichlet process mixture model (DPMM) [2, 15], a Bayesian non-parametric approach that can be viewed as an elegant extension to finite mixture models. The Dirichlet process (DP) acts as a prior on *distributions* over the cluster parameter space. A random distribution over cluster parameters G is first drawn from the DP, then a countably infinite number of cluster parameters are drawn from G, from which the measurement data is finally drawn according to our assumed observation models. Although the model can potentially be infinite, the number of clusters is finite in practice, as they will be bounded by the total number of measurements (typically significantly fewer if the data exhibits clustering behavior). The flexibility of the DPMM clustering model lies in its ability to 'discover' the appropriate number of clusters from the data.

We now derive the DPMM model specifics and inference procedure for the colored-lights domain. A few more assumptions need to be made and parameters defined first. Our model assumes that the cluster parameter distribution G is drawn from a DP prior DP(α, H), where H is the base distribution and α is the concentration hyperparameter (controlling the similarity of G and H, and also indirectly the number of clusters). H acts as a 'template' for the DP, and is hence also a distribution over the space of cluster parameters. We set it to be the product distribution of π, the prior on colors, and a uniform distribution over the explored region. To accommodate false positives, which occur with probability p_{FP}, we scale G from the DP prior by a factor of $(1 - p_{\text{FP}})$ for true positives.

For ease of notation when deriving the inference procedure, we express the DP prior in an equivalent form based on the stick-breaking construction [19]. The idea is that the sizes of clusters are determined by a random process that first selects some proportion of the whole ('breaks the stick'), uses one part to define the size of a cluster, and then recursively subdivides the rest. Parameters are drawn from the base distribution H and associated with each cluster. More formally:

$$\theta \sim \text{GEM}(\alpha);\ (c_s, l_s) \sim H \triangleq \pi \times \text{Unif}[A; B], \tag{5}$$

where GEM (Griffiths-Engen-McCloskey) is the distribution over stick weights θ, and $\pi \in \Delta^{(T-1)}$ is a prior distribution on colors, reflecting their relative prevalence. By defining $G(c, l) \triangleq \sum_{s=1}^{\infty} \theta_s \times \mathbb{I}\left[(c, l) = (c_s, l_s)\right]$, G is a distribution over the cluster parameters and is distributed as DP(α, H). The rest of the generative process is:

$$\theta'_k = \begin{cases} p_{\text{FP}}, & k = 0 \\ (1 - p_{\text{FP}})\,\theta_k, & k \neq 0 \end{cases}; \qquad \text{Cluster proportions (with FPs)} \tag{6}$$

$$\mu_k, \tau_k \sim \text{NormalGamma}(\nu, \lambda, \alpha, \beta); \qquad \text{Cluster location distr. params.}$$

$$z_m^v \sim \theta', \qquad m \in [M_v],\ v \in [V]; \qquad \text{Cluster assignment (for each obs.)}$$

Color observation: Location observation:

$$o_m^v \sim \begin{cases} \phi^0, & z_m^v = 0 \\ \phi^{c_z}, & z_m^v \neq 0 \end{cases}; \qquad x_m^v \sim \begin{cases} \text{Unif}[a_v, b_v], & z_m^v = 0 \\ \mathcal{N}\left(\mu_k, \tau_k^{-1}\right), & z_m^v \neq 0 \end{cases}. \tag{7}$$

The most straightforward way to perform inference in a DPMM is by Gibbs sampling. In particular, we derive a collapsed Gibbs sampler for the cluster correspondence variables z and integrate out the other latent variables c, μ, τ, θ. In Gibbs sampling, we iteratively sample from the conditional distribution of each z_m^v, given all other correspondence variables (which we will denote by z^{-vm}). By Bayes' rule:

$$\begin{aligned} &\mathbb{P}\left(z_m^v = k \,\middle|\, z^{-vm}, \{\{(o, x)\}\}\right) \\ &\quad \propto \mathbb{P}\left(o_m^v, x_m^v \,\middle|\, z_m^v = k, z^{-vm}, \{\{(o, x)\}\}^{-vm}\right)\ \mathbb{P}\left(z_m^v = k \,\middle|\, z^{-vm}, \{\{(o, x)\}\}^{-vm}\right) \\ &\quad \propto \mathbb{P}\left(o_m^v, x_m^v \,\middle|\, \{\{(o, x)\}\}_{z=k}^{-vm}\right) \mathbb{P}\left(z_m^v = k \,\middle|\, z^{-vm}\right). \end{aligned} \tag{8}$$

In the final line, the first term can be found from the posterior predictive distributions described in the appendix (Eqs. 15 and 18), noting that the observations being conditioned on *exclude* (o^v_m, x^v_m) and depend on the current correspondence variable samples (to determine which observations belong to cluster k).

The second term is given by the Chinese restaurant process (CRP), obtained by integrating out the DP prior on θ. Together with our prior on false positives:

$$\mathbb{P}\left(z^v_m = k \,\middle|\, z^{-vm}\right) = \begin{cases} (1 - p_{\text{FP}})\,\frac{N_k^{-vm}}{\alpha+N-1}, & k \text{ exists} \\ (1 - p_{\text{FP}})\,\frac{\alpha}{\alpha+N-1}, & k \text{ new} \\ p_{\text{FP}}, & k = 0 \end{cases}, \tag{9}$$

where N_k^{-vm} is the number of observations currently assigned to cluster k (excluding (v, m)), and N is the total number of non-false-positive observations across all views.

By combining Eqs. 8 and 9, we can sample from the conditional distribution of individual correspondences z^v_m. Although the model supports an infinite number of clusters, the modified CRP expression (Eq. 9) shows that we only need to compute $k + 2$ values for one sampling step, which is finite as clusters without data are removed. One sampling sweep over all correspondence variables $\{\{z\}\}$ constitutes one sample from the DPMM. Given the correspondence sample, finding the posterior configuration is simple. Each non-empty cluster corresponds to a light. For each cluster, applying Eqs. 15 and 17 to its associated data provides the posterior distributions on the light's color and location (with observation model precision) respectively. The posterior marginal distribution on the light's location is a t-distribution with mean ν', precision $\frac{\alpha'\lambda'}{\beta'(\lambda'+1)}$, and $2\alpha'$ degrees of freedom.

5 Incorporating View-Level Information and Constraints

The DPMM-based solution to the colored-lights problem is a straightforward application of the DPMM, but ignores two fundamental pieces of information:

- **False negatives (FN)**: The DPMM does not consider which clusters are visible when a measurement is made. It may therefore posit a cluster for a spurious measurement when its absence in other views would have suggested otherwise.
- **One-measurement-per-object (OMPO) assumption**: Consider the scenario depicted in Fig. 1c, where two blue lights are placed close to each other and hence easily confusable. The DPMM ignores the OMPO assumption and may associate both to the same cluster, even if they were both observed in every view.

Both are consequences of the DPMM's conditional independence assumptions.

To see this, consider the concrete example depicted in Fig. 1, where we wish to sample cluster assignments for an entire view's $M_v = 4$ measurements. The DPMM Gibbs sampler samples the cluster assignment for each measurement *individually*, as shown in Fig. 1b. This causes the two right-most measurements to be assigned

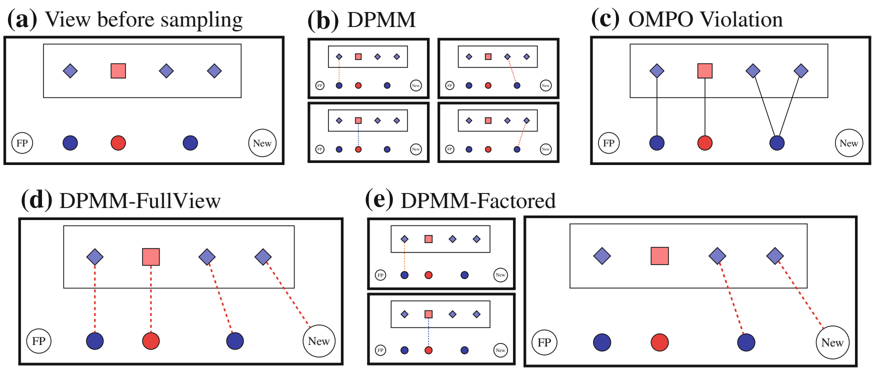


Fig. 1 A concrete example for illustrating concepts in Sect. 5. **a** Each *thick outer box* depicts measurements in the same single view (*inner box*), and the clusters that each measurement can be assigned to (*row below inner box*). The view we consider has 4 measurements of lights' locations and colors. The existing clusters within the field of view are shown as *colored circles* (these were determined from other views). Measurements can also be assigned to the two 'clusters' to the *left* and *right*, for false positives and new clusters respectively. The task is to assign one of the 5 clusters in the *bottom row* to each measurement in the *inner box*. **b** The DPMM samples cluster assignments for each measurement independently. **c** This causes potential violations of the one-measurement-per-object (OMPO) assumption, where each cluster generates at most one observation within each view. **d** One solution is to consider all measurement assignments in the view jointly. However, as explained in Sect. 5.1, this is inefficient. **e** A more efficient approximation is derived in Sect. 5.2 by jointly considering *only* measurements that are OMPO-violating. Measurements that are unlikely to cause constraint violation, such as the two left ones in the example, are considered independently. This provides a trade-off between DPMM and DPMM-FullView

to the same cluster, a violation of the OMPO assumption. The assumption states that *at most one* measurement in a single view can be assigned to each cluster; this view-level constraint cannot be incorporated on the level of individual measurements (DPMM). Likewise, a false negative only arises if *none* of the measurements in a view are assigned to a cluster within the field of view. To handle these constraints we must couple the measurements and sample their assignments *jointly*.

5.1 *DPMM-FullView*

More formally, consider the view's joint correspondence vector $\mathbf{z}^v$. The induced conditional distribution $\mathbb{P}\left(\mathbf{z}^v \,\middle|\, \mathbf{z}^{-v}\right)$ that the DPMM Gibbs sampler uses is, by conditional independence, the product of M_v copies of Eq. 9:

$$\mathbb{P}_{\text{DPMM}}\left(\mathbf{z}^v \,\middle|\, \mathbf{z}^{-v}\right) = \frac{p_{\text{FP}}^{n_0}\ (1-p_{\text{FP}})^{(n_1+n_\infty)}\ \alpha^{n_\infty}\ \left[\prod_{\{m\}_1} N_{\mathbf{z}_m}^{-v}\right]}{\prod_{m=1}^{(n_1+n_\infty)} \alpha + N - m}, \tag{10}$$

where the definitions of n_0, n_1, n_∞ are given in Eq. 3, and $\{m\}_1$ is the set of indices that are matched to existing targets (i.e., $n_1 = |\{m\}_1|$). To incorporate absence information, suppose we knew which K_v of the existing K lights are within the field of view, i.e., $\{k\}^v$ from Sect. 3.[2] This, together with $\mathbf{z}^v$, allows us to determine the detection indicator variables $\{\delta_k\}$ (Eq. 3) and their probabilities:

$$\mathbb{P}\left(\{\delta_k\}\right) = \prod_{k \in \{k\}^v} \left[p_{\mathrm{D}}(k)\right]^{\delta_k} \left[1 - p_{\mathrm{D}}(k)\right]^{1-\delta_k}, \tag{11}$$

where p_{D} is the (target-specific) detection probability defined in Eq. 16. We combine the additional information with the DPMM conditional distribution in a conceptually simple fashion:

$$\mathbb{P}_{\text{FullView}}\left(\mathbf{z}^v \,\middle|\, \mathbf{z}^{-v}, \{k\}^v\right) \propto \mathbb{P}_{\text{DPMM}}\left(\mathbf{z}^v \,\middle|\, \mathbf{z}^{-v}\right) \, \mathbb{P}\left(\{\delta_k\}\right) \, \mathbb{I}\left[\mathbf{z}^v \text{ satisfies OMPO}\right]. \tag{12}$$

The final term evaluates to 1 if the joint correspondence satisfies the OMPO assumption, and 0 otherwise. Hence by construction the correspondence variables sampled from this conditional distribution will incorporate the FN and OMPO constraints.

Although $\mathbb{P}_{\text{FullView}}$ combines all the desired information, the inherent difficulty is hidden in the '$\propto$' sign. The distribution first needs to be normalized before we can sample from it, which is inefficient now because the support of the distribution is the set of correspondence vectors satisfying the OMPO assumption. The OMPO constraint fully couples the measurements' cluster assignments, and all assignments must be considered jointly, as depicted in Fig. 1d. We have essentially reverted to the high branching factor of the MHT! (The exponential blowup of the hypothesis tree is still avoided by sampling.) In the Fig. 1 example, $\mathbb{P}_{\text{FullView}}$ must be evaluated for 304 different values of $\mathbf{z}^v$, compared to the $4 \times 5 = 20$ required for the DPMM.

5.2 *DPMM-Factored*

A closer look at the nature of the OMPO violation suggests a potential approximation to $\mathbb{P}_{\text{FullView}}$. In Fig. 1c, the violation is caused by *only* the two right-most measurements; the two measurements on the left are not easily confusable with the others and hence are easy to handle from a data association perspective. This suggests coupling *only* those measurements that cause OMPO violations. More generally, suppose we can partition each view's set of measurements into 'violating' subsets, where all OMPO violations are contained within a single subset (with high probability). That is, a good partition has the property that any two measurements belonging to different subsets will have low probability of being assigned to the same cluster (and hence

[2]The correct Bayesian approach is to integrate over the posterior distribution of each light's location, which is intractable. This can be approximated by sampling the locations, then averaging the subsequent computations. In practice we found that using the posterior mean was sufficient.

causing an OMPO violation). Let $\mathscr{P}$ denote such a partition, and let $\left\{\mathbf{z}_p^v\right\}_{p\in\mathscr{P}}$ denote the restrictions of $\mathbf{z}^v$ to each subset $p \in \mathscr{P}$. Then:

$$\mathbb{I}\left[\mathbf{z}^v \text{ satisfies OMPO}\right] \approx \prod_{p\in\mathscr{P}} \mathbb{I}\left[\mathbf{z}_p^v \text{ satisfies OMPO}\right]. \tag{13}$$

Returning to Fig. 1c, the most refined partition contains three subsets, where the sole non-singleton contains the two right-most OMPO-violating measurements.

The other two terms in $\mathbb{P}_{\text{FullView}}$ (Eq. 12) are product distributions that factor nicely according to $\mathscr{P}$. We therefore arrive at the following *factored* approximation:

$$\mathbb{P}_{\text{Factored}}\left(\mathbf{z}^v \,\middle|\, \mathbf{z}^{-v}, \{k\}^v\right) \propto \prod_{p\in\mathscr{P}} \mathbb{P}_{\text{DPMM}}\left(\mathbf{z}_p^v \,\middle|\, \mathbf{z}^{-vp}\right)\ \mathbb{P}\left(\{\delta_k\}\,|_p\right)\ \mathbb{I}\left[\mathbf{z}_p^v \text{ OMPO}\right]. \tag{14}$$

This form makes clear that each factor can be normalized and sampled independently. With a good partition, this breaks up the large joint computation in DPMM-FullView into several smaller ones within each subset of $\mathscr{P}$. Using the partition described above for the concrete example in Fig. 1 gives us the sampling process depicted in Fig. 1e, where only the OMPO-violating measurement pair is considered jointly. This results in computing $5 + 5 + 22 = 32$ values, which is slightly greater than DPMM (20) but significantly fewer than DPMM-FullView (304).

One issues remains: Where does the partition come from? This is crucial for all factored approximation: the aggressiveness of partitioning determines the trade-off between approximation error and efficiency. On one extreme, the DPMM model is similar to a fully-factored model (but does not take into account false negatives); on the other extreme, DPMM-FullView is equivalent to a one-set partition. The example in Fig. 1c once again provides an answer: 'violating' subsets can be found by examining clusters in the DPMM samples. Specifically, if measurements tend to be assigned to the same cluster across samples, then clearly they are strong violators and should be considered jointly. We therefore group measurements together if the proportion of samples in which they are assigned to the same cluster exceeds some threshold value. This proportion allows one to select an appropriate trade-off level.

6 Application to Object Type-and-Pose Estimation

As mentioned in Sect. 2, the colored-lights domain is representative of the semantic world-model estimation problem by considering lights as objects and locations and colors as attributes. Extension to additional attributes and higher-dimensional locations (3-D locations, 4-D or 6-D poses) is straightforward since the correspondence priors described in Sects. 3–5 do not depend on the observations. If attributes are independent, we simply take the product of their observation models when determining their posterior or predictive distributions, e.g., in Gibbs sampling (Eq. 8). Dependent

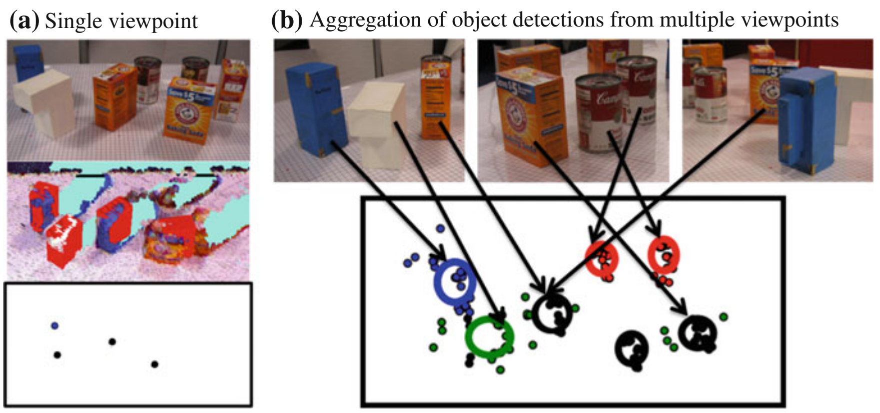

(a) Single viewpoint
(b) Aggregation of object detections from multiple viewpoints

Fig. 2 **a** Given a tabletop scene (*top*), we want to estimate the types and poses of objects in the scene using a black-box object detector. From a single Kinect RGB-D image, however, objects may be occluded or erroneously classified. In the rendered image (*middle*; detections superimposed in *red*), three objects are missing due to occlusion, and the bottom two objects have been misidentified. The semantic attributes that result in our representation are very sparse (*bottom*; dot location is measured 2-D pose, color represents type). **b** Aggregation of measurements from many different viewpoints (*top*) is therefore needed to construct good estimates. However, this introduces data association issues of the type addressed in this work, especially when multiple instances of the same object type are present. From all the object detection data, as shown (botto*m*) by dots (each dot is one detection), our goal is to estimate the object types and poses in the scene (shown as *thick circles* centered around location estimate; color represents type, circle size reflects uncertainty). The estimate above identifies all types correctly with minimal error in pose

attributes will need to be jointly considered as a single unit. For example, for pose estimates with non-diagonal error covariances, the normal-gamma prior needs to be replaced with a normal-Wishart prior.

We applied our discussed approaches to object type-and-pose estimation on table-top scenes, illustrated in Fig. 2. This is similar to the colored-lights domain, where ‘type’ is equivalent to ‘color’, and ‘pose’ is a 3-D version of ‘location’.[3] 3-D point cloud data was obtained from a Kinect sensor mounted on a mobile robot. A ROS perception service attempts to detect instances of the known shape models in a given point cloud. This is done by locating horizontal planes in the point cloud, finding clusters of points resting on the surface, and then doing stochastic gradient descent over the space of poses of the models to find one that best matches the cluster [12]. Example matches for a scene are illustrated in Fig. 2a.

As shown, multiple instances of the same object type are present (increasing association difficulty), objects may be partially or fully occluded from a single view-point (cyan patches are occluded regions), object types can be confused (the white L-shaped block on the left), and pose estimates are noisy (the orange box in the

[3]For simplicity, we assume that the error covariance is axis-aligned and use an independent normal-gamma prior for each dimension, but it is straightforward to extend to general covariances.

center). Aggregation of object detections across different viewpoints and solving the subsequent data association issues, as depicted in Fig. 2b, was therefore essential.

For our scenarios, objects of 4 distinct types were placed on a table. A robot moved around the table in a circular fashion, obtaining 20–30 views in the process. We constructed 12 scenes of varying object and occlusion density to test our approaches; results for 4 representative scenarios are described in the next section.

7 Results

Qualitative results for 4 representative scenarios are shown in Fig. 3. Images from above are for comparison convenience only; the camera's viewing height is much closer to the table height, as shown in Fig. 2a, so in each view only a subset of objects is observable. We compare three approaches: multiple hypothesis tracking (**MHTF** from Sect. 3), generic DPMM clustering (**DPMM** from Sect. 4), and the factored approximation to DPMM-FullView (**DPMM-Factored** from Sect. 5.2). In Fig. 3, the most likely hypothesis is shown for MHTF, and the maximum a posteriori (MAP) sample (out of 100) is shown for the clustering-based approaches.

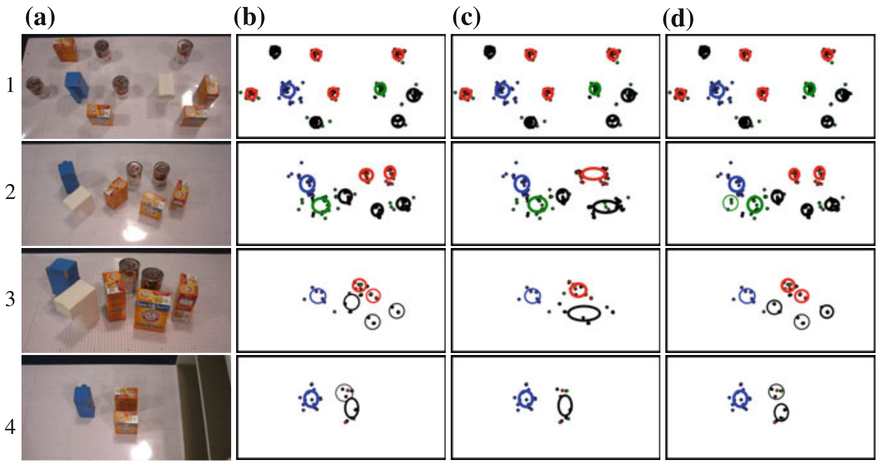


Fig. 3 Qualitative results for three world-model estimation approaches in four scenarios. The bird's-eye view of the scenes is for comparison convenience only; the actual viewing height is much closer to the table. The most likely hypothesis is shown for MHTF, and the maximum a posteriori sample is shown for the clustering-based approaches. Each small colored dot is a semantic (object type-and-pose) detection. Each target / cluster is depicted by an ellipse, centered at the posterior mean location. Ellipse axis lengths are proportional to the standard deviation in their respective dimensions. Ellipses are color-coded by the most likely posterior object type: *red* = red soup can, *black* = orange baking soda box, *green* = white L-shaped block, *blue* = blue rectangular cup. Line thickness is proportional to cluster size. See text in Sect. 7 for qualitative comparisons. **a** Scene from above. **b** MHTF. **c** DPMM. **d** DPMM-Factored

All approaches work well for scenario 1, where objects are spaced far apart. As objects of similar type are placed near each other, **DPMM** tends to combine clusters since it ignores the OMPO assumption. This is most apparent in scenario 3, where two soup cans (red) and three soda boxes (black) are combined into large clusters. By reconsidering the OMPO assumption, **DPMM-Factored** performs significantly better and is on par qualitatively with the **MHTF**, except for an extra cluster (bottom left, green) in scenario 2. In this case, the measurements corresponding to the white L-shaped object are dispersed, causing the shown extra-cluster error to be likely. Examining more samples reveals that a significant proportion (31 %) do not have the extra cluster; they just happen not to be MAP samples. This means that the estimator has significant uncertainty as to whether or not the extra object exists. Although in this case the **DPMM-Factored** MAP sample is wrong, it highlights a feature of our approach. Consider a task, e.g., grasping, that requires an accurate estimate of this object's neighborhood. Given the high uncertainty in the samples, the robot should decide to gather more observations of the region instead of operating based on the incorrect MAP sample. In contrast, the **MHTF** is over 90 % certain of its estimate because most other possibilities have been pruned. Although **MHTF** would have been less certain as well if all hypotheses were retained during filtering, the necessary aggressive pruning tends to make **MHTF** overconfident in its estimates.

Scenario 4 highlights another related difference between the tracking filter and batch approaches. Here two closely-arranged orange boxes are placed near a shelf, such that from most views at most one of the two boxes can be seen. Only in the final views of the sequence can both be seen (imagine a perspective from the bottom-left corner of the image). Due to the proximity of the boxes, and the fact that at most one was visible in the early views, **MHTF** eventually pruned all the then-unlikely hypotheses positing that measurements came from two objects. When finally both are seen together, although a hypothesis with two orange boxes resurfaces, it is too late: the remaining association hypotheses already associate all previous measurements of the boxes to the same target, in turn giving an inaccurate location estimate. In contrast, **DPMM-Factored** re-examines previous associations (in the next sampling iteration) after the two boxes are seen together, and can correct such errors. One way to consider this difference is that **DPMM-Factored** is a batch algorithm, whereas **MHTF** is simply a forward filter and does not have this capability.

Quantitative metrics are given in Table 1, averaged over the association hypotheses for **MHTF** and over 100 samples (after discarding burn-in) for **DPMM** and **DPMM-Factored**. To evaluate predicted targets and clusters against our manually-collected ground truth, for each ground truth object, the closest cluster within a 5 cm radius is considered to be the estimate of the object. If no such cluster exists, then the object is considered missed; all predicted clusters not assigned to objects at the end of the process are considered spurious. We also compare against a baseline approach, **Raw**, that does not perform any data association. It uses the object types and poses perceived in each view directly as a separate prediction of the objects present within the visible field of view. The metrics in the table are evaluated for each view's prediction, and the **Raw** table rows show the average value over all views. The location and type

Table 1 Average accuracy metrics and computation wall times for the four Fig. 3 scenarios

Metric →	Error (cm) in location estimate				% most likely type is correct				Num. missed objects (FNs)				Num. spurious clusters (FPs)				Computation wall time (s)			
Scenario	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Raw	2.5	2.7	1.9	2.1	98	93	67	56	2.0	3.7	5.4	2.0	0.8	1.3	0.3	0.7	N/A			
MHTF	2.1	2.8	**1.5**	2.6	**100**	**100**	**97**	**100**	**0.0**	**0.0**	**0.9**	0.6	0.6	**0.7**	**0.1**	0.6	492	263	833	3.0
DPMM	**1.8**	2.7	3.2	2.6	95	95	91	92	2.8	4.9	4.1	0.7	1.3	2.1	0.5	**0.1**	89	23	4	2.6
Factored	2.0	**2.6**	**1.5**	**2.0**	95	95	94	94	0.5	0.4	1.8	**0.2**	**0.2**	1.6	0.3	**0.1**	146	83	13	4.3

"**DPMM-Factored**" is denoted "Factored" below in the final row due to space constraints. Wall times are computed on a single core of an 2.66 GHz Intel Core i7 processor, using implementations in Python. Times are not provided for **Raw** since no processing on the measurements is required metrics are only computed for clusters assigned to detected objects, i.e., the clusters whose number is being averaged in the third metric.

The need for aggregating measurements across views is exemplified by **Raw**'s tendency to miss objects or confuse their types within single views. **DPMM** overcomes the latter issue by clustering across views, but still misses many objects because it ignores the OMPO assumption and agglomerates nearby similar objects. **DPMM-Factored** approximately respects this constraint and performs significantly better, missing few objects while maintaining accuracy in the posterior type-and-pose estimates. Although quantitatively it is slightly behind **MHTF**, this extra improvement comes at a several-factor computational expense, and potentially introduces filtering-related overconfidence issues mentioned earlier.

8 Related Work

Cox and Leonard [7] first considered data association for world modeling, using an MHT approach as well, but for low-level sonar features. The motion correspondence problem, which is similar to ours, has likewise been studied by many (e.g., [8, 9]), but typically again using low-level geometric and visual features only. Perhaps most similar to our problem is the recent work of Elfring et al. [11], which considers attribute-based anchoring and semantic world modeling with an MHT approach.

To our knowledge, our application of clustering to semantic world modeling is novel. More generally, sampling-based approaches have been applied to data association [9, 16], and may be applicable to approximate our DPMM-FullView model.

The important role of objects in spatial representations and semantic mapping was explored by Ranganathan and Dellaert [17], although their focus was on place modeling and recognition. Anati et al. [1] have also used the notion of objects for robot localization, but did not explicitly estimate their poses.

Object recognition and pose estimation has received widespread attention from the computer vision and robotics communities. Hager and Wegbreit [13] provide a good review as well as a unique approach. For pose estimation from multiple viewpoints, active perception has also been popular recently (e.g., [3, 10]). Our work differs in that we place no assumptions on the choice of camera poses, and we focus on data association issues. Moreover, we emphasize that object type-and-pose estimation was only chosen as a concrete and familiar proof of concept application, and our framework is applicable to many other semantic attributes and tasks.

9 Discussion

We have presented several clustering-based data association approaches for estimating semantic world models. We use Dirichlet process mixture models (DPMM) as our underlying framework. However, DPMMs perform poorly in their generic form because they ignore a crucial view-level constraint. Two improvements were therefore developed by incorporating the constraint exactly and approximately respectively. In preliminary experiments based on tabletop object type-and-pose estimation, the latter approach (**DPMM-Factored**) achieved performance comparable to an existing tracking-based approach using a fraction of the computation time.

As discussed in the introduction, semantic world models are useful in many object-centric tasks, involving a diverse set of attributes. We are currently exploring applications involving attributes beyond object type and pose. To be truly applicable, world models must also cope with objects moving over extended periods of time. Since the presented sampling procedure for inference iterates through all views, it is clearly impractical to apply it to the entirety of the robot's observation history. Instead, a hybrid approach combining the benefits of filtering and batch data association is desirable. Extending our framework to handle temporal dynamics while maintaining tractability over long horizons is the subject of future work.

Acknowledgments This work was supported in part by the NSF under Grant No. 1117325. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. We also gratefully acknowledge support from ONR MURI grant N00014-09-1-1051, from AFOSR grant FA2386-10-1-4135, and from the Singapore Ministry of Education under a grant to the Singapore-MIT International Design Center.

Appendix: Posterior and Predictive Distributions for a Single Light

In this appendix, we verify the claim from Sect. 2 that finding the posterior and predictive distributions on color and location for a single light is straightforward, given that we know which observations were generated by that light. Let $\{(o, x)\}$ denote the set of light color-location detections that correspond to a light with unknown parameters (c, l). Color and location measurements are assumed to be independent given (c, l) and will be considered separately. We assume a known discrete prior distribution $\pi \in \Delta^{(T-1)}$ on colors, reflecting their relative prevalence. Using the color noise model (Eq. 1), the posterior and predictive distributions on c are:

$$\mathbb{P}(c \mid \{o\}) \propto \left[\prod_{o} \phi_{o}^{c}\right] \times \pi_c; \quad \mathbb{P}\left(o' \mid \{o\}\right) = \sum_{c=1}^{T} \mathbb{P}\left(o' \middle| c\right) \, \mathbb{P}(c|\{o\}) = \sum_{c=1}^{T} \phi_{o'}^{c} \, \mathbb{P}(c \mid \{o\}) \,. \tag{15}$$

We can use this to find the light's probability of detection:

$$p_{\mathrm{D}} \triangleq 1 - \mathbb{P}\left(o' = 0 \,\middle|\, \{o\}\right) = 1 - \sum_{c=1}^{T} \phi_0^c \, \mathbb{P}\left(c \mid \{o\}\right). \tag{16}$$

Unlike the constant false positive rate p_{FP}, the detection (and false negative) rate is dependent on the light's color posterior.

For location measurements, we emphasize that both the mean μ *and* precision $\tau = \frac{1}{\sigma^2}$ of the Gaussian noise model is unknown. Modeling the variance as unknown allows us to attain a better representation of the location estimate's empirical uncertainty, and not naïvely assume that repeated measurements give a known fixed reduction in uncertainty each time. We use a standard conjugate prior, the distribution NormalGamma$(\mu, \tau; \lambda, \nu, \alpha, \beta)$.[4] It is well known (e.g., [5]) that after observing n observations with sample mean $\hat{\mu}$ and sample variance $\hat{s}^2$, the posterior is a normal-gamma distribution with parameters:

$$\begin{aligned} \lambda' = \lambda + n;\ \ \nu' = \frac{\lambda}{\lambda + n}\nu + \frac{n}{\lambda + n}\hat{\mu};\ \ \alpha' = \alpha + \frac{n}{2};\ \ \beta' \\ = \beta + \frac{1}{2}\left(n\hat{s}^2 + \frac{\lambda n}{\lambda + n}\left(\hat{\mu} - \nu\right)^2\right). \end{aligned} \tag{17}$$

The upshot of using a conjugate prior for location measurements is that the marginal likelihood of location observations has a closed-form expression. The posterior predictive distribution for the next location observation x' is obtained by integrating out the latent parameters μ, τ, and has the following expression:

$$\begin{aligned} \mathbb{P}\left(x' \,\middle|\, \{x\}\ ;\ \lambda, \nu, \alpha, \beta\right) &= \int_{(\mu,\tau)} \mathbb{P}\left(x \mid \mu, \tau\right) \mathbb{P}\left(\mu, \tau \mid \{x\}\right) \\ &= \frac{1}{\sqrt{2\pi}} \frac{\beta'^{\alpha'}}{\beta^{+\alpha^+}} \frac{\sqrt{\lambda'}}{\sqrt{\lambda^+}} \frac{\Gamma(\alpha^+)}{\Gamma(\alpha')}, \end{aligned} \tag{18}$$

where the hyperparameters with '′' superscripts are updated according to Eq. 17 using the empirical statistics of $\{x\}$ only (excluding x'), and the ones with '+' superscripts are likewise updated but including x'. The ratio in Eq. 18 assesses the fit of x' with the existing observations $\{x\}$ associated with the light.

[4]The typical interpretation of normal-gamma hyperparameters is that the mean is estimated from λ observations with mean ν, and the precision from 2α observations with mean ν and variance $\frac{\beta}{\alpha}$.

References

1. Anati, R., Scaramuzza, D., Derpanis, K., Daniilidis, K.: Robot localization using soft object detection. In: IEEE International Conference Robotics and Automation (2012)
2. Antoniak, C.: Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. Ann. Stat. **2**(6), 1152–1174 (1974)
3. Atanasov, N., Sankaran, B., Ny, J.L., Koletschka, T., Pappas, G., Daniilidis, K.: Hypothesis testing framework for active object detection. In: International Conference Robotics and Automation (2013)
4. Bar-Shalom, Y., Fortmann, T.: Tracking and Data Association. Academic Press, New York (1988)
5. Bernardo, J., Smith, A.: Bayesian Theory. John Wiley, New York (1994)
6. Cox, I., Hingorani, S.: An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. IEEE Trans. Pattern Anal. Mach. Intell. **18**(2), 138–150 (1996)
7. Cox, I., Leonard, J.: Modeling a dynamic environment using a Bayesian multiple hypothesis approach. AI J. **66**(2), 311–344 (1994)
8. Cox, I.J.: A review of statistical data association techniques for motion correspondence. Int. J. Comput. Vis. **10**(1), 53–66 (1993)
9. Dellaert, F., Seitz, S., Thorpe, C., Thrun, S.: EM, MCMC, and chain flipping for structure from motion with unknown correspondence. Mach. Learn. **50**(1–2), 45–71 (2003)
10. Eidenberger, R., Scharinger, J.: Active perception and scene modeling by planning with probabilistic 6D object poses. In: IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (2010)
11. Elfring, J., van den Dries, S., van de Molengraft, M., Steinbuch, M.: Semantic world modeling using probabilistic multiple hypothesis anchoring. Robot. Auton. Syst. **61**(2), 95–105 (2013)
12. Glover, J., Popovic, S.: Bingham Procrustean alignment for object detection in clutter. In: IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (2013)
13. Hager, G., Wegbreit, B.: Scene parsing using a prior world model. Int. J. Robot. Res. **30**(12), 1477–1507 (2011)
14. Kurien, T.: Issues in the design of practical multitarget tracking algorithms. In: Y. Bar-Shalom (ed.) Multitarget-Multisensor Tracking: Advanced Applications, pp. 43–84. Artech House (1990)
15. Neal, R.: Markov chain sampling methods for Dirichlet process mixture models. J. Comput. Graph. Stat. **9**(2), 249–265 (2000)
16. Oh, S., Russell, S., Sastry, S.: Markov chain Monte Carlo data association for multi-target tracking. IEEE Trans. Autom. Control **54**(3), 481–497 (2009)
17. Ranganathan, A., Dellaert, F.: Semantic modeling of places using objects. In: Robotics: Science and Systems (2007)
18. Reid, D.: An algorithm for tracking multiple targets. IEEE Trans. Autom. Control **24**(6), 843–854 (1979)
19. Sethuraman, J.: A constructive definition of Dirichlet priors. Stat. Sin. **4**, 639–650 (1994)

Driven Learning for Driving: How Introspection Improves Semantic Mapping

Rudolph Triebel, Hugo Grimmett, Rohan Paul and Ingmar Posner

Abstract This paper explores the suitability of commonly employed classification methods to action-selection tasks in robotics, and argues that a classifier's *introspective* capacity is a vital but as yet largely under-appreciated attribute. As illustration we propose an active learning framework for semantic mapping in mobile robotics and demonstrate it in the context of autonomous driving. In this framework, data are selected for label disambiguation by a human supervisor using uncertainty sampling. Intuitively, an introspective classification framework—i.e. one which moderates its predictions by an estimate of how well it is placed to make a call in a particular situation—is particularly well suited to this task. To achieve an efficient implementation we extend the notion of introspection to a particular sparse Gaussian Process Classifier, the Informative Vector Machine (IVM). Furthermore, we leverage the information-theoretic nature of the IVM to formulate a principled mechanism for forgetting stale data, thereby bounding memory use and resulting in a truly life-long learning system. Our evaluation on a publicly available dataset shows that an introspective active learner asks more informative questions compared to a more traditional non-introspective approach like a Support Vector Machine (SVM) and in so doing, outperforms the SVM in terms of learning rate while retaining efficiency for practical use.

R. Triebel (✉) · H. Grimmett · R. Paul · I. Posner
Mobile Robotics Group, Oxford University, Oxford, UK
e-mail: rudolph.triebel@in.tum.de

H. Grimmett
e-mail: hugo@robots.ox.ac.uk

R. Paul
e-mail: rohanp@robots.ox.ac.uk

I. Posner
e-mail: hip@robots.ox.ac.uk

R. Triebel
Computer Vision Group, Technical University of Munich, Munich, Germany

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_26

1 Introduction

In answering the question ‘where am I?’ roboticists have gone to great lengths to model, manage and, indeed, exploit uncertainty. This, however, is not as yet the case when it comes to asking ‘what is this?’. As we aspire to robust, long-term autonomous operation our systems have to contend with vast amounts of continually evolving, non-i.i.d. data from which information needs to be assimilated. This presents a challenge and an opportunity particularly to the robotics community as here the real cost of failure can be significant. We believe that *realistic* estimates of uncertainty are pivotal to achieving robust and efficient decision making in robotics. In particular, classification as a precursor to *action-selection* seems to be largely disregarded by the community.

We frame our argument in the context of offline semantic mapping. Significant progress in autonomous driving in recent years has inspired a view that successful autonomous operation in complex, dynamic environments critically depends on a-priori available semantic maps representing ostensibly permanent aspects of the environment such as lane markings, traffic light positions and road sign information (see, for example, [3, 22]). Owing to their safety-critical nature, these maps are typically created manually for particular routes [5]. This is, of course, an expensive process which scales badly with the number of routes for which autonomous operation is to be provided. Much, therefore, can be gained by reducing human involvement in this process and thus providing a robust and scalable solution.

A prominent approach to tackling such a challenge is that of *active learning*, where classification results are iteratively refined by asking a human supervisor for ground-truth labels in ambiguous cases and incorporating the added information into classifier training. To the best of our knowledge this paper is the first in robotics to present an efficient and scalable active learning framework for the task of offline semantic mapping. Crucially, however, our work is also set apart from the vast majority of the related works in active learning by the unusual stance we take with regards to uncertainty estimates in the system. Commonly, active learning relies on selecting data for human labelling using a variant of *uncertainty sampling*, by which data are selected according to how confident a classifier is in individual predictions (see, for example, [17]).

However, Grimmett et al. [7] show that several of the classification frameworks commonly used in robotics are unrealistically overconfident in their assessment of class membership. To characterise this attribute, the authors introduce the notion of the *introspective capacity* of a classification framework: the ability to estimate a classification confidence which realistically reflects how qualified the classifier is to make a particular class decision in each individual test instance. In this paper we show that *introspective classification* harbours significant benefits for active learning as compared to more traditional, non-introspective approaches. In particular, our contributions are

- the application of an active learning framework to semantic mapping in robotics,
- the application of the notion of introspection to the Informative Vector Machine (IVM) [10] as an efficient extension to [7],
- the application of the IVM specifically to achieve *introspective* active learning, which is demonstrated to lead to more effective information extraction over more traditional approaches, and
- the introduction of a principled mechanism for the IVM to *forget* less important data to provide for scalable, life-long active learning on a mobile robot.

The work presented here first appeared as a workshop paper by the same authors [21]. However, here we offer a more detailed treatment as well as the following significant extensions:

- the introspective capacity of the IVM is established, including the effects of varying the sparsity factor,
- qualitative results are included of when the IVM is confident (correctly and incorrectly) in its classifications, and
- timing information is provided regarding the training of an IVM.

We apply our framework to the detection of traffic lights in a real, third-party vision dataset and demonstrate iteratively improved semantic mapping, which makes

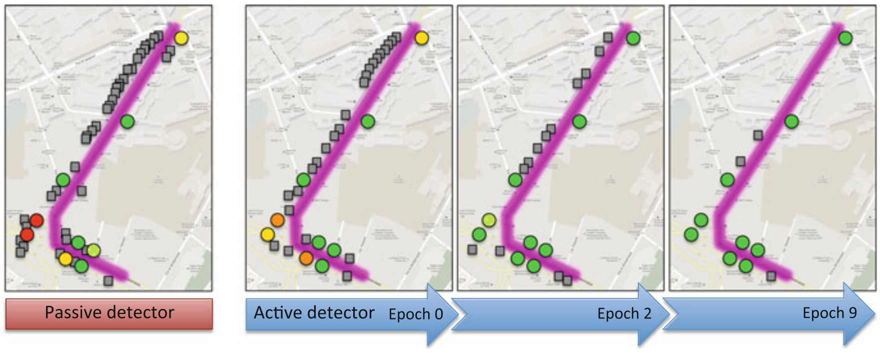


Fig. 1 Active learning in a semantic mapping context. This figure shows semantic maps indicating the positions of traffic lights along a street in Paris. *Circles* denote the locations of ground-truth traffic lights. The *shading* encodes the correctness of the classification output as provided by a probabilistic classifier: *red* denotes a recall of 0 (no detections), and *green* denotes a recall of 1 for that particular traffic light (all views of that object correctly detected). False positives are shown as *grey squares*. From *left to right*, we first see a typical passive detector, followed by our active-learning framework at epochs 0, 2, and 9 respectively. Note that in the active learning setting the shading of the *circles* progresses from *red to green* as a greater proportion of traffic lights are correctly detected with increasing confidence. Similarly the number of false positives reduces dramatically. By epoch 2 the active learning framework already outperforms the passive detector. In this paper we show that our formulation of an *introspective* active learning approach provides for more efficient information extraction—and thus a higher learning rate—over conventional active learning approaches

efficient use of available label information. A typical qualitative example of our system output is shown in Fig. 1.

2 Related Works

Active learning is an established and vibrant field of research spanning a significant number of application domains. Consequently, a variety of methods have been proposed for selecting informative measurements for labelling and/or for incrementally training a learning algorithm. For example, Freund et al. [6] propose disagreement among a committee of classifiers as a criterion for active data selection. McCullum and Nigam [12] apply this to text classification using high label inconsistency as a query criterion coupled with expectation maximisation (EM) for online learning. More recently, Joshi et al. [8] address multi-class image classification using SVMs and propose criteria based on entropy and best-versus-second-best (BvSB) measures based on the hyperplane-margin for determining uncertain points. Tong and Koller [19] pick unlabelled data for query based on minimising the version space within a margin-based SVM formulation. Kapoor et al. [9] propose an active learning system for object categorization using a GP classifier where data points possessing large uncertainty (using posterior mean and variance) are queried for labels and used to improve classification.

Within the robotics community, active learning and directed information acquisition has received attention in recognition, planning and mapping tasks. For example, Dima et al. [4] present unlabelled data filtering for outdoor terrain classification tasks with the aim of reducing the amount of training data to be human-labelled. The approach relies on kernel density estimation over unlabelled data and estimating a "surprise" score for image patches, hence only querying the least likely samples given the density estimate for human labelling. In [13] the authors present a learning approach for continually improving place recognition performance by actively learning an appearance model of a robot's operating environment. The method uses probabilistic topic models and a measure of perplexity to identify least explained images which further drives retrieval of thematically linked samples leading to an improved workspace representation. Recent work by Tellex et al. [18] explores active information gathering for human-robot dialog. The authors formulate an information-theoretic strategy for asking clarifying questions to disambiguate the robot's belief over the mapping between phrases and aspects of the workspace.

While, to the best of our knowledge, this is the first work in robotics applying active learning to a semantic mapping task, our work is also set apart significantly from prior art in active learning in that we introduce and demonstrate the benefits of efficient *introspective* active learning. In this respect, the work most closely related to ours is that of [9] above, in which an inherently introspective classifier is used but its use is not motivated by its introspective qualities.

3 Introspective Classification

The introspective capacity of a classifier characterises its ability to *realistically* estimate the uncertainty in its predictions. Grimmett et al. [7] define the introspective capacity as a classifier's ability to moderate its output by an appropriate measure as to how 'qualified' it is to make a call given its own prior experience, usually in the form of training data. The intuition is that test data, which are in some form 'similar' to that seen in training, are classified with higher certainty than data which are more dissimilar. This points towards non-parametric approaches potentially being more introspective than parametric ones, as all the training data are available for inference in the former, whereas inference in the latter is based on parametric models learned from the data. Grimmett et al. [7] investigated several commonly used classification frameworks providing probabilistic output and found that a Gaussian Process classifier (GPC) [15] indeed is significantly more introspective than, for example, the more commonly used Support Vector Machine (see, for example, [1]) with a probabilistic calibration (such as, for example, provided by Platt et al. [14]).

In [7], this quality is attributed to a GPC's Bayesian treatment of predictive variance. Consider a set of training data $\{X, \mathbf{y}\}$, where $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_{|X|}\}$ denotes the set of feature vectors and $\mathbf{y}$ denotes the set of corresponding class labels. Probabilistic predictions for a test point, $\mathbf{x}_*$, are obtained in two steps. First, the distribution over the latent variable corresponding to the test input is obtained by

$$p(f_* \mid X, \mathbf{y}, \mathbf{x}_*) = \int p(f_* \mid X, \mathbf{x}_*, f)p(f \mid X, \mathbf{y})df, \tag{1}$$

where $p(f \mid X, \mathbf{y})$ is the posterior distribution over latent variables. This is followed by applying a sigmoid function $\sigma(\cdot)$, which in our implementation is the cumulative Gaussian, and *marginalising* over the latent f_* to yield the class likelihood $p(y_* \mid X, \mathbf{y}, \mathbf{x}_*)$ as

$$p(y_* \mid X, \mathbf{y}, \mathbf{x}_*) = \int \sigma(f_*)p(f_* \mid X, \mathbf{y}, \mathbf{x}_*)df_*. \tag{2}$$

It is this *marginalisation* over all models induced by the training set, as opposed to relying on a single *minimisation*-based estimate, which accounts for a more accurate estimate of the inherent uncertainty in class distribution, and therefore endows GP classification with a high introspective capacity.

3.1 Efficiency by Sparsification

A key drawback of a GPC is its significant computational demand in terms of memory and run time. This is due to the fact that the GPC maintains a mean $\boldsymbol{\mu}$, as well as a covariance matrix Σ, which is computed from a kernel function and has size $|\mathbf{y}|^2$. A number of sparsification methods have been proposed in order to mitigate this computational burden. For efficiency, in this work we adopt one such sparsification method: the Informative Vector Machine (IVM) [10]. The main idea of this algorithm is to only use a subset of the training points denoted the *active set*, $\mathcal{I}$, from which an approximation $q(f \mid X, \mathbf{y}) = \mathcal{N}(f \mid \boldsymbol{\mu}, \Sigma)$ of the posterior distribution $p(f \mid X, \mathbf{y})$ is computed. The IVM algorithm computes $\boldsymbol{\mu}$ and Σ incrementally, and at every iteration j selects the training point $(\mathbf{x}_k, y_k)$ which maximizes the entropy difference ΔH_{jk} between q_{j-1} and q_j for inclusion into the active set. Because q is Gaussian, ΔH_{jk} can be computed by

$$\Delta H_{jk} = -\frac{1}{2}\log|\Sigma_{jk}| + \frac{1}{2}\log|\Sigma_{j-1}|. \tag{3}$$

The details of the implementation can be found in Lawrence et al. [11]. The algorithm stops when the active set has reached a desired size. In our implementation, we choose this size to be a fixed fraction γ of the training set q.

To find the kernel hyper-parameters $\boldsymbol{\theta}$ of an IVM, two steps are iterated a given number of times: the estimation of $\mathcal{I}$ given $\boldsymbol{\theta}$, and minimising the marginal likelihood $q(\mathbf{y} \mid X)$ given $\mathcal{I}$. Although there are no convergence guarantees, in practice already a small number of iterations are sufficient to find good kernel hyper-parameters.

Importantly for our work, since inference with the IVM is similar to that with a GPC, the IVM retains the model averaging described in Eq. (2). We argue therefore, that the IVM provides a significant and well-established improvement in processing speed over a GPC while maintaining its introspective properties (see Sects. 5 and 5.4 for details).

4 Scalable Active Learning: Drive, Ask, Improve

The power of an active learning framework lies in its ability to select a suitable training set in an application-oriented way. It thus inherently allows the system to adapt naturally to the non-stationarity of the data often encountered in long-term robotics applications. The active learning framework considered here is a supervised learning process by which a human operator provides class labels for machine-selected test data, which are then fed back into classifier training to improve the classification result of the next round. We examine performance over successive *epochs*, which each consist of (re-)training, classification, and user-feedback. The implementation of a scalable active learning framework requires two problems to be addressed: firstly, a subset of test data has to be selected for re-training such that classification performance increases in the next epoch. Secondly, measures have to be taken that guarantee that the training set is bounded in size, since otherwise the algorithm will sooner or later exhaust the resources of a finite-memory, real robotic system. We compare this active learning approach with a more conventional "passive" alternative, that is, training a classifier once without any subsequent human-feedback improvement.

We now outline the specific active learning algorithm employed in this work, before providing details of both our data selection strategy and our approach to forgetting (bounding the training set size).

4.1 The Active Learning Algorithm

Algorithm 1 describes our active learning framework which, for reasons given in Sect. 3, uses an IVM as the underlying classifier. It requires five different input parameters: the initial hyper-parameters $\boldsymbol{\theta}_0$ used for training the IVM, the fraction γ of training points that are used for sparsification, the batch size b, the normalised entropy (NE) threshold ϑ that a test point needs to exceed to be considered for retraining, and the maximum number of questions r that the algorithm may ask. The last is intended to minimise nuisance to a human operator due to being asked too many questions. The sub-routines in the algorithm are explained as follows.

Algorithm 1: Active Learning with an IVM

```
Data: training data $\mathcal{D} = (X, \mathbf{y})$, stream of test data $X^*$
Input: initial kernel parameters $\boldsymbol{\theta}_0$, batch size $b$, active set size fraction $\gamma$, minimal
       retraining score $\vartheta$, maximum number of questions $r$
Output: stream of output labels $\mathbf{y}^*$
$i \leftarrow 0$
while $X^* \neq \emptyset$ do
    $(\boldsymbol{\theta}_{i+1}, \mathcal{I}_{i+1}) \leftarrow$ TrainIVM($X, \mathbf{y}, \gamma, \theta_0$)
    move next $b$ test points from $X^*$ into $X_i^*$
    $\mathcal{P} \leftarrow \emptyset$
    forall the $\mathbf{x}^* \in X_i^*$ do
        $z \leftarrow$ IVMPrediction($\mathcal{I}_{i+1}, \boldsymbol{\theta}_{i+1}, \mathbf{x}^*$)
        $s \leftarrow$ ComputeRetrainingScore($z$)
        if $s > \vartheta$ then $\mathcal{P} \leftarrow \mathcal{P} \cup \{(\mathbf{x}^*, s)\}$

    sort $\mathcal{P}$ by decreasing values of $s$
    $\mathcal{D}^+ \leftarrow \emptyset$
    for $j \leftarrow 1$ to MIN($r, |\mathcal{P}|$) do
        $(\mathbf{x}_j^+, s_j) \leftarrow$ element $j$ of $\mathcal{P}$
        $y_j^+ \leftarrow$ AskLabelFromUser ($\mathbf{x}_j^+$)
        $\mathcal{D}^+ \leftarrow \mathcal{D}^+ \cup (\mathbf{x}_j^+, y_j^+)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}^+$,  $i \leftarrow i + 1$
```

`TrainIVM` uses the current training set, the active set fraction γ, and the initial kernel parameters to find optimal kernel parameters $\boldsymbol{\theta}_{i+1}$ and an active set $\mathcal{I}_{i+1}$ as described in Sect. 3.1. Throughout this work we employ a squared exponential kernel (which is the same as the Radial Basis Function kernel) with additive white noise:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 e^{-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2l^2}} + \sigma_n^2 \delta_{ij}, \tag{4}$$

where δ_{ij} is the Kronecker delta, and $\boldsymbol{\theta} = \{\sigma_f^2, l, \sigma_n^2\}$ are the signal variance, the length scale, and the noise variance.

`IVMPrediction` returns an estimate of the probability z that the next test datum $\mathbf{x}^*$ has a particular class label, as given in Eq. (2). Based on this probability, the normalised entropy measure is then computed. The top ranked r test data exceeding the retraining threshold ϑ are labelled by the user and added to the training set for the next epoch.

4.2 Data Selection Strategy: What Questions to Ask?

The key element of an active learning algorithm is the strategy by which a new test point $\mathbf{x}^*$ is considered for re-training. In Algorithm 1, this is done in the sub-routine `ComputeRetrainingScore`. An intuitive and well-explored indicator of which data might be suitable for inclusion is the classification *uncertainty* associated with $\mathbf{x}^*$. To characterise the uncertainty of the classification from the given class prediction $z = p(y_* \mid X, \mathbf{y}, \mathbf{x}_*)$, we adopt the measure of *normalised entropy* $H(z)$, such that for the binary case,

$$H(z) = -z \cdot \log_2(z) - (1 - z) \cdot \log_2(1 - z), \tag{5}$$

where $H(z) \in [0, 1]$, with high values representing high uncertainty.

This, indeed is central to our work. While, in principle, any classification framework which provides a distribution over class labels as output can be used in our active learning framework, intuitively we expect those with more realistic estimates of these probabilities to be more effective for active learning. Thus, we expect more introspective classifiers to perform better in the sense that they will ask more informative questions, leading to a higher learning rate. In Sect. 5, we will show that this is indeed the case when comparing the proposed framework based on an IVM with one based on a more commonly used, probabilistically calibrated SVM.

4.3 Forgetting Uninformative Data to Bound Memory Use

The main problem with the active learning framework as we presented it so far is that in theory the training set can grow indefinitely, because there are no guarantees that the algorithm will stop asking new questions. This makes the algorithm less flexible, especially if the input data can not be guaranteed to be within certain locality bounds, for example in a life-long learning application. Therefore, and for run time efficiency, we bound the size of the training set by removing points from it when it exceeds a given target size n_t. To decide which points to remove, we leverage the information-theoretic instruments that the IVM already provides. After each training round, we keep the entropy differences given in Eq. (3) for all training points and sort them in increasing order. Those training data which correspond to the first $n_i - n_t$ values, where n_i is the current training set size, are then removed before training in the next epoch. Intuitively, this method discards the data that were least informative during the last training round. One caveat with this method is that it assumes independence between the training data, which is not generally given. For example, two data may both have small individual ΔH values, but when removing both of them the entropy could change significantly. In this work we acknowledge but do not explore this phenomenon. Instead, we note that in our experiments we did not observe a deterioration in classification performance when we applied our method for forgetting.

5 Experimental Results

In this section we investigate the performance of our introspective active learning approach in terms of learning rate, data selection strategy, classification performance and tractability. We compare and contrast our approach with one based on the much more commonly used SVM classifier (calibrated to provide probabilistic output). The task we set both learners is to detect traffic lights in a third-party image dataset. Specifically, we use the publicly available Traffic Lights Recognition (TLR) data set [16], which comprises 11,179 colour images taken at 25 Hz from a car driven through central Paris at speeds under 31 mph. It has ground-truth labels for traffic light positions and subtype labels 'green', 'orange', 'red', 'ambiguous' (though here we are only concerned with the detection of traffic lights, irrespective of their state). As recommended by the authors of the dataset, we disregard labels of type 'ambiguous' and exclude sections where the vehicle was stationary for long periods of time. We use data from the first 5,800 frames for training and the remainder for testing. We compute a template-based feature set inspired by Torralba et al. [20] which has a successful track record in the detection of traffic lights [7]. Each training or test window is represented by a feature vector of length 200.

When training the IVM we used an active set fraction γ of 0.2, which means that informative points will be added to the active set until its size is 20 % of the training set size. We use a Squared Exponential (SE) with white noise kernel. Training such a classifier takes approximately 1.5 s on a single 3.4 GHz core.

The SVMs used here are trained using *libsvm* [2], and use the isotropic Radial Basis Function (RBF) kernel, which is equivalent to the SE kernel used by the IVM. They are trained using 10-fold cross-validation on top of a grid-search over the parameters C (the penalty parameter for the error term) and γ (the inverse of the length scale for the isotropic RBF kernel), both in the space 2^k where $k = \{-7, -6, \ldots, +4\}$. Training takes approximately 10 min.

5.1 Does Introspection Improve Active Learning?

One of the central claims of this paper is that the use of an introspective classifier will lead to more informative questions being asked of the human expert. In order to test this claim we perform a cross-over experiment (see Fig. 2) which starts with both an IVM and an SVM are initially trained on the same data, 200 traffic lights (positive) and 200 background patches (negative). Then, 1,000 new data (with a class fraction of 1:1, the same as during training) are shown to both classifiers for testing. Each chooses up to 50 data points (providing their normalised entropies are over a threshold empirically set to be $\vartheta = 0.97$) to add to their own training set for the next round, resulting in *two* new and different training sets: the 'IVM set' and the 'SVM set'. A new IVM and SVM are now trained on *each* of the two new sets and evaluated on a further 1,000 new data points. This process thus gives rise to four classifiers: two IVMs trained on data selected by an IVM and a SVM respectively, and two equivalent SVMs. We compute precision and recall for all four classifiers. The results after 100 repetitions of this experiment are shown in Fig. 3. As expected, both the IVM and the SVM perform better when trained on the dataset chosen by the initial IVM, suggesting that the questions asked by the IVM tend to be more informative. An unpaired t-test shows this result to be significant to a level of over 95 %.

The overall effect of introspection in an active learning setting seems to be an increased learning rate, a claim which we support with the following active learning

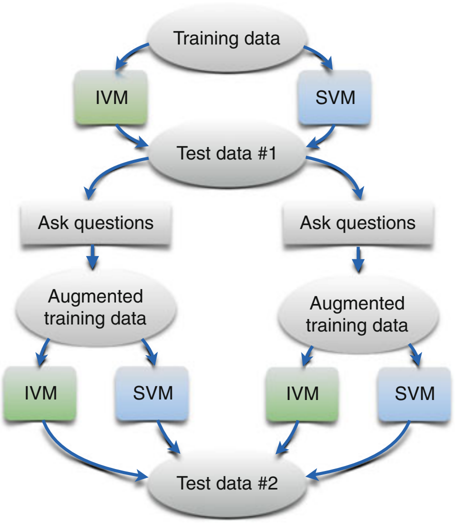


Fig. 2 Here we show the procedure for the cross-over experiment, designed to test whether one classifier chooses points which do not only benefit itself in the next round, but are consistently more useful for the other type of classifier as well. We compare an IVM and an SVM, and choose the test points with highest normalised entropy to be labelled to augment the original training set

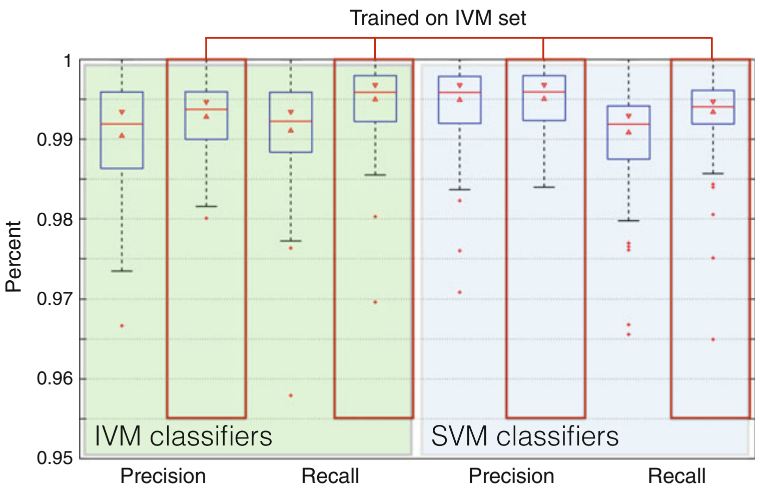


Fig. 3 Data selected by the IVM lead to an improved learning rate in terms of precision and recall for both an IVM and SVM over those selected by the SVM. Results are shown for 100 experimental runs, and increases are significant to the 95 % level. See text and Fig. 2 for more details

experiment, performed over 11 epochs. As described in Sect. 4, our active learning algorithm is retrained after having seen a batch of test points, as opposed to running the training algorithm after every new datum encountered. Every epoch consists of a training phase, a classification phase, and a feedback phase. At the very start of epoch 0, the classifiers are trained on 50 positive (traffic light) windows and 500 negative (background) windows extracted at random from the training frames. We choose this class fraction disparity to reflect the fact that in real data sets, negative examples are much more prevalent than positive examples. During each classification phase, the classifiers are then tested on a batch of 1,000 windows extracted from the test frames. The class fraction for these test windows is 1:10, the same as for training. Next, the 50 points with the highest normalised entropy (providing they are over $\vartheta = 0.97$) are added to the training set, ready for retraining at the start of the next epoch. Note that each classifier (IVM and SVM) makes its own choices regarding which points to add for the next epoch.

The results are shown in Fig. 4, where the IVM learner starts off with a worse f_1 measure at epoch 0 but has already exceeded the SVM by epoch 2, and is better (with non-overlapping 95 % confidence bounds) in the steady state from then onwards. The gradient of the plot in Fig. 4 is shown in Fig. 5, and shows that the rate of increase of f_1 measure (the learning rate) for the IVM is better than that of the SVM over the first few epochs, and then always at least as good subsequently.

Figure 4 further serves to justify empirically our choice of normalised entropy as a valid criterion for data selection, by comparing it to randomly selecting new training data. Intuitively, both methods should improve classification by virtue of the fact that they increase the training set size. However, the results indicate that for both

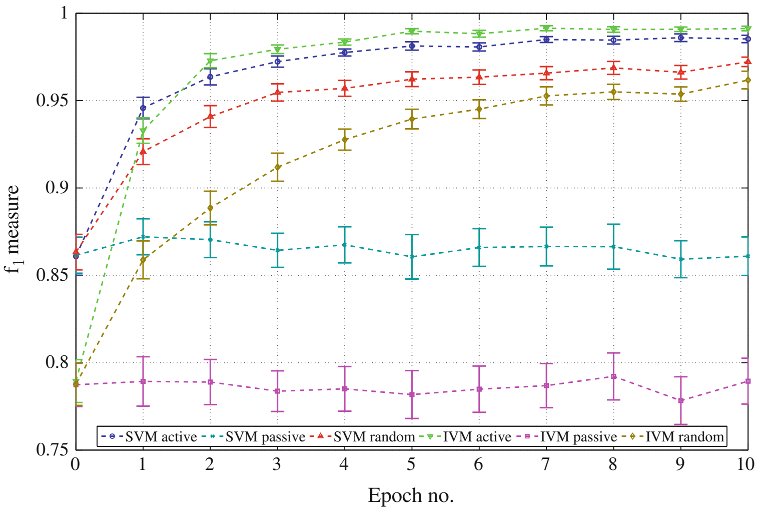
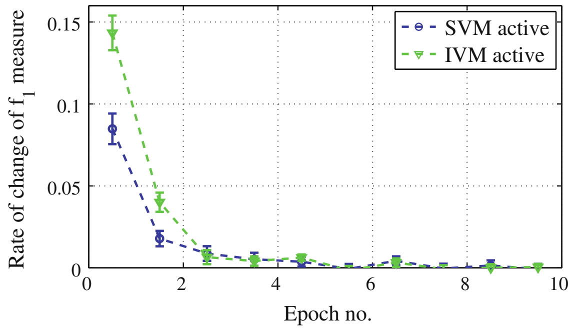


Fig. 4 Classification performance for both IVM and SVM variants as indicated by the f_1-measure after each epoch. Measurements are averaged over 100 runs. Error bars indicate the 95 % confidence region of the mean. The IVM using a normalised entropy-based data selection strategy (IVM-active) consistently outperforms all other active learning variants in terms of learning rate and final classification performance

Fig. 5 The gradient of the f_1 measure of the active learners from Fig. 4



the IVM and the SVM, using normalised entropy leads to more rapidly improving classification performance.

5.2 *Does Forgetting Affect the Performance?*

Our work aims to contribute an introspective active learning algorithm that is efficient in terms of computational effort and scalable with respect to its memory requirements. In this section we investigate the efficacy of the mechanism we have put in place to

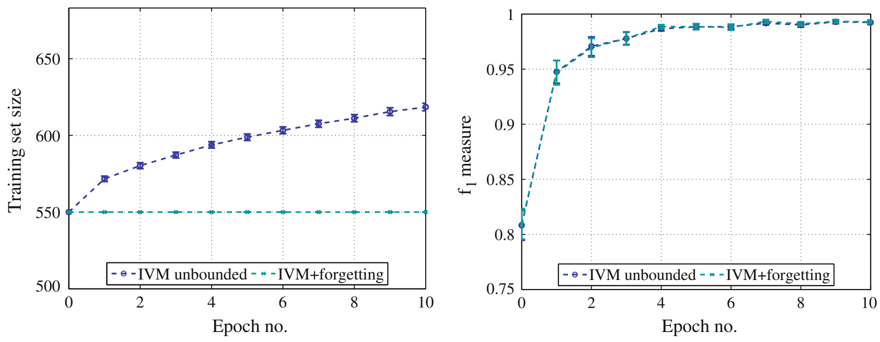


Fig. 6 *Forgetting* results in commensurate classification performance while successfully bounding the active set size of the classifier. Each datum represents the mean (and associated 95 % confidence interval) over 100 experimental runs. *Left* The evolution of the training set size. The IVM+forgetting learner has a target training set size $n_t = 550$, the initial training set size. *Right* Classification performance with and without *forgetting*. For corresponding SVM results, see Fig. 5

provide this tractability: forgetting. In experiments thus far, new training data were added in each epoch. The IVM active set size is a fixed proportion of the training set size, which has the benefit of increasing classification performance, but is detrimental to processing time. In the context of a life-long-learner, this is not a scalable solution.

We therefore elect to cap the size of the training set at $n_t = 550$ data, which makes the computational effort constant. This '*IVM with forgetting*' learner can add new data, but only by simultaneously discarding enough data to reduce the training set size to the target size n_t. Figure 6 (left) shows the training set size for the normal IVM with unbounded training set, and an IVM with forgetting, capped at 550 data (the initial training amount). Figure 6 (right) shows the corresponding classification performance as characterised by the f_1 measure. It indicates that in this scenario, the IVM with forgetting mechanism has the same performance as the unbounded IVM. We note that this is likely to be dataset dependent.

5.3 What Does the Active Learner Ask?

In Fig. 7 we show the 27 most certain and 27 least certain test cases for an IVM at epochs 0, 3, and 10, and whether they were correctly classified or not. Firstly, it is reassuring to confirm that the certain classifications are always correct. At epoch 0 we see that the confident classifications are all of the background class, almost entirely of fairly uniformly textured surfaces like tarmac, and that the unconfident classifications are all regarding traffic lights. As the learners gather more data, the traffic lights which at epoch 0 were uncertain, are now very confident at epoch 3. At epoch 10, the uncertain group are more balanced in terms of traffic lights and background, and we see that although there is a little more variation in terms of the

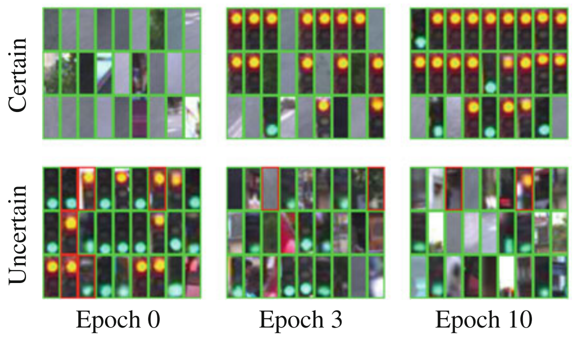


Fig. 7 The 27 most certain and 27 least uncertain test classifications of an IVM at epochs 0, 3, and 10 during the active learning experiment. A *green border* indicates a correct classification, and a *red border* indicates an incorrect classification

confident patches, they are very similar to the confidence classifications at epoch 3. This is consistent with the learning algorithm having reached an equilibrium after epoch 3 in Fig. 4.

5.4 The Effects of Sparsity

In [7] we showed that the GPC is more introspective than other more commonly used classification frameworks. In this paper we have argued the necessity of using a sparse formulation for the sake of computational complexity, however, it is necessary to ensure that the IVM is introspective in its own right. The useful characteristic of an introspective classifier is that it tends to be confident when it is making true predictions, and uncertain when it may be making false predictions. In addition, we would like to see whether the introspective quality changes with the active set size; intuitively, a truly introspective classifier will be more confident if it is exposed to more data, and vice versa.

Similarly to the approach in [7] we have plotted the cumulative true and false classifications against uncertainty in Fig. 8 for a single round of training and testing. In the legend, “IVM $\gamma = 0.4$” indicates an IVM with an active set fraction of 0.4, such that the active set contains 40 % of the training set. These particular IVMs have been trained on 550 data and tested on 11,000, with the ratio 1:10 positive:negative. There are several things to notice from the graph. Firstly, we can see that by looking at the curves for the IVMs with $\gamma = \{0.2, 0.4, 0.6, 0.8, 1.0\}$, indeed as we would hope, having a larger active set results in a more confident classifier; however it is interesting to see that there are diminishing returns: very little confidence is gained between an active set fraction of 0.6 and 1.0. Secondly and most importantly, *the IVM is introspective*: the incorrect classifications occur with *high* uncertainty, whereas the majority of the correct classifications occur with *low* uncertainty. Thirdly, we would expect that as the level of sparsity decreases, we approach the behaviour of the GPC, which is indeed what happens; the full GPC is commensurate with the IVMs with $\gamma = \{0.6, 0.8, 1.0\}$.

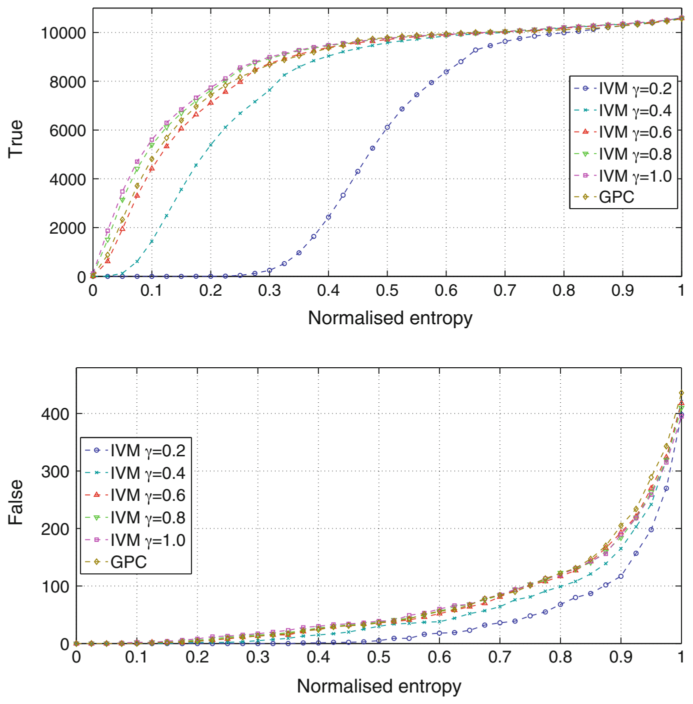


Fig. 8 The introspective capacity of the IVM. We show the number of true (*top*) and false (*bottom*) classifications (positive and negative classes together) which are made with a normalised entropy lower than a chosen value. For instance, if we were to threshold at NE $= 0.5$, we would have 6000 correct classifications with the IVM $\gamma = 0.2$ and <10 incorrect classifications

6 Conclusion

The contributions of this paper are three-fold: firstly, the notion of introspective classification introduced earlier shows promise in the context of active learning, where a reliable estimate of the classification uncertainty is required. We do this by showing an improvement in both classification performance and learning rate over a non-introspective classifier (Sect. 5.1). Secondly, an efficient version of the Gaussian Process Classifier, namely the Informative Vector Machine is used, which makes the approach particularly useful for robotics applications with large amounts of data. We show visual examples of where it is confused and where it is confident (Sect. 5.3), and use it to create the first offline semantic mapping algorithm via active learning. Finally, we present an information-theoretic solution to the problem of increasing memory requirements by forgetting the least informative data, which maintains a high classification performance in our experiments, but more extensive experimentation is required to confirm the success of this approach for the wider scope of mobile robotics applications.

Acknowledgments This work is funded under the European Community's Seventh Framework Programme (FP7/2007–2013) under Grant Agreement Number 269916 (V-CHARGE) and by the UK EPSRC Grant Number EP/J012017/1.

References

1. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Min. Knowl. Discov. **2**(2), 121–167 (1998)
2. Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. ACM Trans. Intell. Syst. Technol. (TIST) **2**(3), 27 (2011)
3. DARPA. Urban challenge route network definition file (RNDF) and mission data file (MDF) formats, March 2007
4. Dima, C., Hebert, M., Stentz. A.: Enabling learning from large datasets: applying active learning to mobile robotics. In: IEEE International Confreence on Robotics and Automation (ICRA), vol. 1, pp. 108–114. IEEE (2004)
5. Fairfield, N., Urmson, C.: Traffic light mapping and detection. In: IEEE International Conference on Robotics and Automation (ICRA) (2011)
6. Freund, Y., Seung, H.S., Shamir, E., Tishby, N.: Selective sampling using the query by committee algorithm. Mach. Learn. **28**(2), 133–168 (1997)
7. Grimmett, H., Paul, R., Triebel, R., Posner, I.: Knowing when we don't know: introspective classification for mission-critical decision making. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA) (2013)
8. Joshi, A.J., Porikli, F., Papanikolopoulos, N.: Multi-class active learning for image classification. In: Computer Vision and Pattern Recognition (CVPR), pp. 2372–2379 (2009)
9. Kapoor, A., Grauman, K., Urtasun, R., Darrell, T.: Gaussian processes for object categorization. Int. J. Comput. Vis. **88**(2), 169–188 (2010)
10. Lawrence, N.D., Seeger, M., Herbrich, R.: Fast sparse gaussian process methods: the informative vector machine. Adv. Neural Inf. Process. Syst. **15**, 609–616 (2002)
11. Lawrence, N.D., Platt, J.C., Jordan, M.I.: Extensions of the informative vector machine. In: Proceedings of the First international conference on Deterministic and Statistical Methods in Machine Learning, pp. 56–87. Springer (2005)
12. McCallum, A., Nigam, K.: Employing em in pool-based active learning for text classification. In: Proceedings of ICML-98, 15th International Conference on Machine Learning, pp. 350–358 (1998)
13. Paul, R., Newman, P.: Self help: Seeking out perplexing images for ever improving navigation. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 445–451. IEEE (2011)
14. Platt, J.: Probabilistic outputs for support vector machines and comparison to regularize likelihood methods. In: Smola, A.J., Bartlett, P., Schoelkopf, B., Schuurmans, D. (eds.) Advances in Large Margin Classifiers, pp. 61–74 (2000)
15. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)
16. Robotics Centre of Mines ParisTech. Traffic lights recognition (TLR) data set
17. Settles, B.: Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin-Madison (2010)
18. Tellex, S., Thaker, P., Deits, R., Kollar, T., Roy, N.: Toward information theoretic human-robot dialog. In: Robotics: Science and Systems (2012)
19. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. **2**, 45–66 (2002)
20. Torralba, A., Murphy, K.P., Freeman, W.T.: Sharing visual features for multiclass and multiview object detection. IEEE Trans. Pattern Anal. Mach. Intell., **29**(5):854–869 (2007). ISSN 0162–8828

21. Triebel, R., Grimmett, H., Paul, R., Posner, I.: Introspective active learning for scalable semantic mapping. In: Workshop. Robotics Science and Systems (RSS) (2013)
22. Urmson, C., et al.: Autonomous driving in urban environments: boss and the Urban Challenge. J. Field Robot., **25**(8):425–466 (2008). ISSN 1556–4967

RatSLAM: Using Models of Rodent Hippocampus for Robot Navigation and Beyond

Michael Milford, Adam Jacobson, Zetao Chen and Gordon Wyeth

Abstract We describe recent biologically-inspired mapping research incorporating brain-based multi-sensor fusion and calibration processes and a new multi-scale, homogeneous mapping framework. We also review the interdisciplinary approach to the development of the RatSLAM robot mapping and navigation system over the past decade and discuss the insights gained from combining pragmatic modelling of biological processes with attempts to close the loop back to biology. Our aim is to encourage the pursuit of truly interdisciplinary approaches to robotics research by providing successful case studies.

1 Introduction

The brain circuitry involved in encoding space in rodents has been extensively tested over the past thirty years, with an ever increasing body of knowledge about the components and wiring involved in navigation tasks. The learning and recall of spatial features is known to take place in and around the hippocampus of the rodent, where there is clear evidence of cells that encode the rodent's position and heading. RatSLAM [1–3] is a robotic navigation system based on current models of the rodent hippocampus, which has achieved several significant outcomes in vision-based Simultaneous Localization And Mapping (SLAM), including mapping of an entire suburb using only a low cost webcam [4, 5], and navigation continuously over a period of two weeks in a delivery robot experiment [6]. These results showed for the first time that a biologically inspired mapping system could compete with or surpass the performance of conventional probabilistic robot mapping systems. The RatSLAM system has recently been open-sourced and published [7].

We have also "closed the loop" back to the neuroscience underpinning the RatSLAM system. In our research, we took a pragmatic approach to modelling the neural mechanisms, and would engineer "better" solutions whenever the underlying

M. Milford (✉) · A. Jacobson · Z. Chen · G. Wyeth
School of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane, Australia
e-mail: michael.milford@qut.edu.au

biology did not appear to meet the robot's needs. However, some of the modifications necessary to make the models of hippocampus work effectively over long periods in large and ambiguous environments raised new questions for further biological study, including a potential neural mechanism for filtering uncertainty in navigation [8]. The research has also led to recent experiments demonstrating that vision-based navigation can be achieved at any time of day or night, during any weather, and in any season using sequences of visual images as small as 2 pixels in size [9–12]. Most recently we have led collaborative research with human- and animal-neuroscience labs leading to novel human-inspired vision-based place recognition algorithms that are starting to rival human capabilities at specific tasks [13, 14].

In this paper we describe two recent biologically-inspired areas of investigation building on the existing RatSLAM system. We first provide a brief but necessary overview of the core RatSLAM system. We then describe research mimicking the hypothesized sensory calibration processes in the rodent brain and present experiments demonstrating autonomous calibration of a place recognition system, a key requirement for mapping and navigation systems. Finally, we describe new research modelling the multi-scale, homogeneous mapping frameworks recently discovered in the rat brain and present results showing the place recognition performance benefits of such an approach. We conclude with a discussion of the key lessons learnt in more than a decade of pursing an interdisciplinary robotics-neuroscience research agenda.

2 RatSLAM

In this section we briefly describe the core RatSLAM algorithms upon which the new research presented here is based. RatSLAM is a SLAM system based on computational models of the navigational processes in the part of the mammalian brain called the *hippocampus*. The system consists of three major modules—the *pose cells, local view cells*, and *experience map*. Further technical details on RatSLAM can be found in [4, 6].

2.1 Pose Cells

The pose cells are a Continuous Attractor Network (CAN) of units connected by both excitatory and inhibitory connections, similar to a recently discovered class of navigation neurons found in many mammals called *grid cells* [15]. The network is configured in a three-dimensional prism (Fig. 1), with cells connected to nearby cells by excitatory connections, which wrap across all boundaries of the network. The dimensions of the cell array nominally correspond to the three-dimensional pose of a ground-based robot—x, y, and θ. The pose cell network dynamics are such that the stable state is a single cluster of activated units, referred to as an *activity packet* or

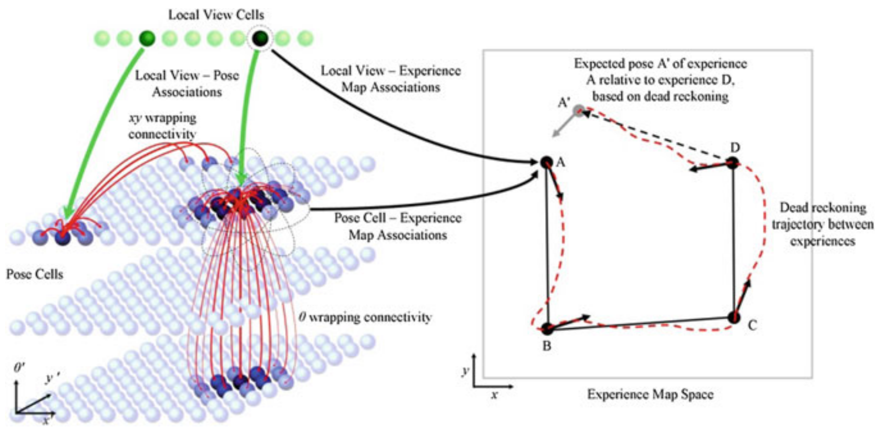

Local View Cells
Local View – Pose Associations
xy wrapping connectivity
Local View – Experience Map Associations
Pose Cell – Experience Map Associations
Pose Cells
θ wrapping connectivity
θ'
y'
x'
Expected pose A' of experience A relative to experience D, based on dead reckoning
A'
A
B
C
D
Dead reckoning trajectory between experiences
y
x
Experience Map Space

Fig. 1 The RatSLAM system, consisting of local view cells, pose cells and the experience map

energy packet. The centroid of this packet encodes the robot's best internal estimate of its current pose. Network dynamics are regulated by the internal connectivity as well as by input from the local view cells.

2.2 Local View Cells

The local view cells are an expandable array of cells or units. Novel scenes drive the creation of a new local view cell which is then associated with the raw sensory data (or an abstraction of that data) from that scene. In addition, an excitatory link is learnt (one shot learning) between that local view cell and the centroid of the dominant activity packet in the pose cells at that time. When that view is seen again by the robot, the local view cell is activated and injects activity into the pose cells via that excitatory link. Re-localization in the pose cell network occurs when a sufficiently long sequence of familiar visual scenes is experienced in the correct sequence, causing constant injection of activity into the pose cells resulting in the re-activation of the pose cells that were associated with that scene the first time.

2.3 Experience Map

Initially the representation of space provided by the pose cells corresponds well to the metric layout of the environment a robot is moving through. However, as odometric error accumulates and loop closure events occur, the space represented by the pose cells becomes discontinuous—adjacent cells in the network can represent physical places separated by great distances. Furthermore, the pose cells represent a finite area but the wrapping of the network edges means that in theory an infinite area can be mapped, which implies that some pose cells represent multiple physical places. The experience map is a graphical map that provides a unique estimate of the robot's pose by combining information from the pose cells and the local view cells. A new experience is created when the current activity state in the pose cells and local view cells is not closely matched by the state associated with any existing experiences. As the robot transitions between experiences, a link is formed from the previously active experience to the new experience. A graph relaxation algorithm runs continuously to evenly distribute odometric error throughout the graph, providing a map of the robot's environment which can readily be interpreted by a human.

3 Brain-Based Sensor Fusion and Calibration

Current state of the art robot mapping and navigation systems produce impressive performance under a narrow range of robot platform, sensor and environmental conditions. In contrast, animals such as rats produce "good enough" maps that enable them to function in an incredible range of situations and environments around the world. From only four days after birth, rat pups start to *learn* how to best sense, map and navigate in their environment [16, 17]. Rat pups have been seen to demonstrate particular movement behaviours such as pivoting that are theorized to help them calibrate their sensory stream. Furthermore, adult rats rapidly adapt to changes in their own sensing equipment or in their environment during their adult life [18]. It has even been shown that it is possible to integrate novel sensory devices into a rat brain and have the rats subsequently learn to utilise this novel input [19]. We investigated the feasibility of adopting a "sensor agnostic" approach to mapping and localization inspired by the adaptation capabilities of rats.

We describe a rat-inspired multi-sensor fusion and calibration system that assesses the usefulness of multiple sensor modalities based on their utility and coherence for place recognition both when a robot is first placed in an environment through calibration behaviors [20] and autonomously while moving [21], without knowledge as to the type of sensor. We demonstrate the system on a Pioneer robot in indoor and outdoor environments with large illumination changes.

3.1 Approach

Here we present our sensor-agnostic approach to multi-sensory calibration and online sensory evaluation. The system is algorithmic in nature; however it is loosely inspired by rodent behavioural and neural processes.

3.1.1 Sensor Pre-processing

Sensor data is pre-processed to enable agnostic evaluation of sensory information through a standardized format. All sensor data is normalized by dividing by the maximum possible sensor reading producing a value between (0, 1). Sensor data in the form of multi-dimensional arrays, such as images, are down-sampled and separated into a single line vector, for example, RGB images are converted to grayscale, down-sampled to 12×9 and separated into a single vector 108 elements long. Sensor pre-processing is applied to all sensor modalities producing a single vector for each sensor called a template.

3.1.2 Multi-sensor Fusion

Sensor data similarity is evaluated utilizing a Sum of Absolute Differences (SAD) comparison, in order to determine the similarity between the current template and all previously stored templates. The best template match to the current sensor template is the previously learnt template with the smallest difference score. We define a template as *familiar* if a previously learnt template has a difference score less than a predetermined recognition threshold, S_{thresh}. The current sensory template is defined as *novel* if the best template match difference score is greater than the recognition threshold. Furthermore, we define a technique for dynamically evaluating the utility and reliability of sensors as the robot moves through the environment. Sensor reliability is determined using two biologically inspired metrics, *spatial coherence* and *template expectation similarity.* These metrics are binary operators and evaluate the agreement between two sensory modalities. Each sensor is compared to each other sensor using these two metrics and combined to produce a single coherence score which is used to determine the utility of each sensor. Spatial coherence builds on the idea of using geometric information to validate place recognition and utilizes the experience map to determine the Euclidean distance between template matches. Two sensors are deemed to be spatially coherent if the Euclidean distance between the location matches is below a geometric threshold, g_{thresh}. Template expectation similarity determines the similarity between the current sensor data and a predicted sensor reading generated from another sensor. Sensors are deemed to be reliable if coherent with at least one other sensor or if no template match has been reported, otherwise the sensor is tagged as unreliable.

Sensor data is fused together through the implementation of "super templates", formed by concatenating each sensor template into a single vector. When comparing super templates, the component of the overall matching score corresponding to each sensor is normalized by the number of readings for the sensor to remove any effect of varying sensor vector sizes (Fig. 2).

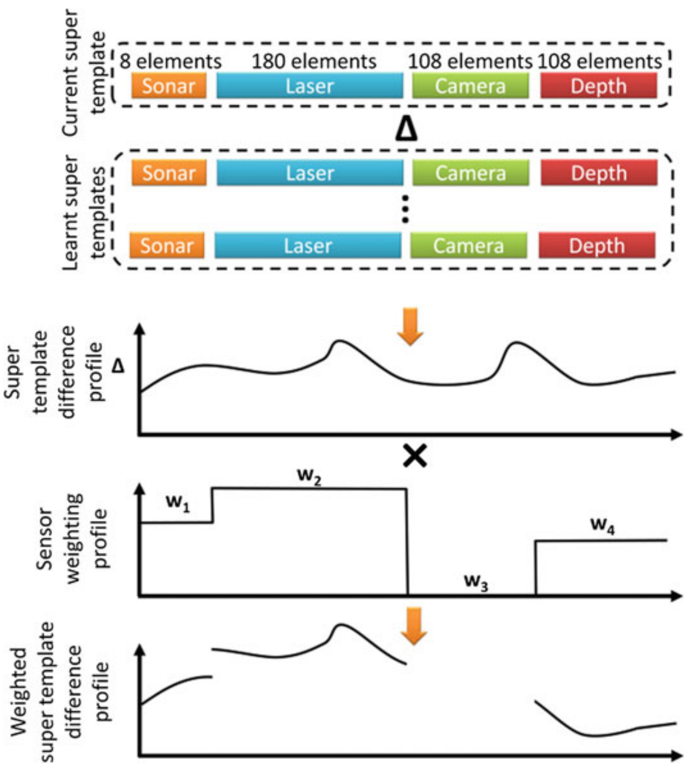


Fig. 2 Super templates are created by the concatenation of individual sensor data and compared to previously learnt super templates using a weighted SAD. Super templates allow the storage of sensory information for a particular scene, allowing all sensory data to be processed in a uniform manner

3.1.3 Movement-Driven Autonomous Calibration

Autonomous calibration of the place recognition processes for each sensor is achieved by mimicking the pivoting behavior of young rat pups when calibrating their sensors. The main requirement of a robot is that it is capable of safely performing two donuts within the operating environment and that the environment is primarily static for the calibration behaviors. The performance of two donut behaviors is required to allow the sensory equipment to experience an environmental scene twice, allowing the distinction of novel and familiar sensory data.

Place recognition calibration is performed by monitoring the difference scores between the current and previous sensory snapshots as the robot completes two revolutions, the first a "novel" revolution and the second a "familiar" revolution, since the robot is repeating a previous movement. The place recognition threshold is set to the maximum difference score for the familiar region of the calibration behavior. This method captures the largest possible variance in difference score for a *familiar*

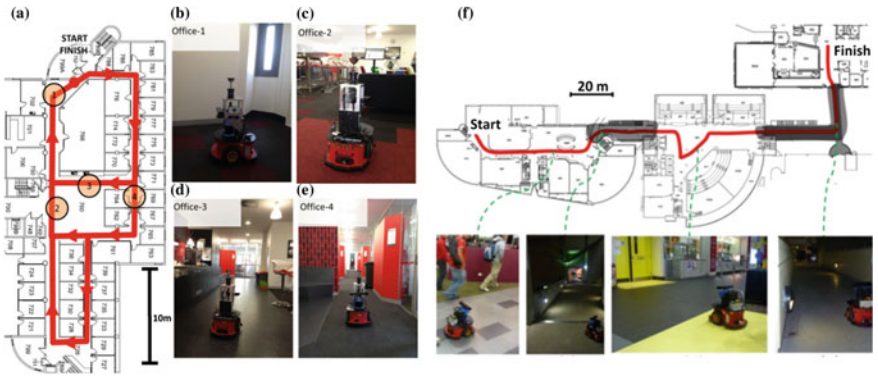


Fig. 3 **a** Map indicating the calibration locations and robot path for the office environment. **b–e** show photos of the calibration locations used within the office environment, which varied between open plan space, corridors and a kitchen. **f** Campus environment. The route was traversed during both day- and night-time conditions, with snapshots of the robot in the environment shown along the route

template match. This process is a conservative one—while it is likely the system will miss place matches in more perceptually challenging environments, false negatives are generally less catastrophic than false positives. The system also calculates a threshold quality score based on analysis of the difference score distribution over the two revolutions.

3.2 Experimental Setup

All the dataset acquisition and testing was performed in ROS groovy, all datasets ROS bags are available for readers to download and process at: https://wiki.qut.edu.au/display/cyphy/Michael+Milford+Datasets+and+Downloads. Detailed system parameters are provided in [20, 21].

3.2.1 Testing Environments

The testing environments were diverse and included a university campus and an office building floor. The Campus dataset was traversed during day and night conditions to test the system's ability to handle varying environmental conditions (Fig. 3).

3.2.2 Robot Platforms

The office robot configuration was built on an Adept MobileRobots Pioneer 3DX utilizing a FireWire PointGrey Camera with Catadioptric mirror, 16 ultrasonic range sensors, SICK laser range finder and Microsoft Kinect with RGB and Depth images. The campus robot configuration was also assembled on the Adept MobileRobots Pioneer 3DX using 16 ultrasonic range sensors, SICK laser range finder and Microsoft Kinect with RGB and Depth images.

3.3 Results

For reasons of brevity, here we present only the maps produced in each experiment—which reveal whether the system was able to produce topologically correct maps without any false connectivity between map locations. Further results can be found in [20, 21].

3.3.1 Office Environment

The calibration behavior was performed in the office environment resulting in the generation of the four sets of sensor thresholds and confidence scores shown in Fig. 4. Evaluation of Fig. 4b illustrates that all the autonomously generated place recognition thresholds are reliable (have a confidence score above 1), except the thresholds for sensors 1 and 2 in office calibration location 1. These low confidence scores were most likely due to the approximately equally distant and bland white walls of office calibration location 1. Figure 5 shows the experience maps are all

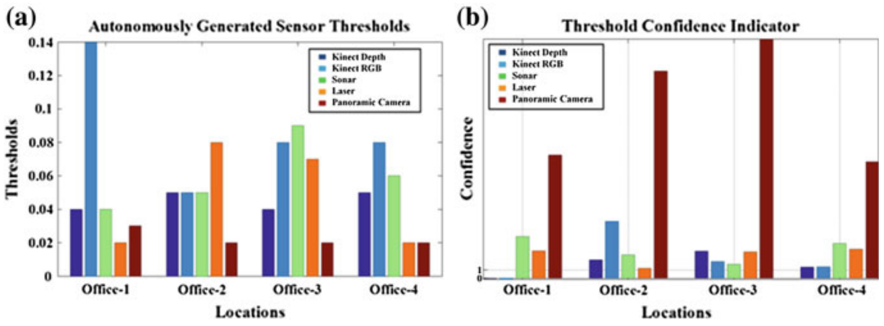


Fig. 4 **a** Autonomously calibrated thresholds from office calibration locations 1–4. Each group of five bars corresponds to the five sensor calibrations at one calibration location. **b** Corresponding calibration confidence scores. For each individual sensor, confidence scores less than 1 indicate a sensor calibration failure

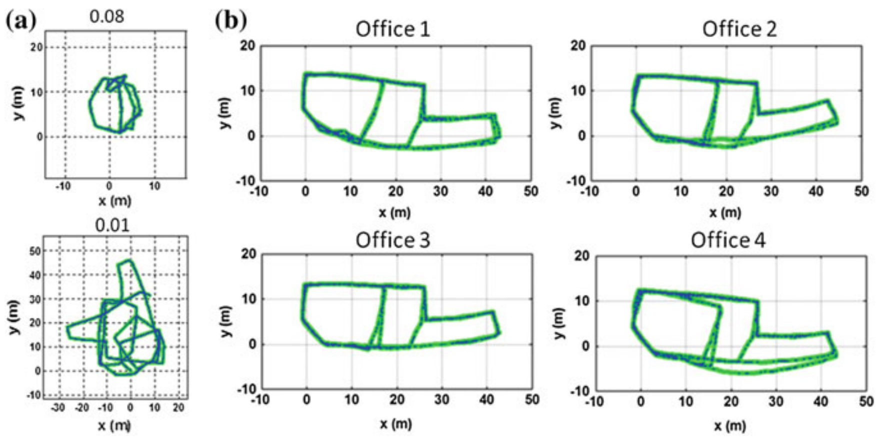


Fig. 5 OpenRatSLAM experience maps for the office environment generated with wheel encoder self-motion information using the (**a**) manually selected super template thresholds of 0.08 and 0.01 and (**b**) reliable autonomously calibrated thresholds from office calibration locations 1–4

topologically correct and have no incorrect loop closures, including the map created using only 3 reliable sensors from office calibration location 1.

3.3.2 Campus Environment

Here we present results for the campus environment experimentation produced from traversing the campus environment twice, first during the day and the second at night. Place recognition thresholds calibrated in the office calibration locations that resulted in a full set of trusted sensors (locations 2–4) were used for testing in the campus environment. Figure 6 shows the resultant OpenRatSLAM maps for the campus environment. All sensors were down weighted at various times during the experiment, removing large amounts of false positive matches from individual sensors. The dynamic sensor fusion system also removed some true positive matches, which resulted in some regions of the map not being connected together. All the maps are topologically correct although the recall rate for Fig. 6c is less than ideal. A reference map without sensor weighting is shown in Fig. 6d.

3.4 Future Work

We are currently investigating the use of a much wider range of sensing modalities such as WiFi. One of the most interesting insights from these multi-sensor fusion experiments is that different sensor types have varying spatial specificities when used in an associative mapping framework such as RatSLAM. Cameras offer the

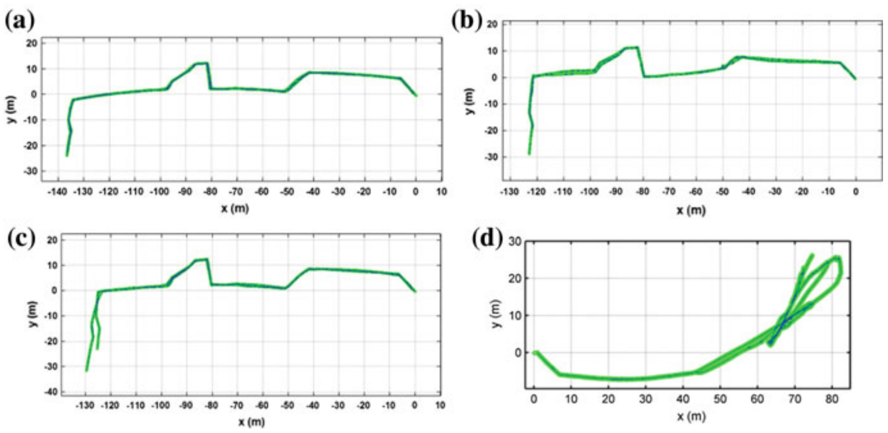


Fig. 6 **a**–**c** OpenRatSLAM experience maps of the campus environment generated with wheel encoder self-motion information using reliable autonomously calibrated thresholds from office calibration locations 2–4. **d** Map without online sensor weighting

potential for spatially precise place recognition performance, while sensors such as WiFi offer broader spatial localization. Attempting to integrate the place recognition information provided by each of these very different sensor types using a single scale mapping framework is likely suboptimal. In the next section, we present a pilot study investigating a multi-scale, homogeneous mapping framework inspired by the multi-scale maps recently found in the rodent brain.

4 Multi-scale Mapping

Most robot navigation systems perform mapping at one fixed spatial scale, or over two scales, often locally metric and globally topological [22–24]. Recent discoveries in neuroscience suggest that animals such as rodents, and likely many other mammals including humans, encode the world using multiple but homogeneous parallel mapping systems, each of which encode the world at a different scale [15, 25]. Although investigated in a theoretical context [26, 27], the potential performance benefits of such a mapping framework have not yet been investigated in a real-world robotics context. In this study, we investigated the utility of combining multiple homogeneous maps at different spatial scales to perform place recognition [14]. The performance of the multi-scale implementation was compared to a single scale implementation using two different vision-based datasets.

4.1 Approach

Our overall approach involves a feature extraction stage, a learning stage using arrays of Support Vector Machines, and a place recognition stage that combines place recognition hypotheses at different spatial scales.

4.1.1 Feature Extraction

Dimensional reduction was performed before camera images were input to the SVMs. We implemented two commonly used feature extraction methods—Principal Component Analysis (PCA) and GIST. PCA [28] is an efficient dimension reduction method which projects the original data into the directions with largest variances. Camera images were down-sampled to 64×48 before applying PCA. The first 38 principal eigenvectors were picked which were shown to already capture 90 % of the data variance. For GIST features, we chose the model proposed by Oliva [29] which provides a holistic description of the scene called Spatial Envelope. GIST was also attractive because of the possibility of generating relevant insights into how the biological visual mapping system may function. We extracted the GIST feature from down sampled 64×48 images which resulted in a 512-dimensional feature. We then extracted the top 32 principal eigenvectors, which captured approximately 90 % of the total variance.

4.1.2 Learning Algorithm

Support Vector Machines (SVM) [30] were chosen as the learning algorithm for two reasons. Firstly, they are an effective method for finding an optimal hyperplane to separate training data whilst simultaneously maximizing the classification margin, making it suited to the task of training recognition of a specific spatial segment and maximizing the difference between the training segment and other similar segments. Secondly, the use of SVMs removes the need for the extensive parameter tuning required of more biologically plausible grid cell models, such as continuous attractor networks [2], although we do intend to eventually adopt these models to maximize biological relevance.

4.1.3 Combining Multi-scale Place Match Hypotheses

Each array of SVMs produces a firing matrix M representing the matching scores of the testing segments on the trained SVMs where element $M(i, j)$ indicates the response of the ith SVM from a training dataset to the jth segment in a test dataset. Firing scores in each column j are then normalized to sum to one for each segment recognition distribution:

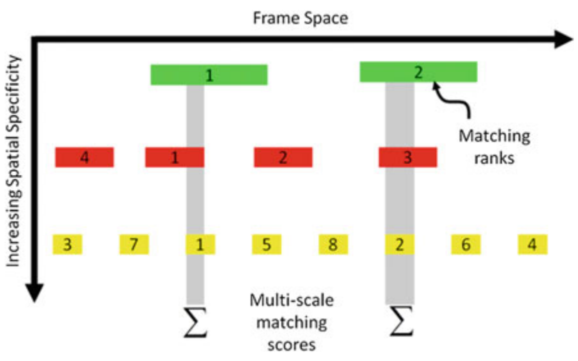


Fig. 7 Overlapping SVM matching scores are combined at the smallest spatial scale in order to accept or reject place match hypotheses. In this case, K = 3

$$M(i,j) = \frac{M(i,j)}{\sum_i M(i,j)} \tag{1}$$

Place recognition hypotheses produced by each array of SVMs are only as accurate as the average size of a segment in that array. To create a common scale in which hypotheses from different spatial scales can be compared and combined, reported place recognition matches are transformed to the scale space of the smallest segment size. For K arrays of SVMs, the matching scores after normalization of each array are:

$$M_p, \quad \mathrm{p} = 1, \ldots, K \tag{2}$$

Suppose there are L_p training segments for the matching score M_p. For a segment j in a test data set, its coherence measurement on each training segment $c\,(i, j),\ i = 1, \ldots, L_p$ is determined by whether spatially overlapping hypotheses exist over all SVMs scales. If not, the system reports "no coherent" match (c = 0):

$$c(i,j) = \begin{cases} 1, & M_p(i,j) > 0,\ \forall p \\ & 0,\ else \end{cases} \tag{3}$$

At the smallest spatial scale, there can be several competing place recognition hypotheses that are supported by all other spatial scales. To determine the most likely hypothesis, we sum the firing scores of the overlapping SVMs at each spatial scale and classify segment j to the class $C(j)$ with the largest accumulated firing score (Fig. 7):

$$C(j) = arg\max_i \sum\nolimits_p M_p(i,j),\ \forall c(i,j) = 1 \tag{4}$$

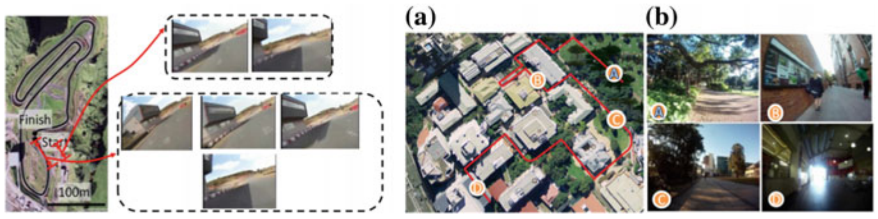


Fig. 8 The Rowrah dataset (*left*) and Campus data (*right*) with example frames

Table 1 Dataset descriptions

Dataset name	Single traverse distance (m)	Number of frames per traverse	Resolution
Rowrah	1000	1570	320 × 240
Campus	800	1000	1280 × 960

4.2 *Experimental Setup*

We used two datasets (Fig. 8) to test the multi-scale algorithms, with details listed in Table 1. Each dataset consists of two traverses along the same route with the first traverse used for training and the second traverse for testing. The Rowrah dataset was collected from a forward-facing camera mounted on a motorbike and can be downloaded at the following link: http://www.youtube.com/watch?v=_UfLrcVvJ5o. The Campus dataset was sourced from a GoPro Hero 1 camera mounted on a bicycle pushed by an experimenter. The bike was pushed through and in-between buildings along a mixed indoor-outdoor path approximately 800 m long. Due to GPS not being viable, datasets were parsed frame by frame to build ground truth correspondence between testing and training data sets for each spatial scale.

4.2.1 Training Procedure

Images from the first traverse of the environment were used for training while images from the second traverse were used to evaluate performance. The overall training procedure consisted of the following three steps: dataset segmentation, feature extraction and SVM training.

Dataset Segmentation

The images in each dataset were grouped into a total of S subsequent segments (S/2 segments per traverse). Larger values of S result in smaller size of each segment. For the sake of intuition, we refer to different SVMs by the size of each segment, not the number of segments. For example, each traverse in the Campus dataset is approximately 800 m and therefore splitting the Campus dataset into 170 segments

(85 segments per traverse) resulted in an average segment size of approximately 9.4 m. We then use “9.4 m system” to refer to the SVMs with 170 segments.

Feature Extraction

Two feature types (as discussed in Sect. 4.1.1) were extracted from each dataset. The feature vectors from all frames in a segment were combined into a single vector and input into each of the SVMs.

SVM Training

To train a SVM model for each segment, we manually labeled the images in that segment as positive examples and those from the other N segments as negative examples. Ideally, all other (S-1) groups would be used as negative examples. However, since in real world situations it may not be possible to train on the entire training dataset, we instead arbitrarily set N to be 9, indicating for each segment, we use 1 frame group as a positive example and 9 other adjacent frame groups as negative samples

4.3 Results

We show three key sets of results—comparison between single and multi-scale place recognition, ground truth plots and illustrative multi-scale place recognition combination plots.

4.3.1 Single- and Multi-scale Place Recognition

This section presents precision recall (PR) curves for the single- and multi-scale place recognition experiments. Each PR curve was generated by sweeping the accepted range in each hypothesis rank. For both single- and multi-scale matching, it is, not surprisingly, easier to perform place recognition when trying to match a spatially broad segment than when trying to match a spatially specific segment. This disparity is most likely due to two reasons; firstly because performance is bound to increase when the false positive spatial error tolerance is bigger, and secondly, because the larger segments are trained on a larger number of frames.

Precision-recall performance at all except very low precision levels improved significantly across all experiments. At 100 % precision, the recall rate was improved by an average of 74.79 % across all experiments. The biologically-inspired feature GIST slightly outperformed PCA—at 100 % precision, the recall rate for GIST was improved by an average of 81.7 % over all experiments, versus 67.9 % for PCA.

4.3.2 Ground Truth Plots

Figures 11a, b present ground truth plots showing the true positives (green circles), false positives (blue squares) and false negatives (red stars) output by the single and multi-scale systems for the Rowrah datasets without (a) and with (b) multi-scale combination. Straight lines connect the matching segments.

4.3.3 Multi-hypothesis Combination Plots

Figure 11c–f show examples of how place match hypotheses at varying scales are combined together. In general, a large number of false positives at the smallest spatial scale (yellow color) are eliminated due to lack of support from larger spatial scales. The examples in (c, d) show how secondary ranked spatially specific matches are correctly chosen as the overall place match due to support from other spatial scales. In (e) the best ranked spatially specific match is correctly supported by the other spatial scales, while (f) shows a failure case where the incorrect 4th ranked spatially specific match is more strongly supported by the other spatial scales that the 1st ranked and correct spatially specific match. Interestingly, the most common failure mode of the system is to report a "minor" false positive match—a place match to a different location at the smallest spatial scale but within the correct place at a larger spatial scale (Figs. 9, 10).

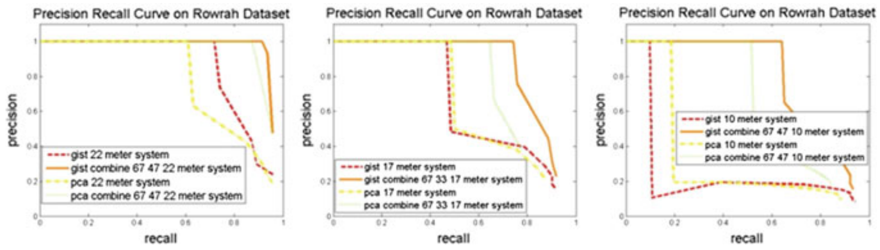


Fig. 9 Precision recall curves demonstrating the single- and multi-scale place recognition performance for the Rowrah dataset

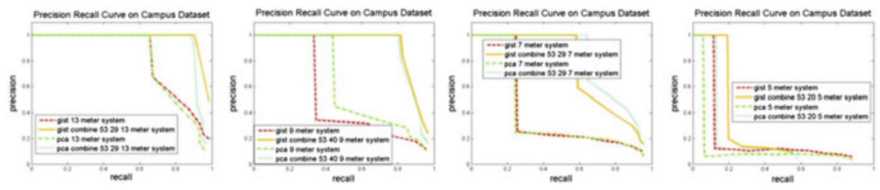


Fig. 10 Precision recall curve demonstrating the results with and without combination for Campus dataset using gist features and PCA features

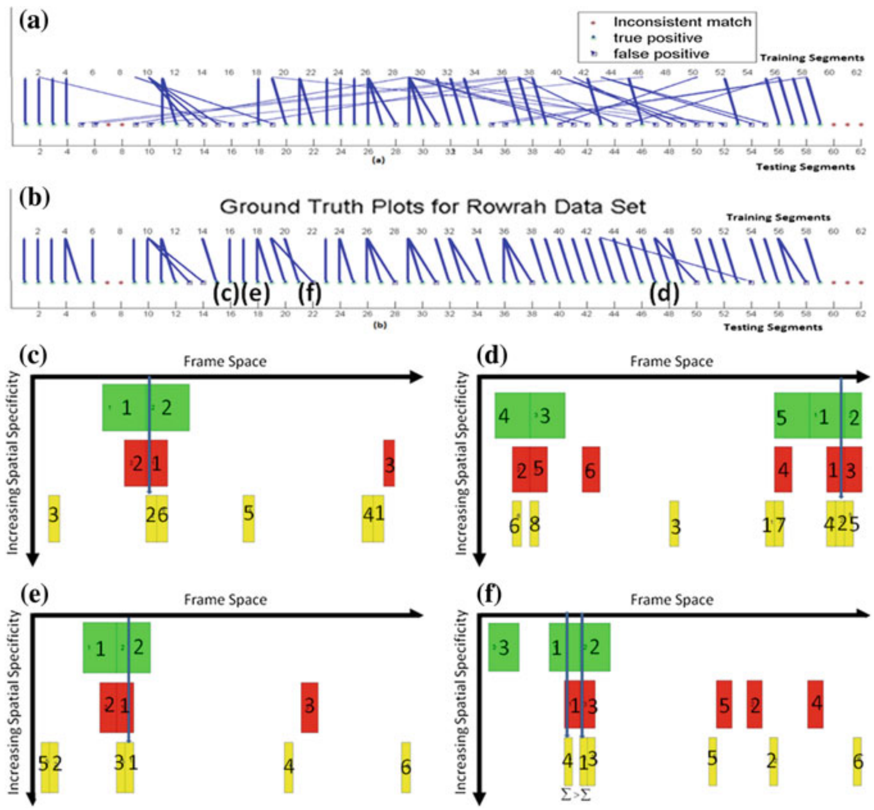


Fig. 11 Ground truth plots for the **a** single and **b** multi-scale Campus dataset. **c**, **d** Show examples of secondary-ranked spatially specific place matches (*yellow*) that became the primary overall place match hypothesis due to support from other spatial scales. In **e** the first ranked spatially specific match is supported, while **f** shows a failure case where a secondary ranked spatially specific match is incorrectly chosen as the overall match due to more significant support from the other spatial scales than the correct, first ranked spatially specific match

4.4 Discussion and Future Work

Place recognition performance was improved by combining the output from parallel systems, each trained to recognize places at a specific spatial scale. Although here we presented a specific implementation of both the vision processing and place recognition framework, we believe that the novel multi-scale combination concept should generalize to other systems. In future work, we will incorporate an odometry source to enable the system to allocate segments directly based on distance travelled rather than (in effect) time. Odometry information may enable us to expand our current system to two-dimensional unconstrained movement in large open environments. Testing the system in open field environments will be more analogous to many current rodent experiments and may increase the likelihood of generating neuroscience insights. An obvious extension to the sensor fusion work presented here and elsewhere [21, 31] would be to use a multi-scale mapping framework approach to exploit the variable spatial specificity of different sensor modalities, such as cameras, range finders and WiFi.

5 Achieving Balance in Interdisciplinary Research

If we were asked to identify the single key issue involved in conducting interdisciplinary (especially biologically-inspired) robotics research it would be this:

How can research achieve the appropriate balance between maintaining a faithful representation of the modelled biological systems and producing state of the art performance in the robotics domain at a relevant task?

To discuss this issue concisely in a paper such as this, one must necessarily make some generalizations. Research focusing on maintaining fidelity to the underlying source of biological inspiration often produces performance that is inferior to conventional mathematical approaches, but can lead to novel insightful predictions about biological systems. Conversely, research that readily abandons any relevance to the biology may lead to better robotics performance but is rarely the cause of new discoveries in biological research. In addition, it becomes an increasingly painful process to generate relevant testable predictions or insights in the biological field as the model becomes more and more abstracted.

In the initial stages of the RatSLAM project, we started with what was then a state of the art neural network model of the mapping processes observed in the rodent brain. As we tested the algorithms in larger and more challenging environments and over longer periods of time, we were forced to make some pragmatic modifications to the algorithms to produce good mapping performance. These modifications seemingly moved the model further away from biology. One example would be the pragmatic decision to engineer the *pose cells*, artificial neurons that encode the complete three-dimensional (x, y, ϑ) pose of a ground-based robot and are re-used at regular intervals to efficiently encode large environments. The decision to move to *pose cells* was made because the neuron types known at that time—*place cells* which represent (x, y) location—and *head-direction cells* which represent orientation—were unable to represent and correctly update multiple robot location hypotheses. Subsequently neuroscientists discovered a new type of spatial neuron called a *grid cell* in the rodent brain sharing similar although not identical characteristics [15, 32]. This discovery demonstrated that a functionally driven investigation (engineering a new cell to produce better mapping performance) could lead to relevant insights or predictions in another discipline, in this case neuroscience. It is interesting to speculate that, had we abandoned the biological neural network completely and moved to a conventional technique such as a particle or Kalman filter, it may have been harder to make this specific prediction. Conversely, if we had maintained a more biological faithful model, we may never have been able to test it in environments that were sufficiently challenging to require the ability to encode and propagate multiple location hypotheses. At least in this particular example, it was only by following the “middle ground” that we were able to make some contribution to both fields.

Acknowledgments This work was supported by an Australian Research Council Discovery Project DP120102775 and Microsoft Research Faculty Fellowship to MM, and an ARC & NHMRC Thinking Systems grant TS0669699 to GW.

References

1. Wyeth, G., Milford, M.: Spatial cognition for robots: robot navigation from biological inspiration. IEEE Robot. Autom. Mag. **16**, 24–32 (2009)
2. Milford, M.J.: Robot Navigation from Nature: Simultaneous Localisation, Mapping, and Path Planning Based on Hippocampal Models, vol. 41. Springer, Berlin (2008)
3. Milford, M.J., Wyeth, G., Prasser, D.: RatSLAM: a hippocampal model for simultaneous localization and mapping. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 403–408. New Orleans, USA (2004)
4. Milford, M., Wyeth, G.: Mapping a suburb with a single camera using a biologically inspired SLAM system. IEEE Trans. Robot. **24**, 1038–1053 (2008)
5. Milford, M., Wyeth, G.: Single camera vision-only SLAM on a suburban road network. In: Proceedings of the International Conference on Robotics and Automation, Pasadena, United States (2008)
6. Milford, M., Wyeth, G.: Persistent navigation and mapping using a biologically inspired SLAM system. Int. J. Robot. Res. **29**, 1131–1153 (2010)
7. Ball, D., Heath, S., Wiles, J., Wyeth, G., Corke, P., Milford, M.: OpenRatSLAM: an open source brain-based SLAM system. Auton. Robots, 1–28 (2013)
8. Milford, M., Wiles, J., Wyeth, G.: Solving navigational uncertainty using grid cells on robots. PLoS Comput. Biol. **6**(11), e1000995 (2010)
9. Milford, M., Turner, I., Corke, P.: Long exposure localization in darkness using consumer cameras. In: Proceedings of the IEEE International Conference on Robotics and Automation (2013)
10. Milford, M.: Vision-based place recognition: how low can you go? Int. J. Robot. Res. **32**, 766–789 (2013)
11. Milford, M., Wyeth, G.: SeqSLAM: visual route-based navigation for sunny summer days and stormy winter nights. In: Proceedings of the IEEE International Conference on Robotics and Automation, St Paul, United States (2012)
12. Milford, M.: Visual route recognition with a handful of bits. In: Robotics: Science and Systems VIII. Australia, Sydney (2012)
13. Milford, M., Vig, E., Scheirer, W., Cox, D.: Towards condition-invariant, top-down visual place recognition. In: Proceedings of the Australasian Conference on Robotics and Automation, Sydney, Australia (2013)
14. Chen, Z., Jacobson, A., Erdem, U.M., Hasselmo, M.E., Milford, M.: Towards bio-inspired place recognition over multiple spatial scales. In: Proceedings of the Australasian Conference on Robotics and Automation, Sydney, Australia (2013)
15. Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., Moser, E.I.: Microstructure of a spatial map in the entorhinal cortex. Nature **11**, 801–806 (2005)
16. Golani, I., Bronchti, G., Moualem, D., Teitelbaum, P.: “Warm-up" along dimensions of movement in the ontogeny of exploration in rats and other infant mammals. In: Proceedings of the National Academy of Sciences of the United States of America, vol. 78, pp. 7226–7229 (1981)

17. Stratton, P., Milford, M., Wyeth, G., Wiles, J.: Using strategic movement to calibrate a neural compass: a spiking network for tracking head direction in rats and robots. PLoS One **6**(10), e25687 (2011)
18. Cheung, A., Ball, D., Milford, M., Wyeth, G., Wiles, J.: Maintaining a cognitive map in darkness: the need to fuse boundary knowledge with path integration. PLoS Comput. Biol. **8**(8), e1002651 (2012)
19. Thomson, E.E., Carra, R., Nicolelis, M.A.: Perceiving invisible light through a somatosensory cortical prosthesis. Nat. Commun. **4**, 1482 (2013)
20. Jacobson, A., Chen, Z., Milford, M.: Autonomous movement-driven place recognition calibration for generic multi-sensor robot platforms. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan (2013)
21. Milford, M., Jacobson, A.: Brain-based sensor fusion for navigating robots. In: Proceedings of the IEEE International Conference on Robotics and Automation, Karlsruhe, Germany (2013)
22. Bosse, M., Newman, P., Leonard, J., Soika, M., Feiten, W., Teller, S.: An atlas framework for scalable mapping. In: Proceedings of the International Conference on Robotics and Automation, Taipei, Taiwan, pp. 1899–1906 (2003)
23. Kuipers, B., Modayil, J., Beeson, P., MacMahon, M., Savelli, F.: Local metrical and global topological maps in the hybrid spatial semantic hierarchy. In: Proceedings of the International Conference on Robotics and Automation, New Orleans, USA (2004)
24. Kuipers, B., Byun, Y.T.: A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. Robot. Auton. Syst. **8**, 47–63 (1991)
25. Stensola, H., Stensola, T., Solstad, T., Froland, K., Moser, M., Moser, E.: The entorhinal grid map is discretized. Nature **492**, 72–78 (2012)
26. Burak, Y., Fiete, I.R.: Accurate path integration in continuous attractor network models of grid cells. PLoS Comput. Biol. **5**(2), e1000291 (2009)
27. Welinder, P.E., Burak, Y., Fiete, I.R.: Grid cells: the position code, neural network models of activity, and the problem of learning. Hippocampus **18**, 1283–1300 (2008)
28. Jolliffe, I.T.: Principal Component Analysis, 2nd edn. Springer, New York (2002)
29. Oliva, A., Torralba, A.: Modeling the shape of the scene: a holistic representation of the spatial envelope. Int. J. Comput. Vis. **42**, 145–175 (2001)
30. Vapnik, V.: The support vector method of function estimation. Nonlinear Modeling. Kluwer, Boston (1998)
31. Jacobson, A., Milford, M.: Towards brain-based sensor fusion for navigating robots. In: Proceedings of the Australasian Conference on Robotics and Automation (2012)
32. Fyhn, M., Molden, S., Witter, M.P., Moser, E.I., Moser, M.-B.: Spatial representation in the entorhinal cortex. Science **27**, 1258–1264 (2004)

Into Darkness: Visual Navigation Based on a Lidar-Intensity-Image Pipeline

Timothy D. Barfoot, Colin McManus, Sean Anderson, Hang Dong, Erik Beerepoot, Chi Hay Tong, Paul Furgale, Jonathan D. Gammell and John Enright

Abstract Visual navigation of mobile robots has become a core capability that enables many interesting applications from planetary exploration to self-driving cars. While systems built on passive cameras have been shown to be robust in well-lit scenes, they cannot handle the range of conditions associated with a full diurnal cycle. Lidar, which is fairly invariant to ambient lighting conditions, offers one possible remedy to this problem. In this paper, we describe a visual navigation pipeline that exploits lidar's ability to measure both range and intensity (a.k.a., reflectance) information. In particular, we use *lidar intensity images* (from a scanning-laser rangefinder) to carry out tasks such as *visual odometry* (VO) and *visual teach and repeat* (VT&R) in realtime, from full-light to full-dark conditions. This lighting invariance comes at the price of coping with motion distortion, owing to the scanning-while-moving nature of laser-based imagers. We present our results and lessons learned from the last few years of research in this area.

1 Introduction

1.1 Motivation

Visual navigation is an enabling technology for mobile robots operating in challenging, real-world environments. Satellite-based positioning, such as GPS, is often insufficient or unavailable in many interesting situations: indoors, in urban canyons, under forest canopies, underground, underwater, and on other planets. As such, passive cameras and/or lidars are employed to provide as position estimation and also help with path following, hazard detection, and object recognition. Cameras and

T.D. Barfoot (✉) · C. McManus · S. Anderson · H. Dong · E. Beerepoot · C.H. Tong · P. Furgale · J.D. Gammell · J. Enright
University of Toronto Institute for Aerospace Studies, 4925 Dufferin St., Toronto, ON, Canada
e-mail: tim.barfoot@utoronto.ca

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_28

lidars are typically viewed as separate-yet-complementary sensors. Roughly speaking, (passive) cameras are used to acquire appearance information and geometry while (active) lidars are used to acquire geometry. The active nature of lidars make them well-suited to working in any lighting condition, while passive cameras struggle with lighting change.

An underexploited capability of 3D lidar is its ability to acquire appearance information through *intensity images*. Intensity data is derived from the amount of transmitted light that is reflected back from the scene. Traditionally, the raw output of a lidar sensor is thought to be a 3D point cloud; instead, we consider the output to be a pair of range and intensity images. Figure 1 provides a comparison between passive camera images and lidar intensity images for the same scene.

This paper describes how we use lidar intensity images (derived from a scanning-laser rangefinder) to build a realtime, lighting-invariant visual pipeline that leverages the heritage of the traditional stereo-camera pipeline. We were motivated to do this for two reasons: (i) to navigate in full-dark conditions, and (ii) to recognize places we had seen before, despite drastic changes in lighting. Much of our work is targeted at future planetary exploration missions. For example, permanently shadowed craters near the south pole of the Moon may contain water-ice and other useful volatiles. Missions to these craters will require robots that are able to navigate in full darkness and in

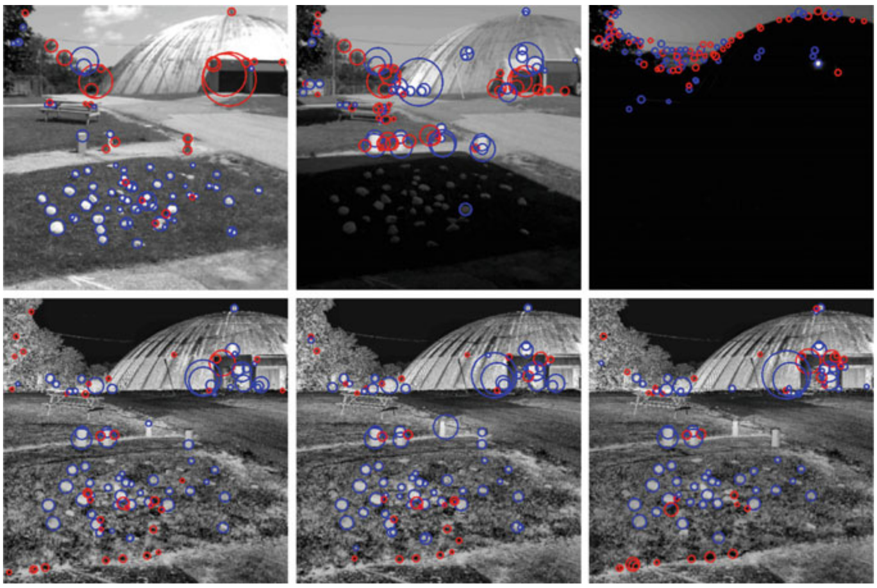


Fig. 1 Examples of passive camera (*top row*) and lidar intensity (*bottom row*) images of the same scene at three different times of day (13h38, 18h12, 05h43). Images acquired using an Optech ILRIS3D survey-grade lidar with built-in passive camera. Raw SURF [5] features are marked in all images. We see that the lidar intensity images and features look very similar regardless of the sunlight conditions, while the passive camera images and features change significantly with lighting

realtime (due to the low-latency communications and short lunar day). However, beyond planetary exploration, robotics generally requires the ability to recognize previously visited places in order to build consistent survey maps and re-drive routes.

We successful built a lidar-intensity-image pipeline suitable for visual odometry (VO) [10, 21, 23] and visual teach and repeat (VT&R) [22, 24]. However, due to the scanning-while-moving nature of laser-based imaging, motion distortion can be significant if the sensor's motion is high relative to its framerate. In order to obtain an accurate motion estimate, we were forced to innovate ways of coping with this distortion [2, 3, 26–29], but we believe that it has been worth the effort to achieve lighting invariance.

The rest of the paper is organized as follows. The remainder of Sect. 1 provides a brief summary of related work and introduces the lidar-intensity-image pipeline. Section 2 discusses motion distortion and our various approaches to overcome it. Section 3 provides the results of some visual-odometry and visual-teach-and-repeat experiments. Section 4 discusses lessons learned and concludes the paper.

1.2 Related Work

We only briefly review other works that have used lidar intensity images (a.k.a., reflectance images) for motion estimation. McManus et al. [23, 24] provide more information. Laser intensity images have been used in the past for surveying applications [8, 18] and some researchers have looked at automated point-cloud registration techniques that use 2D interest points in the intensity images [1, 6].

The SwissRanger sensor, a 'flash lidar', also produces intensity/range images but using a different principle than laser-based scanners. Unlike a laser scanner, the SwissRanger uses an array of 24 LEDs to simultaneously illuminate a scene, offering the advantage of higher framerates. However, the SwissRanger has a limited FOV, short maximum range, and is very sensitive to environmental noise. Weingarten et al. [30] used images from the SwissRanger for robotics applications; however, their method, as well as others that followed [11, 32], only used range data from the sensor and not the intensity data.

May et al. [20], and later Ye and Bruch [31], were the first to develop 3D mapping and appearance-based egomotion techniques using a SwissRanger. May et al. [20] used intensity images to employ two feature-based methods for motion estimation: a Kanade–Lucas–Tomasi (KLT)-tracker and frame-to-frame VO using SIFT features. Their results indicated that the SIFT approach yielded more accurate motion estimates than the KLT approach, but less accurate than the iterative closest point (ICP) method. Although May et al. [20] demonstrated that frame-to-frame VO might be possible with the SwissRanger, the largest environment in which they tested was a 20 m long indoor hallway, with no groundtruth. Furthermore, laser scanners are very different from flash lidars in that they scan the scene with a single light source, introducing new problems such as image formation and image distortion caused by

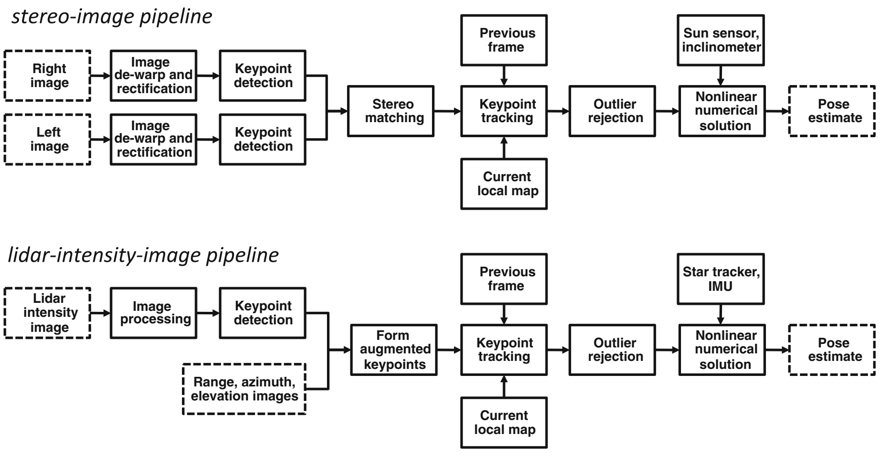


Fig. 2 Stereo-image (*above*) and lidar-intensity-image (*below*) visual pipelines. Both pipelines show the main steps required to go from raw images on the *left* to a pose solution on the *right*. On the surface, we see that switching to lidar intensity images only alters the initial steps. However, due to the scanning-while-moving nature of the lidar-intensity images, most of the blocks require modification to compensate for motion distortion

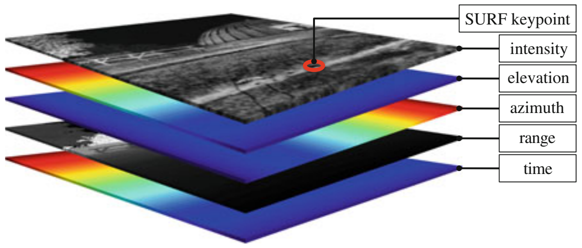


Fig. 3 Image stack concept. We detect sparse keypoints in the intensity image (at subpixel locations) then, for each keypoint, pierce through the stack to look up the associated azimuth, elevation (based on lidar's mirror angles), range, and time (using bilinear interpolation). The result is a keypoint augmented with its 3D position and timestamp

moving and scanning at the same time. We believe our work is the first and only one to use laser-based intensity images in a realtime visual pipeline.

1.3 Lidar-Intensity-Image Pipeline

Figure 2 provides an overview of our lidar-intensity-image pipeline, as well as the typical stereo-image pipeline for comparison. The purpose of these pipelines is to determine the robot's motion from a sequence of images. The main steps of the lidar pipeline are:

(i) acquire a lidar intensity image,
(ii) carry out preprocessing to improve contrast in the image—*either adaptive histogram equalization or a linear range correction*,
(iii) detect sparse keypoints in the intensity image—*we use SURF implemented on a GPU but other methods should work*,
(iv) build an 'image stack' (cf., Fig. 3) and for each keypoint pierce through to form augmented keypoints—*this provides the 3D position (azimuth, elevation, range) and time of each keypoint*,
(v) track keypoints—*we match to the previous frame (VO) and a local map (VT&R)*,
(vi) detect and reject outliers—*we use RANSAC*,
(vii) solve for the robot's motion using a nonlinear, least-squares method—*we can incorporate an attitude sensor such as a star tracker and/or IMU to help determine orientation*.

We carry out these steps every time a new image is acquired.

Figure 4 compares the lidar-intensity-image and stereo-image pipelines on the VO problem. With the robot stopping every time it gathered images (approximately every 0.5 m), we can see that the lidar and stereo pipelines both provide good estimates of the robot motion compared to GPS groundtruth. However, if we allow the lidar to acquire images while in motion, the VO performance degrades quickly as the images becomes distorted. We discuss this motion distortion and our efforts to compensate for it in detail in the next section.

2 Coping with Motion Distortion

2.1 Nature of Distortion

In this paper, we are concerned with laser-based 3D rangefinders that are capable of producing high-quality intensity images. The main sensor we use for realtime oper-

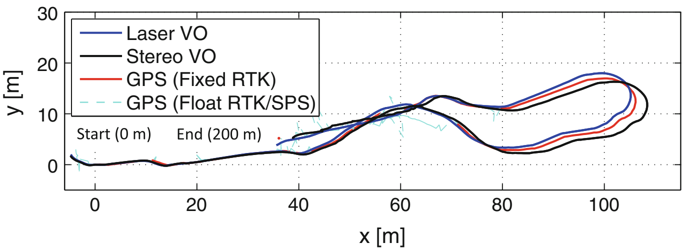


Fig. 4 Comparison of lidar-intensity-image (laser) and stereo-image VO. In this experiment the robot stopped every time images were acquired (approximately every 0.5 m). Both algorithms are able to provide reasonable estimates (compared to GPS groundtruth) in this stop-scan-go paradigm

ations is the Autonosys LVC0702, which uses a combined nodding and polygonal mirror assembly to steer a single laser beam through a raster pattern in order to build intensity/range images (cf., Fig. 5; left). It works out to about 50 m in range.

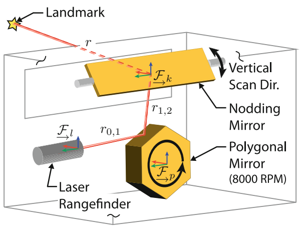
To our knowledge, the Autonosys unit has the highest pulse repetition frequency (PRF) of the single-laser scanners on the market at 500,000 points/s. We use the unit in a mode that produces 480 × 360 pixel intensity images at 2 Hz. Even for a robot moving at a modest speed, say 0.5 m/s, there is noticeable distortion in the images if they are acquired during motion (cf., Fig. 5; right). This is essentially an exaggerated 'rolling shutter' effect; an image is gathered over a fraction of a second, with every pixel acquired at a unique (known) time.

We considered using a Velodyne HDL64E for our work, but found that the narrow vertical field of view (16°) and resolution (64 pixels) made the resulting intensity images unsuitable for our approach; it would also require careful inter-calibration of intensity values derived from the 64 separate laser sources.

Flash lidar does not suffer from the same motion distortion issues as laser-based rangefinders and eventually may be provide lighting-invariant imagery. Currently, however, inexpensive units such as the SwissRanger SR4000 have limited range capabilities (less than 10 m), struggle to cope with sunlight, and have low resolution (e.g., 176 × 144 pixels). More expensive units, such as the ASC TigerEye, have longer range and work outside but still have limited resolution (e.g., 128 × 128 pixels) and thus are not suitable for our image pipeline as of yet.

The Microsoft Kinect sensor produces similar data products to lidar (i.e., intensity and depth images) but it does not work outside in direct sunlight, its range is limited to approximately 7 m, and the intensity image comes from a passive camera and thus is not lighting-invariant.

Thus, for the time being, we need to be able to handle motion distortion in laser-based lidar images. The next sections describe our efforts to cope with this issue.



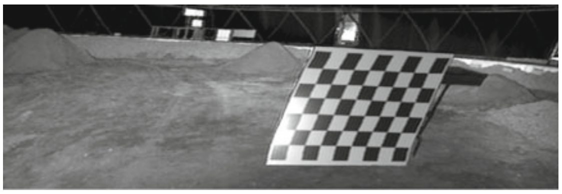


Fig. 5 3D range sensors based on a single laser use a mirror assembly (*left*) to steer the beam through a raster pattern in order to build an image. If the lidar is mounted on a moving robot, the scene can undergo a non-affine transformation during imaging; in the checkerboard example (*right*), some of the straight lines are distorted because the lidar was in motion during acquisition

2.2 *Effect on Features*

In our pipeline, we extract SURF features from the raw, motion-distorted images and track them on a frame-to-frame basis (cf., Fig. 6). The effect of scanning while moving has not been so severe as to cause feature tracking to fail catastrophically in our experiments so far. Thus, we have avoided carrying out full-image motion compensation. However, if the same scene is imaged twice at two very different vehicle speeds, it is intuitive that feature matching will fail as the images will experience different amounts of motion distortion. We are currently carrying out a study to characterize how large we can make the speed differential and still successfully match features.

2.3 *Motion-Compensated RANSAC*

The next step is to identify outlying feature tracks and remove them from the pipeline. We originally used the out-of-the-box random sample and consensus (RANSAC) algorithm [12] typically found in the stereo-camera pipeline. This worked reasonably well most of the time, but we found that the threshold used to separate inliers from outliers was difficult to tune. We would typically end up with a lot of false negatives (i.e., throwing away many good feature tracks) or a handful of false positives (i.e., letting some bad feature tracks in). We see these two cases in Fig. 7 (left; middle). Note that when the threshold is tight the inliers are restricted to a horizontal band, implying near-simultaneous capture. This is expected as the lidar scans quickly left-to-right while slowly scanning up-and-down.

We found the reason for RANSAC's difficulty to be our choice of model. Typically, RANSAC in the stereo-camera pipeline seeks to find a rigid, frame-to-frame pose change that explains the most data. Due to the motion distortion present in the

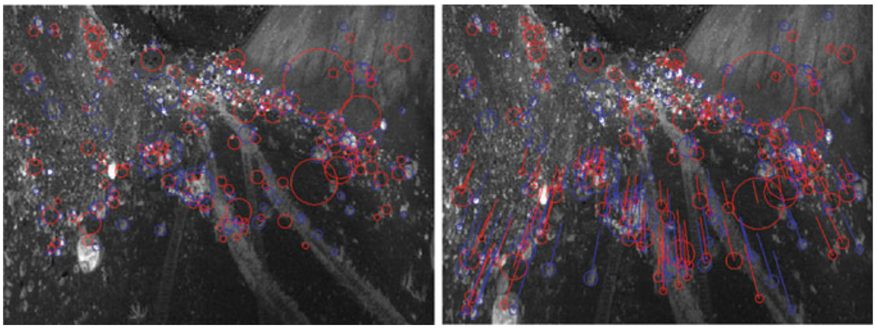


Fig. 6 We extract SURF features from the raw (i.e., motion-distorted) images (*left*) and track them on a frame-to-frame basis (*right*). This allows us to only motion-compensate a sparse number of points rather than the entire image

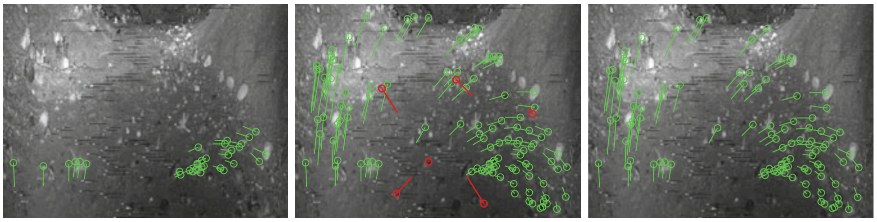


Fig. 7 In a VO pipeline, outliers are usually detected/rejected on a frame-to-frame basis using RANSAC [12], which is used to solve for the rigid pose change between two frames that explains the most features. Unfortunately, for intensity images gathered during motion, this model is insufficient and results in false negatives if the matching threshold is too tight (*left*) or false positives (*red*) if the threshold is too loose (*middle*). To overcome this, we created a motion-compensated version of RANSAC that solves for the 6DOF velocity that explains the most features (*right*) [2]

imagery, we found that it was much more effective to have RANSAC find the *constant velocity* that explains the most data over a two-frame time interval. It turns out that, as with the rigid pose change, the minimum number of feature tracks needed to fit a constant velocity is still three. Thus, RANSAC proceeds as usual, but with this change of model. We see the improved outlier rejection in Fig. 7 (right). We are now able to keep the decision threshold tight and still obtain lots of good feature tracks without false positives. Anderson and Barfoot [2] provide further details.

2.4 Continuous-Time Estimation for Pose

After removing outlying feature tracks, the last major step in the pipeline is to solve for the pose change using an iterative, nonlinear, least-squares method such as *bundle adjustment* [7]. It is again important in this step to account for the proper timestamps of all observed features in order to combat the motion distortion. Traditional approaches represent the robot's trajectory in discrete time (cf., Fig. 8; left). This is sufficient because the exposure times associated with passive-camera image capture are so short that they can be thought of as instantaneous.

Unfortunately, for lidar intensity images, this is not a good assumption. If we simply put a discrete-time pose at nominal image capture times, we cannot account for the actual (and varying) timestamps of the measurement observations and our VO solution ends up being poor. If we put a discrete-time pose at every unique measurement timestamp, the problem is computationally intractable but also under-constrained (without including an additional prior).

We found a better idea was to consider the robot's trajectory to be a continuous function of time (cf., Fig. 8; right) so that we could query it at any particular time at which a feature was observed. Thus, in general, we write the robot's trajectory as $\mathbf{x}(t)$ and then build a measurement reprojection error term as

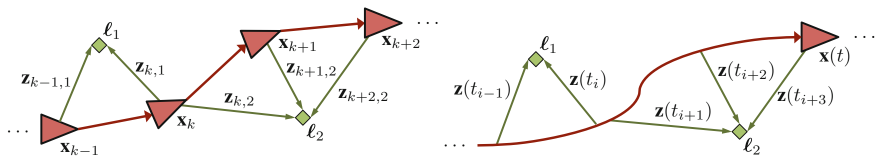


Fig. 8 Typically in robotics, the robot's trajectory is represented in discrete time (*left*). We represent the robot's trajectory in continuous time (*right*), $\mathbf{x}(t)$, which allows us to query the pose at the exact times the landmarks were observed; this is important due to the scanning-while-moving nature of laser-based scanners

$$\mathbf{e}_{k,j} = \mathbf{z}_{k,j} - \mathbf{g}\left(\mathbf{x}(t_k), \boldsymbol{\ell}_j\right), \tag{1}$$

where $\mathbf{z}_{k,j}$ is the observation of landmark j at time t_k, $\mathbf{g}(\cdot, \cdot)$ is the lidar's observation model, and $\boldsymbol{\ell}_j$ is the position of landmark j. We sum over all the measurements to build a nonlinear, least-squares cost function, J, that we seek to minimize with respect to $\mathbf{x}(t)$:

$$J(\mathbf{x}(t), \boldsymbol{\ell}) = \frac{1}{2}\sum_{k,j} \mathbf{e}_{k,j}^T \mathbf{R}_{k,j}^{-1} \mathbf{e}_{k,j}, \tag{2}$$

where $\mathbf{R}_{k,j}$ is the measurement noise covariance associated with $\mathbf{e}_{k,j}$. We then seek to solve the following optimization problem,

$$\left\{\mathbf{x}(t)^\star, \boldsymbol{\ell}^\star\right\} = \underset{\mathbf{x}(t),\,\boldsymbol{\ell}}{\operatorname{argmin}}\ J(\mathbf{x}(t), \boldsymbol{\ell}) \tag{3}$$

for the optimal robot trajectory, $\mathbf{x}(t)^\star$, and landmark positions, $\boldsymbol{\ell}^\star$. We use Gauss-Newton optimization with a robust kernel to find the best trajectory estimate.

Naturally, we still need to discretize $\mathbf{x}(t)$ in some way to make the state estimation problem tractable and have considered a few ways of doing so:

(i) *linear interpolation*: represent the trajectory using discrete-time poses, $\mathbf{x}_i$, with linear pose interpolation in between to evaluate measurement error terms at their appropriate times [10]

(ii) *spline*: represent the trajectory as a weighted sum of a finite number of known basis functions, $\mathbf{x}(t) = \sum_i c_i \boldsymbol{\phi}(t)$, and solve for the optimal coefficients, c_i [13, 15, 25]

(iii) *spline velocity*: represent the trajectory in terms of velocity (i.e., a relative pose trajectory), which we still consider to be a weighted sum of a finite number of known basis functions: $\dot{\mathbf{x}}(t) = \sum_i c_i \boldsymbol{\phi}(t)$, and solve for the optimal coefficients, c_i [3]

(iv) *Gaussian process*: represent the robot trajectory nonparametrically as a Gaussian process, $\mathbf{x}(t) \sim \mathcal{GP}(\boldsymbol{\mu}(t), \boldsymbol{\Sigma}(t, t'))$ and solve for the pose at desired times [27, 29]

Regardless of the method, we solve for only a *finite* number of variables, in each case optimizing the robot trajectory based on the observed feature tracks. Our preferred approach is to do this online in a sliding-window style estimator where we estimate a small temporal section of the robot's trajectory (i.e., several seconds) and then slide the optimization window along to incorporate the next batch of measurements.

2.5 Celestial Attitude Corrections

While the continuous-time trajectory estimation approach is already quite accurate, we can also incorporate absolute attitude (i.e., orientation) corrections into our pose solution, as depicted in Fig. 2. As our motivation has been planetary exploration, we have primarily investigated celestial observations (with ephemeris, coarse location on the planet, and time/date) as a source of absolute attitude data.

In the daytime, we can use a dedicated sun sensor/inclinometer to provide attitude corrections very inexpensively [19]. Alternatively, we have found that it is actually possible with some laser-based imagers to use intensity/range images as a make-shift sun sensor [16]; the sun appears as a blob of points with maximum intensity and zero range (cf., Fig. 9; middle).

At nighttime, a small star tracker/inclinometer (cf., Fig. 9; right) is the preferred source of absolute attitude information. As star measurements can be provided frequently and during motion [17], they seem to be a natural pairing for lidar to support dark navigation.

Regardless of the source of absolute attitude measurements, we typically incorporate them into the VO pipeline by introducing additional error terms in Eq. (2). These celestial attitude sensors can be included at very little additional mass, power, and computational cost and are very beneficial to the accuracy of the VO solution over long distances [19].

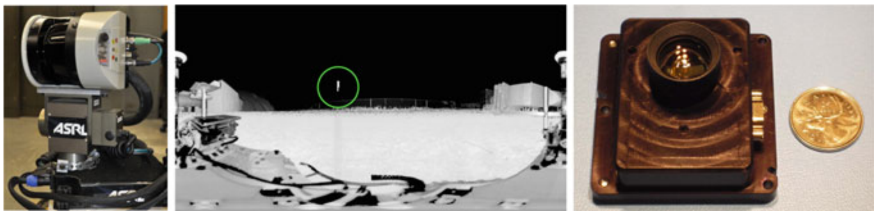


Fig. 9 Celestial/gravity observations can be used to correct rover orientation in a VO pipeline. In the daytime, we use the sun and either a dedicated sun sensor (not shown) or lidar intensity images [16]. For example, a SICK laser (*left*) was swept 360° to produce a panoramic intensity image (*middle*); the sun shows up as an artifact (*circled in green*) with maximum intensity and zero range. At nighttime, we use a small star tracker (*right*), which directly outputs full 3DOF orientation while the robot is motion [17]

3 Experimental Results

3.1 Setup

We gathered a large-scale lidar intensity image dataset at a sand and gravel pit near Sudbury, Ontario, Canada [4]. The ROC6 robot was equipped with the Autonosys lidar and DGPS for groundtruth positioning. The robot travelled the same 1.1 km course 10 times in a diurnal cycle, or approximately every 2.5 h for a 25 h period. Figure 10 depicts the experimental setup and the path the robot took based on DGPS groundtruth. The dataset is available for download from our webpage: http://asrl.utias.utoronto.ca/datasets/abl-sudbury.

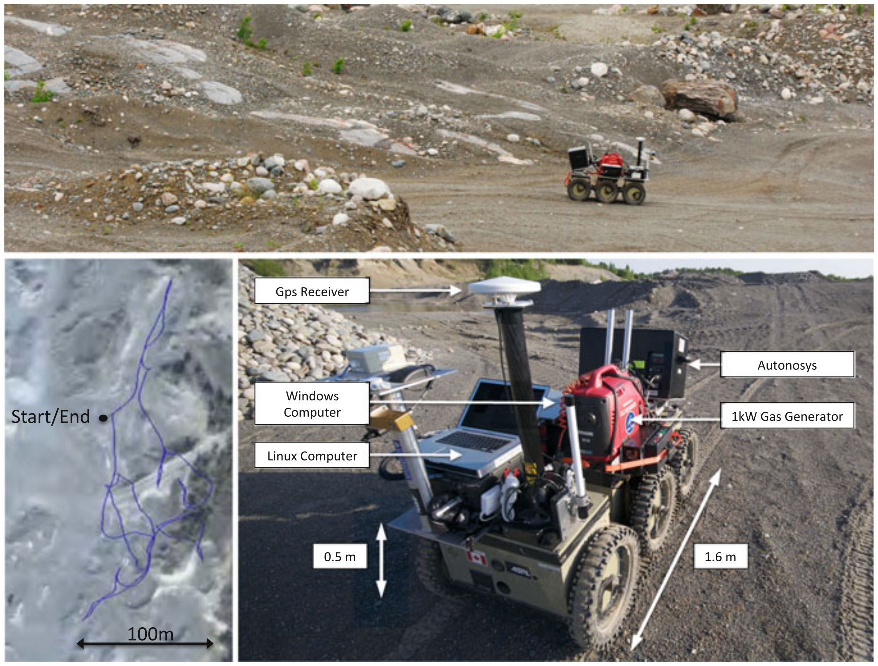


Fig. 10 We gathered 11 km of lidar-intensity-image data and DGPS groundtruth at the Ethier Sand and Gravel Pit near Sudbury, Ontario (*top*). The Autonosys lidar was mounted on the ROC6 field robot (*right*) and the same 1.1 km circuit (*left*) was repeated every 2.5 h for 25 h straight, ensuring data was gathered across an entire diurnal cycle (full-light to full-dark). The entire dataset is available on our webpage: http://asrl.utias.utoronto.ca/datasets/abl-sudbury [4]

3.2 Visual Odometry

Figure 11 shows an example of our motion-compensated visual odometry algorithm running on one of the *full-dark* 1.1 km circuits from our Sudbury dataset. It was a very cloudy night and therefore pitch black during the experiment. It should be noted that while the robot frequently revisited places it had been before, we are not detecting and exploiting loop closures in this experiment (i.e., we are not doing SLAM), merely dead-reckoning from sequential lidar data (in a sliding window).

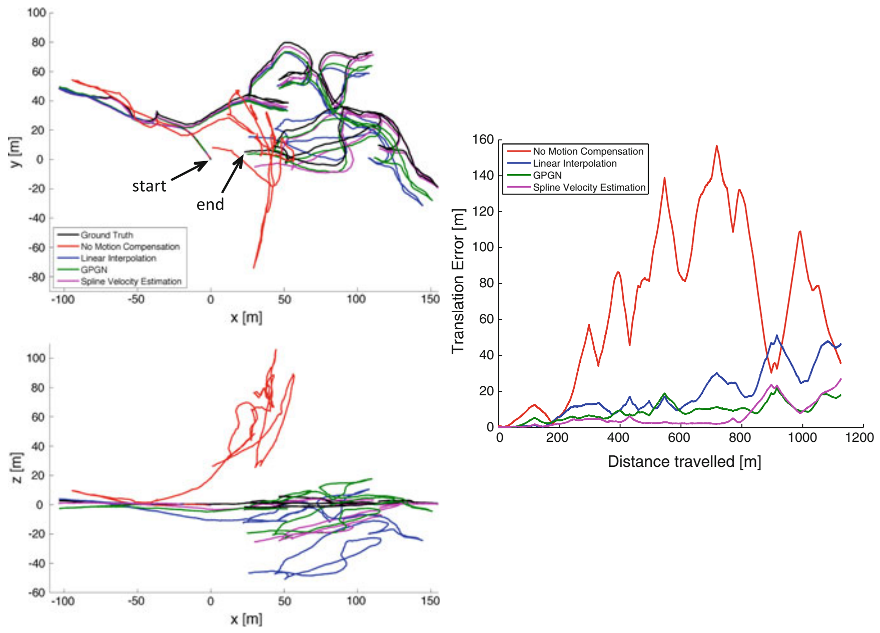


Fig. 11 VO results for one of the *full-dark* 1.1 km gravel-pit circuits, comparing all of the various motion-compensation strategies we have used over the last few years. Roughly speaking, all the methods are a big improvement compared to not compensating for the motion distortion. The spline-velocity estimation of Anderson and Barfoot [3] (*purple*) and the GPGN method of Tong and Barfoot [27] (*green*) do the best. Total Euclidean error (*right*) is much lower for the motion-compensated methods

The plot compares (i) GPS groundtruth, (ii) no motion compensation (i.e., traditional discrete-time VO), (iii) linear interpolation, (iv) Gaussian process Gauss-Newton (GPGN), and (v) spline velocity estimation (integrated after the fact to produce $\mathbf{x}(t)$). All the algorithms use the same VO pipeline, except for the last step involving the nonlinear, numerical pose solution. To provide a fair comparison we used our motion-compensated RANSAC feature tracks for all the algorithms; with traditional feature tracks the performance would be poor for all algorithms, even the discrete-time estimator.

We see some variability in performance across the algorithms, but there are different tuning parameters in each algorithm, making the comparison rough at best. At a high level, the total Euclidean error (cf., Fig. 11; right) shows all that the motion-compensated methods have much lower error than the traditional non-motion-compensated algorithm. On this particular dataset, the GPGN and spline velocity methods fared the best, with linear interpolation performing worse.

The motion compensation in the pose solution clearly helps and comes at little extra computational cost over the discrete-time estimator; we still do nonlinear, iterative least-squares with a robust cost function and estimate a similar number of variables, but the accuracy is higher.

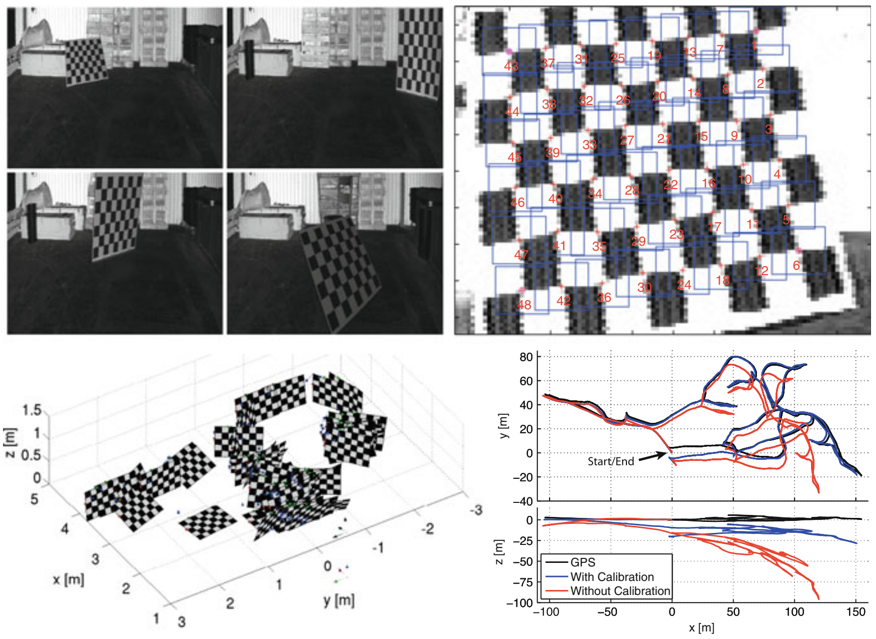


Fig. 12 We developed a lidar calibration tool inspired by the standard passive camera calibration approach. We present a number of views of checkboards to the lidar and capture intensity (*top-left*) and range images (not shown). We then automatically extract the locations of the checkboard corners from the intensity images (*top-right*). We simultaneously solve for the checkerboard poses and intrinsic parameters of the lidar (*bottom-left*). Calibration greatly improves the quality of our VO solution (*bottom-right*); comparison carried out using the spline-velocity approach

3.3 Lidar Calibration

As with any imaging sensor, our lidar-intensity-images require calibration to make the 3D positions of the landmarks accurate (cf., Fig. 12). This procedure is described by Dong et al. [9], but briefly the calibration uses multiple images of checkerboards at known locations to solve for intrinsic parameters by calculating the poses of the checkerboards. We effectively calibrate the azimuth, elevation, and range images in the lidar image stack of Fig. 3. Figure 12 (bottom-right) shows the effect of this calibration on the quality of our VO solution; we see that the calibration is just as important as motion compensation to producing an accurate VO solution.

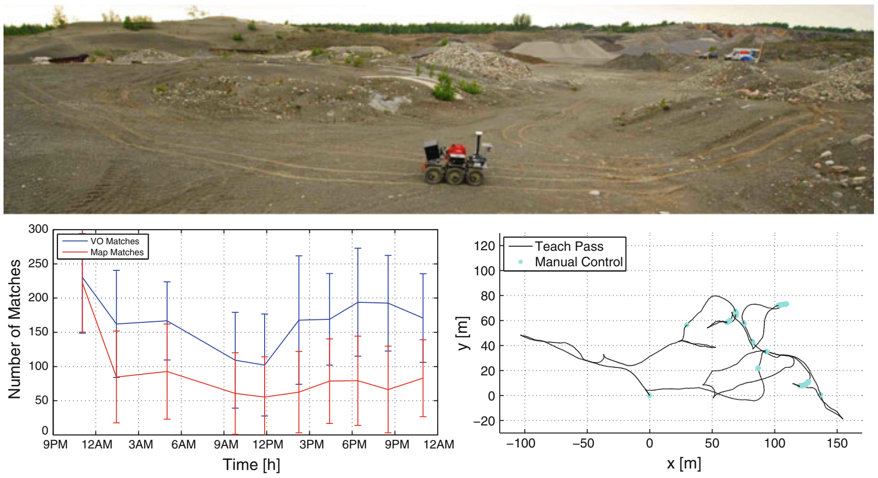


Fig. 13 Our visual-teach-and-repeat method was used to autonomously repeat the 1.1 km gravel-pit circuit 10 times. The path was taught in full daylight and repeated throughout an entire diurnal cycle. The method allowed the ROC6 robot to drive autonomously almost directly in its taught tracks (*top*) for 99.7 % of the distance. To localize relative to the path, the system matched features to the previous frame (VO) and to the map built during the teach pass; we see good numbers of features for all ten repeats (*left*), independent of the time of day. Only when both matching methods failed simultaneously were we required to exert manual control (*right*) to move the robot past small difficult sections (0.3 % of distance)

3.4 Visual Teach and Repeat

The second (and perhaps more important) experiment we will discuss is visual teach and repeat (VT&R). Chronologically, we carried out this experiment before our work to motion-compensate VO and so when we refer to the VO pipeline in this section, it is the basic, discrete-time version without motion compensation. In fact, the lidar dataset [4] described above was actually gathered as by-product of the VT&R experiment. It turns out for path repeating, the non-motion-compensated solution is almost good enough, but we decided to work on motion compensation primarily to improve the robustness of VT&R.

The idea behind VT&R is to pilot a robot manually along a route once to 'teach' it, and then to autonomously repeat the route many times. We accomplish this by running the VO pipeline during the teaching phase to estimate motion, but we save all of the features used to estimate VO, relative to the camera view from which they were first observed. During 'repeat', we match features from the live camera view to those stored in the map (as well as to the previous live view; cf., Fig. 2). This allows the robot to determine its pose relative to the taught path. A feedback controller then steers the robot to bring the path-tracking errors to zero over time. If the robot cannot match to the map, then VO is used to propagate the previous path-tracking errors forward in time. The end result is a robot that can drive directly in its taught tracks, using only visual feedback (i.e., no GPS).

We originally carried out this work using a stereo camera [14] before switching to lidar. However, we found that if the lighting changed too much between the teach and repeat phases, the robot would be unable to match its live view to the map reliably. This was the main reason we decided to explore using lidar intensity images, which can be matched across a wide variety of lighting conditions.

To demonstrate the lighting-invariant capabilities of our lidar pipeline, we taught a 1.1 km route in daylight (cf., Fig. 13; top) and then repeated it autonomously every 2.5 h for the next 25 h [24]. This meant we were matching light-to-light, light-to-dusk, light-to-dark, and light-to-dawn. Figure 13 (left) shows how many features we were on average able to match to the map (red) and previous image (blue) across all ten repeat runs; both numbers remain fairly constant. By distance, our system was 99.7 % successful, with the remaining 0.3 % requiring some minor manual interventions. Figure 13 (right) shows the union of the few places requiring manual interventions across all 10 repeat runs. Our average path-tracking error was about 8 cm RMS as measured by DGPS.

We found that using a VO pipeline without motion compensation inside our VT&R system meant we could not drive very far without matching to the map. We have yet to put our VO motion-compensation improvements back into VT&R, but believe this will further increase robustness by handling more of the cases where it is difficult to match to the map.

4 Conclusion and Future Work

We have discussed our experiences in building a visual pipeline based on lidar intensity images for both visual odometry and visual teach and repeat. Our major lessons learned along the way are:

(i) lidar intensity images offer excellent lighting invariance and can be used successfully in a visual pipeline,
(ii) lidar image stacks require careful calibration to achieve high-quality VO results (i.e., comparable to the stereo-camera pipeline),
(iii) the scanning-while-moving nature of laser-based imagers results in motion distortion that affects the accuracy of VO if left unchecked,
(iv) visual teach and repeat is possible even without motion compensation but will be more robust with it,
(v) it is possible to compensate for motion distortion in the RANSAC and pose solution steps of the VO pipeline,
(vi) it is possible to extract features from the raw intensity images, but this may no longer work if the motion distortion becomes too high,
(vii) absolute attitude corrections can be used to correct the robot's orientation and further improve the accuracy of the pipeline.

We believe our work shows not only that it is possible to build a VO pipeline that will work in the dark (and any other lighting condition), but that we can successfully match features across lighting conditions. We used this matching ability to build a lighting-invariant, visual-teach-and-repeat system, but we see this enabling other lighting-invariant robotics capabilities as well. For example, our next step is to do place recognition across lighting conditions. We hope that an affordable version of the lidar we used in this work becomes available within a few years, as we believe this could have a big impact in enabling real-world applications.

References

1. Abymar, T., Hartl, F., Hirzinger, G., Burschka, D., Frohlich, C.: Automatic registration of panoramic 2.5D scans and color images. In: Proceedings of the International Calibration and Orientation Workshop EuroCOW, vol. 54, Castelldefels, Spain (2007)
2. Anderson, S., Barfoot, T.D.: RANSAC for motion-distorted 3D visual sensors. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan (2013)
3. Anderson, S., Barfoot, T.D.: Towards relative continuous-time SLAM. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1025–1032, Karlsruhe, Germany (2013)
4. Anderson, S., McManus, C., Dong, H., Beerepoot, E., Barfoot, T.D.: The gravel pit lidar-intensity imagery datase. Technical report ASRL-2012-ABL001, University of Toronto (2012)
5. Bay, H., Ess, A., Tuytelaars, T., Gool, L.: SURF: speeded up robust features. Comput. Vis. Image Underst. (CVIU) **110**(3), 346–359 (2008)
6. Bohm, J., Becker, S.: Automatic marker-free registration of terrestrial laser scans using reflectance features. In: Proceedings of the 8th Conference on Optical 3D Measurement Techniques, pp. 338–344, Zurich, Switzerland (2007)
7. Brown, D.C.: A solution to the general problem of multiple station analytical stereotriangulation. Rca-mtp data reduction technical report no. 43, Patrick Airforce Base, Florida (1958)
8. Dold, C., Brenner, C.: Registration of terrestrial laser scanning data using planar patches and image data. In: Proceedings of the ISPRS Commission V Symposium 'Image Engineering and Vision Metrology', vol. XXXVI, Dresden, Germany (2006)

9. Dong, H., Anderson, S., Barfoot, T.D.: Two-axis scanning lidar geometric calibration using intensity imagery and distortion mapping. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 3657–3663, Karlsruhe, Germany (2013)
10. Dong, H.J., Barfoot, T.D.: Lighting-invariant visual odometry using lidar intensity imagery and pose interpolation. In: Proceedings of the International Conference on Field and Service Robotics (FSR), Matsushima, Japan (2012)
11. Droeschel, D., Holz, D., Stuckler, J., Behnke, S.: Using time-of-flight cameras with active gaze control for 3D collision avoidance. In: Proceedings of the IEEE International Conference on Robotics and Automation, Anchorage, Alaska, United States (2010)
12. Fischler, M., Bolles, R.: Random sample and consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)
13. Furgale, P., Rehder, J., Siegwart, R.: Unified temporal and spatial calibration for multi-sensor systems. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan (2013)
14. Furgale, P.T., Barfoot, T.D.: Visual teach and repeat for long-range rover autonomy. J. Field Robot., Special issue on visual mapping and navigation outdoors, **27**(5):534–560 (2010)(video1), (video2), (video3)
15. Furgale, P.T., Barfoot, T.D., Sibley, G.: Continuous-time batch estimation using temporal basis functions. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 2088–2095, St. Paul, USA (2012)
16. Gammell, J.D., Tong, C.H., Barfoot, T.D.: Blinded by the light: exploiting the deficiencies of a laser rangefinder for rover attitude estimation. In: Proceedings of the 10th Conference on Computer and Robot Vision (CRV), pp. 144–150, Regina, Canada (2013)
17. Gammell, J.D., Tong, C.H., Berczi, P., Anderson, S., Barfoot, T.D., Enright, J.: Rover odometry aided by a star tracker. In: Proceedings of the IEEE Aerospace Conference, pp. 1–10, Big Sky, MT (2013)
18. Kretschmer, U., Abymar, T., Thies, M., Frohlich, C.: Traffic construction analysis by use of terrestrial laser scanning. In: Proceedings of the ISPRS WG VIII/2 Laser-Scanners for Forest and Landscape Assessment, vol. XXXVI, pp. 232–236, Freiburg, Germany (2004)
19. Lambert, A., Furgale, P.T., Barfoot, T.D., Enright, J.: Field testing of visual odometry aided by a sun sensor and inclinometer. J. Field Robot., Special issue on Space Robotics, **29**(3):426–444 (2012) (video)
20. May, S., Fuchs, S., Malis, E., Nuchter, A., Hertzberg, J.: Three-dimensional mapping with time-of-flight cameras. J. Field Robot. **26**(11–12), 934–965 (2009)
21. McManus, C., Furgale, P.T., Barfoot, T.D.: Towards appearance-based methods for lidar sensors. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1930–1935, Shanghai, China (2011)
22. McManus, C., Furgale, P.T., Stenning, B.E., Barfoot, T.D.: Visual teach and repeat using appearance-based lidar. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 389–396, St. Paul, USA (2012)
23. McManus, C., Furgale, P.T., Barfoot, T.D.: Towards lighting-invariant visual navigation: an appearance-based approach using scanning laser-rangefinders. Robot. Auton. Syst. **61**, 836–852 (2013)
24. McManus, C., Furgale, P.T., Stenning, B.E., Barfoot, T.D.: Lighting-invariant visual teach and repeat using appearance-based lidar. J. Field Robot. **30**(2), 254–287 (2013)
25. Oth, L., Furgale, P., Kneip, L., Siegwart, R.: Rolling shutter camera calibration. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1360–1367 (2013)
26. Tong, C., Dong, H., Barfoot, T.D.: Pose interpolation for laser-based visual odometry. Submitted to the J. Field Robot., 27 May 2013. Manuscript # ROB-13-0044
27. Tong, C.H., Barfoot, T.D.: Gaussian process Gauss-Newton for 3D laser-based visual odometry. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 5184–5191, Karlsruhe, Germany (2013)

28. Tong, C.H., Furgale, P.T., Barfoot, T.D.: Gaussian process Gauss-Newton: non-parametric state estimation. In: Proceedings of the 9th Conference on Computer and Robot Vision (CRV), pp. 206–213, Toronto, Canada (2012)
29. Tong, C.H., Furgale, P.T., Barfoot, T.D.: Gaussian process Gauss-Newton for non-parametric simultaneous localization and mapping. Int. J. Robot. Res. **32**(5), 507–525 (2013)
30. Weingarten, J.W., Gruner, G., Siegwart, R.: A state-of-the-art 3D sensor for robot navigation. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robotics and Systems, vol. 3, pp. 2155–2160, Lasuanne, Switzerland (2004)
31. Ye, C., Bruch, M.: A visual odometry method based on the SwissRanger SR4000. In: Proceedings of the SPIE—Unmanned Systems Technology XII, vol. 7692 (2010)
32. Yuan, F., Swadzba, A., Philippsen, R., Engin, O., Hanheide, M., Wachsmuth, S.: Laser-based navigation enhanced with 3D time-of-flight data. In: Proceedings of the IEEE International Conference on Robotics and Automation, Kobe, Japan (2009)

Automatic Differentiation on Differentiable Manifolds as a Tool for Robotics

Hannes Sommer, Cédric Pradalier and Paul Furgale

Abstract Automatic differentiation (AD) is a useful tool for computing Jacobians of functions needed in estimation and control algorithms. However, for many interesting problems in robotics, state variables live on a differentiable manifold. The most common example are robot orientations that are elements of the Lie group SO(3). This causes problems for AD algorithms that only consider differentiation at the scalar level. Jacobians produced by scalar AD are correct, but scalar-focused methods are unable to apply simplifications based on the structure of the specific manifold. In this paper we extend the theory of AD to encompass handling of differentiable manifolds and provide a C++ library that exploits strong typing and expression templates for fast, easy-to-use Jacobian evaluation. This method has a number of benefits over scalar AD. First, it allows the exploitation of algebraic simplifications that make Jacobian evaluations more efficient than their scalar counterparts. Second, strong typing reduces the likelihood of programming errors arising from misinterpretation that are possible when using simple arrays of scalars. To the best of our knowledge, this is the first work to consider the structure of differentiable manifolds directly in AD.

1 Introduction

Computation of the Jacobian matrix of a nonlinear function is an essential part of many estimation and control algorithms and, as such, it is a ubiquitous task within robotics and computer vision. When faced with the task of implementing Jaco-

C. Pradalier (✉)
GeorgiaTech Lorraine, Metz, France
e-mail: cedric.pradalier@gatech.edu

H. Sommer · P. Furgale
ETH Zürich, Zürich, Switzerland
e-mail: Hannes.Sommer@mavt.ethz.ch

P. Furgale
e-mail: Paul.Furgale@mavt.ethz.ch

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_29

bian computation within a computer program, there are essentially four choices.[1] First, one can hand-compute the analytical expression. This is easy for simple functions, careful attention may be paid to the correct handling of singularities in the operations, and the computations may be hand-optimized for speed. However, it may become arduous to compute and verify Jacobians every time a small change is needed. Second, one may approximate the derivatives numerically. This is easy to implement but the resulting Jacobians may be inaccurate for highly nonlinear functions and finite differencing schemes can fail completely when the functions include conditional statements. Third, one may use symbolic differentiation tools to generate code from the nonlinear functions. The resulting Jacobians are accurate and highly efficient to evaluate but this method involves a pre-processing step and there is no guarantee that the automatically generated code correctly handles singularities. Finally, one may use Automatic Differentiation (AD) to compute the Jacobian matrices algorithmically. AD algorithms compute derivatives by exploiting the deterministic nature of derivative computation. The derivatives of atomic operations are implemented by the AD toolkit. The derivatives of more complex functions are constructed by applying the chain rule at each operation and bookkeeping the results. The application of AD is extremely easy using one of the many tools available.[2] Jacobians computed by this method are as accurate as their hand-coded counterparts but they are less efficient to compute and, again, may not handle all singularities.

In terms of accuracy and evaluation speed, hand coding or code generation are the clear choices. However, they share the common drawback that any change in the original function requires the Jacobian computation to be updated in lockstep. When prototyping, AD alleviates this requirement as changes in the nonlinear function are automatically reflected in the Jacobian computation. In our experience, researchers live continually in the prototyping phase and so, for problems in robotics focusing on estimation and control, AD is a tool to accelerate research.

However, for many interesting problems in robotics, state variables live on a Differentiable Manifold (DM). In robotics the most important DMs are the proper rotation and Euclidean groups in two- and three-dimensional Euclidean space ({SO(2), SO(3)} and {SE(2), SE(3)} respectively) for rigid-body kinematics, as well as the two-dimensional sphere, S^2, (e.g. for bearing vectors in sensor models [6]). Unit-length quaternions (elements of $S^0(\mathbb{H})$) are another popular choice for representing orientations.

The handling of state variables on DMs within estimation and control has been the subject of active research for many years in robotics and aerospace [1, 3–5, 12], but this analysis has not made it into the AD literature. The theory and implementation of AD is decidedly focused on the derivatives of *scalar operations* as the computational atoms [11]. It is possible to coax AD packages to compute the correct

[1]This paper will focus on the scale of problem generally encountered in estimation and control algorithms in robotics and not deal with methods used for larger-scale problems such as finding the Jacobian of a fluid simulation with respect to airfoil parameters (c.f. [7]).

[2]See http://www.autodiff.org/ for an extensive list of AD tool-kits spanning many popular programming languages.

Jacobians for functions that operate on elements of a DM, but this involves lifting the derivative computations to the the outer space. Working in the outer space precludes the possibility to utilize the special structure of the manifold to simplify derivative computation.

In this paper we extend the theory of AD to DMs by considering block operations as the atoms of computation, and exploiting the specific structure of the manifold to increase the efficiency of computation. Throughout this paper we will use expressions involving unit-length quaternions as our main example but the method is applicable to any DM. A pictorial representation of our contribution is presented in Fig. 1. This figure compares scalar AD with our approach by plotting the computation graph for a simple example in which a unit-length quaternion, q, is used to rotate a point, $\mathbf{v}$. We see the contributions of this work to be the following:

- We extend the theory of AD to operate on DMs. Because the atoms of computation become block operations, we call this Block Automatic Differentiation (BAD). To the best of our knowledge, this is the first work to make this extension.
- We present a prototype C++ implementation of BAD with a number of desirable properties. First, it is faster than AD that does not explicitly consider differentiation with respect to the underlying manifold. Second, rather that simply overloading operators on scalar or matrix types (a standard method of developing AD), we develop a type-safe system where every value belongs to a specific mathematical type with a well-defined tangent space for differentiation. This makes the system easier to use than standard scalar AD and guards against coding errors possible from the accidental misinterpretation of arrays of scalar types.

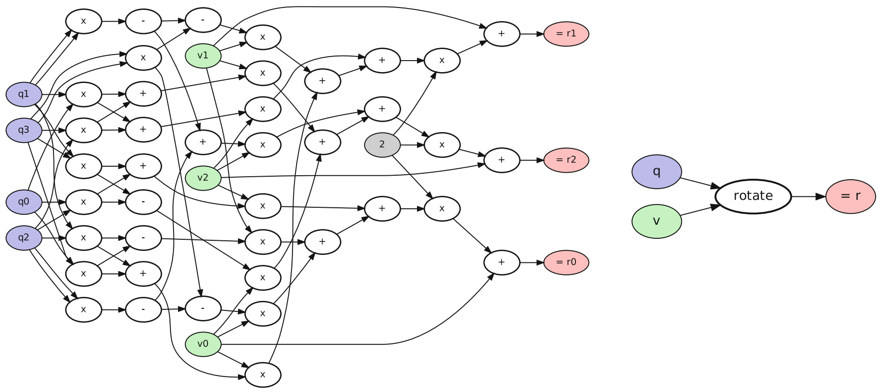


Fig. 1 A simple example in which a unit-length quaternion, q, is used to rotate a point, $\mathbf{v}$. *Left* The computational graph associated with the calculation from the point of view of standard scalar-focused AD algorithms. *Right* The computational graph from the point of view of our method. Although the evaluation of these two graphs is the same, computing the Jacobian from the left graph with a dual-number approach requires **168 multiplication and 156 additions**, whereas only **3 multiplications** are required by taking advantage of the DM structure on the right (see Sect. 2.3.2 for details)

- We perform a rigorous comparison of our method against the dual-number approach to AD implemented in the Google Ceres optimization package [2].

The rest of the paper is organized as follows. Section 2 reviews the background theory and presents the basis for our BAD algorithm. Section 3 describes our prototype implementation of BAD as a C++ library. Our experimental results are presented in Sect. 4 and we conclude in Sect. 5.

2 Theory

This section reviews the relevant theory on AD and DMs, and then describes our extension to bring the two concepts together.

2.1 Automatic Differentiation

An excellent introduction to Automatic (or Algorithmic) Differentiation is presented in [11] and [10]. This section attempts to summarize the content therein to bring the minimum of context on the topic. The interested reader is referred to these documents and the references therein for further details.

In the realm of computer science, AD is an ancient field, first developed to compute derivatives on specialized computers in the late 1960s. Widespread application of the technique came with the development of numeric tools in Fortran in the last two decades of the 20th century.

The basic idea of AD is easy to explain by imagining the evaluation of mathematical expressions represented by a *computation graph*. When interpreted by a compiler, a numeric expression is represented as a computation tree, with constants or variables as leaves and operators (‘$+$’,‘$-$’,...) and functions (cos, sin, exp, ...) as interior nodes. Because of sub-expressions being reused or compiler optimization, this tree becomes a computation graph (more precisely, a directed acyclic graph). Figure 1 depict such computation graph for rotating a 3D point with a quaternion.

The foundation of AD stems from the fact that the rules of differentiation are deterministic and they can be applied mechanically and recursively to the computation graph. Rather than deriving a formula for the derivative of an expression, the derivative is computed algorithmically by traversing the graph and using a combination of the chain rule and known operator differentiation rules. Traversing the graph from the leaves to the root is known as ‘forward’ AD and traversing it from the root to the leaves is known as ‘backward’ AD. Backward AD is more complex to implement and requires more storage but typically requires fewer operations, whereas forward AD can be implemented in a straightforward manner with operator overloading and/or dual numbers. In the latter case, the AD system evaluate the expression of interest, initially designed for numeric values, on a specific data structure grouping the expression value and its derivative(s) at each node of the graph. A recent example of such dual-numbers is the Jet class used by Google Ceres [2] to implement its AD feature.

In traditional languages not supporting operator overloading (e.g. Fortran 77), AD has been implemented by a pre-processor stage that would parse an expression in the original language and generate code to compute the expression and its derivative(s), to be compiled and linked in the final program.

Our contribution lies in the fact that adding knowledge about the underlying DM, we can dramatically simplify the computational graph (e.g. Fig. 1). By considering differentiation and the chain rule at the block level, we are able to exploit specific problem structure and increase computational efficiency. We compare our approach to the "dual-number" AD [9] implemented in Google Ceres [2].

The basic idea of the dual-number concept to do AD for scalar functions is to calculate the derivatives value alongside the functions values through the computation graph from the leaves to the root. This is done by applying all scalar operations to pairs of value v and derivative's value d $\langle v, d\rangle$. The derivative's value starts with 1, while the values start with the value the function is evaluated at. The scalar operations apply normally on the value parts, but a special corresponding operation to the derivative. For example $\langle v_1, d_1\rangle * \langle v_2, d_2\rangle := \langle v_1 v_2, d_1 v_2 + d_2 v_1\rangle$. To calculate gradients of functions in multiple scalar variables one simply keeps one second number for each variable and initializes each variable with the corresponding second number as 1 and the others as zero. For example $a * b$ one would then calculate the value and gradient at $a = 2, b = 3$ as follows : $\langle 2, 1, 0\rangle * \langle 3, 0, 1\rangle = \langle 2 * 3, 1 * 3 + 0 * 2, 0 * 3 + 1 * 2\rangle = \langle 6, 3, 2\rangle$. The gradient can then be extracted as the row vector of all the resulting second numbers (3, 2).

2.2 Differentiable Manifolds and Jacobians

The concepts of differential geometry have been adopted by the robotics community as the mathematical underpinnings of kinematics and dynamics. An excellent introduction is available in Murray et al. [8] but we will provide a brief overview of the concepts necessary to understand the idea of BAD.

For the scope of this paper a m-dimensional DM (with $m \in \mathbb{N}$) is a set, $\mathscr{M}$, together with an *m-atlas*, $A_{\mathscr{M}}$, inducing a second-countable Hausdorff topology on $\mathscr{M}$. To precisely define the notion of an atlas is beyond the scope of this paper, but informally $A_{\mathscr{M}}$ is a collection of *charts*, where each chart, $\varphi : U_\varphi \to \mathscr{M}$, is an invertible mapping defined on an open subset U_φ of $\mathbb{R}^m$ onto a subset of $\mathscr{M}$. The atlas provides a *differentiable structure* to the manifold $\mathscr{M}$. Using these charts, we can do differential calculus for functions between DMs, which include all finite dimensional vector spaces (by assigning DMs that reproduce the usual calculus on vector spaces).

2.2.1 Defining a Local Jacobian

Consider a mapping $f : \mathscr{M} \to \mathscr{N}$ between the m-dimensional DM $\mathscr{M}$ and the n-dimensional DM $\mathscr{N}$. At a point $p \in \mathscr{M}$, for f there exist a notion of differentiability and a formal local linearization, the *differential*, denoted with $\mathrm{d}_p f$. Its rigorous definition is also beyond the scope of this paper. Informally it plays the role of a derivative in a DM context. For algorithmic treatment, one requires a matrix representing this differential, but the usual Jacobian of nonlinear functions is only defined for mappings between vector spaces.

However, after choosing charts $\varphi_{\mathscr{M}}$ and $\varphi_{\mathscr{N}}$ around p and $f(p)$ respectively, one may define $\hat{\mathbf{f}} := \varphi_{\mathscr{N}}^{-1} \circ f \circ \varphi_{\mathscr{M}}$ at $\hat{\mathbf{p}} := \varphi_{\mathscr{M}}^{-1}(p)$, a mapping $\mathbb{R}^m \supset \mathrm{dom}(\varphi_{\mathscr{M}}) \to \mathbb{R}^n$, where $\mathrm{dom}(\varphi_{\mathscr{M}})$ denotes the domain that $\varphi_{\mathscr{M}}$ is defined on. We call f differentiable at p iff $\hat{\mathbf{f}}$ is differentiable at $\hat{\mathbf{p}}$ and define f's *Jacobian* in these charts with,

$$\mathrm{J}_p f := \mathrm{J}_{\hat{\mathbf{p}}} \hat{\mathbf{f}} = \mathrm{J}_{\varphi_{\mathscr{M}}^{-1}(p)} (\varphi_{\mathscr{N}}^{-1} \circ f \circ \varphi_{\mathscr{M}}). \tag{1}$$

These Jacobians can then play the same role in algorithms dealing with manifolds as the usual Jacobians do for nonlinear mappings between vector spaces. The algorithmic complexity to calculate them depends not only on f but also on the chosen charts. The latter feature is one of the underlying principles that BAD tries to exploit.

2.2.2 Defining a Global Jacobian

Usually manifolds are algorithmically represented as *embedded sub-manifold* of an *outer* $\mathbb{R}$*-vector space* $\mathscr{O}_{\mathscr{M}} := \mathbb{R}^{O_{\mathscr{M}}}$, with $O_{\mathscr{M}} > m$. This means informally that $\mathscr{M}$ is represented by a subset of $\mathscr{O}_{\mathscr{M}}$ for which the differentiable structure inherited from $\mathscr{O}_{\mathscr{M}}$ makes it a DM diffeomorphic to $\mathscr{M}$. Hence, points of $\mathscr{M}$ can easily be represented as the corresponding elements of $\mathscr{O}_{\mathscr{M}}$. A simple example is the Lie group of unit-length quaternions, a three-dimensional manifold that is often stored and manipulated as the S^3 sub-manifold of $\mathbb{R}^4$. In the following, we will assume that $\mathscr{M}$ and $\mathscr{N}$ are embedded sub-manifolds of $\mathbb{R}^{O_{\mathscr{M}}}$ and $\mathbb{R}^{O_{\mathscr{N}}}$ respectively.

This situation allows for a special way to acquire a Jacobian for f. One can choose a mapping $\tilde{\mathbf{f}} : V \to \mathscr{O}_{\mathscr{N}}$ defined on an open environment $V \subset \mathscr{O}_{\mathscr{M}}$ of p such that $f|_{\mathscr{M} \cap V} = \tilde{\mathbf{f}}|_{\mathscr{M} \cap V}$ and calculate the Jacobian $\mathrm{J}_p \tilde{\mathbf{f}} \in \mathbb{R}^{O_{\mathscr{N}} \times O_{\mathscr{M}}}$ of $\tilde{\mathbf{f}}$ instead. Here, $F|_A$ denotes the restriction of a mapping F on a set A. We will call $\mathrm{J}_p \tilde{\mathbf{f}}$ a *global Jacobian* of f. Note that it does not require a choice of charts but depends on the choice of $\tilde{\mathbf{f}}$ instead. For some applications this global Jacobian is already usable but for many it introduces instabilities and performance loss because it calculates a matrix that is bigger than necessary (recall that the *local* $\mathrm{J}_p f \in \mathbb{R}^{n \times m}$) introducing extra degrees of freedom. In those cases one can calculate $\mathrm{J}_p f$ in a second step after choosing charts from $\mathrm{J}_p \tilde{\mathbf{f}}$ by exploiting,

$$\varphi_{\mathscr{N}}^{-1} \circ f \circ \varphi_{\mathscr{M}}|_U = \vartheta \circ \tilde{\mathbf{f}} \circ \varphi_{\mathscr{M}}|_U \tag{2}$$

$$\implies \mathrm{J}_p f = \mathrm{J}_{f(p)}\vartheta \cdot \mathrm{J}_p\tilde{\mathbf{f}} \cdot \mathrm{J}_{\hat{\mathbf{p}}}\varphi_{\mathscr{M}}, \tag{3}$$

for an open set $U \subset \mathrm{dom}(\varphi_{\mathscr{M}}) \subset \mathbb{R}^m$ containing $\hat{\mathbf{p}}$, small enough, and a differentiable $\vartheta : \tilde{\mathbf{f}}(\varphi_{\mathscr{M}}(U)) \to \mathbb{R}^n$, such that $\vartheta|_{\mathrm{dom}(\varphi_{\mathscr{N}}^{-1})\cap\mathrm{dom}(\vartheta)} = \varphi_{\mathscr{N}}^{-1}|_{\mathrm{dom}(\varphi_{\mathscr{N}}^{-1})\cap\mathrm{dom}(\vartheta)}$.

Even though this introduces an unnecessary step in the algorithm, it is precisely the path suggested by a scalar-based AD. Because the manifolds are represented as embedded sub-manifolds and thus the algorithm evaluating f maps, in practice, elements of $\mathscr{O}_{\mathscr{M}}$ to elements of $\mathscr{O}_{\mathscr{N}}$. Hence, applying a scalar-based AD approach directly on this evaluating algorithm will calculate a global Jacobian, $\mathrm{J}_p\tilde{\mathbf{f}}$, typically concealing the fact that a choice of $\tilde{\mathbf{f}}$ does happen in this step.

The missing two Jacobians in (3) can either be calculated using scalar-based AD or by manually supplied algorithms. The latter is exactly the concept behind the current Ceres implementations when one uses its *local parametrization*, which plays here the role of the chosen chart $\varphi_{\mathscr{M}}$. As Ceres does not currently support manifold-valued error terms it can only be used in cases where $\mathscr{N}$ is a vector space and thus ϑ can be chosen trivial.

2.3 *Block Automatic Differentiation*

This section provides an overview of the BAD concept and a simple worked example showing the magnitude of speedup that is possible.

2.3.1 Overview

The motivation behind the development of BAD is to be able to inject knowledge from the specific structure of a DMs into the AD process in order to speed up the calculation of Jacobian matrices. The speedups gained by this approach will be specific to each manifold.

To enable an AD library to do that, it needs to know which DMs are involved in a mapping $f : \mathscr{M} \to \mathscr{N}$ that should be differentiated. To achieve that, the primary idea is to consider a computation graph on a mathematically higher level. Instead of primitive scalar operations $(+, *, \dots)$ on a single scalar type, we consider a set of basic operations that operate on points in manifolds. For example, this encompasses the usual vector and matrix operations, but also geometric operations like exponential maps, or special operations like rotation of a 3D point by a unit-length quaternion.

In practice most differentiable functions of interest between DMs can be deconstructed into a computation graph compounding such basic operations. These are the computational atoms that we think in when building up mathematical models and, in robotics, there is a surprisingly small set of such basic operations needed to implement many fundamental algorithms.

Such a high level computation graph can then be grouped by a BAD library into sub-blocks for which there is optimized (Jacobian) evaluation code. This optimized code can either be manually written or the output of a code generator of a symbolic toolkit.

In this way, the BAD concept is a mixture of manual, symbolic, and automatic differentiation, allowing a series of novel opportunities to optimize the computational complexity:

- to derive a suitable intermediate value-cache structure for the evaluation of the compound function and Jacobian evaluation;
- to automatically apply mathematically simplified algorithms for whole compound functions;
- to choose, based on the expression, the Jacobian evaluation direction or even mixtures of forward and backward steps; and
- to use highly optimized matrix manipulation libraries to do the final calculations.

However, there is one important obvious drawback when compared to AD: there are many more combinations of manifolds and basic operations on them than scalar operations. While it is easily possible to make a scalar AD library complete, a BAD library will never be. Because of this it is very important for a BAD library to be user extensible and to grow over time, eventually reverting to scalar AD as a last resort. To address this, our implementation efforts have been focused on building up a core framework that tries to make it as easy as possible to add support for manifolds and operations.

Ultimately, a BAD library will be less optimal than the output of an ideal symbolic tool optimizing the whole function f. Nevertheless, current symbolic tools have many important drawbacks compared to the BAD concept:

- the work flow from the mathematical model to the running code involves extra steps (i.e. code generation during the compilation phase) that may take much more processing time, especially when the expressions get quite complex;
- they are bad in factoring out repeated blocks to functions and thus produce huge code that is impossible to read and hard to maintain;
- they don't allow dynamic (at run-time) construction of the function f, which can be essential for special problems; and
- they usually do not incorporate matrix manipulation libraries and thus neglect a very important intermediate level of optimization.

2.3.2 Example

Here we provide the simple example of the rotation of a 3×1 vector, $\mathbf{v}$, by a rotation, $\mathbf{C}$, represented by a unit-length quaternion, q, to illustrate the potential of mathematical simplification of the Jacobian evaluation by exploiting knowledge about the underlying structures. Let v denote the pure imaginary quaternion corresponding to $\mathbf{v}$. The function we will analyze can then be defined as,

$$\mathbf{r} : S^0(\mathbb{H}) \to \mathbb{R}^3, \quad q \mapsto \operatorname{Im}(qv\bar{q}), \tag{4}$$

where multiplication of non-bold quantities is quaternion multiplication, the overbar denotes the quaternion conjugate operation, and $\operatorname{Im}(\cdot)$ extracts the imaginary components of the quaternion as a 3×1 vector. Here, $\mathbf{r}(\cdot)$ is only defined on the unit-length quaternions.

To be able to write an algorithm that evaluates this function or its Jacobian, we have to define how to represent the involved quantities. We will represent q with a 4-tuple (q_0, q_1, q_2, q_3) identified with q's coordinates in default ordered basis of $\mathbb{H}$, $(1, i, j, k)$. $\mathbf{v}$ and $\mathbf{r}(q)$'s value will be represented by the usual 3-tuples. To evaluate (4), the scalar calculations shown in Listing 1 can be executed (taken directly from the Ceres source code). (see Fig. 1 for the equivalent computation graph).

Listing 1 Algorithm evaluating $\mathbf{r}(q)$ defined in Eq. (4)

```
t1=-q3*q3; t2= q0*q1; t3= q0*q2; t4= q0*q3; t5=-q1*q1;
t6= q1*q2; t7= q1*q3; t8=-q2*q2; t9= q2*q3;
r0=2*((t8+t1)*v0+(t6-t4)*v1+(t3+t7)*v2)+v0;
r1=2*((t4+t6)*v0+(t5+t1)*v1+(t9-t2)*v2)+v1;
r2=2*((t7-t3)*v0+(t2+t9)*v1+(t5+t8)*v2)+v2;
```

These are $21m + 18a$ Floating Point Operation (FLOP)s for a single evaluation.[3]

To calculate the local Jacobian matrix $\mathrm{J}_q\mathbf{r}$ at an arbitrary $q \in S^0(\mathbb{H})$ using a dual-number approach, we follow the method utilizing the outer vector space encompassed by (3). To start we analyze the complexity of the first step in which one would calculate the Jacobian of $\tilde{\mathbf{r}} : \mathbb{H} \to \mathbb{R}^3$, defined by the pseudo code in Listing 1 (corresponding to $\tilde{\mathbf{f}}$ in Sect. 2.2). Using the dual-number approach, it is necessary to propagate four extra variables through each operation (one partial derivative per component of (q_0, q_1, q_2, q_3)) (see Sect. 2.1). For each extra variable, a multiplication in the original code requires an extra $2m + 1a$ and each addition requires an extra $1a$. This results in $21(2m + 1a) + 18a = 42m + 39a$ FLOPs per variable and $4(42m + 39a) = 168m + 156a$ for the full $4x3$ global Jacobian matrix. To build the local $\mathrm{J}_q\mathbf{r}$ we have to choose a chart around q. We will use the common exponential chart $\varphi_q : U \subset \mathbb{R}^3 \to S^0(\mathbb{H})$, $\mathbf{w} \mapsto \exp(w)q$, where w denotes the pure imaginary quaternion with vector part $\mathbf{w}$ and U is an environment of $\mathbf{0}$ small enough to make the map injective.

[3]Here, m represents multiplication operations and a represents addition operations. We ignore negations.

Its Jacobian at $\mathbf{w} = \mathbf{0}$ can be calculated from q with negations only. The remaining step is to multiply these matrices, making the total cost $9(4m + 3a) = 36m + 27a$, assuming a straightforward implementation.

Our approach would exploit the following simplification of the general directional derivative in direction $\mathbf{w} \in \mathbb{R}^3$ to calculate $\mathrm{J}_q\mathbf{r} = \mathrm{J}_\mathbf{0}(\mathbf{r} \circ \phi_q)$.

$$\begin{aligned} \mathrm{J}_\mathbf{0}(\mathbf{r} \circ \varphi_q) \cdot \mathbf{w} &= (\mathrm{d}_q \mathrm{Im}(qv\bar{q}))(wq) && (5a)\\ &= \mathrm{Im}(wqv\bar{q} + qv\overline{wq}) && (5b)\\ &= \mathrm{Im}(wqv\bar{q} + qv\bar{q}\bar{w}) && (5c)\\ &= \mathrm{Im}(wqv\bar{q} - qv\bar{q}w) && (5d)\\ &= \mathrm{Im}([w, qv\bar{q}]) && (5e)\\ &= 2\mathbf{w} \times \mathrm{Im}(qv\bar{q}) && (5f)\\ &= 2\mathbf{w} \times \mathbf{r}(q) && (5g) \end{aligned}$$

In the above, $[\cdot, \cdot]$ is the commutator of quaternions and $\times$ represents the cross product of 3×1 vectors. Note that the equations above just sketch the proof, sometimes taking advantage of concepts requiring more in-depth knowledge of DM and specifically of $\mathbb{H}$. For the sake of brevity we will stay at this level of details here.

To calculate the columns of $\mathbf{r}$'s Jacobian matrix, one would evaluate the last expression for w substituted with each default basis vector of $\mathbb{R}^3$. In these evaluations, the cross product becomes trivial (i.e. only requiring negations), reducing the FLOP count to $3m$ needed to scale the components by a factor 2.

The example above shows how we can go from $204m + 183a$ FLOPs to $3m$ FLOPs to calculate the Jacobian by choosing a beneficial local chart and injecting knowledge about the underlying DM and the outer space at the block level. Any manifold has the potential to benefit from this method based on its specific structure, or by choosing advantageous embeddings or tangent space basis vectors. For unit-length quaternions representing rotations, one would implement multiplication and inversion, along with the log and exponential map, resulting in a full-featured BAD system for manipulation of expressions for this specific DM. For any other manifold of interest, one would follow the same procedure, writing down the operations to be supported, deriving some efficient analytical expressions for the Jacobians induced by these operations, and implementing these as operations supported by the library.

3 Implementation

This section describes the usage of our implementation and compares it to the AD package provided by the Ceres optimizer.

Using the Ceres AutoDiffCostFunction to calculate the local Jacobian of the expression $\mathbf{r}(q)$, one would write the C++ code presented in Listing 2. Please note the necessary extra step on line 27–29 to convert the Jacobian in global coordinates (of the embedding space) to the Jacobian in local parametrization.

Listing 2 Calculating the local Jacobian of the function $\mathbf{r}(\cdot)$ with Ceres AutoDiffCostFunction

```
using namespace ceres;
struct Cv {
    const double *v;
    Cv(const double *v) : v(v) {}

    template <typename T>
    bool operator()(const T* const q, T* residuals) const
    {
        UnitQuaternionRotatePoint(q, v, residuals);
        return true;
    }
};

QuaternionParameterization quaternionParameterization;

void calcJacobian(const double *q, const double *v,
        double *result, double *qJ)
{
    const double *parameters[] = {q};
    double qGlobalJ[3 * 4];
    double *jacobians[] = {qGlobalJ};
    double qLocalParamJ[4 * 3];

    AutoDiffCostFunction<Cv,3,4> r(new Cv(v));

    r.Evaluate(parameters, result, jacobians);

    quaternionParameterization.ComputeJacobian(q,
        qLocalParamJ);
    internal::MatrixMatrixMultiply<3, 4, 4, 3, 0>(
        qLocalParamJ, 3, 4, localParamJ, 4, 3, qJ, 0, 0,
            3, 3);
}
```

Listing 3 illustrates how our approach could be used to solve the same problem. In this example, we use the "auto" keyword from C++ 2011 to simplify the code and let the compiler deduce the type of an expression.[4]

In line 6, the values pQ is converted into something one can later take a derivative with respect to (=Diffable). The Ref template only enforces capturing per reference (as also in line 7 with v an pV).
In line 9, the high-level computation graph converted to a type and becomes (thanks to *auto*) the type of r. The data of r will only contain the references to pQ and pV.
In line 11, an intermediate derived-value cache will be created. Its type will depend on r's type and essentially include storage to store intermediate values that could be needed repeatedly in the (Jacobian) evaluation (e.g. the conjugate of q or its rotation matrix counterpart). The resulting storage is embedded per reference into the cache, to have a unified storage for all values, without the need to copy the result data back.
In line 13, intermediate values that are needed will be calculated (e.g. always the final evaluation result—here the rotated vector). In line 14, the Jacobian is computed directly into the provided storage qJ.

[4]In fact without "auto" the whole concept would become quite cumbersome to use because the type names quickly become huge and unreadable. To allow the use of the library without knowing about these generated types is one of the big implementation challenges.

Listing 3 Using the BAD to calculate the local Jacobian of the function $\mathbf{r}(\cdot)$

```
using namespace tex; using namespace Eigen;
void calcJacobian(const UnitQuaternion & pQ,
        const EuclideanPoint<3> & pV,
        EuclideanPoint<3> & result, Matrix3d & qJ)
{
    Diffable<Ref<UnitQuaternion>, 0> q(pQ);
    Ref<EuclideanPoint<3>> v(pV);

    auto r = q.rotate(v);

    auto cache = createCache(r, result);

    cache.update(r);
    evalFullDiffInto(r, q, cache, qJ);
}
```

Because the mathematical types of all variables and operations in the expression are encoded in their C++ types, it is possible to do the following:

- to derive a suitable intermediate and derived value cache structure for the evaluation of this expression and especially its Jacobian;
- to automatically apply mathematically simplified algorithms (using template specialization and overload resolution to look them up) to evaluate the expression and especially its derivatives using this cache structure;
- to implement the Jacobian evaluation using (block) forward or backward evaluation (or some mixture of the two), whatever is the fastest for this particular fragment of the computation graph.

4 Experiments

In this section we describe two experiments comparing our BAD prototype implementation with the dual-number approach implemented by Ceres. In the first experiment, we compare the timing of Jacobian evaluations on increasingly large problems. In the second experiment, we use our approach on real-world data in a nonlinear optimization problem whose goal is to estimate the time-varying orientation of an elephant seal in a wildlife monitoring context.

4.1 Comparison with State-of-the-Art Automatic Differentiation

In the first experiment, we compared the AD implementation of Ceres to our prototype BAD implementation. To measure performance on an increasingly complex problem, we measured the time required to calculate the value and the Jacobian of expressions of the form,

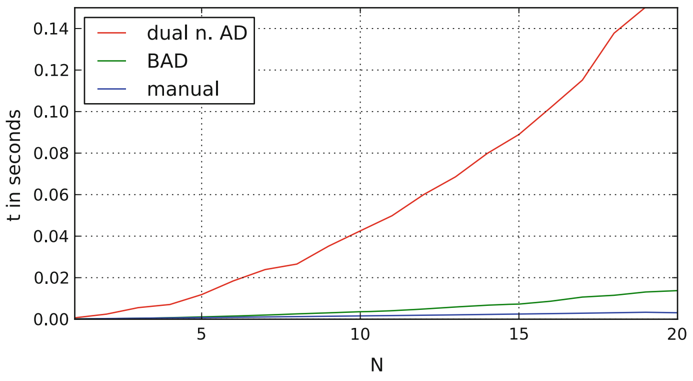


Fig. 2 Performance comparison for evaluating the value and Jacobian of $\prod_{i=1}^{N} \mathbf{C}_i\mathbf{v}$, as a function of N. The *red line* (dual n. AD) refers to the dual-number approach implemented by Ceres, the *green line* corresponds to BAD and the *blue line* is a hand-tuned version of the evaluation

$$\begin{matrix} \mathbf{C}_1\mathbf{v} & \\ \mathbf{C}_2\mathbf{C}_1\mathbf{v} & \quad \text{for} \quad N \in 1 \ldots 20 \\ \mathbf{C}_N \ldots \mathbf{C}_1\mathbf{v}, & \end{matrix} \tag{6}$$

where $\mathbf{C}_i$ is a rotation (represented in our case by a unit-length quaternion), and $\mathbf{v}$ is a 3×1 vector. The time required by Ceres (denoted as dual-number AD) and by our approach, as a function of the problem size, is shown in Fig. 2. On these artificial problems, there is an obvious benefit for using the specific structure of the DM to compute the Jacobian. However, we also compared it with a hand-tuned implementation of the Jacobian calculation. This results in the lowest curve (blue) in Fig. 2. Numerically, on this specific example, we find that BAD is 12 times faster than Ceres, but still 4 time slower than the hand-tuned evaluation, independently of the problem size.

4.2 *Application: Elephant-Seal Attitude Estimation*

In this section, we will compare the performance of the different optimization solutions on a specific application: the estimation of the attitude of elephant seals based on accelerometer and magnetometer recording collected over weeks or months with sensors attached to the animals while they are at sea. Although the data-set is peculiar and might seem far outside the field of robotics, the problem of batch attitude or position estimation from initial measurement unit is a common issue for underwater and flying robots.

The sensors attached to the animals record the following data at 1 Hz: depth, acceleration, magnetic field. A sound-based system estimates the animal linear velocity but due to energy saving considerations, this is switched on only for 3 h every 12 h.

At the surface, global positioning is retrieved from the ARGOS satellites. To illustrate the point of this paper, we will focus on using the accelerometer and the magnetometer to retrieve the animal attitude. In a later stage, this could be used in combination with the velocity measurement to estimate a 3D trajectory, or in combination with GPS data to estimate sea currents. In a real situation, it is also necessary to use the batch estimation process to estimate some sensor calibration parameters. All these extensions will be kept out of scope for this paper.

4.2.1 Problem Statement

Formally, the state we are estimating is the seal attitude as a rotation matrix C_t at time t and the propulsion force P_t it applies along the x axis in its body frame. To this end, we have the measurements of two constant vectors in the world frame: the acceleration $G = [0, 0, 9.81]^T$ and the magnetic field $B = [B_x, B_y, B_z]^T$. The exact value of the magnetic field can be found using the IGRF[5] magnetic field model, which depends on the time, the latitude and the longitude. For sake of simplicity, we will assume here that the reference magnetic field is constant between two GPS fix. We will denote b_t the magnetic field reported by the sensor in the body frame, and a_t the accelerometer output. Note that the measured acceleration results from the combination of gravity and the propulsion force $P_t \cdot \mathbf{x}$ applied by the seal, where $\mathbf{x}$ is the longitudinal axis of the animal.

With these variables we can build the following error terms for the accelerometer and magnetometer:

$$F_{acc}(t) = C_t \cdot [a_t - P_t \cdot \mathbf{x}] - G \quad \text{and} \quad F_{mag}(t) = C_t \cdot b_t - B \tag{7}$$

In addition, we can make some continuity hypothesis on the attitude and the propulsion and express them as the following error terms:

$$F_p(t) = P_t - P_{t-1} \quad \text{and} \quad F_C(t) = C_t \cdot C_{t-1} \tag{8}$$

The initial values used for the optimizer is propulsion at zero and Euler angles (roll, pitch and yaw) estimated independently from the accelerometer and magnetometer.

4.2.2 Comparison of Optimization Performance

Figure 3a, b shows the time, t, required to calculate the Jacobians necessary to solve the simplified seals optimization problem using single-threaded execution on a benchmark PC after loading the first N input lines of the measured data. In Fig. 3a, we measure the time (using the x86 RDTSC instruction) in total spent in the Jacobian calculation function per error term. In Fig. 3b, we show the time measured

[5]http://www.mathworks.fr/matlabcentral/fileexchange/28874-igrf-magnetic-field.

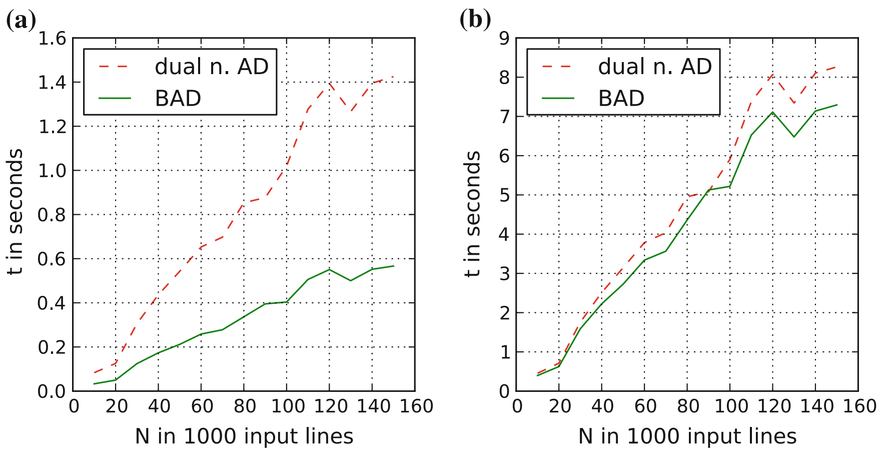


Fig. 3 Time required to compute the Jacobian as a function of the number of error terms for the simplified seal problem. On the *left*, only Jacobian computation time is included, on the *right*, the time for the creation of the full sparse Jacobian matrix is also included. See text for details

by the Ceres optimizer for the whole Jacobian evaluation step (with the clock used in Fig. 3a disabled). The solid line represents our implementation of BAD and the dashed one shows Ceres implementation of the dual-number approach. Both show the performance improvement by BAD. In Fig. 3a, BAD is approximately 2.5-times faster, in Fig. 3b it is only about 13.5 %. The explanation for this example is that Ceres spends a relatively large amount of time constructing the overall Jacobian for the complete optimization problem from the Jacobians of the individual error term, which are rather simple error terms. Ceres includes the full sparse Jacobian matrix construction in what it reports as the Jacobian calculation time.

To be complete, we must state here that the comparison here is slightly unfair in favor of Ceres for two reasons. Ceres expects global Jacobians (the natural output of a scalar AD) instead of the local Jacobians that they later calculate from these global ones by multiplying by the Jacobian of the local parametrization (see (3)). To fit with this requirement, we calculate $n \times 4$ matrices from our $n \times 3$ Jacobian and Ceres then calculates the $n \times 3$ back (n is the dimension of the error term). Each of these conversions requires a matrix multiplication (first 3×4 then 4×3). The first multiplication delays our implementation in Fig. 3. It would be fairer to skip these artificial steps but it would require changes in the Ceres code, which would further improve the performance of BAD in Fig. 3b. In addition, Ceres uses non-aligned Eigen::Vectors, while our implementation needs aligned ones, which requires us to copy the data into aligned vectors. The time spent copying these vectors is also counted in the data depicted in Fig. 3.

5 Conclusions

In this paper we have brought together the concepts of AD and DM to develop a block AD approach that we call BAD. This approach has the potential to be much more computationally efficient than traditional scalar AD by exploiting the specific structure of the DM. We presented a worked example of a unit-length quaternion rotating a 3×1 vector and showed how it can greatly reduce the number of instructions needed for Jacobian computation. Finally, we presented experimental results on simulated and real data that show that our prototype implementation of BAD is indeed faster than a state-of-the-art AD approach.

Our next step will be to finalize the interface and implement a full-featured BAD library in C++ encompassing the most common DMs and operations needed in robotics.

Acknowledgments Cédric Pradalier is supported by the European Commission within the Noptilus project, EC Grant agreement 270180. The elephant-seal data-set was kindly provided by Ch. Guinet and B. Picard from the CNRS Center for Biological Study in Chizé, France, based on data collected on the Kerguelen Islands, South Indian Ocean.

References

1. Absil, P.A., Mahony, R., Sepulchre, R.: Optimization Algorithms on Matrix Manifolds. Princeton University Press, Princeton (2009)
2. Agarwal, S., Mierle, K., Others: Ceres solver. https://code.google.com/p/ceres-solver/ (2013)
3. Barfoot, T.D., Forbes, J.R., Furgale, P.T.: Pose estimation using linearized rotations and quaternion algebra. Acta Astronaut. **68**(1–2), 101–112 (2011). doi:10.1016/j.actaastro.2010.06.049
4. Gwak, S., Kim, J., Park, F.C.: Numerical optimization on the euclidean group with applications to camera calibration. IEEE Trans. Robot. Autom. **19**(1), 65–74 (2003)
5. Hertzberg, C., Wagner, R., Frese, U., Schrder, L.: Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds. Inf. Fusion **14**(1), 57–77 (2013). doi:10.1016/j.inffus.2011.08.003. http://www.sciencedirect.com/science/article/pii/S1566253511000571
6. Lambert, Furgale: Barfoot, enright: field testing of visual odometry aided by a sun sensor and inclinometer. J. Field Robot. **29**(3), 426–444 (2012). doi:10.1002/rob.21412
7. Martins, J.R., Sturdza, P., Alonso, J.J.: The complex-step derivative approximation. ACM Trans. Math. Softw. (TOMS) **29**(3), 245–262 (2003)
8. Murray, R.M., Sastry, S.S.: A Mathematical Introduction to Robotic Manipulation. CRC Press, Boca Raton (1994)
9. Piponi, D.: Automatic differentiation, c++ templates, and photogrammetry. J. Gr. Tools **9**(4), 41–55 (2004)
10. Rall, L.B.: Perspectives on automatic differentiation: past, present, and future? Automatic Differentiation: Applications, Theory, and Implementations, pp. 1–14. Springer, Berlin (2006)
11. Rall, L.B., Corliss, G.F.: An introduction to automatic differentiation. In: Berz, M., Bischof, C.H., Corliss, G.F., Griewank, A. (eds.) Computational Differentiation: Techniques, Applications, and Tools, pp. 1–17. SIAM, Philadelphia (1996)
12. Shuster, M.D.: A survey of attitude representations. J. Astronaut. Sci. **41**(4), 439–517 (1993)

Minimal Solutions for Pose Estimation of a Multi-Camera System

Gim Hee Lee, Bo Li, Marc Pollefeys and Friedrich Fraundorfer

Abstract In this paper, we propose a novel formulation to solve the pose estimation problem of a calibrated multi-camera system. The non-central rays that pass through the 3D world points and multi-camera system are elegantly represented as Plücker lines. This allows us to solve for the depth of the points along the Plücker lines with a minimal set of 3-point correspondences. We show that the minimal solution for the depth of the points along the Plücker lines is an 8 degree polynomial that gives up to 8 real solutions. The coordinates of the 3D world points in the multi-camera frame are computed from the known depths. Consequently, the pose of the multi-camera system, i.e. the rigid transformation between the world and multi-camera frames can be obtained from absolute orientation. We also derive a closed-form minimal solution for the absolute orientation. This removes the need for the computationally expensive Singular Value Decompositions (SVD) during the evaluations of the possible solutions for the depths. We identify the correct solution and do robust estimation with RANSAC. Finally, the solution is further refined by including all the inlier correspondences in a non-linear refinement step. We verify our approach by showing comparisons with other existing approaches and results from large-scale real-world datasets.

G. Hee Lee (✉) · B. Li · M. Pollefeys
Department of Computer Science, ETH Zürich, Universitätstrasse 6,
8092 Zurich, Switzerland
e-mail: glee@student.ethz.ch

B. Li
e-mail: libo@student.ethz.ch

M. Pollefeys
e-mail: marc.pollefeys@inf.ethz.ch

F. Fraundorfer
Faculty of Civil Engineering and Surveying, Technische Universität München,
Arcisstrasse 21, 80333 Munich, Germany
e-mail: friedrich.fraundorfer@tum.de

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_30

1 Introduction

The pose estimation problem of a multi-camera system refers to the problem of determining the rigid transformation between the world frame and multi-camera frame, given a set of 3D points defined in the world frame and its corresponding 2D image points. In contrast with a single camera that has a single center of projection, the multi-camera system is an imaging sensor where light rays passing through the 3D world points and camera are non-central, i.e. the light rays do not meet at a single center of projection. An advantage of the multi-camera system is that it provides the flexibility to be set in a configuration which gives a maximum coverage of the environment. The solution to the pose estimation problem of a multi-camera system has important applications in robotics such as finding the initial camera pose estimates in structure-from-motion (SfM) / visual Simultaneous Localization and Mapping (SLAM), geometric verification and place recognition for loop-closures, and visual localization of a robot with respect to a given map that contains visual descriptors. Figure 1 shows our robotic car platform and the images from the multi-camera system mounted on it.

The fact that the light rays from a multi-camera system do not meet at a single center of projection means that all the classical approaches [6, 13, 17] for solving the perspective pose problem cannot be used. An alternative approach has to be proposed to handle the non-central nature of the multi-camera system. In addition, it is important that the proposed approach is a minimal solution and requires minimal correspondences that makes it efficient to be used within robust estimators such as RANSAC [5] (see Sect. 5 for more detail).

In this paper, we proposed a novel formulation to solve the pose estimation problem of a multi-camera system. In particular, we adopt the representation of non-central light rays from a multi-camera system with the Plücker line coordinates from

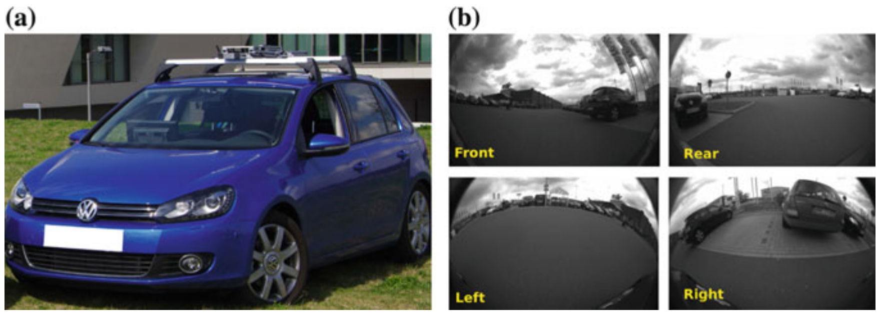


Fig. 1 **a** Our robotic car platform with a multi-camera system made up of 4 separate fish-eye cameras looking *front*, *rear*, *left* and *right* (cameras are embedded in the car logos and side mirrors). **b** Sample images from the 4 cameras

existing works [10–12, 16] for relative motion estimation of the multi-camera system. We show that this allows us do a two-step approach for solving the pose estimation problem—(a) solve for the unknown depth of the points along the Plücker lines and (b) compute the multi-camera pose from the known depths with absolute orientation [6, 8]. We show that with a minimal number of 3-point correspondences, it leads to an 8 degree polynomial minimal solution that yields up to a maximum of 8 real solutions for the unknown depths. Each of these possible solutions of the depth is used to compute the coordinates of the 3D world points in the multi-camera frame. The known 3D points in the multi-camera frame are used to compute the pose of the multi-camera system using absolute orientation.

The standard approach for solving the absolute orientation requires an expensive step of SVD and it is inefficient to perform the SVD multiple times to evaluate all the possible solutions of the depths. We circumvent this problem by deriving an efficient minimal solution for the absolute orientation, which allows us to compute the rigid transformation between the world and multi-camera frames from 3-point correspondences in closed-form without the need for SVD. Once we have obtained all the possible solutions for the rigid transformation, we compute the depths of all the other 3D world points. This allows us to choose the correct solution within a robust estimator such as RANSAC. Finally, the solution is further refined by including all the inlier correspondences in a non-linear refinement step that minimizes the reprojection errors (see Sect. 6 for more detail). We verify our approach by showing comparisons with other existing approaches and results from large-scale real-world datasets.

2 Related Works

The method proposed by Chen et al. [2] is most related to our method. In this work, they proposed a 3-point minimal solution and N-point solution to the multi-camera pose estimation problem. Similar to our method, their proposed solution is also a two-step approach. First, the coordinates of the 3D points in the multi-camera frame are determined. The 3D points in the multi-camera frame are determined by solving three distance parameters defined on the rays that passes through the 3D points. Next, the rigid transformation between the 3D points in the world and multi-camera frames is solved by absolute orientation. The formulation in the first step resulted in two 8 degree polynomials where a total of up to 16 real solutions are computed by root finding. In comparison, our method resulted in only one 8 degree polynomial that gives up to 8 real solutions, which has the advantage of less computational time needed to identify the correct solution. Another drawback of [2] is that the representations of the rays used to define the distance parameters breaks down when the three rays are respectively lying on parallel planes and in the case of linear pushbroom cameras [7] (see Sect. 4.3 for more detail). As a result, an alternative representation has to be made. In contrast, our representation of the rays as the Plücker lines is holistic and does not require any alternative formulation in any case. In addition, we also derive an efficient closed-form minimal solution for absolute orientation.

In [14], Nister proposed a formulation that directly solves for the rotation and translation parameters. His formulation gives an 8 degree polynomial minimal solution. This method is of special interest as the coefficients for the equation system can be computed with a low number of computations making it a fast method. He also proposed the use of Sturm sequencing for root finding and stated that the execution times is in the order of microseconds. The method is evaluated with simulations and compared to the single camera case. Similar to Nister's method, our method also ends up with an 8 degree polynomial minimal solution, which can also be solved with the Sturm sequencing to achieve the same execution time. Despite the computational efficiency, as also noted in [9], the derivation of Nister's method is not intuitive and requires laborious geometry and algebraic reasonings.

Kneip et al. presented that most recent work on pose estimation using a multi-camera system in [9]. In this work, the authors presented a 3-point minimal solution and N-point solution. They first solved for the rotations and point depths with a Gröbner Basis [3] solver followed by solving for the translation. They showed simulation experiments, comparisons to single camera perspective pose methods and a real-world visual odometry experiment using a two-camera setup. The exact process of solving the pose estimation problem with the Gröbner Basis approach is a black-box process which is not described in detail in [9]. Hence, Kneip's method cannot be reproduced easily. In comparison, our method is based on several algebraic equations which are intuitive and easy to implement. They mentioned that the generated solution from the Gröbner Basis solver has a length of 8000 lines of code and the execution time in the order of milliseconds. This makes it slower than Chen's, Nister's and our methods which solve an 8 degree polynomial that can be done in the order of microseconds as noted in Nister's paper [14].

In contrast to the minimal solvers for the pose estimation problem of the multi-camera system, there also exist linear [4] and iterative N-point [18, 19] solutions. The linear solution needs 6 or more point correspondences and thus less efficient in RANSAC [5] compared to our minimal solution which requires only 3 point correspondences. Since the iterative N-point solutions involves computationally expensive iterations, they are usually used to refine the pose estimation after all the inlier point correspondences have been found by RANSAC coupled with a minimal solution.

We adopt the Plücker lines representation for a multi-camera system from existing works on motion estimation [10–12, 16]. However, it is important to note that we adopt the Plücker lines representation to solve the multi-camera pose estimation problem, which is a completely different problem from the multi-camera motion estimation problem in [10–12, 16]. The objective of multi-camera motion estimation is to compute the relative transformation between two multi-camera frames given the 2D-2D image point correspondences, while the multi-camera pose estimation problem ask for the rigid transformation between a given world frame and the multi-camera frame given the 2D image point to 3D world point correspondences. To the best of our knowledge, no other work has adopted the Plücker lines representation to solve the multi-camera pose estimation problem.

3 Problem Definition

Figure 2 shows an illustration of the pose estimation problem of the multi-camera system. It is made up of multiple cameras denoted by (C_1, C_2, C_3) that are rigidly fixed onto a single body. Note that we show only 3 cameras in Fig. 2 but our proposed method works for any multi-camera system that has any number of cameras. Our method also works even if there was only 1 single camera (see perspective case in Sect. 4.3). We denote the reference frame of the multi-camera system and the world frame as F_G and F_W. The intrinsics and extrinsics of the respective cameras are assumed to be known from calibration and are denoted by K_i and $T_{C_i} = [R_{C_i}\ t_{C_i};\ 0\ 1]$ with respect to the multi-camera frame F_G, where $i = 1, 2, 3$. The pose estimation problem of a multi-camera system is defined as follows:

Definition 1 Given a set of three 3D points defined in F_W denoted by (X_1, X_2, X_3) that are seen by arbitrary cameras on the multi-camera system and their corresponding 2D image coordinates denoted by (x_1, x_2, x_3), find the rigid transformation R and t that brings the multi-camera frame F_G into the world frame F_W.

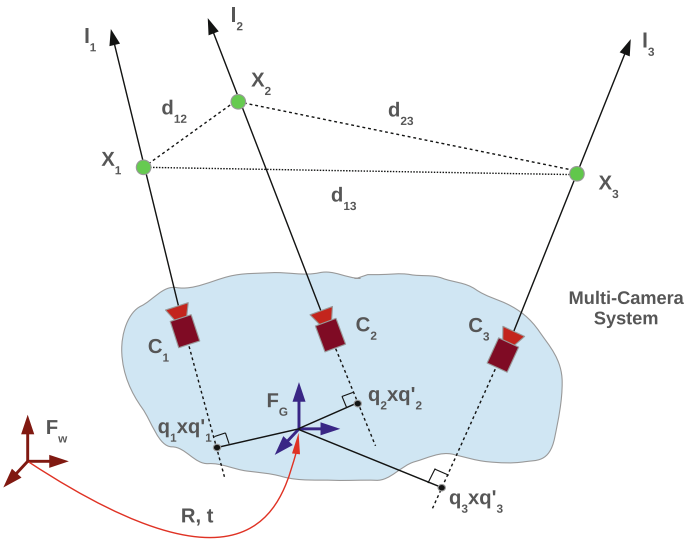


Fig. 2 Illustration of the pose estimation problem for a multi-camera system

4 Multi-Camera Pose Estimation

Figure 3 shows an illustration of our formulation for pose estimation of the multi-camera system. We first express the rays that join the respective three 2D-3D correspondences as Plücker line coordinates with respect to the multi-camera frame F_G (see Sect. 4.1 for more detail). Next, we solve for the unknown depths associated with each of the Plücker line using our minimal solution that leads to an 8 degree polynomial giving up to 8 real solutions (see Sect. 4.2 for more detail). Lastly, we compute the coordinates of the 3D points in the multi-camera frame F_G with the known depths and solve for the rigid transformation R and t between the world and multi-camera frames using our efficient minimal solution of absolute orientation in closed-form (see Sect. 4.4 for more detail).

4.1 Plücker Line Representation

We saw in Sect. 1 that the main problem with a multi-camera system is the absence of a single projection center for the camera. Following [16], we remove the need for a single projection center by representing the rays that pass through the 3D world points and the multi-camera system as Plücker line coordinates expressed in the multi-camera frame F_G. The Plücker line is a 6-vector $l_i = [q_i^T \ q_i'^T]^T$ where $i = 1, 2, 3$ as shown in Fig. 2. $q_i = R_{C_i}\hat{x}_i$ is the unit direction of the ray expressed in the multi-camera frame F_G where $\hat{x}_i = K_i^{-1}x_i$ is the normalized image coordinate of the point x_i. The closest point from the Plücker line to F_G is given by $q_i \times q_i'$ as shown in Fig. 2 and it is the point that forms a perpendicular intersection on the Plücker line from the multi-camera frame F_G. q_i' is defined as the cross product $q_i' = t_{Ci} \times q_i$. Any point X_i^G that is expressed in the multi-camera frame F_G is given by

$$X_i^G = q_i \times q_i' + \lambda_i q_i \tag{1}$$

where λ_i is the depth of the point X_i^G along the Plücker line, i.e. the signed distance from $q_i \times q_i'$ to X_i^G. Note that λ always has to be positive for the 3D point to appear in front of the camera.

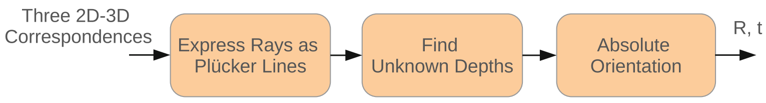


Fig. 3 Our formulation for the pose estimation of the multi-camera system

4.2 Minimal Solution for Depths

The distances d_{ij} where $(i, j) \in \{(1, 2), (1, 3), (2, 3)\}$ between the 3D points X_i in the world frame F_W shown in Fig. 2 have to be the same as the distances between the 3D points X_i^G in the multi-camera frame F_G , i.e.

$$||X_i - X_j||^2 = ||X_i^G - X_j^G||^2 \tag{2}$$

where $(i, j) \in \{(1, 2), (1, 3), (2, 3)\}$. By making use of the preservation of the 3D point distances given by Eq. 2 and the Plücker line equation from Eq. 1, we get three constraints

$$||X_i - X_j||^2 = ||(q_i \times q_i' + \lambda_i q_i) - (q_j \times q_j' + \lambda_j q_j)||^2 \tag{3}$$

where $(i, j) \in \{(1, 2), (1, 3), (2, 3)\}$ with three unknown depths λ_1, λ_2 and λ_3 from the Plücker lines. Expanding and rearranging the unknowns in Eq. 3, we get

$$k_{11}\lambda_1^2 + (k_{12}\lambda_2 + k_{13})\lambda_1 + (k_{14}\lambda_2^2 + k_{15}\lambda_2 + k_{16}) = 0 \tag{4a}$$

$$k_{21}\lambda_1^2 + (k_{22}\lambda_3 + k_{23})\lambda_1 + (k_{24}\lambda_3^2 + k_{25}\lambda_3 + k_{26}) = 0 \tag{4b}$$

$$k_{31}\lambda_2^2 + (k_{32}\lambda_3 + k_{33})\lambda_2 + (k_{34}\lambda_3^2 + k_{35}\lambda_3 + k_{36}) = 0 \tag{4c}$$

where k are the coefficients made up from the known Plücker line coordinates q_i and q'_i, and 3D world points X_i. We drop the full expressions of the coefficients for brevity. Using the Sylvester Resultant [3] to eliminate λ_1 from Eqs. 4a and 4b, we get a polynomial $f(\lambda_2, \lambda_3) = 0$ which is a function of only λ_2 and λ_3. We do another Sylvester Resultant on $f(\lambda_2, \lambda_3) = 0$ and Eq. 4c to eliminate λ_2, we get an univariate 8 degree polynomial dependent on only λ_3.

$$A\lambda_3^8 + B\lambda_3^7 + C\lambda_3^6 + D\lambda_3^5 + E\lambda_3^4 + F\lambda_3^3 + G\lambda_3^2 + H\lambda_3 + I = 0 \tag{5}$$

where A, B, C, D, E, F, G, H and I are coefficients made up from k from Eq. 4. The roots of Eq. 5 can be obtained from the eigen-values of the Companion matrix [3] made up of the coefficients. A maximum of up to 8 real roots can be obtained for λ_3. As suggested in [14], a more efficient way to solve for the roots of the 8 degree polynomial is by using the Sturm sequences.

λ_2 can be found by back-substituting λ_3 in Eq. 4c. After completing the square on Eq. 4c and making λ_2 the subject, we get

$$\lambda_2 = \frac{1}{2a}(-b \pm \sqrt{b^2 - 4ac}) \tag{6}$$

where $a = k_{31},\ b = k_{32}\lambda_3 + k_{33},\ c = k_{34}\lambda_3^2 + k_{35}\lambda_3 + k_{36}$. Similarly, λ_1 can be found by back-substituting λ_2 into Eq. 4a which takes similar form as Eq. 6 after completing the square and making λ_1 the subject. A total of up to 32 (i.e. $8 \times 2 \times 2$) solution triplets of λ_1, λ_2 and λ_3 can be obtained. A solution triplet is discarded if any one of the λs is an imaginary or negative value. A further step to disambiguate the solutions is by doing a redundancy check on λ_1 using Eq. 4b. The solution pairs of λ_2 and λ_3 should produce consistent λ_1 from both Eqs. 4a and 4b. Any solution pair of λ_2 and λ_3 which produces λ_1 with discrepancy from Eqs. 4a and 4b is discarded. In our simulations, we observed that these disambiguation checks are capable of reducing the maximum number of solutions to two for most of the time. All the other existing 2D-3D point correspondences are used to identify the correct solution within RANSAC, i.e. the correct solution yields the highest number of inliers in RANSAC.

4.3 Special Cases

In this section, we look at five special cases for the multi-camera pose estimation problem mentioned in [2]. In particular, we compare the similarities and differences between the existing methods and our method under these five different cases.

Case 1: Partially Parallel. Two out of the three light rays are parallel in this case as illustrated in Fig. 4. This means that two of the unit directions must be equal, i.e. $q_1 = q_2 \neq q_3$. From Fig. 4, we can see that $\lambda_2 = \lambda_1 + c_{12}$, where c_{12} is a known value from the Plücker line coordinates and distance between the 3D points (X_1, X_2). Applying the Sylvester Resultant for variable elimination together with Eqs. 4b and 4c, we get a 4 degree polynomial minimal solution that can be solved in closed-form. Similar 4 degree polynomial minimal solution was obtained for Chen's and Nister's methods.

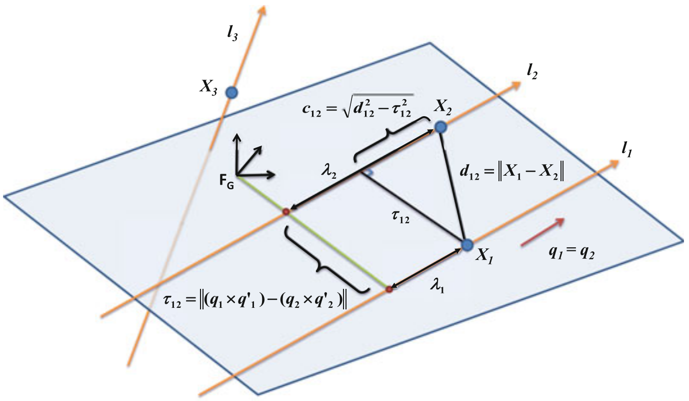


Fig. 4 Illustration of the partially parallel case

Case 2: Perspective. The three light rays pass through a common center of projection in the perspective case, i.e. all the 2D-3D correspondences are from one single camera in the multi-camera system. Let us choose the camera reference frame to be the center of projection. This implies that the camera extrinsics become $t_{C_1} = t_{C_2} = t_{C_3} = 0$ and $R_{C_1} = R_{C_2} = R_{C_3} = I$. Substituting these values into Eq. 3 and applying the Sylvester Resultant for variable elimination, we get a 4 degree polynomial minimal solution that can be solved in closed-form. This result is similar to Chen's and Nister's method, and the P3P solution for a perspective camera [6]. Note that a 4 degree polynomial is obtained even if the reference frame was not chosen as the center of projection of the camera.

Case 3: Parallel Plane. This is the case where the three light rays lie on three different planes that are parallel to each other. It is important to note that these light rays however do not have the same unit direction, i.e. $q_1 \neq q_2 \neq q_3$ from the Plücker lines. It can be observed that the constraints from our method in Eq. 3 does not break down. In contrast, the representations of the rays used by Chen et al. [2] to define the distance parameters cannot be used in the case where all the three rays respectively lie on parallel planes. As a result, an alternative representation has to be made.

Case 4: Linear Pushbroom. There is only one camera in the case of linear pushbroom [7]. Here, the camera moves through a straight line of motion with a known speed and takes images at regular intervals. Hence, the transformations between any three camera locations (similar to the extrinsics of a multi-camera system) are known and the rays that observed unique 3D world points from these locations lie on parallel planes. This implies that the linear pushbroom case is the same as the parallel plane case where our method does not break down. In comparison, an alternative representation has to be made for Chen's method.

Case 5: Orthographic. For orthographic projection, all the light rays are parallel. Hence, all the unit directions of the Plücker lines are equal, i.e. $q_1 = q_2 = q_3$. Similar to the partially parallel case, we have the following 3 constraints $\lambda_2 = \lambda_1 + c_{12}$, $\lambda_3 = \lambda_1 + c_{13}$ and $\lambda_3 = \lambda_2 + c_{23}$, where infinite solutions exist for λ_1, λ_2 and λ_3. Intuitively, we can move the multi-camera system anywhere along the direction of the parallel light rays and the constraints are still fulfilled, hence infinite solutions. This degeneracy is independent of the formulation and holds for all works [2, 4, 9, 14, 18, 19] on pose estimation for the multi-camera system.

4.4 Minimal Solution for Absolute Orientation

Absolute Orientation can be solved using the methods from [6, 8]. However, these methods require a computationally inefficient step of SVD which becomes an overhead when it is used numerous times within RANSAC to compute all the hypothesis solutions. We present a minimal solution which allows us to compute the absolute orientation in closed-form without the need for SVD. The proposed method computes the transformation R and t to align the two point sets P and Q consisting of three correspondence 3D points as shown in Eq. 7.

$$P_i = RQ_i + t, \quad i = 1,2,3 \tag{7}$$

First, two local frames F_M and F_N are defined on the point sets P and Q respectively. The origins of the local frames are defined on the first points, i.e. P_1 and Q_1. We can now write the transformed points in the newly defined local frames F_M and F_N as $M_i = P_i - P_1$ and $N_i = Q_i - Q_1$. Next, we define the x-axis of each local frame to pass through the second point respectively, i.e. M_2 and N_2. The x-axis of F_M and F_N can be aligned by applying the following transformations

$$M_2 = \begin{bmatrix} M_{2x} \\ M_{2y} \\ M_{2z} \end{bmatrix} = R_M \begin{bmatrix} \|M_2\| \\ 0 \\ 0 \end{bmatrix}, \; N_2 = \begin{bmatrix} N_{2x} \\ N_{2y} \\ N_{2z} \end{bmatrix} = R_N \begin{bmatrix} \|N_2\| \\ 0 \\ 0 \end{bmatrix} \tag{8}$$

where R_M and R_N are unknown rotation matrices that align the two x-axis. Here, we only describe the steps to solve for R_M since R_N can be computed in an analogous fashion. Since the alignment of the x-axis only involves two rotations around the y- and z-axis, R_M can be written as

$$R_M = R_{Mz} R_{My} = \begin{bmatrix} ce & -f & de \\ cf & e & df \\ -d & 0 & c \end{bmatrix} \tag{9}$$

where c and d are sine and cosine of the rotation angle around the y-axis, and e and f are sine and cosine of the rotation angle around the z-axis. Putting Eq. 9 into Eq. 8, we get the following three constraints

$$\|M_2\|ce - M_{2x} = 0 \tag{10a}$$

$$\|M_2\|cf - M_{2y} = 0 \tag{10b}$$

$$-M_{2z} - \|M_2\|d = 0 \tag{10c}$$

where d can be calculated directly from Eq. 10c and c can then be computed with the Pythagoras identity. e and f can be solved by substituting c into Eqs. 10a and 10b. The full expressions for solving a, b, c and d are given as follows

$$d = \frac{-M_{2z}}{\|M_2\|}, \; c = \sqrt{1-d^2}, \; e = \frac{M_{2x}}{\|M_2\|c}, \; f = \frac{M_{2y}}{\|M_2\|c} \tag{11}$$

Finally, we apply R_M and R_N to align the x-axis of both point sets. The new sets of transformed points are given by

$$U_i = R_M^T M_i, \; V_i = R_N^T N_i, \quad i = 1, 2, 3 \tag{12}$$

The last step is to find the remaining rotation R_V around the x-axis which would complete the alignment of the two local frames F_M and F_N. This gives the constraint in Eq. 13 which can be expanded into three independent constraints in Eqs. 14a–14c.

$$U_3 = R_V V_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & a & -b \\ 0 & b & a \end{bmatrix} V_3 \tag{13}$$

$$V_{3x} - U_{3x} = 0 \tag{14a}$$

$$V_{3y}a - U_{3y} - V_{3z}b = 0 \tag{14b}$$

$$V_{3z}a - U_{3z} - V_{3y}b = 0 \tag{14c}$$

where a and b are sine and cosine of the rotation angle. $U_3 = [U_{3x}\ U_{3y}\ U_{3z}]^T$ and $V_3 = [V_{3x}\ V_{3y}\ V_{3z}]^T$. We do variable elimination on Eqs. 14b and 14c to solve for a which can be back-substituted to solve for b. The full expressions for a and b are

$$a = \frac{U_{3y}V_{3y} + U_{3z}V_{3z}}{V_{3y}^2 + V_{3z}^2},\ b = \frac{-U_{3y} + V_{3y}a}{V_{3z}} \tag{15}$$

Finally, the full transformation R and t is given by

$$R = R_N^T R_V R_M,\ t = -RP_1 + Q_1 \tag{16}$$

5 Robust Estimation

Outlier 2D-3D point correspondences are rejected from our proposed method using RANSAC [5]. We compute the reprojection errors of all the 2D-3D point correspondences based on the hypotheses generated from random sets of unique 3-point correspondences. The hypothesis that yields the highest inlier count, i.e. highest number of 2D-3D point correspondences with the respective reprojection error lower than a given threshold, is chosen as the correct solution. As defined in [5], the number of RANSAC iterations needed is given by $\eta = \frac{\ln(1-p)}{\ln(1-w^n)}$, where p is the probability that all selected correspondences are inliers, w is the probability that any selected correspondence is an inlier and n is the number of correspondences needed for the hypothesis. Assuming that $p = 0.99$ and $w = 0.5$, a total of 35 iterations are needed for our 3-point algorithm, i.e. $n = 3$. In contrast, the linear 6-point algorithm [4] where $n = 6$ would require 293 iterations. The efficiency in having less iterations within RANSAC highlights the importance of using the minimal number of point correspondences.

Each hypothesis generated by RANSAC often give rise to more than one real solution from solving the polynomial equation in Sect. 4.2. We do additional iterations within RANSAC to check the inlier count for each of these solutions from each hypothesis, where the correct solution gives the highest inlier count. It is therefore desirable to have the minimal solution to keep the number of additional RANSAC iterations low. The number of additional RANSAC iterations for our method is halved compared to Chen's method [2] since our method has a minimal solution up to 8 real solutions while Chen's method yields up to 16 real solutions.

6 Non-Linear Refinement

We further refine the estimated pose R and t by minimizing the total reprojection errors from all the inlier point correspondences found from RANSAC. The cost function is given by

$$\underset{R,t}{\operatorname{argmin}} \sum_i \sum_j ||\pi(P_i, X_j) - \mathbf{x}_{ij}||^2 \tag{17}$$

where $\mathbf{x}_{ij}$ is the 2D image point with X_j as its 3D point correspondence and seen by the ith camera C_i that makes up the multi-camera system. $\pi(.)$ is the camera projection function that projects a 3D point onto the 2D image. P_i is the camera projection matrix given by

$$P_i = K_i[R_{C_i}^T R^T \quad -R_{C_i}^T(R^T t + t_{C_i})] \tag{18}$$

where K_i is the camera intrinsics, R_{C_i} and t_{C_i} are the camera intrinsics as defined in Sect. 3. The minimization of Eq. 17 is done with the Google Ceres solver[1] using the Levenberg-Marquardt algorithm.

7 Results

We evaluate our proposed multi-camera pose estimation algorithm with both simulations and large-scale real-world datasets.

7.1 *Simulations*

We compare the accuracy and stability of our algorithm with Nister's [14], Chen's [2] and Kneip's [9] algorithms based on the simulation setup suggested in [17]. The simulated multi-camera system is made up of 4 separate cameras looking front,

[1] http://code.google.com/p/ceres-solver/.

right, left and right with no overlapping field-of-views. Note that the chosen camera configuration and simulated rays are free from the parallel ray degeneracy mentioned in Sect. 4.3. The absolute orientation used in Chen's method is from [6] while the minimal solution proposed in Sect. 4.4 is used in our method.

We randomly generate a ground truth camera pose within a given range of $[-1\ 1]$ m for (x, y, z) and $[-0.1\ 0.1]$ rad for all angles, i.e. roll, pitch and yaw. We also randomly generate three 3D world points within a given range of $[-10\ 10]$ m for (x, y, z). The image coordinates are found by reprojecting the 3D points into the respective camera where it is visible. We corrupt the image coordinates with noise ranging from 0.1 to 1 pixel with a 0.1 pixel interval. The pose of the camera in the world frame is computed based on the corrupted image coordinates using the four algorithms. Following [17], we compute the relative translational error as $2||t_{est} - t_{gt}||/(||t_{est}|| + ||t_{gt}||)$ where t_{est} and t_{gt} are the estimated and ground truth translations. The relative rotational error is computed as the norm of the Euler angles from $R_{est} R_{gt}^T$ where R_{est} and R_{gt} are the estimated and ground truth rotation matrices.

Figure 5a, b shows the plots of the average relative translational and rotational errors from 500 random trials per image coordinate noise level. It can be seen that Chen's and our algorithms produced very similar errors for all noise levels. The errors from both Chen's and our algorithms are also significantly lower than Nister's and Kneip's algorithms. The results imply that the two-step approaches, i.e. Chen's and our algorithms, that solves for the depths and absolute orientation are less susceptible to the influences of noise compared to Nister's direct approach and Kneip's Gröbner basis method.

Time Efficiency of Minimal Solution for Absolute Orientation: We compare the time efficiency of our minimal solution for absolute orientation proposed in Sect. 4.4 with the standard approach that requires SVD [6, 8]. We randomly generate 500 camera poses in the world frame. For each of these poses, we randomly generate 3

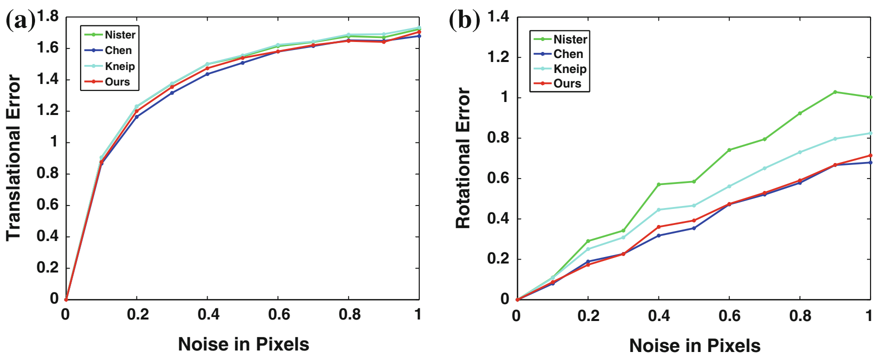


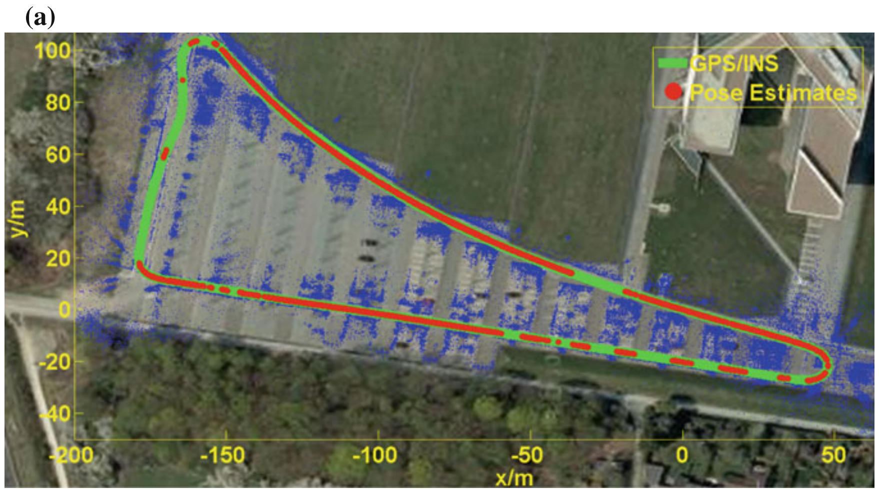
Fig. 5 Average (**a**) translational and (**b**) rotational errors from 500 random trials at different pixel noise levels using Nister's [14], Chen's [2], Kneip's [9] and our algorithms. Note that a large part of the translational error for Nister's method (*green line*) is hidden behind the translational error for Kneip's method (*cyan line*)

points in the world frame and compute the coordinates of these points in the camera frame. The rigid transformation between the camera and world frames is computed with both approaches. Note that the poses estimated from both methods are always the same as the groundtruth and there is no need for RANSAC in this comparison since the points are noise-free. We obtain the time needed for each trial with the respective method and compute the efficiency of our minimal solution for absolute orientation method over the SVD method as the ratio of the mean time taken by our method to the mean time taken by the SVD method for all the 500 trials. The efficiency ratio is found to be 1.23 and this means that on the average our proposed method is 1.23 times faster than the standard SVD approach.

7.2 Real Datasets

Figure 1 shows our car platform with 4 fish-eye cameras looking front, rear, left and right with minimal overlapping field-of-views used to collect the datasets for testing our algorithm. The GPS/INS system is also available for ground truth. Figure 1b shows 4 sample images from the respective cameras. Figures 6a and 7a shows two areas for testing our algorithm. TestArea01 and TestArea02 are car parks besides an office building and a supermarket, and covers an area of approximately 140×280 m and 160×150 m respectively. We collect two datasets separately from each of the test area, i.e. 2×2 datasets—one for building a map and the other for testing our pose estimation algorithm on the map in each test area. To build the maps, we extract the SURF [1] features, and triangulate the 3D points based on the GPS/INS readings. We apply bundle adjustment (implemented with Google Ceres solver) on the GPS/INS poses and triangulated 3D points to get the final maps. The maps also contains all the 2D-3D correspondences of the SURF and 3D points. The blue dots on Figs. 6a and 7a are the 3D points from the maps after bundle adjustment.

The green trajectories on Figs. 6a and 7a are the GPS/INS ground truth readings from the second datasets for testing our pose estimation algorithm on both areas. A total of 2500 and 2100 frames are used for testing. We first create a vocabulary-tree [15] with all the SURF features from the map. For every frame from the test dataset, we extract the SURF features, and query for the frame from the map with the highest similarity score with the vocabulary-tree. We obtain the 2D-3D correspondences of the test and map frames by matching the SURF features. Finally, we compute the pose of the test frame in the map with our multi-camera pose estimation algorithm. Note that a frame refers to a set of 4 images from all the cameras. The red dots on Figs. 6a and 7a are the estimated poses with our algorithm with at least 20 2D-3D correspondences. It can be seen that the poses estimated from our algorithm follows the GPS/INS ground truth closely. Figures 6b and 7b show the distributions of the translational and rotational errors. We can see that the error distributions are sufficiently low. The translational error is computed as $||t_{est} - t_{gt}||$ where t_{est} and



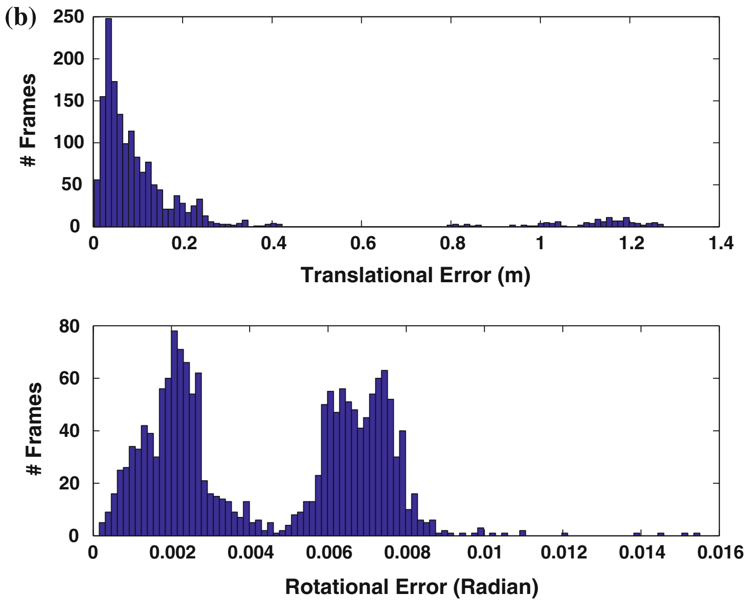
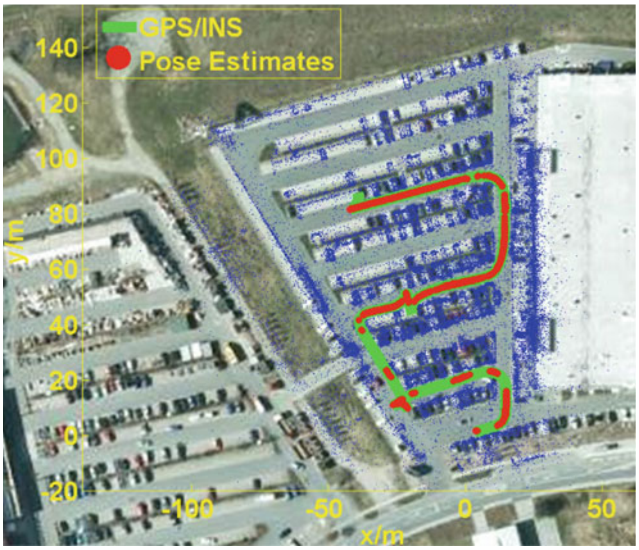


Fig. 6 **a** Localization results for TestArea01. Results from frames with <20 correspondences are discarded. **b** Plots showing the distribution of the translational and rotational errors against GPS/INS ground truths

t_{gt} are the translations from the pose estimation and GPS/INS ground truth. The rotational error is computed as the norm of the Euler angles from $R_{est}R_{gt}^T$ where R_{est} and R_{gt} are the rotation matrices from the pose estimation and GPS/INS ground truth.

(a)



(b)

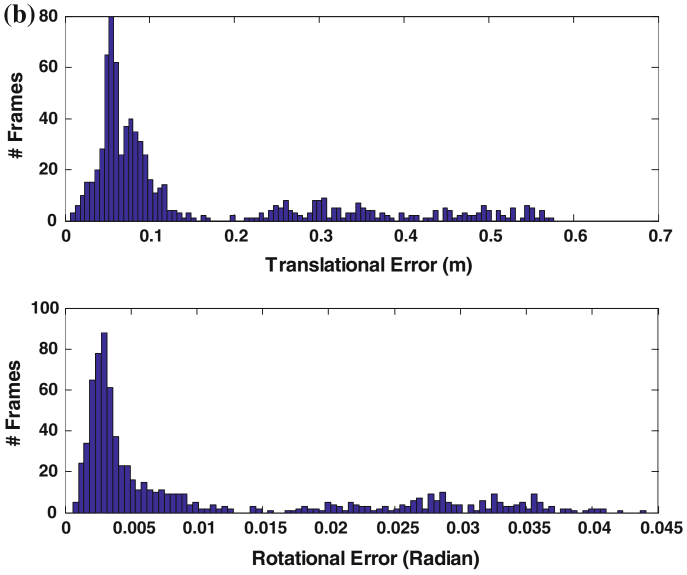


Fig. 7 **a** Localization results for TestArea02. Results from frames with <20 correspondences are discarded. **b** Plots showing the distribution of the translational and rotational errors against GPS/INS ground truths

8 Conclusion

We showed a new formulation to solve the pose estimation problem of a multi-camera system. Our formulation is intuitive and easy to implement. It is based on the Plücker line coordinates which solves the pose estimation problem in two steps—(a) solve for the depth and (b) solve for the rigid transformation with absolute orientation. We showed that the depths can be solved with a minimal number of 3-point correspondences and leads to an 8 degree polynomial minimal solution. We identified a degenerated case for our method in the case of orthographic projection. We also derived an efficient analytical closed-form minimal solution for the absolute orientation. Our method is verified with both simulations and large-scale real-world datasets from a robotic car platform.

Acknowledgments This work is supported in part by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant #269916 (v-charge) and 4DVideo ERC Starting Grant Nr. 210806.

References

1. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (surf). Comput. Vis. Image Underst. **110**, 346–359 (2008)
2. Chen, C.S., Chang, W.Y.: On pose recovery for generalized visual sensors. Pattern Anal. Mach. Intell. **26**, 848–861 (2004)
3. Cox, D.A., Little, J., O'Shea, D.: Ideals, Varieties, and Algorithms—an Introduction to Computational Algebraic Geometry and Commutative Algebra, 2 edn. Springer, Berlin (1997)
4. Ess, A., Neubeck, A., Van Gool, L.: Generalised linear pose estimation. In: British Machine Vision Conference (2007)
5. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**, 381–395 (1981)
6. Haralick, R.M., Lee, D., Ottenburg, K., Nolle, M.: Analysis and solutions of the three point perspective pose estimation problem. In: Computer Vision and Pattern Recognition, pp. 592–598 (1991)
7. Hartley, R.I., Gupta, R.: Linear pushbroom cameras. IEEE Trans. Pattern Anal. Mach. Intell. **19**, 963–975 (1994)
8. Horn, B.K.P.: Closed form solutions of absolute orientation using unit quaternions. JOSA-A **4**, 629–642 (1987)
9. Kneip, L., Furgale, P., Siegwart, R.: Using multi-camera systems in robotics: efficient solutions to the npnp problem. In: International Conference on Robotics and Automation (2013)
10. Lee, G.H., Fraundorfer, F., Pollefeys, M.: Motion estimation for a self-driving car with a generalized camera. In: Computer Vision and Pattern Recognition (2013)
11. Lee, G.H., Fraundorfer, F., Pollefeys, M.: Structureless pose-graph loop-closures with a multi-camera system on a self-driving car. In: International Conference on Intelligent Robots and Systems (2013)
12. Li, H.D., Hartley, R., Kim, J.H.: A linear approach to motion estimation using generalized camera models. In: Computer Vision and Pattern Recognition, pp. 1–8 (2008)
13. Moreno-Noguer, F., Lepetit, V., Fua, P.: Accurate non-iterative o(n) solution to the pnp problem. In: International Conference on Computer Vision, pp. 1–8 (2007)

14. Nistér, D.: A minimal solution to the generalised 3-point pose problem. Comput. Vis. Pattern Recognit. **1**, 560–567 (2004)
15. Nistér, D., Stewénius, H.: Scalable recognition with a vocabulary tree. Comput. Vis. Pattern Recognit. **2**, 2161–2168 (2006)
16. Pless, R.: Using many cameras as one. Comput. Vis. Pattern Recognit. **2**, 587–593 (2003)
17. Quan, L., Lan, Z.D.: Linear n-point camera pose determination. Pattern Anal. Mach. Intell. **21**, 774–780 (1999)
18. Schweighofer, G., Pinz, A.: Globally optimal o(n) solution to the pnp problem for general camera models. In: British Machine Vision Conference, pp. 1–10 (2008)
19. Tariq, S., Dellaert, F.: A multi-camera 6-dof pose tracker. In: International Symposim on Mixed and Augmented Reality, pp. 296–297 (2004)

Recursive Inference for Prediction of Objects in Urban Environments

Cesar Cadena and Jana Košecká

Abstract Future advancements in robotic navigation and mapping rest to a large extent on robust, efficient and more advanced semantic understanding of the surrounding environment. The existing semantic mapping approaches typically consider small number of semantic categories, require complex inference or large number of training examples to achieve desirable performance. In the proposed work we present an efficient approach for predicting locations of generic objects in urban environments by means of semantic segmentation of a video into object and non-object categories. We exploit widely available exemplars of non-object categories (such as road, buildings, vegetation) and use geometric cues which are indicative of the presence of object boundaries to gather the evidence about objects regardless of their category. We formulate the object/non-object semantic segmentation problem in the Conditional Random Field framework, where the structure of the graph is induced by a minimum spanning tree computed over a 3D point cloud, yielding an efficient algorithm for an exact inference. The chosen 3D representation naturally lends itself for on-line recursive belief updates with a simple soft data association mechanism. We carry out extensive experiments on videos of urban environments acquired by a moving vehicle and show quantitatively and qualitatively the benefits of our proposal.

1 Introduction

In recent years the research trends in robotic mapping, navigation and localization focused on developing methods for better understanding of the surrounding environment in order to facilitate more reliable lifelong navigation and mapping. The goal of this work is to endow the models of urban environments with semantic information, which would enable reasoning about presence of different semantic classes (objects)

C. Cadena (✉)
University of Adelaide, Adelaide, Australia
e-mail: cesar.cadena@adelaide.edu.au

J. Košecká
George Mason University, Fairfax, USA
e-mail: kosecka@gmu.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_31

in an on-line setting. We propose to tackle this problem by means of an on-line recursive semantic segmentation of a video stream into object and non-object (ground, vegetation, buildings) categories.

The most common techniques for semantic segmentation of urban environments focus on a small number of commonly encountered semantic classes (e.g. road, sky, buildings, trees, cars). While the state of the art of the semantic parsing approaches in outdoors settings achieve relatively high average accuracy of 85–90 % on some datasets [28], it is largely due to the fact that majority of 2D or 3D regions belong to non-object semantic categories. These categories such as buildings, roads, vegetation and sky often exhibit lower intra class variability, have strong location priors and ample of training data available. With more detailed scrutiny, the existing approaches consider either very small number of object categories (e.g. cars, trees) or exhibit poorer performance when number of object categories grows and the training examples are sparse [18]. The performance has been notably improved by using a specialized sliding window based object detectors [24] or explicit models of higher order dependencies between individual regions captured by higher order potentials in MRF (CRF) framework [7, 14]. These approaches however are not suitable for an on-line setting and typically require a large amount of training examples and/or an expensive training and inference procedure.

In our approach instead of modeling complex spatial and label dependencies or requiring large number of training examples for object categories, we propose an intermediate semantic representation of urban scenes into a single generic *object* category and non-object categories of *ground, building* and *vegetation*. In order to effectively gather evidence about generic objects regardless of their category, we use informative 3D features indicative of occlusion boundaries and depth ordering cues, which were found previously useful in research on perceptual grouping.

Contribution

The main contribution of the proposed work is the development of novel representation, features and associated efficient inference algorithm for the problem of semantic labeling of outdoors urban environments into object and multiple non-object categories. Similarly to the existing approaches we formulate the semantic labeling problem in the Conditional Random Field (CRF) framework, where the dependencies between random variables are represented by a graph, induced by different partitions of an image or a 3D point cloud. The distinguishing features of our approach are: (a) the use of a tree graph structure in the CRF setting which is induced by the 3D scene structure and allows exact and efficient inference amenable for real-time implementation; (b) the use of simple and efficient features and geometric cues, providing evidence about discontinuities and depth ordering; (c) an explicit model of temporal coherency enabling an on-line inference; (d) a flexible model structure easily adoptable to a single or multi-frame settings, without a need for extra training. The semantic output of our method produces detections and associated confidences about the presence of isolated generic objects and semantic labels of non-object categories. The output can be used effectively for priming *specific* object detectors and as a starting point of additional reasoning about various object/scene attributes (e.g. static/dynamic, movable, undergoing seasonal change etc.).

In the next section, we provide an overview of the related work. In Sect. 3 we describe the details of our approach. Section 4 describes the experiments on street scene sequences and compares our approach with the state of the art methods. Finally, in Sect. 5 we present discussion and conclusions of the presented work and discuss possible future directions.

2 Related Work

The presented work on semantic segmentation of images and 3D point clouds into object and non-object categories is motivated by several previous approaches developed both in Computer Vision and Robotics communities. The existing approaches vary in the number and types of semantic classes they consider, the sensing modality, features and inference algorithms.

The approaches developed in the context of robotics applications rely mostly on 3D measurements from laser range finder or dense depth reconstruction and have been also explored in the context of analysis of urban scenes acquired by a moving vehicle. In these methods the graph structure is typically induced by a partitioning of 3D point clouds. Authors in [5] consider 2D semantic mapping of street scenes with laser and image data providing computationally expensive solution with a graph induced by Delaunay triangulation. Both laser and image measurements are used in [20], where efficient solution is provided considering only vehicle object class. Dense stereo reconstruction was used on CamVid urban sequences by [29] further improving the performance with seven specific object classes.

In computer vision community the problem of simultaneous segmentation and categorization of image regions was typically considered in a single view setting. Non-parametric approaches of [6, 23] treated the representations of both object and no-object categories in the same manner and used both the SIFT Flow dataset with 33 semantic labels and Label Me with 253 labels to evaluate the performance of their approaches.

In addition to a single view setting several approaches explicitly modeled temporal relationships between the frames in the inference problem or exploited 3D structure obtained from visual reconstruction [28]. These strategies further improved the labeling performance while still considering a small number of object categories, with objects being trees, cars, persons and recycle bins. Recently, Sengupta et al. [22] have obtained 3D semantic models in urban scenes where the labeling of every single image was transferred to the 3D reconstruction by voting. However, neither the 3D information nor the sequential nature was used for the inference itself. Floros and Leibe [7] segment urban scenes using images and 3D in a CRFs setting using high order potentials between 3D points with their reprojection in several images in the sequence. Their system is only applicable to static environments and is not amenable for real-time implementation. In our case we directly formulate the inference on a graph induced by 3D structure of the scene, which enables us to use the odometry to guide a soft data association between consecutive frames. Our formulation hence naturally fits the standard probabilistic recursive filtering approach suitable for variety of perceptual tasks.

Additional ideas related to the problem of generic object detection can be found in works which exploit different models of saliency, various perceptual grouping cues and unsupervised object discovery. Alexe et al. in [2] propose an approach for generic object detection motivated by a notion of saliency; objects are salient regions surrounded by background and delimited from it by strong contour edges. This approach only exploits the appearance cues, is applicable to a single view setting and more suitable in the context of image based retrieval applications, where images of scenes are well composed, containing little clutter. Our approach is motivated by work of [3], which explicitly reasons about evidence of occlusions boundaries extracted from optical flow and relative depth ordering cues. Also related to our work are several attempts to discover objects in urban scenes. In [25] authors propose a completely unsupervised approach to semantic parsing of outdoors scenes based on the idea of online clustering, demonstrating a capability of discovering categories of car, vegetation, building and ground place in the absence of any labelled data. While this approach is very effective for large objects, generalization to a large number of categories and possibly small objects is be more difficult. Alternative approach for parsing the environments into static parts and moving objects has been recently proposed in [27]. The authors demonstrated successful detection of cars, pedestrians and bicyclists in 3D laser scenes, using the independent motion cue. We view our approach as complementary to the previously proposed techniques. The proposed semantic segmentation of video into object and non-object categories yields a representation that can be used effectively as priors for detection of more broader class of categories (e.g. mailboxes, traffic/road signs, fire hydrants, moving objects).

3 The Approach

3.1 Method

We formulate the semantic parsing in the framework of Conditional Random Fields (CRFs) with a tree graph structure encoding the pairwise relationships. We assume that an image and a 3D point cloud of the scene are available. Our approach starts by over-segmenting the image using the efficient *simple linear iterative clustering* (SLIC) algorithm [1]. Every superpixel in the image is interpreted as a cluster in the 3D point cloud for further computations. The 3D centroid of each cluster is used to compute a minimum spanning tree over Euclidean distances, defining the edges for the graphical model. The data and pairwise terms are determined using simple yet discriminative appearance and geometric cues for the classes that we are interested in. The learning and the final inference process is carried out over the graph in the CRFs framework. In the remainder of this section we detail the components of our approach and explain the intuition behind them.

3.2 Framework: Conditional Random Fields

CRFs directly model the *conditional* distribution over the hidden variables given observations $p(\mathbf{x}|\mathbf{z})$. The nodes in a CRF are denoted $\mathbf{x} = \langle \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n \rangle$, and the observations are denoted $\mathbf{z}$. In our framework the hidden states correspond to the m possible classes: $\mathbf{x}_i = \{ground, objects, \ldots\}$. A CRF factorizes the conditional probability distribution over hidden states into a product of *potentials*, as:

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z(\mathbf{z})} \prod_{i \in \mathcal{N}} \phi(\mathbf{x}_i, \mathbf{z},) \prod_{i,j \in \mathcal{E}} \psi(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z}) \tag{1}$$

where $Z(\mathbf{z})$ is the normalizing partition function, and $\langle \mathcal{N}, \mathcal{E} \rangle$ are the set of nodes and edges on the graph. The unary, or data-term, and pairwise potentials are represented by $\phi(\mathbf{x}_i, \mathbf{z})$ and $\psi(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z})$, respectively, as their domains span over one and two random variables or nodes in the graphical model. Potentials are described by log-linear combinations of *feature functions*, $\mathbf{f}$ and $\mathbf{g}$, i.e., the conditional distribution in Eq. 1 can be rewritten as:

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z(\mathbf{z})} \exp \left(w_1 \sum_{i \in \mathcal{N}} \mathbf{f}(\mathbf{x}_i, \mathbf{z}) + w_2 \sum_{i,j \in \mathcal{E}} \mathbf{g}_c(\mathbf{x}_{i,j}, \mathbf{z}) + w_3 \sum_{i,j \in \mathcal{E}} \mathbf{g}_m(\mathbf{x}_{i,j}, \mathbf{z}) \right) \tag{2}$$

where $\mathbf{w} = [w_1, w_2, w_3]$ is a weight vector, which represents the importance of each term. CRFs learn these weights discriminatively by maximizing the conditional likelihood of labeled training data. We will describe every term of Eq. 2 in detail in Sect. 3.4. With this formulation we can obtain either the marginal distribution over the class of each variable $\mathbf{x}_i$ by solving Eq. 2, or the most likely classification of all the hidden variables $\mathbf{x}$. The latter can be formulated as the *maximum a posteriori* (MAP) problem, seeking the assignment of $\mathbf{x}$ for which $p(\mathbf{x}|\mathbf{z})$ is maximal.

3.3 Minimum Spanning Tree over 3D Distances

Instead of computing the graph at the pixel level, we over segment the image into superpixels. The CRF graph structure is typically determined by the neighbourhood relations between superpixels, often connecting unrelated semantic classes (e.g. a person's head with the sky in the background). We define the graph structure for the CRF as a minimum spanning tree (MST) over the Euclidean distances between 3D superpixel's centroids in a scene. By definition, the minimum spanning tree connects points that are close in the measurement space, highlighting intrinsic localities in the scene, see Fig. 1b. Given that our graph structure is a tree we use the *belief propagation* (BP) algorithm [12] to infer the probability class of each node.

3.4 Feature Functions Description

Now we define the feature functions $\mathbf{f}(\mathbf{x}, \mathbf{z})$ and $\mathbf{g}(\mathbf{x}, \mathbf{z})$ in Eq. 2. Starting by the data-term for each superpixel s that is computed as:

$$\mathbf{f}(\mathbf{x}_s, \mathbf{z}) = -\log P_s(\mathbf{x}_s|\mathbf{z}) \tag{3}$$

where the local prior $P_s(\mathbf{x}_s|\mathbf{z})$ is the output of a k-nearest neighbours (k-NN) classifier from a set of observations $\mathbf{z}$. We compute $P_s(\mathbf{x}_s|\mathbf{z})$ as proposed by [23] in Eq. 4.

$$P_s(\mathbf{x}_s = l_j|\mathbf{z}) = \frac{1}{\sum_{j=1}^{m}\left(\frac{f(l_j)}{\overline{f}(l_j)}\frac{\overline{F}(l_j)}{F(l_j)}\right)}\frac{f(l_j)}{\overline{f}(l_j)}\frac{\overline{F}(l_j)}{F(l_j)} \tag{4}$$

where $f(l_j)$ (resp. $\overline{f}(l_j)$) is the number of neighbours to s with label l_j (resp. not l_j) in the kd-tree. And $F(l_j)$ (resp. $\overline{F}(l_j)$) is the counting of all the observations in the training data with label l_j (resp. not l_j). The observations $\mathbf{z}$ computed for every superpixel s capturing the appearance cues obtained from the image (*Image Features*) and the depth cues (*3D Features*) are summarized in Table 1 and described next.

Image Features: From the appearance of the superpixel we only use color cues and its vertical location, as follows: the mean and standard deviation of each channel in

Table 1 Local observations

Source	Default	Observation	Dim.	Comments
Image		LAB color	6	Mean and standard deviation
		RGB color	6	Mean and standard deviation
		v_s	1	Vertical pixel location
3D		$\tilde{\mathbf{p}}_s$	3	$[x_s, abs(y_s), z_s]$
	0	$(\mu_{\Delta d_s}, \sigma_{\Delta d_s})$	2	if $d_s < \frac{1}{\|N\|}\sum_{j\in N}(d_j)$, $\Delta d_s = \|d_s - d_{j\in N}\|$
	0	$1 - mean(\|\mathbf{n}_s\mathbf{n}_N\|)$	1	Neighbouring planarity
	0	$\frac{3\sigma_3}{\sigma_1+\sigma_2+\sigma_3}$	1	Superpixel planarity
	0	$\|\mathbf{n}_s \cdot \hat{\mathbf{k}}\|$	1	Superpixel vertical orientation

the LAB and RGB color spaces for the superpixel, and the vertical pixel coordinate for the superpixel's centroid.
3D Features: For the 3D point cloud we use cues from the position and planarity, for the superpixel itself and for the superpixel with respect to its neighbourhood. The cues are: the modified 3D coordinate $\tilde{\mathbf{p}}$ for the superpixel's centroid with the absolute value in its lateral coordinate (y_s), then we have depth (x_s), height (z_s) and positive lateral distance; the mean and standard deviation of the absolute difference between the depth d_s and the neighbourhood's depths: $\|d_s - d_{j\in N}\|$, but these are only computed if $d_s < \frac{1}{\|N\|}\sum_{j\in N}(d_j)$, with this condition we encode the *in-front-of* property; the superpixel planarity encoded by the curvature of a superpixels' point cloud [11] using the SVD and sort the singular values such that $\sigma_1 > \sigma_2 > \sigma_3$; the neighbourhood planarity computed as one minus the mean of the dot product between the normal to the plane against the neighbourhood normals [29], where the normal corresponds to the singular vector associated to σ_3; and, the superpixel vertical orientation as an absolute value of its normal's vertical component.

The superpixel neighbourhood N refers to all the superpixels in contact with superpixel s in the image. In Table 1 we also show the default values and the dimensionality of these observations. As a result we compute for each superpixel a very simple 21 dimensional feature vector with 13 elements from *Image features* and 8 from *3D features*.

Pairwise Potentials

We define two pairwise potentials, one capturing the color proximity $\mathbf{g}_c(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z})$ and the other the metric proximity $\mathbf{g}_m(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z})$ of two superpixels. The potentials are:

$$\mathbf{g}_c(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z}) = \begin{cases} 1 - \exp\left(-\|\mathbf{c}_i - \mathbf{c}_j\|_2\right) \rightarrow l_i = l_j \\ \exp\left(-\|\mathbf{c}_i - \mathbf{c}_j\|_2\right) \quad \rightarrow l_i \neq l_j \end{cases} \tag{5}$$

$$\mathbf{g}_m(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z}) = \begin{cases} 1 - \exp\left(-\|\mathbf{p}_i - \mathbf{p}_j\|_2\right) \rightarrow l_i = l_j \\ \exp\left(-\|\mathbf{p}_i - \mathbf{p}_j\|_2\right) \quad \rightarrow l_i \neq l_j \end{cases} \tag{6}$$

where $\|\mathbf{c}_i - \mathbf{c}_j\|_2$ and $\|\mathbf{p}_i - \mathbf{p}_j\|_2$ are the L2-Norm of the difference between the mean colors in the LAB-color space, and centroid's 3D positions, respectively, of two superpixels and l is the class label.

3.5 Recursive Inference

So far we have described all the components to carry out the inference in a single frame composed by an image and a 3D point cloud. But in a normal operation of a robot, this information comes streamed. We would like to take an advantage of the sequential nature of the data to carry out the inference without losing the single frame if needed, while keeping the robustness against, for instance, lost frames or odometry failures.

Let C_k be the coordinate frame of reference in time k and ${}^k\mathbf{p}^{k-1}$ the 3D coordinates of the nodes $\mathbf{x}$ in frame $k-1$ in C_k. Let ${}_{k-1}^{\;\;k}\mathscr{T}$ be the transformation from C_{k-1} to C_k given by the robot odometry. In the first frame $k=0$, we infer the state of each node in the graph as described before for a single frame. For $k=1$ we follow the procedure as in single frame case, computing the superpixels, the features and the data-term. Then we transform the 3D coordinates of nodes at $k=0$ to C_1 by computing ${}^1\mathbf{p}^0 = {}_0^1\mathscr{T} \times {}^0\mathbf{p}^0$. With the set $[{}^1\mathbf{p}^0, {}^1\mathbf{p}^1]$ a new MST is computed. These steps are described in Fig. 1. Now, we can proceed to compute the pairwise potentials

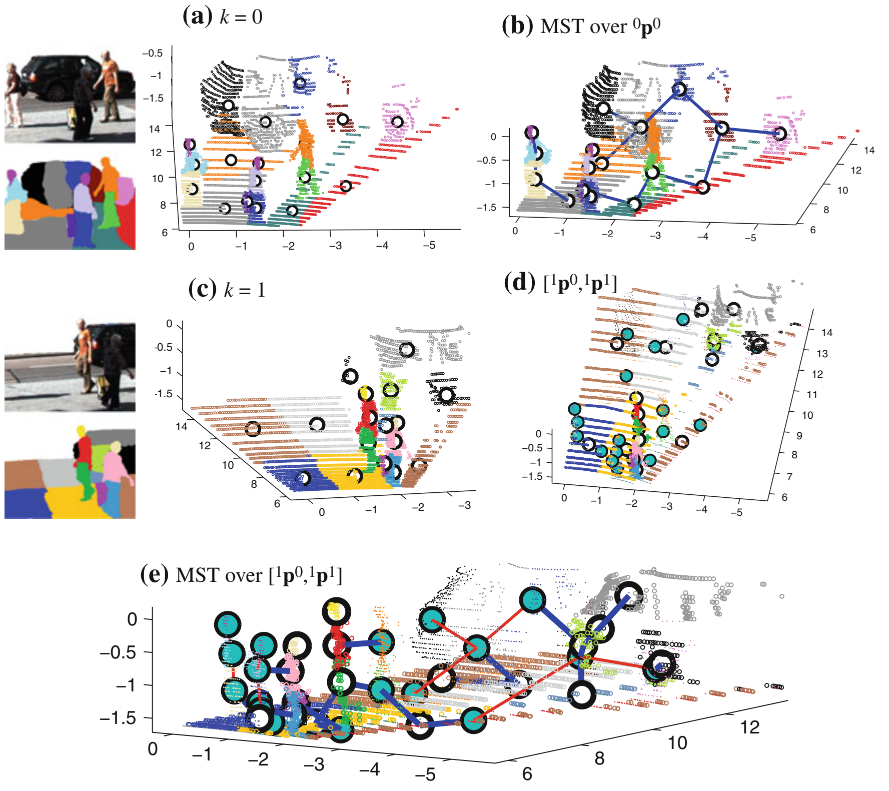


Fig. 1 Toy example for the sequence semantic segmentation process. **a** Suppose that the robot gathers the image and point cloud at time $k=0$, the image is over-segmented and the nodes are placed in the clusters's centroids. **b** The Euclidean MST provide the graph structure for our CRF, where the inference takes place. **c** The same as (**a**) in the next time step. **d** We transform the centroids from the previous time step to the current one, the state of the corresponding nodes is now known (blue filled circles). **e** A new MST is computed between all the nodes, known and unknown. The new graphical model is a forest as the edges between known nodes (*red lines*) are not used in the inference process

over the edges in this new graph and carry out the inference, estimating the state of ${}^1\mathbf{x}^1$ conditioned over the states of ${}^0\mathbf{x}^0$. This process is repeated for the next time k, conditioning only over the state in $k-1$. The nodes in $k-2$ are no more taken into account as we assume a filtering approach for the inference, Eq. 7.

$$p(\mathbf{x}^k|\mathbf{x}^{k-1}, \mathbf{x}^{k-2}, \ldots, \mathbf{x}^0, \mathbf{z}^k, \mathbf{z}^{k-1}, \ldots, \mathbf{z}^0) = p(\mathbf{x}^k|\mathbf{x}^{k-1}, \mathbf{z}^k) \tag{7}$$

Note that we have omitted the left superscript in Eq. 7 to denote the coordinate frame of reference as this choice does not affect the inference as long as all the nodes are expressed in a common reference frame.

Given the observed nodes the graph structure becomes a forest, see Fig. 1e. We can find two extreme cases when we carry out the inference to compute $p(\mathbf{x}^k|\mathbf{x}^{k-1}, \mathbf{z}^k)$. The first one is when the robot and the scene are static, in this case the spatial locations of $\mathbf{x}^k$ are expected to be very close to $\mathbf{x}^{k-1}$. Since the $\mathbf{x}^{k-1}$ is now treated as evidence, we would obtain at most a forest with n^k (number of nodes in time k) trees of size 2 connecting the corresponding nodes between $k-1$ and k. The inference for each node $\mathbf{x}_i^k$ is just a weighted average between the state from the local evidence $\mathbf{z}^k$ through the data term and the already inferred state $\mathbf{x}_i^{k-1}$ through the pairwise terms. The second case is when the motion between consecutive frames exceeds the range of the sensor, in which case would be only one edge connecting $\mathbf{x}^k$ and $\mathbf{x}^{k-1}$. In this case $p(\mathbf{x}^k|\mathbf{x}^{k-1}, \mathbf{z}^k)$ approaches a single frame case $p(\mathbf{x}^k|\mathbf{z}^k)$ as the distance between frames increases.

Our MST connecting point clouds from two different timestamps gives us a robust way to avoid common errors from data association algorithms in dynamic environments. Even more, this MST graph structure is only telling us that there is a relation between the connected nodes and that it is likely, up to their 3D distance, they belong to the same category; but not, whether they are the same physical entity or a landmark.

4 Experiments

For our experiments we use the KITTI dataset [9], which contains images (1240×380) and 3D laser data taken for a vehicle in different scenarios. We demonstrate the performance of our approach on data from urban residential and city scenes. There are 70 manually labelled non-sequential images as ground truth made available by [22], 45 for training and 25 for testing. The original classes released by [22] are: road, building, vehicle, pedestrian, pavement, vegetation, sky, signal, post/pole and fence. We have mapped those to the four classes: ground (road and pavement), building, vegetation, and objects (vehicle, pedestrian, signal, pole and fence). The sky class is omitted as we carry out the inference only in the portion of the image where we have 3D data. For our system the effective manually labeled region in the image is reduced as the laser measurements span up to 3.2 m above the ground and the maximum range that we consider is 30m. As such we use only that ground truth image region for training and for the quantitative evaluations in testing, see Fig. 2.

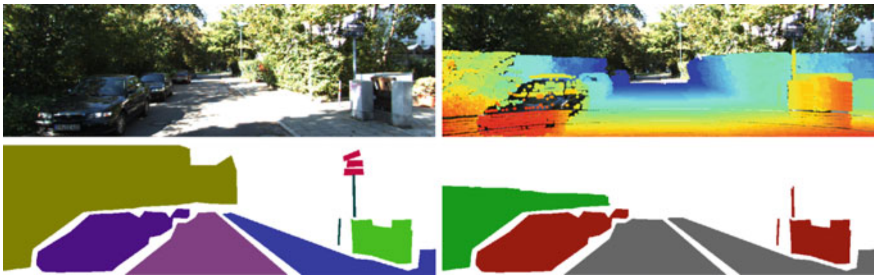


Fig. 2 Original ground truth as released by [22] (*left*) and the effective ground truth with 3D information used in this paper (*right*)

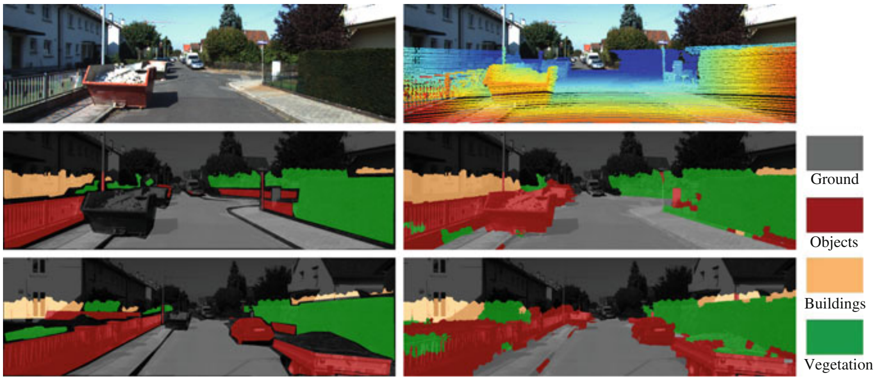


Fig. 3 First row: Original image (*left*) with the reprojected 3D point cloud (*right*). Other rows: Ground truth labeling (*left*) and MAP result from our approach (*right*)

We obtain the superpixel segmentation using SLIC implementation from the VLFeat library of [26], followed by the computation of the features described in Table 1. With the computed features in the training set we build a kd-tree using the implementation of [4] with the default parameters. We obtain the per-class data terms with k-NN classification using Eq. 4 with the $k = 10$ nearest neighbours.

Now, using the MST graph, the output of the local classifier in Eq. 3 and the pair-wise potentials, Eq. 5, we learn the parameters in the CRF setting. For the learning, inference and decoding with CRFs we use the Matlab code for undirected graphical models (UGM).[1]

At the testing time, to obtain the most likely label assignment for the superpixels we solve the MAP problem for the model. This problem does not require any threshold selection and all the parameters are learned from the data. The inference results give us the labeling assignments over superpixels, we transfer those to every pixel in the superpixel to compute the pixel-wise recall accuracy of semantic labeling. In Fig. 3 we show several examples of the output of our approach in the single frame setting.

[1]Code made available by Mark Schmidt at http://www.di.ens.fr/~mschmidt/Software/UGM.html.

Table 2 Semantic segmentation for single view in pixel-wise percentage recall accuracy

	Ground	Objects	Building	Vegetation	Average	Global
Data-term: k-NN (Eq. 3)	96.8	75.9	80.7	77.6	82.8	83.5
CRF_MST_k-NN						
only Image Features	96.8	49.2	64.6	95.5	76.5	76.8
only 3D Features	95.9	84.2	80.5	46.7	76.8	78.8
Im. and 3D Features	97.3	82.9	82.8	86.9	87.5	88.4

Table 3 Results and timing for single view vs recursive segmentation

		Ground	Objects	Building	Vegetation	Time MST	Time BP
Single view	Recall	0.973	0.829	0.828	0.869	21 ms	164 ms
	Precision	0.981	0.881	0.916	0.759		
	F_1	0.977	0.854	0.870	0.811		
Recursive Inference	Recall	0.975	0.836	0.832	0.855	57 ms	69 ms
	Precision	0.980	0.871	0.931	0.767		
	F_1	0.977	0.853	0.879	0.809		

In Table 2 we show the pixel-wise recall accuracy along with the average and global accuracy for our approach: CRF_MST_k-NN which uses image and 3D features. To study the importance of image features vs 3D features, we remove one set at a time keeping the rest of the system intact. The rows only *Image Features* and *3D Features* show the corresponding performance when using one kind of features. The main contribution of the images features is placed in the *vegetation* class, while the 3D features improve the *objects* and *building* classes. Our full system, with both sets of features obtains the best trade off between all performance measures.

The row *data-term* in Table 2 shows the result of the k-NN classification using the image and 3D information. It is clear that the MST and the CRF framework improve the general performance.

Given that the manually labeled testing frames belong to the same KITTI sequence (000015), we run our approach over the full sequence and compute the impact on the accuracy by our recursive inference processing. To that end we compute the odometry using the open-source libviso2 [10], using the default parameters provided with the library. Table 3 shows the results and the timing for the stages that change with respect to single view segmentation. We can see that there is no significant difference in the performance and even when the cost of computing the MST with greater number of nodes has increased, the belief propagation is now even more efficient. Our approach is able to reach simultaneously high recall and precision for all the classes, with $F_{0.5}$ of 0.96 for *ground*, 0.87 for *objects*, 0.90 for *buildings*, and 0.78 for *vegetation*, compared for example with a $F_{0.5}$ of 0.62 and 0.52 for grass and bush, 0.91 and 0.67 for tarmac and dirt path, and 0.78 and 0.71 for textured and

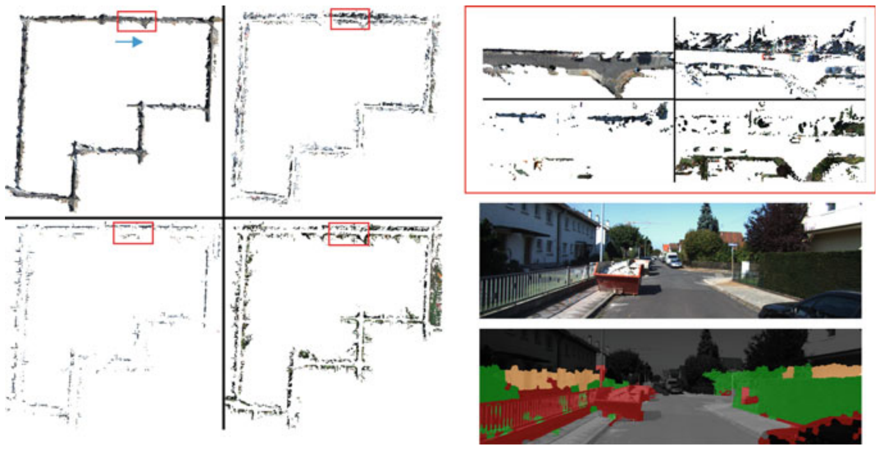


Fig. 4 Results in KITTI sequence 000015 in a residential environment after optimize the trajectory with the detected loop closures. In every image is shown the segmented points belonging to *ground, objects, building* and *vegetation* (*left* to *right* and *up* to *down*). On the right we show a zoom up along with a image of the corresponding region and the segmented results

smooth walls, reported by [20] with a labeling system using spatio-temporal context and spending 4 s per frame.

Although our approach is not directly comparable to [22] given the differences in the classes, sensor modalities and the effective region used as ground truth we can see similar performance in the average and global accuracies, 81.7 and 88.4 % reported by [22] compared to 87.5 and 88.4 % for us.

The results over the full sequence 000015 of 1900 frames is shown in Fig. 4. The 3D laser returns are reprojected over the odometry. For clarity we split the point cloud in four views corresponding to each class. This trajectory corresponds to a loop in a residential part of the city. To detect the loop closures we use the DLoopDetector library [8], with the default vocabulary and parameters for BRIEF and geometrical checking with the epipolar constraint. To find the transformation between candidates to loop closures we use ICP over the two point clouds excluding those assigned as *objects*. This choice is because this class contains the entities that could be dynamic (cars, people, bikes), hence seems reasonable enough aligning two point clouds by their more stable components during one day: *ground, buildings, vegetation*. The result after optimize the pose graph with g2o [13] is shown in Fig. 4 left.

To test how our system performs facing new environments, we carry out the same semantic recursive inference over a sequence taken in the downtown with high density of dynamic objects, see Fig. 5. In Fig. 5a two images are shown, at the beginning and at the middle of the sequence. The probability of belonging to the *objects* class is shown in the second row. We can see how the pedestrians, cars, poles, chairs are all included in this class even when we do not have some of these specific instances in the training data. In the third row of Fig. 5a we show the final reprojected point cloud with texture of *objects* class from locations close to images in the first row, the

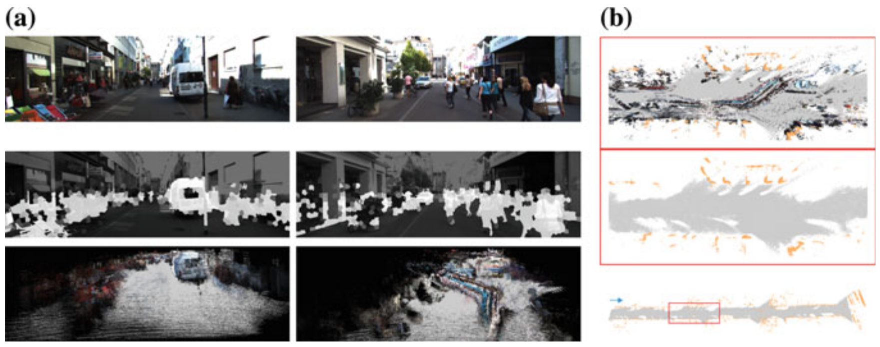


Fig. 5 Segmentation by our recursive inference process in a high dynamic street in downtown. **a** A couple of frames with the highlighted *objects* class. **b** Reprojected point cloud for the *ground* and *building* class, on the top we show also the *objects* class

ground is also shown in grey only for reference. In the second column we can see the trajectories of the pedestrians. In Fig. 5b below, we show the reprojected point cloud over the odometry trajectory for classes *ground* and *building*. A zoom up in the second location of Fig. 5a is shown on the top and middle. In the first row the *objects* is included to highlight the importance of recognizing this class to allow a better and reliable 3D mapping.

LEUVEN Dataset:

We have also tested our system on the LEUVEN dataset [16], where a set of 70 labeled images and disparity maps were released by [15]. The images were gathered in a residential neighbourhood at 316×216 pixels in resolution. We map the original eight classes to three as follows: ground (road and sidewalk), building, and objects (car, person and bike). The sky again was omitted and the vegetation class was not present in the labeling data. We show some of our results in Fig. 6.

We obtain a pixel-wise recall of 97.6 % (*ground*), 79.3 % (*objects*) and 98.4 % (*buildings*). Our average and global accuracies are 91.8 % and 95.9 %, [7] report 82.4 % and 95.4 %, and [15] report 84.9 % and 95.8 %, respectively. Note the difference for each system, [7] and [15] included the sky but omitted the person class. Floros and Leibe [7] use at least 5 frames for the 3D reconstruction and manually tune the parameters for the CRF. While Ladický et al. [15] solve a more complicated problem inferring the disparity map jointly with the segmentation.

4.1 Pedestrian Detection

We can combine our segmentation with different object detectors. As an example of this idea we present the results of combination of the efficient approximation of HOG based pedestrian detection with sliding windows (IKSVM) of [17], with our

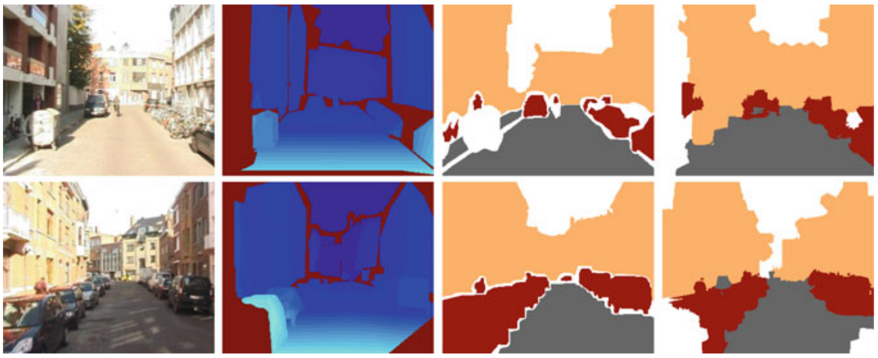
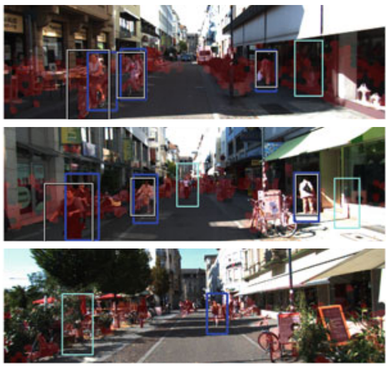


Fig. 6 LEUVEN dataset. From the left, original image, disparity map, ground truth labeling, MAP result from our proposal

(a) Bounding box detections.



(b) Precision-Recall.

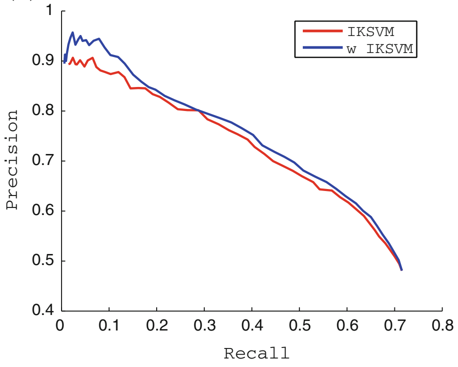


Fig. 7 Pedestrian detection on a high dynamic sequence. **a** Bounding box for ground truth (*white*), IKSVM [17] (*blue* and *cyan*) and weighted IKSVM with the evidence for the *objects* class inside the box (*blue*). The *objects* class is highlighted in *red*. **b** The precision-recall curve for this sequence with the IKSVM alone and weighted IKSVM by our objectness evidence

evidence of objects for the downtown sequence in Fig. 5. We compute the proportion of pixels inside the bounding box belonging to the *objects* class to measure the "objectness" of the box. By weighting the score from the pedestrian detector with this objectness measure the precision is improved, see Fig. 7b. We can see in Fig. 7a how detections in walls or vegetation are rejected (cyan) by the weighted IKSVM. Note in the second image the bounding box in the center is too large for the actual size of the corresponding pedestrian resulting in rejection because a low objectness. We are currently working to integrate the outcome of our segmentation to guide the pedestrian (and other objects) detection, to improve the efficiency and performance of the sliding window approach.

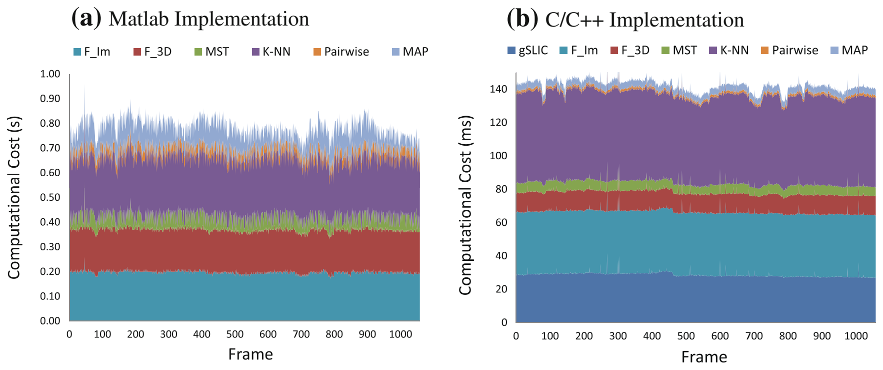


Fig. 8 Computational timing performance for a sequence in downtown with our recursive inference approach, see Fig. 5. We show the corresponding cost for image features (F_{Im}), 3D features (F_{3D}), MST, unary term (K-NN), pairwise term (Pairwise), and MAP assignment. **a** Matlab Implementation. **b** C/C++ Implementation

4.2 *Timing*

We compute the timing on the sequence of Fig. 5. The computational cost in Matlab is detailed in Fig. 8a, excluding the superpixel over-segmentation and the odometry computation. The on-line system runs at 1 fps in a single-thread of a 3.4 GHz IntelCore i7-2600 CPU M350 and 7.8GB of RAM. For the whole system, the average and the maximum times are 778ms and 960ms, respectively. In this case, the cost to obtain the SLIC superpixels is a bottleneck with almost 1.5 seconds. With a C/C++ implementation and using the GPU SLIC version (gSLIC) of Ren and Reid [21] the full system takes only 142ms per frame, Fig. 8b. The libviso2 library spends 50ms per frame to compute the odometry. Solving the MAP problem has the same computational cost than obtaining the marginals with the BP algorithm.

5 Discussion

We have presented a computationally efficient approach for semantic labeling of urban street view sequences into structural, natural and object categories. The proposed approach uses effectively 3D cues to generate evidence about the presence of objects.

We have shown that our graph structure induced by the MST over 3D does not sacrifice the labeling accuracy, and keeps the intra-class components coherently connected. Furthermore, this choice enables an exact and efficient inference. The computational cost is constant with respect to the length of the trajectory. The computational complexity for the inference is $\mathcal{O}(nm^2)$, where n is the number of nodes in the graph, and m the number of classes.

Our recursive inference process consistently propagates the semantic segmentation over time in a efficient way, keeping a robust performance with no necessity of

expensive data association techniques. The alignment transformation between two consecutive frames is not needed to be perfect as we can assume with the current state of the art in (visual, laser, IMU) odometry systems a reasonable accuracy. Still, three possible failures could affect the recursive processing: frames are lost during the data acquisition, the odometry system fails but it is detectable, and finally the odometry fails resulting in a different motion model than the actual. In the two first scenarios, the next acquired frame or during the detected failure our approach can handle it as a single frame inference problem. In the last case, the data term is not affected by this failure but the propagation of evidence through the graph and hence the state estimation would be affected. An analysis of this influence is part of our future research. So far we have shown a basic research implementation for a very efficient recursive inference process and as expected a C/C++ implementation offers us real time capabilities (5fps).
We demonstrated that our method can work in real scenarios, with dynamic objects, in a different kind of streets from the training and with objects never seen before.

We see our proposal as the first stage of a scalable semantic understanding system for mobile robots. The subsequent stages can use the obtained representation for finding objects or use it for isolating the stationary part from the dynamic part of the environment. This can further improve other tasks such long-term place recognition or dynamic objects detection/estimation. The presented model can be extended in a hierarchical manner to incorporate additional information about specific objects of interest if those become available. Our method can be used in conjunction with [19] to estimate the motion model of generic objects even if they are static in the scene.

Acknowledgments This research has been funded by the US Army Research Office Grant W911NF-1110476 and the Australian Research Council grant DP130104413.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Patt. Anal. Mach. Intell. **34**(11), 2274–2282 (2012)
2. Alexe, B., Deselaers, T., Ferrari, V.: What is an object?. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 73–80, June 2010
3. Ayvaci, A., Soatto, S.: Detachable object detection: segmentation and depth ordering from short-baseline video. IEEE Trans. Patt. Anal. Mach. Intell. **34**(10):1942–1951 (2012)
4. Buchanan, A.M., Fitzgibbon, A.W.: Interactive feature tracking using K-D trees and dynamic programming. IEEE Conf. Comput. Vis. Patt. Recognit. **1**, 626–633 (2006)
5. Douillard, B., Fox, D., Ramos, F., Durrant-Whyte, H.: Classification and semantic mapping of urban environments. Int. J. Rob. Res. **30**, 5–32 (2011)
6. Eigen, D., Fergus, R.: Nonparametric image parsing using adaptive neighbor sets. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2799–2806, June 2012
7. Floros, G., Leibe, B.: Joint 2d–3d temporally consistent semantic segmentation of street scenes. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2823–2830, 2012

8. Galvez-Lopez, D., Tardos, J.D.: Bags of binary words for fast place recognition in image sequences. IEEE Trans. Robot. **28**(5), 1188–1197 (2012)
9. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Conference on Computer Vision and PatternRecognition (CVPR), 2012
10. Geiger, A., Ziegler, J., Stiller, C.: Stereoscan: dense 3d reconstruction in real-time. In: Intelligent Vehicles Symposium (IV), 2011
11. Klasing, K.: Aspects of 3D perception, abstraction, and interpretation in autonomous mobile robotics. Ph.D. thesis, Technical Univeristy of Munich, Germany (2010)
12. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press, USA (2009)
13. Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g2o: A general framework for graph optimization. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, May 2011
14. Ladický, L., Sturgess, P., Alahari, K., Russell, C., Torr, P.H.S.: What, where and how many? combining object detectors and CRFs. In: Computer Vision—ECCV 2010. Springer, Berlin (2010)
15. Ladický, L., Sturgess, P., Russell, C., Sengupta, S., Bastanlar, Y., Clocksin, W., Torr, P.H.S.: Joint optimization for object class segmentation and dense stereo reconstruction. Int. J. Comput. Vis. **100**(2), 122–133 (2012)
16. Leibe, B., Cornelis, N., Cornelis, K., Van Gool, L.: Dynamic 3d scene analysis from a moving vehicle. In: IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07. pp. 1–8 (2007)
17. Maji, S., Berg, A.C., Malik, J.: Classification using intersection kernel support vector machines is efficient. In: IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008. pp. 1–8 (2008)
18. Micusik, B., Košecká, J.: Semantic segmentation of street scenes by superpixel co-occurrence and 3d geometry. In: 2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops), 625–632 Oct 2009
19. Moosmann, F., Stiller, C.: Joint self-localization and tracking of generic objects in 3d range data. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1138–1144, Karlsruhe, Germany (2013)
20. Posner, I., Cummins, M., Newman, P.: A generative framework for fast urban labeling using spatial and temporal context. Auton. Robot. **26**, 153–170 (2009)
21. Ren, C.Y., Reid, I.: gSLIC: a real-time implementation of SLIC superpixel segmentation. Technical report, University of Oxford, Department of Engineering (2011)
22. Sengupta, S., Greveson, E., Shahrokni, A., Torr, P.H.S.: Urban 3D Semantic Modelling Using Stereo Vision. In: ICRA (2013)
23. Tighe, J., Lazebnik, S.: Superparsing: scalable nonparametric image parsing with superpixels. In: Computer Vision—ECCV 2010. Springer, Berlin (2010)
24. Tighe, J., Lazebnik, S.: Finding things: image parsing with regions and per-exemplar detectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2013)
25. Triebel, R.A., Paul, R., Rus, D., Newman, P.: Parsing outdoor scenes from streamed 3d laser data using online clustering and incremental belief updates. In: Twenty-Sixth AAAI Conference on Artificial Intelligence (2012)
26. Vedaldi, A., Fulkerson, B.: VLFeat: an open and portable library of computer vision algorithms (2008). http://www.vlfeat.org/
27. Wang, D., Posner, I., Newman, P.: What could move? finding cars, pedestrians and bicyclists in 3d laser data. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Minnesota, USA (2012)
28. Xiao, J., Quan, L.: Multiple view semantic segmentation for street view images. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 686–693 Oct 2009
29. Zhang, C., Wang, L., Yang, R.: Semantic segmentation of urban scenes using dense depth maps. In: Computer Vision—ECCV 2010. Springer, Berlin (2010)

A New Approach to Model-Free Tracking with 2D Lidar

Dominic Zeng Wang, Ingmar Posner and Paul Newman

Abstract This paper presents a unified and model-free framework for the detection and tracking of dynamic objects with 2D laser range finders in an autonomous driving scenario. A novel state formulation is proposed that captures joint estimates of the sensor pose, a local static background and dynamic states of moving objects. In addition, we contribute a new hierarchical data association algorithm to associate raw laser measurements to observable states, and within which, a new variant of the Joint Compatibility Branch and Bound (JCBB) algorithm is introduced for problems with large numbers of measurements. The system is calibrated systematically on 7.5K labeled object examples and evaluated on 6K test cases, and is shown to greatly outperform an existing industry standard targeted at the same problem domain.

1 Introduction

In this paper, we describe a unified framework for the detection and tracking of moving objects from a 2D laser range finder for autonomous driving applications. The aim of this work is to formulate a light-weight standalone system that takes a minimal number of sensory inputs to produce reliable motion estimates for *only* objects that are dynamic at the time of observation. Moreover and central to this work, we place no requirement on the shape or parametric form of the tracked objects.

Recent years have seen a succession of triumphs of autonomous navigation systems on the road. Both the successes of the DARPA Grand [11] and Urban [14] Challenges and recent demonstrations across the world heighten our community's

D.Z. Wang (✉) · I. Posner · P. Newman
Mobile Robotics Group, University of Oxford, Oxford, UK
e-mail: dominic@robots.ox.ac.uk

I. Posner
e-mail: ingmar@robots.ox.ac.uk

P. Newman
e-mail: pnewman@robots.ox.ac.uk

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_32

belief that self-driving vehicles are truly within our reach. Safe navigation of such systems is a challenging but yet arguably the most critical task that requires sensible interactions with complex dynamic environments. To be able to perceive the dynamic aspects of the environment and predict future movements of the manoeuvring objects is thus essential to every successful autonomous vehicle on the road.

It has been observed by many authors [8, 16] that the problems of sensor pose estimation, map-building and detection and tracking of dynamic objects are closely related to each other. Removal of dynamic objects from the map-building process enhances the quality of the map, while knowledge about the static structure of the environment helps significantly in the successful detection of dynamic objects. Both are in turn tightly coupled with sensor pose estimation because all observations are made relative to the sensor. To this end, our proposed system also estimates jointly the sensor pose, a local static background that maps the static structure around the sensor and the dynamic states of the tracked moving objects, but with an emphasis on the task of dynamic object detection.

Figure 1 shows a typical output of the proposed system. As can be noted, detection and tracking of dynamic objects is particularly challenging in the urban driving scenario due to the presence of a significant amount of background clutter, further complicated by the fact that the sensor itself is also attached to a moving vehicle. Despite of the difficulties, our system is able to successfully identify and track moving objects *without* any restriction to their type.

In what follows, Sect. 2 reviews existing approaches to detection and tracking of dynamic objects. Section 3 states our main contributions in the paper. Section 4 presents the details of our proposed system. We then quantitatively evaluate the

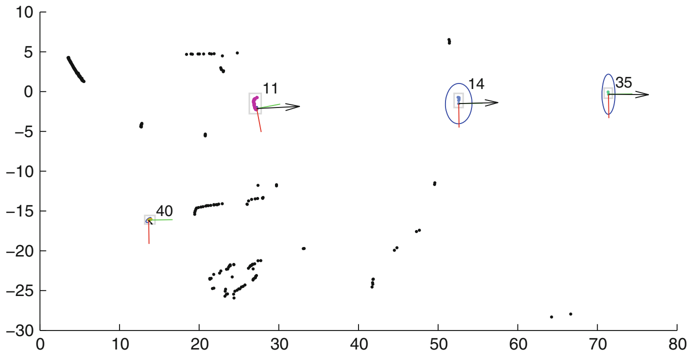


Fig. 1 A typical system output. Detections are highlighted with bounding boxes. Frame axes attached to each detection show objects' local reference frames, and *coloured* points indicate estimated locations of object boundary points (cf. Sect. 4.1). Uncertainty *ellipses* are shown for each object's estimated position, and *numbers* next to each detection denote unique tracking ID's. Three manoeuvring cars and one walking pedestrian are detected, of which two cars' motions are being predicted in the absence of direct observation. Note the scene clutter and it is far from easy to say what is a car and what is not. Note also that the vehicle itself is moving from frame to frame

performance of the proposed system with real-world data, and show that it outperforms an industry standard solution that was designed for the same problem domain of object tracking from a moving sensor in Sect. 5. Finally we conclude the paper in Sect. 6.

2 Related Works

The problem of detection and tracking of multiple manoeuvring targets has been under active research for decades. Early efforts have been focused on the tracking of disjoint point-like targets, and it was soon realised that the challenge lies in obtaining the correct association between noisy measurements and object tracks [2].

In the autonomous driving application domain, however, further complications arise when moving targets are usually buried deep within significant background clutter and they exhibit more complex motions than single point targets. The fact that all observations are made relative to a *moving* sensor adds additional difficulty because static obstacles may also appear dynamic due to occlusion and noise. Most existing practical dynamic tracking systems, for example, systems deployed in the Urban Grand Challenge [6, 7], function by first segmenting measurements from multiple laser range finders, and then extracting geometric features from the segments, which are used to compile a list of object hypotheses. Then, the dynamic objects are extracted as objects having a significant manoeuvring speed.

Most related to our work is a body of work that jointly estimates a static map of the environment along side detecting and tracking of moving objects. Examples include Toyota's tracking system [8] and Wang's system [16] that combines SLAM with dynamic object tracking. Both approaches take an occupancy grid representation of the environment, and use knowledge of occupancy probabilities from the map to propose likely moving object detections.Yang and Wang [18] propose a system that jointly estimates the vehicle pose and moving object detections using a variant of RANSAC, and track merging and splits are handled via a decision tree based on spatiotemporal consistency tests. Works by Tipaldi et al. [12, 15] focus on the detection part of the problem, and formulate it under a joint Conditional Random Field (CRF) framework for solving both the data association and moving object detection problems.

We also mention another body of work that is targeted at detection and tracking of particular object classes of interest. For example, Arras et al. [1] and Topp and Christensen [13] focus on the detection and tracking of people from a laser range finder by first detecting legs from a segmented laser scan and group them into person tracks. Our proposed system is different in that moving objects of any class and shape may be modelled.

3 Contributions

Our main contribution in this paper is the formulation of a unified framework that jointly estimates the pose of the sensor, a continuously updated local static background, and the motion states of dynamic objects, with the focus on reliable detection of moving objects. All three aspects are tightly coupled through a novel joint state representation that allows for objects of arbitrary shapes and sizes to be modelled and tracked.

In addition, we propose a hierarchical data association algorithm to assign raw laser measurements to potential state updates, and present a greedy variant of the Joint Compatibility Branch and Bound (JCBB) algorithm [9] that is suitable for associating a large number of measurements.

4 Model-Free Tracking of Moving Objects

The system we propose is run within a recursive Bayesian framework (implemented as an Extended Kalman Filter). In this section, we describe in detail the system formulation in terms of state representation, prediction and measurement models as well as how data association is handled within the same framework.

4.1 An Unusual State Representation

The motions of dynamic objects can be arbitrary and independent of each other. The sensor, however, does not observe their motions directly but ranges and bearings of points on the surface of the objects. Thus once conditioned on the measurements, motions between different objects become correlated, due to the fact that these observations are taken from a moving sensor.

In order to correctly account for this correlation, the states of the objects and that of the sensor have to be estimated in a single joint distribution. A local static background is also simultaneously estimated as part of the joint state which is essential to distinguishing measurements belonging to dynamic objects from those from static objects. The state therefore consists of three parts: the sensor pose, the dynamic objects, and the static map.

Sensor Pose Representation and Related The sensor pose $\mathbf{x}_S = [\alpha, \beta, \psi]^T$ is represented by a 2D transform from the sensor's frame of reference to a stationary world frame of reference as depicted in Fig. 2a, which is updated by vehicle odometry measurements at the prediction stage, and by laser measurements as part of the update stage as will be described in Sect. 4.2.

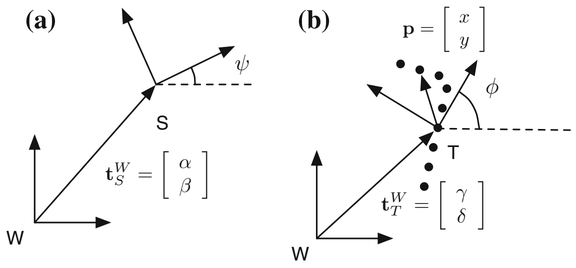

(a)
ψ
S
$\mathbf{t}_S^W = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$
W
(b)
$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix}$
ϕ
T
$\mathbf{t}_T^W = \begin{bmatrix} \gamma \\ \delta \end{bmatrix}$
W

Fig. 2 Illustrations of frame conventions and variable definitions. **a** Transform from the sensor frame to the world frame. **b** Transform from a track frame to the world frame, with boundary points represented locally to the track frame

Since the holonomic constraints apply to the vehicle but not to the sensor directly, and odometry measurements are naturally referenced to the vehicle's frame of reference, the transform between the sensor and vehicle's frames of reference are required. To account for uncertainties in this estimated transform, we include it as part of the state as $\mathbf{x}_C = [\delta\alpha, \delta\beta, \delta\psi]^T$. This is the 2D transform that transforms points from the sensor frame into the vehicle frame.

Model-Free Object Representation For convenience of description, in what follows, we will also refer to dynamic objects as "tracks", since their motion state is continuously being tracked. Each dynamic object i has its own set of axes T_i, thus its motion state is represented by the 6-vector $\mathbf{x}_T^i = [\gamma_i, \delta_i, \phi_i, \dot{\gamma}_i, \dot{\delta}_i, \dot{\phi}_i]^T$ as shown in Fig. 2b (the subscript i is dropped to avoid clutter). What is unusual about our representation is however, that *none* of these states are directly observed according to the observation model. Instead, each object has *additional* state parameters attached, named the "boundary point" coordinates, that are 2D cartesian coordinates represented *locally* to the object's frame of reference as illustrated in Fig. 2b. It is these boundary points that are directly observed according to our observation model.

To understand the intuition behind boundary points, consider the case of a moving object being illuminated by the lidar for the first time. The set of raw range and bearing measurements Z is used to initialise a new track with its 6-vector states *plus* boundary points at the locations of the raw measurements in Z but transformed into the object's frame of reference (hence the name "boundary points" because the lidar impinges on the boundary of the object). All subsequent measurements (lidar illuminations) will be taken to be noisy observations of these boundary points on the object.

This model-free representation raises an interesting and central data association question. We must decide whether or not to extend the object's boundary by initialising additional boundary points with new raw lidar measurements or simply associate the laser returns to the existing boundary points as it stands. Furthermore, which of the laser returns belong to the static background and hence have nothing to do with dynamic objects whatsoever? Our approach to data association lies at the heart of this work and is detailed in Sect. 4.3.2.

We make the assumption that dynamic objects observed in the 2D scanning plane of the sensor behave as rigid bodies. This assumption, though does not hold strictly true due to deformable bodies such as a walking pedestrian, is a close approximation when observations are constrained to the 2D plane. Under this assumption, boundary points stay fixed relative to the object's frame of reference and hence their states are unaltered at forward-prediction.

With the introduction of boundary points, each object is thus parameterised with a partial outline of its perimeter allowing objects of arbitrary shapes and dimensions to be modelled under the same representation.

Static Background Representation The representation for the static part of the state is simply a collection of boundary points as in the case of a dynamic object, except boundary points on the static background are represented with their global 2D cartesian coordinates in the *world*'s reference frame.

The Complete State Structure The complete state $\mathbf{x}$ consists of all parts described above, and is arranged as follows:

$$\mathbf{x} = [\mathbf{x}_S^T, \mathbf{x}_T^T, \mathbf{x}_b^T, \mathbf{x}_p^T, \mathbf{x}_C^T]^T, \tag{1}$$

where $\mathbf{x}_S$ is the sensor pose, $\mathbf{x}_T$ the collection of all 6-vector motion states of dynamic objects, $\mathbf{x}_b$ the collection of boundary points on the static background, $\mathbf{x}_p$ the collection of all boundary points of all dynamic objects arranged sequentially, and finally, $\mathbf{x}_C$, the extrinsic calibration parameters of the sensor.

4.2 Top Level Algorithm Description

In this section, we give a top level description of the algorithm. Each time a new measurement arrives, the mean $\hat{\mathbf{x}}$ and covariance $\mathbf{P}$ of the joint state are updated differently according to the type of the measurement (odometry or laser).

Odometry Measurement Processing In general, odometry measurements arrive at a much higher frequency than laser measurements, they need to be processed very efficiently, and therefore only forward-prediction of the sensor pose state taking the odometry measurement as a noisy control input is carried out in this case.

Laser Measurement Processing When a new laser scan frame arrives, the current state is first tested against out-of-date dynamic tracks and boundary points on the static background that have fallen out of the sensor's field of view. These are removed from the joint state. It also determines parts of the static background that have changed due to a static object transitioning into a dynamic state, hence must be removed to allow a new dynamic track to be initiated. Next, the motion part of all dynamic tracks is forward-predicted according to an appropriate motion model as will be described in Sect. 4.3.1, and followed by data association and measurement updates

(Sect. 4.3.2). Finally, any tracks appear to be static are merged with the map, and adjacent tracks following the same rigid body motion are merged into a single track. The latter is to account for the situation that occasionally a large object is tracked as different "pieces", and this allows for the pieces to be put back into a single object. This merging procedure is described in Sect. 4.3.3.

4.3 Detailed Algorithm Description

4.3.1 Dynamic Object Motion Prediction

At the prediction step after conducting a laser measurement, all dynamic tracks are predicted forward according to a generic motion model before being updated with the measurements. To capture a wide range of dynamic objects, it is desirable to use a general motion model. In this work, all dynamic tracks are assumed to follow the constant velocity model [2, Chap. 6].

4.3.2 Hierarchical Data Association

Not all state variables in the joint state (Sect. 4.1) are directly observable, for the ones that are, namely boundary points on either the static background or any dynamic object, it is ambiguous which is being observed, which is not, and indeed, whether a new boundary point needs to be initialised. Thus when new laser measurements arrive, it has to be determined for each measurement that

1. It is an observation on a static object.
    a. It is an observation of an existing boundary point.
    b. It is an observation of a new boundary point.
2. It is an observation on an existing dynamic object.
    a. It is an observation of an existing boundary point on the object.
    b. It is an observation of a new boundary point on the object.
3. It is an observation on a new dynamic object.

In addition, in case 2, it has also to be determined to which of the existing dynamic objects the measurement belongs to, and in case 3, how many new tracks need to be initialised.

This data association problem naturally breaks down into two levels. The first level operates at the coarse scale, in which measurements are first divided into clusters, and each cluster is assigned to either the static background, or a dynamic object, or used to initialise a new dynamic track. At the fine level, for each object (or the static background), measurements from the associated clusters are further associated with its existing boundary points or used to initialise new boundary points.

Coarse Level Data Association The measurements in a given laser scan are first divided into a set of clusters $\mathscr{C} = \{C_1, C_2, \ldots, C_{|\mathscr{C}|}\}$. The clusters are then assigned to the static background and dynamic objects recursively with the ICP [3] algorithm as follows. First, boundary points on the static background are aligned to the set of measurements Z with ICP, and clusters in $\mathscr{C}$ which contains measurements matched to any boundary points on the static background in this way are associated to the static background, and used to update or initialise new boundary points at the fine level for the static background. Then the associated clusters are removed from $\mathscr{C}$ and a similar procedure follows recursively for each dynamic track. The clusters that remain in $\mathscr{C}$ at the end of this process are thus not associated with any existing track (or the static background), and each cluster will initialise a new tentative dynamic track as will be detailed in Sect. 4.3.3.

Fine Level Data Association Given a set of clusters associated with a certain track (or the static background), the fine level data association must find a matching satisfying certain desirable criteria that assigns measurements contained in the clusters to boundary points on the track. Correct assignment is critical to successful tracking, and, the stability of the system as a whole, due to the fact that correlation is introduced between all pairs of variables in the joint state. In particular, all state variables we would like to infer: the sensor pose, the dynamic states of the tracked objects, are not directly observed.

Joint Compatibility Branch and Bound (JCBB) [9] is a well-known data association algorithm that takes into account the correlations between observations. Explained in our nomenclature, an association between the set of measurements and the set of boundary points is called a *feasible* association if:

1. Each measurement is associated to at most one boundary point, and no two measurements are associated to the same boundary point (one-one association).
2. Each matching of a measurement to a boundary point is individually compatible as described below (individual compatibility).
3. The overall data association is jointly compatible as described below (joint compatibility).

To clarify the concepts of individual and joint compatibilities, consider a boundary point whose observation model has the standard form $\mathbf{z}_j = \mathbf{h}_j(\mathbf{x}) + \mathbf{w}_j$. Here $\mathbf{x}$ is the joint state defined in Sect. 4.1, and $\mathbf{w}_j$ is the additive zero-mean measurement noise. Thus its innovation covariance matrix is $\mathbf{S}_j = \mathbf{H}_j\mathbf{P}\mathbf{H}_j^T + \mathbf{R}$. Here $\mathbf{H}_j$ is the Jacobian of the function $\mathbf{h}_j$ evaluated at the current state mean, and $\mathbf{R}$ is the measurement noise covariance matrix (we assume all measurements have the same noise covariance matrix).

Individual Compatibility Individual compatibility requires the assigned measurement $\hat{\mathbf{z}}_i$ must fall within a certain confidence region of boundary point j's validation gate, i.e. an assignment of $\hat{\mathbf{z}}_i$ to $\mathbf{z}_j$ is individually compatible if:

$$(\hat{\mathbf{z}}_i - \mathbf{h}_j(\hat{\mathbf{x}}))^T \mathbf{S}_j^{-1} (\hat{\mathbf{z}}_i - \mathbf{h}_j(\hat{\mathbf{x}})) \leq \chi^2_{d,\alpha}, \tag{2}$$

where $\chi^2_{d,\alpha}$ is the χ^2 validation gate threshold of degree of freedom d and confidence level α. Here, d is the measurement dimension, hence $d = 2$, because each measurement contains a range and a bearing $\hat{\mathbf{z}} = [\hat{r}, \hat{\theta}]^T$.

Joint Compatibility Under the assumption of independent observations, the joint observation model of a complete association σ is given by

$$\mathbf{h}_\sigma(\mathbf{x}) = [\mathbf{h}^T_{\sigma(1)}(\mathbf{x}), \mathbf{h}^T_{\sigma(2)}(\mathbf{x}), \ldots]^T, \tag{3}$$

with the innovation covariance

$$\mathbf{S}_\sigma = \begin{bmatrix} \mathbf{H}_{\sigma(1)}\mathbf{P}\mathbf{H}_{\sigma(1)}{}^T + \mathbf{R} & \mathbf{H}_{\sigma(1)}\mathbf{P}\mathbf{H}_{\sigma(2)}{}^T & \cdots \\ \mathbf{H}_{\sigma(2)}\mathbf{P}\mathbf{H}_{\sigma(1)}{}^T & \mathbf{H}_{\sigma(2)}\mathbf{P}\mathbf{H}_{\sigma(2)}{}^T + \mathbf{R} & \\ \vdots & & \ddots \end{bmatrix}, \tag{4}$$

where $\sigma(1)$ denotes the index of the boundary point associated to the first *assigned* measurement and so on. Thus the joint measurement has dimension $N_a d$ if the number of assigned measurements is N_a. An association σ is jointly compatible if

$$(\hat{\mathbf{z}}_\sigma - \mathbf{h}_\sigma(\hat{\mathbf{x}}))^T \mathbf{S}_\sigma^{-1} (\hat{\mathbf{z}}_\sigma - \mathbf{h}_\sigma(\hat{\mathbf{x}})) \le \chi^2_{N_a d,\alpha}. \tag{5}$$

Here $\hat{\mathbf{z}}_\sigma$ is the collection of the measurements that are *assigned* to some boundary point according to association σ.

The JCBB algorithm then finds the *feasible* association that has the largest number of assigned measurements N_a^*. Since there are in general many feasible associations with $N_a = N_a^*$, the algorithm finds the association σ^* that gives the lowest joint Normalised Innovation Squared (jNIS, defined to be the expression to the left of the inequality in Eq. 5).

The JCBB-Refine Algorithm Unfortunately, the JCBB algorithm is an exponential algorithm in the number of measurements to be assigned. This means it is not directly applicable to our application domain, since in our case observations are raw laser measurements.

We introduce the JCBB-Refine algorithm, which instead of aiming to find the optimum assignment σ^*, we only find a *good* association $\tilde{\sigma}$ that is *feasible*. Of course, there are many *feasible* associations, a *good* association must be measured relative to some gauge. The JCBB-Refine algorithm we propose here takes an initial association σ_0 as a starting point, and finds a feasible association that has as many assigned measurements and as low a jNIS as possible in a greedy manner while respecting the initial association σ_0. The initial association σ_0 can be arbitrary, i.e. it does *not* have to be feasible. In fact, none of the feasibility conditions has to be satisfied.

Given σ_0, the algorithm first removes assignments that do not comply with individual compatibility (i.e. noncompliant measurements become unassociated with any boundary point), and then removes duplicate assignments with a single pass through the measurements. After these, the resulting association satisfies feasibility conditions 1 and 2. The algorithm then proceeds to iteratively removing the assignment that leads to the most jNIS reduction until condition 3 is satisfied. Starting from this minimal set of assignments that is now feasible, the unassociated measurements are then tried in turn, and assigned to the boundary point (among the boundary points that are individually compatible and yet unassigned) that gives the lowest jNIS if the assignment does not violate joint compatibility. The resulting association is thus guaranteed to remain feasible.

The JCBB-Refine algorithm can be initialised with any sensible starting assignment σ_0. In our particular application, the assignment as a result of the ICP matching at the coarse level association is a natural starting point. The association after the refinement is then used to update the joint state with the associated measurements, and all unassociated measurements initialise new boundary points to extend the object boundary. Explicit forms of our observation models for different types of boundary points are stated in an appendix.

Recursive Updates in Triangular Form It is shown [9] that the innovation covariance matrix $\mathbf{S}$, its inverse, and the jNIS can be computed recursively as hypotheses are being tested. However in its direct form, the recursion suffers from numerical stability issues when the number of measurements becomes large because both $\mathbf{S}$ and $\mathbf{S}^{-1}$ have to be maintained to be positive definite. We show the same computation can be achieved in the triangular form, which is a numerically stable representation for positive definite matrices.

To begin with, at step k, assume a decomposition for $\mathbf{S}_k$ is given such that $\mathbf{S}_k = \mathbf{U}_k^T \mathbf{U}_k$ for some upper triangular matrix $\mathbf{U}_k$, for example through Cholesky decomposition, so that $\mathbf{S}_k^{-1} = \mathbf{G}_k \mathbf{G}_k^T$, $\mathbf{G}_k = \mathbf{U}_k^{-1}$ (note $\mathbf{G}_k$ is also upper triangular). And the next iteration selects a new boundary point to be assigned to a measurement such that

$$\mathbf{S}_{k+1} = \begin{bmatrix} \mathbf{S}_k & \mathbf{W}_k^T \\ \mathbf{W}_k & \mathbf{N}_k \end{bmatrix}. \tag{6}$$

Then it can be shown that

$$\mathbf{U}_{k+1} = \begin{bmatrix} \mathbf{U}_k & \mathbf{R}_k^T \\ \mathbf{0} & \mathbf{F}_k \end{bmatrix}, \tag{7}$$

where $\mathbf{R}_k = \mathbf{W}_k \mathbf{G}_k$, $\mathbf{F}_k = \mathrm{chol}(\mathbf{N}_k - \mathbf{R}_k \mathbf{R}_k^T)$, and

$$\mathbf{G}_{k+1} = \begin{bmatrix} \mathbf{G}_k & -\mathbf{G}_k \mathbf{R}_k^T \mathbf{M}_k \\ \mathbf{0} & \mathbf{M}_k \end{bmatrix}, \tag{8}$$

where $\mathbf{M}_k = \mathbf{F}_k^{-1}$. In addition, we keep track of a vector $\boldsymbol{\xi}_k = \mathbf{G}_k^T \boldsymbol{\nu}_k$. Its update equation is given by $\boldsymbol{\xi}_{k+1} = [\boldsymbol{\xi}_k^T, \boldsymbol{\mu}_k^T]^T$ where $\boldsymbol{\mu}_k = \mathbf{M}_k^T(\tilde{\boldsymbol{\nu}}_k - \mathbf{R}_k \boldsymbol{\xi}_k)$. Here, $\tilde{\boldsymbol{\nu}}_k$ is the innovation vector of the newly assigned measurement. And the jNIS can be updated simply as $\mathrm{jNIS}_{k+1} = \mathrm{jNIS}_k + \boldsymbol{\mu}_k^T \boldsymbol{\mu}_k$. Given $\mathbf{U}_k$, $\mathbf{G}_k$ and $\boldsymbol{\xi}_k$ at any stage of the computation, the innovation covariance $\mathbf{S}_k$, its inverse and the innovation vector $\boldsymbol{\nu}_k$ can be easily retrieved if needed.

We note this form has the same computational complexity as the recursion introduced in [9], but is more numerically stable.

4.3.3 Track Initialisation and Merging

The initialisation of new dynamic tracks is non-trivial because we have to ensure that only new *dynamic* objects are initialised into new tracks and static objects are merged with the static background. To this purpose, we apply the technique of constrained initialisation [17], where each new track's motion status is deferred until it has accumulated enough evidence to make the correct decision. Specifically, a new track is first marked as "tentative" when initialised, and becomes "mature" only if it is continuously being observed for more than a fixed number of frames (otherwise it is dropped). Then it is tested against the static background, and each existing dynamic track in turn for merging. The test and merging are all handled consistently within the same Bayesian filtering framework. If all merging tests fail, it is declared "established" and added to the set of existing dynamic tracks.

In the case of testing against merging with the static background, a fictitious noiseless measurement of values all zero on the absolute velocity (including the angular velocity) of the tentative track is considered. If this measurement passes the validation gate, the tentative track is considered to be a static object, therefore should be merged into the static background. The fictitious measurement is hence used to update the joint state $\mathbf{x}$ as if it was an actual sensor measurement. After the update, all available information from the track will have been transferred to the rest of the joint states, and it can be safely marginalised out after copying over its boundary points to the static background to complete the merge. A similar procedure applies to merging tests with an existing dynamic track. In this case, the fictitious measurement applies to the relative motion of the tentative track to the existing track under consideration.

The same merging procedure is also conducted at the end of each processing cycle for testing each existing track against merging with the static background and other existing tracks as mentioned in Sect. 4.2.

5 System Evaluation

In this section, we quantitatively evaluate the proposed system, and compare its performance against an industrial standard solution for benchmarking. We note there exists a large body of work on similar application domains (For example, [7, 8, 16]), however it is often difficult to obtain a fair quantitative comparison to the methods due to either a lack of quantitative results or difficulty of a direct comparison using a common dataset.

Table 1 Details of the training and test datasets

Dataset	No. laser frames	Duration (min)	Drive length (km)	No. objects
Training	3508	4.68	1.04	7517
Test	2151	2.87	0.82	5928

Here each count of an "object" is a single observation of an object instance in a single laser scan frame

5.1 Experiment Setup

Our experiment platform is a modified Nissan Leaf that is equipped with a SICK LDMRS laser scanner, which is a scanner targeted at object tracking applications on mobile platforms. It scans the environment in four vertically separated scanning planes at 12.5 Hz and produces native object tracking information at the same time. Odometry information is provided internally as part of the vehicle state at 100 Hz.

We collected data of busy traffic at the centre of Oxford containing a variety of dynamic objects including pedestrians, cars, bicyclists, buses, trucks, motorcycles and so on, and extracted two busy sections of the log right at the centre of the city for evaluation. One dataset is used to find the best-performing parameter set, and is hence named the *training* set, and the other, the test set, is used to obtain unbiased test results running under the trained parameter set for fair comparison. Table 1 lists the details of the two datasets respectively. Ground truth detections of dynamic objects are obtained from both datasets by manual labelling.

5.2 Evaluation Metric and System Training

We evaluate the system's ability to detect dynamic objects against the ground truth using the standard Precision and Recall metrics. Specifically, Precision and Recall are computed over the detected object boxes against the hand-labeled ground truth object boxes using the overlapping criterion as is commonly used in the computer vision community [4]. An object box is marked as a true detection if it overlaps with a ground truth object box by more than a fixed percentage threshold. In all our results, we use 0.5 as the percentage overlap threshold. And a detection is matched to at most one ground truth object, and multiple detections of the same ground truth object are treated as false positives.

To train the system for best performing parameter sets, we follow an approach similar to that described in [5] as follows: both Precision P and Recall R are functions of system parameters, thus if the number of system parameters exceeds one, the set of all feasible (R, P) pairs will in general occupy a continuous 2D space in the R-P plane. The best parameters are then the parameters that give rise to the (R, P) pairs at the *frontier* of the feasible region (conceptually corresponds to the top-right boundary of the feasible region, see Fig. 3a for an example).

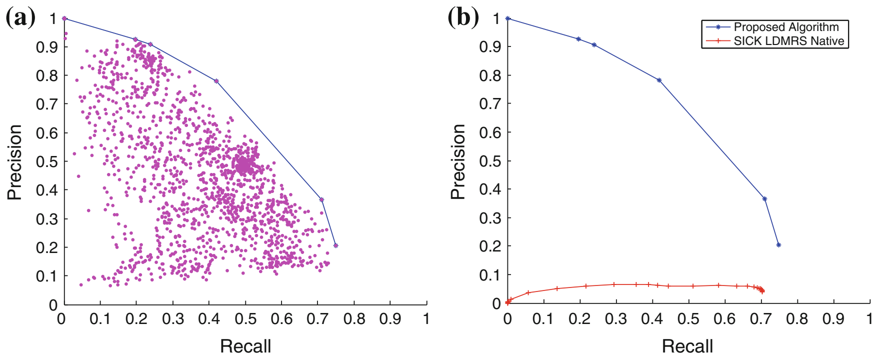


Fig. 3 **a** Scatter plot of the obtained 1803 sample parameter settings using [10] with the estimated frontier overlaid. **b** Precision-Recall tuning curves for the proposed system and the SICK LDMRS native tracking system

Formally, the 2D feasible region parameterised by the set of all possible parameters $\mathscr{P}$ is given by $\mathscr{F} = \{(R(\mathbf{p}), P(\mathbf{p}))\colon \mathbf{p} \in \mathscr{P}\}$, the frontier parameter set of $\mathscr{F}$ is given by $F = \{\mathbf{q} \in \mathscr{P}\colon \forall \mathbf{p} \in \mathscr{P},\ \ R(\mathbf{p}) \leq R(\mathbf{q}) \text{ or } P(\mathbf{p}) \leq P(\mathbf{q})\}$. In other words, a parameter set is in F if and only if it is not possible to achieve both a higher Precision and a higher Recall.

To find this frontier parameter set, we apply a Bayesian parameter tuning algorithm developed by Snoek et al. [10] to bias the search in the high-dimensional parameter space to look for satisfactory parameter settings, and obtain an approximation to the frontier parameter set by finding the upper part of the convex hull of the obtained (R, P) scatter plot. Figure 3a shows the obtained 1803 sample parameter settings with the algorithm, and the blue curve shows the extracted frontier.

Since the SICK LDMRS native tracking system clusters each incoming scan and keeps track of every cluster, it makes no distinction between static and dynamic objects. To compare the systems under the same setting, we take tracks with estimated speeds higher than a given threshold to be the detected dynamic objects. It would be desirable to be able to fine-tune the parameters of the LDMRS's native tracking system. However, the most critical parameters are fixed internally to the sensor, and modifications are unfortunately not feasible.

Figure 3b presents the Precision-Recall curves for the proposed, and LDMRS's native tracking systems. The curve of the LDMRS's native system is generated by varying the speed threshold as described above. As can be seen, the proposed system outperforms the LDMRS's native system by a significant margin. This is somewhat expected, since the LDMRS's native system tracks only the cluster centroids, which are not stable reference points on the objects to track due to occlusions and dependency on the sensor viewpoint. On the other hand, the proposed system enforces each track's frame of reference to be attached rigidly to the object, and dynamic objects are explicitly handled differently to static ones.

5.3 Test Case Performance

Given a range of operating points along the Precision-Recall curve, we choose empirically a single parameter setting that achieves the best balanced performance from Fig. 3 for each system. Specifically we choose the parameter setting that gives $R = 0.53$ and $P = 0.57$ for the proposed system and the speed threshold that achieves $R = 0.69$ and $P = 0.05$ for the LDMRS's native system. All experiments that follow report metrics evaluated on the *test* dataset using these chosen operating points.

Figure 4a, b show performance metrics for the two systems on the test dataset as the detection range is varied. Both show a decreasing trend on both Precision and Recall as the detection radius increases. Figure 4c places the systems under common axes for comparison. From the figure, although the close-range performances are similar (with the proposed system slightly outperforming), the difference is significant from 20 m onwards.

Figure 4d compares the instantaneous performance at each frame of the two systems. F_1-measures are evaluated at each frame based on detections of the past 100 frames for each system, and results are plotted against the frame number. While the proposed system outperforms the LDMRS at most frames, there are occasional performance drops. Closer inspection into the dataset reveals that around Frame 400 there exists a period of driving with very few number of dynamic objects present, hence the apparent low performance from both systems. However, near to Frame 1300, many walking pedestrians close to background clutter are present which are missed out by the proposed system due to segmentation failure. The LDMRS performs better in this scenario but in sacrifice of Precision.

Our current prototype implementation of the proposed system in MATLAB runs in real-time at 2 Hz on a MacBook Pro equipped with a duo-core 2.4 GHz Intel i5 CPU and 4 GB of RAM.

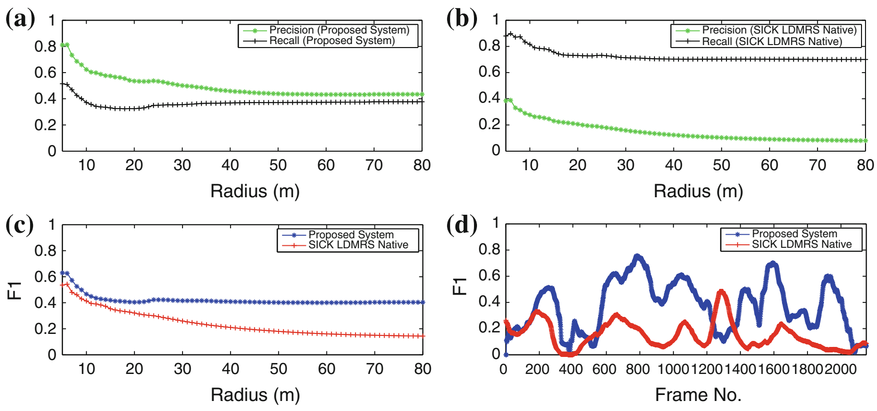


Fig. 4 **a** Precision and Recall versus operating radius for the proposed system. **b** Precision and Recall versus operating radius for the SICK LDMRS's native system. **c** F_1-measure versus operating radius for both systems. **d** F_1-measure over past 100 frames versus frame number for both systems

6 Conclusions

We presented a unified framework for jointly estimating the sensor pose, a local static background and dynamic states of moving objects, focused mainly on accurate moving object detection. Observations in our formulation are raw sensor measurements and object states are inferred as hidden variables under a rigid body constraint.

Within the same unified framework, we proposed a novel two-level data association algorithm that takes benefits of both the density of observations and strong correlations between them. A new variant of the JCBB [9] algorithm was suggested to tackle with large numbers of measurements, and a solution to numerical stability issues under such scenarios was also presented.

The proposed system was tuned systematically, and demonstrated to outperform an existing industry standard.

Acknowledgments This work is supported by the Clarendon Fund. Paul Newman is supported by an EPSRC Leadership Fellowship, EPSRC Grant EP/I005021/1. The authors wish to thank Jasper Snoek for making the Spearmint Bayesian optimisation package publicly available.

Appendix

In this appendix, we state the exact forms of the observation models applied to boundary points on the static background and dynamic objects respectively. All variables involved in what follows are defined in Sect. 4.1, and the function **u** maps a pair of 2D cartesian coordinates into polar coordinates.

Each boundary point j on the static background may potentially generate a laser measurement $\mathbf{z} = [r, \theta]^T$, and hence its measurement model is the boundary point's location in polar coordinates in the *sensor*'s frame of reference:

$$\mathbf{h}_j(\mathbf{x}) = \mathbf{u}(\mathbf{g}(\mathbf{x}_S, \mathbf{b}_j)),\ \ \mathbf{g}(\mathbf{x}_S, \mathbf{b}_j) = \mathbf{R}^T(\psi)\left(\begin{bmatrix} x_j \\ y_j \end{bmatrix} - \begin{bmatrix} \alpha \\ \beta \end{bmatrix}\right). \tag{9}$$

Each boundary point j on any dynamic track i may also give rise to a laser measurement, and the measurement model in this case is the 2D polar coordinates of the boundary point in the *sensor* frame, and is given by:

$$\mathbf{h}_j(\mathbf{x}) = \mathbf{u}(\mathbf{g}(\mathbf{x}_S, \mathbf{x}_T^i, \mathbf{p}_j^i)),\ \ \mathbf{g}(\mathbf{x}_S, \mathbf{x}_T^i, \mathbf{p}_j^i) = \mathbf{R}^T(\psi)\left(\mathbf{R}(\phi_i)\begin{bmatrix} x_j^i \\ y_j^i \end{bmatrix} + \begin{bmatrix} \gamma_i \\ \delta_i \end{bmatrix} - \begin{bmatrix} \alpha \\ \beta \end{bmatrix}\right). \tag{10}$$

References

1. Arras, K., Grzonka, S., Luber, M., Burgard, W.: Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2008, pp. 1710–1715 (2008)
2. Bar-Shalom, Y., Kirubarajan, T., Li, X.R.: Estimation with applications to tracking and navigation. Wiley, New York (2002)
3. Besl, P., McKay, N.D.: A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Mach. Intell. **14**(2), 239–256 (1992)
4. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The Pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)
5. Gavrila, D.M., Munder, S.: Multi-cue pedestrian detection and tracking from a moving vehicle. Int. J. Comput. Vis. **73**(1), 41–59 (2007)
6. Leonard, J., How, J., Teller, S., Berger, M., Campbell, S., Fiore, G., Fletcher, L., Frazzoli, E., Huang, A., Karaman, S., Koch, O., Kuwata, Y., Moore, D., Olson, E., Peters, S., Teo, J., Truax, R., Walter, M., Barrett, D., Epstein, A., Maheloni, K., Moyer, K., Jones, T., Buckley, R., Antone, M., Galejs, R., Krishnamurthy, S., Williams, J.: A perception-driven autonomous urban vehicle. J. Field Robot. **25**(10), 727–774 (2008)
7. Mertz, C., Navarro-Serment, L.E., MacLachlan, R., Rybski, P., Steinfeld, A., Suppe, A., Urmson, C., Vandapel, N., Hebert, M., Thorpe, C., Duggins, D., Gowdy, J.: Moving object detection with laser scanners. J. Field Robot. **30**(1), 17–43 (2013)
8. Miyasaka, T., Ohama, Y., Ninomiya, Y.: Ego-motion estimation and moving object tracking using multi-layer LIDAR. In: Proceedings of the Intelligent Vehicles Symposium, pp. 151–156. IEEE (2009)
9. Neira, J., Tardos, J.: Data association in stochastic mapping using the joint compatibility test. IEEE Trans. Robot. Autom. **17**(6), 890–897 (2001)
10. Snoek, J., Larochelle, H., Adams, R.P.: Practical Bayesian optimization of machine learning algorithms. In: Neural Information Processing Systems (2012)
11. Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., Mahoney, P.: Stanley: the robot that won the DARPA grand challenge. J. Field Robot. **23**(9), 661–692 (2006)
12. Tipaldi, G., Ramos, F.: Motion clustering and estimation with conditional random fields. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009, pp. 872–877 (2009)
13. Topp, E., Christensen, H.: Tracking for following and passing persons. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), pp. 2321–2327 (2005)
14. Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M.N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T.M., Kolski, S., Kelly, A., Likhachev, M., McNaughton, M., Miller, N., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Salesky, B., Seo, Y.W., Singh, S., Snider, J., Stentz, A., Whittaker, W.R., Wolkowicki, Z., Ziglar, J., Bae, H., Brown, T., Demitrish, D., Litkouhi, B., Nickolaou, J., Sadekar, V., Zhang, W., Struble, J., Taylor, M., Darms, M., Ferguson, D.: Autonomous driving in urban environments: boss and the urban challenge. J. Field Robot. **25**(8), 425–466 (2008)
15. van de Ven, J., Ramos, F., Tipaldi, G.: An integrated probabilistic model for scan-matching, moving object detection and motion estimation. In: Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 887–894 (2010)
16. Wang, C.C., Thorpe, C., Thrun, S.: Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded

urban areas. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Taipei, Taiwan (2003)
17. Williams, S.B.: Efficient solutions to autonomous mapping and navigation problems. Ph.D. thesis, Australian Centre for Field Robotics, The University of Sydney (2001)
18. Yang, S.W., Wang, C.C.: Simultaneous egomotion estimation, segmentation, and moving object detection. J. Field Robot. **28**(4), 565–588 (2011)

Part VI
Planning

Task-Oriented Grasp Planning Based on Disturbance Distribution

Yun Lin and Yu Sun

Abstract One difficulty of task-oriented grasp planning is task modeling. In this paper, a manipulation task was modeled by building a non-parametric statistical distribution model from disturbance data captured during demonstrations. This paper proposes a task-oriented grasp quality criterion based on distribution of task disturbance and uses the criterion to search for a grasp that covers the most significant part of the disturbance distribution. To reduce the computational complexity of the search in a high-dimensional robotic hand configuration space, as well as to avoid a correspondence problem, the candidate grasps are computed from a reduced configuration space that is confined by a set of given thumb placements and thumb directions. The proposed approach has been validated with a Barrett hand and a Shadow hand on several objects in simulation. The resulting grasps in the evaluation generated by our approach increase the coverage of frequently-occurring disturbance rather than the coverage of a large area with a scattered distribution.

1 Introduction

Manipulation and grasp have been active research topics in robotics. One of the primary goals of the research is the choice of an appropriate grasp, in terms of task requirement and stability properties, given an object associated with a manipulation task to be performed [1]. Such a problem is referred to as the grasp synthesis problem. To solve this problem, different approaches and algorithms have been developed for the robotic hand to execute a stable manipulation task.

One solution to grasp synthesis problem is grasp planning. Grasp planning uses optimization mathematics to search for the optimal contact placement on an object, which gives rise to difficulty in choosing a quality criterion for the optimization

Y. Lin · Y. Sun (✉)
University of South Florida, 4202 E. Fowler Ave., ENB118, Tampa, FL 33620-5399, USA
e-mail: yunlin@mail.usf.edu

Y. Sun
e-mail: yusun@cse.usf.edu

procedure. One widely-used quality criterion is the force-closure property, which measures the capability of a grasp to apply appropriate forces on an object to resist disturbances in any direction, defined as the radius of the largest six-dimensional wrench space sphere centered at the origin and enclosed with the unit grasp wrench space [2]. Related research was developed in [3–6], etc. Nevertheless, they are task-independent, in which an evenly distributed disturbance in all directions is assumed.

In many manipulation tasks, however, such as drinking, writing and handling a screwdriver, a task-related grasp criterion has to be applied for the choice of appropriate grasp configurations for different task requirements. A typical task-oriented grasp method is to choose a suitable task wrench space (TWS) and then measure how good a task wrench space can be fitted into a grasp wrench space [4, 7–10]. Few works have considered the task information in grasp planning due to the difficulty of modeling a task [7, 9, 11]. To obtain the task wrench space in reality, necessary sensors are required to measure the contact regions and contact normals, which remains a challenge. This is the main reason why most works empirically approximate the task wrench space rather than actually measure it. Instead of a wrench space ball used in force-closure quality measure, Li and Sastry [7] developed a quality criterion to measure the ability of a grasp to perform a task wrench space using a six-dimensional wrench space ellipsoid to better approximate a task. The research in [10] approximated the task wrench space as a task polytope and focused on the computation of task-oriented quality measures.

Pollard [4] proposed the object wrench space (OWS) that takes the complete object geometry into consideration. The OWS integrates all disturbance wrenches that can be exerted anywhere on the object. Borst et al. [9] presented an algorithm to approximate the OWS by an ellipsoid and to measure how good the OWS ellipsoid can be fitted into a Grasp Wrench Space (GWS). The idea of OWS takes all possible disturbances into account, which is good for unknown tasks but is not task-specific; for a specific task, a grasp does not need to perform the whole OWS but to perform the required subset TWS of the task wrench space.

Another difficulty of task-oriented grasp planning is the computational complexity of the searching in the high-dimensional hand configuration space. It is, therefore, natural to introduce human experience relative to a task [12–17]. Aleotti and Caselli [18] used data gloves to map human-hand to robotic-hand workspace and captured the task wrench space in virtual reality. They also considered a database of candidate grasps, and grasps were evaluated by a task-related quality measure. However, the correspondence problem has been a crucial issue to map between different configuration spaces of the human hand and the robotic hand. Research in [19] searched for candidate grasps by a shape-matching algorithm and evaluated the grasps by a task-oriented criterion. However, the same modeling problem of the TWS still exists and the work also relies on empirical modeling.

This paper proposes a grasp quality criterion, called the task coverage grasp quality metric, to compute the proportion of task disturbance that a grasp covers. Instead of assuming an evenly-distributed task wrench space, this approach takes into account the task disturbance distribution measured from human demonstration, since it is possible that disturbance wrenches in some directions occur more frequently than in other areas, even if they may be smaller than wrenches that occur less frequently. In two tool manipulations, for example, a knife and a fork have similar shapes but have disturbance wrench distribution along different directions, hence favoring different grasps. Therefore, a targeted grasp is prone to increasing the coverage of most frequent disturbances, rather than a grasp with the same coverage of the area with scattered distributed disturbance. To reduce the computational complexity of the search in high-dimensional robotic hand configuration space, as well as to avoid a correspondence problem, the candidate grasp is computed from a set of given thumb placements rather than contact points [20–24] on an object surface. One advantage of thumb placement is that it is independent of the physical constraints of a given hand, which has the problem of solving the inverse kinematics that satisfies the constraints imposed by contact points [25]. Every thumb placement is associated with the direction thumb should point to, which further reduce the search space of wrist positions and orientations.

2 Grasp Analysis

2.1 Grasp Preliminaries

Considering a multi-fingered robotic hand grasping an object, a grasp comprises multiple contact points. Assuming a hard finger model of the grasp [26], i.e., point contact with friction (PCWF), the most common friction model is Coulomb's friction model; at each local contact, the tangential force is bounded by the normal force, $f^t \leq \mu f^n$, where f^t is the tangential force component, f^n is the normal force component, and μ is the coefficient of friction. Thus, all feasible contact forces are constrained to the friction cone. The friction has a vertex at the contact point, and the axis is along the contact normal, with an opening angle of $2\tan^{-1}\mu$. For the convenience of computation, the circular friction cone is usually approximated with an m-sided pyramid. Then, any contact force f_i at the ith contact that is within the constraint of friction cone can be represented as a convex combination of the m force vectors on the boundary of the cone:

$$f_i \approx \sum_{j=1}^{m} \alpha_j f_{ij} \tag{1}$$

where coefficient $\alpha_j \geq 0$, and $\sum_{j=1}^{m} \alpha_j = 1$.

The 3-d force vector f_i and torque vector τ_i can be written as a wrench w_i. Each contact can be described with a six-dimensional vector of wrench w_i:

$$w_i = \begin{bmatrix} f_i \\ \tau_i = \lambda(d_i \times f_i) \end{bmatrix} \tag{2}$$

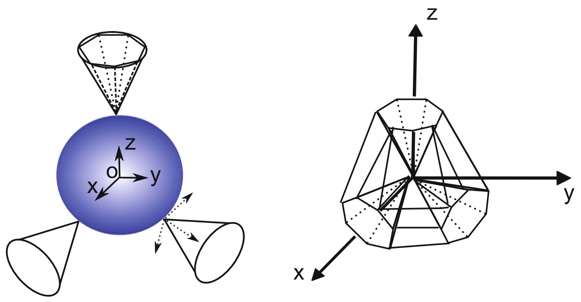


Fig. 1 The wrench space of a grasp

where d_i is the vector from global origin of the object to the contact point and λ is the scale factor of torque to force conversion. λ can be set to be the inverse of the maximum radius from the torque origin so that torque is independent of the object scale [4].

Given n contact points of a grasp, the unit GWS, written as $W(G)$, can be defined as the linear combination of the unit wrench space at each contact:

$$W(G) = \left\{ w | w = \sum_{i=1}^{mn} \alpha_i w_i, \alpha_i \geq 0, \sum_{i=1}^{mn} \alpha_i = 1, |w_i| = 1 \right\} \tag{3}$$

In other words, UGWS is the set of all possible resultant wrenches that can be applied to the object by all the contacts if applying unit magnitude of contact force, i.e., the convex hull of the contact wrenches (Fig. 1).

A typical way of evaluating grasp quality is to compute force-closure, i.e., the ability of a grasp to equilibrate external force and moment in any directions by applying appropriate forces. It implies that if the origin of the wrench space is in the convex hull, then the grasp is force closure. Similar to the grasp wrench space, a task can also be described as the space of disturbance wrenches that must be applied to the object. Ferrari and Canny [3] quantified the force-closure property by the magnitude of the contact wrenches that can compensate the disturbance wrench in the worst case. If no task-oriented information is provided to form a subset of the whole space of wrenches, a typical task wrench space is a 6D ball T_{ball} centered at the wrench space origin, where external disturbance is uniformly weighted (Left of Fig. 2). The grasp quality is the reciprocal of the scale to enlarge the grasp wrench space so that it contains the whole task wrench space:

$$Q(G) = \frac{1}{k_m} \tag{4}$$

$$k_m(G) = min(k) | T_{ball} \in k \cdot W(G), \tag{5}$$

In other words, the $k_m(G)$ is the minimum magnitude of contact force in order to be capable of resisting all task wrenches. The larger k_m is, the greater effort is

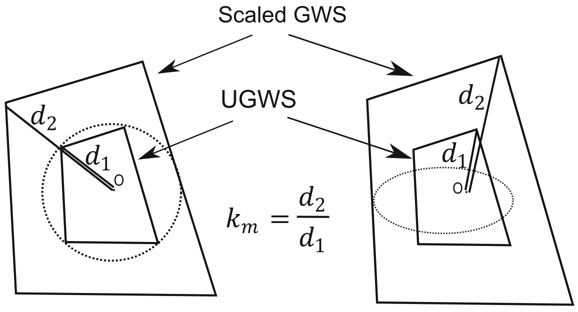


Fig. 2 Grasp quality measures for (*left figure*) task ball represented by the *dashed circle*, and (*right figure*) task ellipsoid represented by the *dashed ellipse*

needed for a grasp to encounter the task wrench along the weakest direction. The grasp planning is to find the maximum $Q(G)$, the reciprocal of $k_m(G)$.

2.2 Measure of Task Wrench

The quality measure in Eq. 4 can also be used for different task requirements instead of using a uniform ball. Related research has been conducted in [4, 7–10]. Li and Sastry [7] developed a quality criterion to measure the ability of a grasp to perform a task wrench space using a six-dimensional wrench space ellipsoid to better approximate a task (Right of Fig. 2). Although this measure takes task requirement into account, they stated that the data acquisition is difficult, so it is challenging to model the task. As reviewed in the Introduction, while most researchers focus on the problems of defining the task wrench space quality and the measurements of how good a grasp can be fitted into a task wrench space, quite few address this practical problem of how to measure the demonstrated task wrench space. Perhaps the only work that measures task wrench space from demonstration was the one conducted by Aleotti and Caselli [18]. In their work, the demonstrated task wrench space was estimated in simulation by mapping the captured hand posture to virtual reality, where a correspondence problem still exists due to two mappings from reality to virtual reality and demonstrated task wrench space from human demonstration to the robot.

Most of the works [7, 10, 19] relied on much experience to estimate the task wrench space by predicting the contact disturbance. Taking tool manipulations such as pen, screwdriver, scoop, fork, toothbrush, etc. for example, the contact disturbance is expected to be applied on the tip of those tools. Then the empirical task-oriented disturbance wrench space is a friction cone applied to the tip. The wrench space is assumed to be uniformly distributed in the space. However, even if the disturbance is applied to the same location of different tools, the disturbance wrench can distribute unevenly over the whole task wrench space. Comparing a writing task and manipulation of a screwdriver, for instance, although both require the grasp to resist disturbance force applied to the tip, they have different disturbance distribution.

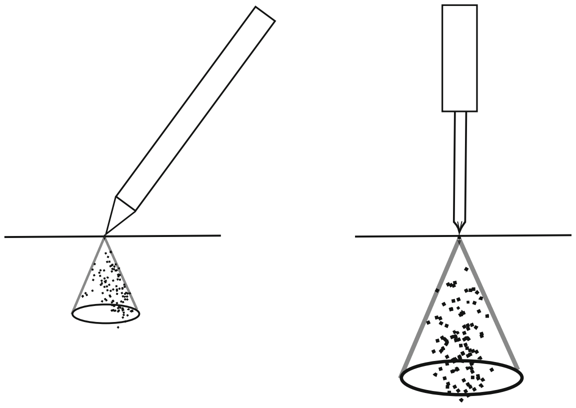


Fig. 3 Disturbance distribution of two tasks. *Left figure* shows a writing task with a pen; *right figure* shows a screwing task with a screwdriver

As illustrated in Fig. 3 the comparison between a pen and a screwdriver, the disturbance distributions of them are different. For the writing task, the main disturbance wrench of a writing task is the force pointed to the upper-left direction, and the torque generated along with the force. Hence, the grasp wrench space should be able to apply the opposite force to resist the disturbance, which is distributed primarily in the right area of the friction cone shown in the figure; whereas the main disturbance wrench of the screwdriver is the normal force to the surface and the rotational friction around the principle axis of the screwdriver. Also, the expected disturbance force of the screwdriver is larger than that of the pen. Therefore, different distributions of wrenches in a task wrench space would result in different preferred grasps.

To measure the distribution of the disturbance wrench space, we provided a user interface consisting of a haptic device Phantom Omni, and a virtual reality environment. For each task, a user is asked to manipulate a tool using the haptic device (see Fig. 4 for example). The haptic device provides the user with a haptic feedback of the interaction force with the virtual environment. The virtual reality environment was developed based on Chai3D [27], an open source C++ library for computer haptics, visualization, and interactive real-time simulation. It integrates C++ library of Open Dynamic Engine (ODE) for collision detection and OpenGL library for graphical visualization. We integrated the QHull library to calculate the convex Hull [28]. The

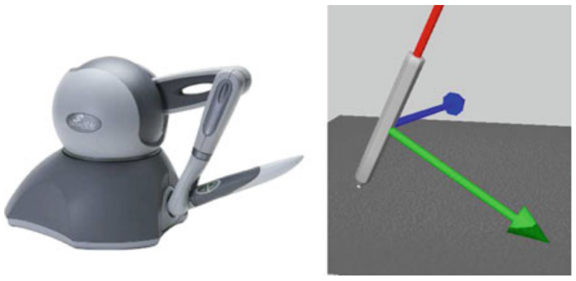


Fig. 4 A user interface for demonstration. *Left figure* A haptic device, Phantom OMNI, to manipulate a virtual object. *Right figure* the virtual environment

collision force of the tool is captured in the environment after each iteration. The task wrench space (*TWS*) is a set of all wrenches measured over time t.

$$TWS = \{w(t)|w(t) = w_c(t) + w_n(t)\} \tag{6}$$

where $w(t)$ is a wrench at time t; $w_c(t)$ is the contact wrench of the tool with the environment; $w_n(t)$ is non-contact wrench. The non-contact wrench $w_n(t)$ is an offset wrench that includes forces not acting on the surface of the object, such as gravity and other force generated by acceleration. Here, we consider only gravity because motion of the tool is assumed to be pseudo-static. Gravity is considered as the force acting on the center of mass of the object. If the center of mass is set as the torque origin, the wrench compensated by the gravity is a wrench with zero torque. If no contact occurs during the manipulation, only gravity is required to be compensated, e.g., when lifting up a book on an open palm, where the task wrench stabilizes the effect of gravity along a single direction. Note that the direction of the gravity disturbance relative to the object coordinate frame is changing with the motion of the object, e.g., when rotating a book by a hand, where the task wrench stabilizes the effect of gravity along multiple directions.

Since the probability distribution model of disturbance is unknown, for each task, we can build a non-parametric statistical distribution of the disturbance from the dataset of TWS measured by demonstration. Then, to reduce the computational complexity, a smaller set of data points can be randomly sampled based on the non-parametric statistical distribution.

2.3 *Quality Measure Based on Distribution of Task Disturbance*

The quality metric k_m in Eq. 4 measures how much effort a grasp needs to cover the whole required task wrench space, which quantifies a constraint in the worst case that the robot should not drop the object. However, the worst case constraint is not always a real guarantee, given that we are modeling the task wrench space from noisy data. Thus, a different quality metric is to be developed that is insensitive to noise.

Furthermore, k_m does not take into account the distribution of a task wrench space. Without considering distribution of a task, it cannot distinguish quality between two task wrenches of the same volume but with different distributions. Consider the scenario of two different GWS for the same TWS shown in Fig. 5. It can be observed that the TWS has a higher distribution in the left area. GWS 1 and GWS 2 in Fig. 5a, b have the same volume and the same k_m. However, GWS 1 has a higher ability than GWS 1 to apply forces that frequently occur in the task, shown in Fig. 5c.

Based on the above two reasons, we propose a new task-oriented grasp quality metric that considers both TWS modeled from noisy data, as well as the distribution of TWS. When developing a grasp quality measurement for task-wrench distribution,

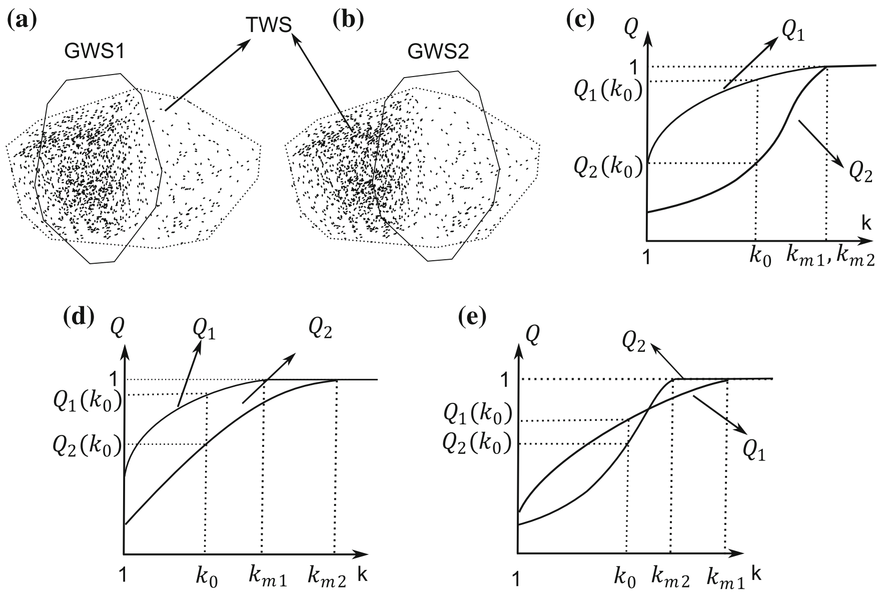


Fig. 5 Comparison of quality measure Q in different scenarios. **a**, **b** two grasp wrenches for the same task wrench space; **c** comparison of quality measures Q versus scale k between grasps in **a** and **b**. $Q_1(k_0) > Q_2(k_0)$, and $k_{m1} = k_{m2}$; Figures **d** and **e** show the other two cases of Q as a function of scale k: case in **d**, $Q_1(k_0) > Q_2(k_0)$, and $k_{m1} < k_{m2}$; case in **e** $Q_1(k_0) > Q_2(k_0)$, and $k_{m1} > k_{m2}$

we must consider the different capabilities along different directions to apply forces. It is preferred that less effort is required of a grasp to apply forces along directions where the disturbance force frequently happens, considering the efficiency of power consumption. The GWS is not necessary to cover the whole TWS, because less capability is required to apply forces for some force directions where force magnitude is large but rarely occurs. Then some noisy outliers may be excluded from the GWS. Intuitively, the grasp quality can be defined as the ratio of TWS that can be covered by the scaled GWS $W(G)$, given a scale k. The set of task wrenches that is in the scaled GWS is represented as:

$$W = \{w(t) | w(t) \in TWS \cap w(t) \in k \cdot W(G)\} \tag{7}$$

The grasp quality can be represented as:

$$Q(G) = \frac{|W|}{|TWS|} \tag{8}$$

where $|W|$ is the size of the task wrenches covered by the scaled GWS, and $|TWS|$ is the size of total task wrenches; $0 \leq Q(G) \leq 1$. The larger $Q(G)$ is, the more disturbance wrenches can be resisted by the grasp G. Therefore, the grasp planning is to find the optimal grasp that maximizes $Q(G)$.

It is noted that as k increases, Q is not linearly increasing with k, and the increasing rate of Q is not the same for different grasps (Fig. 5c–e). Therefore, the choice of k affects the result of the optimal grasp. Figure 5c compares quality Q_1 and Q_2 of the two grasps G1 and G2 shown in Fig. 5a, b as a function of k. It can be seen that Q_1 increases faster at the beginning. As k becomes larger, the increasing of Q_1 is slowed down. For all $k < k_m$, $Q_1 > Q_2$, when $k \geq k_m$, $Q_1 = Q_2 = 1$. It is also possible that different Q can intersect at some $k < k_m$, as illustrated in Fig. 5e. Also, if choosing a very large value of k, Q of different G is equal to 1. Therefore, it is important to choose a reasonable k that results in a desired Q.

Scale k stands for the amount of force the robotic hand is expected to apply. We suggested a scale k_0 by considering both the capability of the robotic hand, as well as task requirement. Suppose a unit vector $\hat{w}$ stands for a fixed direction for the disturbance wrench $w(t)$. Let $a(t) = \|w(t)\|$, the magnitude of $w(t)$, so that the disturbance wrench can be written as $w(t) = a(t)\widehat{w(t)}$. For a given task wrench set, k_0 is determined by the smaller value between the maximum magnitude $a(t)$ of all wrenches in the task, and the maximum forces that can be applied by the robotic hand—typically the capability ω_{max} of robot motors, written as:

$$k_0 = min(max(a(t)), \omega_{max}) \tag{9}$$

for all $t = 1, \ldots, T$, where T is the number of data samples. In this paper, we used a Barrett hand for the experiment. The maximum finger force of the Barrett hand is $20N$, so we set $\omega_{max} = 20$ in order to bound k_0. k_0 can also be set to other empirical value, e.g., the amount of force that humans usually apply in a manipulation.

2.4 Incorporation of Thumb Placement Constraint into Grasp Planning

Since a number of anthropomorphic hands have a high number of degrees of freedom (DOF) in order to be as dexterous as human hand, introducing complexity to the search in the optimization, much work has focused on providing constraints to the search space in order to reduce the computational complexity of the search in high dimensional robotic hand configuration space, for example, imposing appropriate contact points on the object (e.g., [20–24]). The constraint on contact points, however, is assumed to be independent of physical constraints of a given hand. It raises the problem of solving the inverse kinematics that satisfies the constraints imposed by contact points [25]. In this paper, therefore, to avoid the problem given rise by the constraints of contact points, the candidate grasp is computed from a set of given thumb placement on the object surface, as well as the direction thumb should point to. Thumb positions offer a general reference of the body part to be gripped; thumb

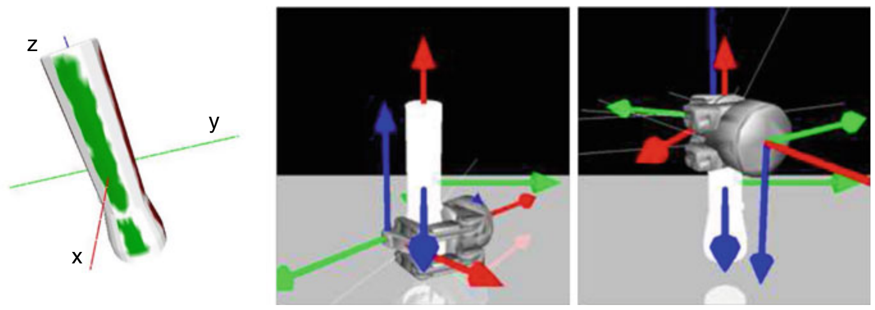


Fig. 6 Illustration of searching procedure constrained by the thumb place and direction. The *colored* area in the *first figure* is the area where the thumb is allowed to be placed. Thumb placement in *red*-colored area can only be pointed to axis x, while thumb placement in *green*-colored area can only be pointed to axis y

direction provides a constraint on wrist positions and orientations. The constraint of thumb placement can be labeled manually on the object, or generated automatically from examples.

The upper-left of Fig. 6 shows an example of labeled area. The thumb can be placed only on the colored area, with different colors specifying different thumb directions. Thumb placement in the red-colored area can be pointed only to axis x, while thumb placement in green-colored area can be pointed only to axis y. Thumb pose together provide partial constraints to wrist positions/orientations; hence, they reduce the search space during the optimization procedure. Moreover, since the thumb position of the robot is directly translated from the thumb position of the human demonstrator, no mapping between the two very different kinematic systems is required, which avoids the complicated correspondence problem. The user can also specify a grasp type, such as power grasp and precision grasp [29], to better satisfy the task requirement. Figure 6 shows snapshots of a searching procedure of a power grasp throughout the constraint area of thumb placement.

3 Results

In the experiment, we tested our approach for several tasks with different objects. Non-expert subjects were asked to manipulate an object in the user interface via Phantom OMNI. The interaction force between the object and the environment was captured during the demonstration with a sample rate of 100 Hz. The data set of the disturbance, compensated by object gravity, was recorded. Then, from the data set, a non-parametric statistical distribution of the disturbance was built. To reduce the computational complexity, a smaller set of data points was randomly sampled based on the non-parametric statistical distribution.

A Barrett hand model and a Shadow hand model were tested during the simulation for task-oriented grasp planning. The desired grasp type and the constraint area of the thumb location and direction were input into the simulator as well, which highly reduce the search space of the robotic hand configuration. In the simulation, we set the friction coefficient μ to be 1. The friction cone is approximated by an eight-sided pyramid. For each hand configuration, the grasp wrench space can be computed from the contact points and contact normals can be obtained by the open dynamics library. Grasp quality Q was calculated based on the grasp wrench space and the distribution of disturbance. The grasp planning searches the best grasp configuration that maximizes Q.

Figures 7, 8 and 9 show three examples of object manipulation. In the first example, the user was asked to perform a writing motion with a pencil, where the pencil interacts with the environment at the tip. The chosen grasp should be excellent for balancing the pressure and friction at the tip. As shown in Fig. 7a–c the distribution of task wrenches, task wrenches are biased to the positive directions of Fy and Fz, other than the full space of the friction cone. The resulting grasp is, therefore, close to the tip. For the hand configuration shown in Fig. 7d, $Q = 0.8459$ at $k = 2.6$, meaning it covers 84.59 % of task wrenches, which is much larger than that of Fig. 7e where $Q = 0.1968$ at the same k, because it is better to apply force along the Fy and Fz

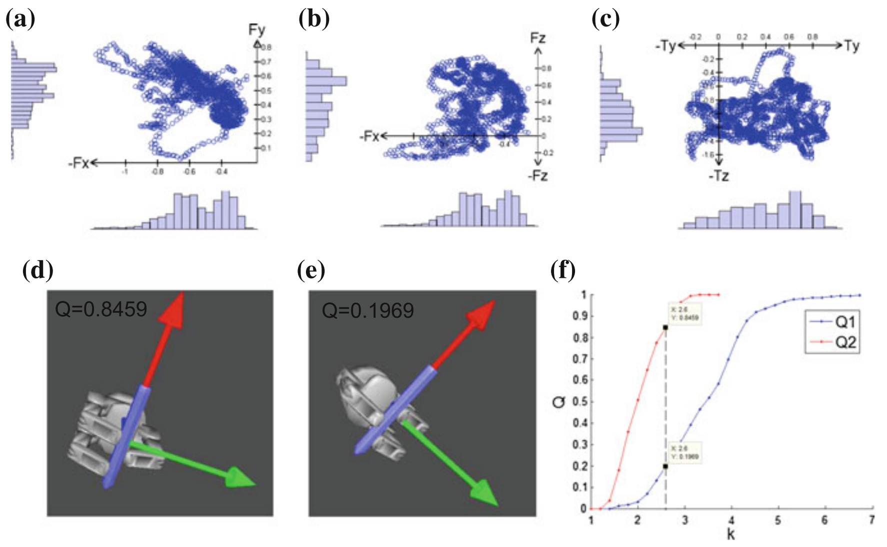


Fig. 7 Planning results for a writing task with a pencil. The center of mass is set to be the origin of the coordinate frame, where axes x, y and z are marked by *red*, *green* and *blue* colors (shown in Fig. **d**). **a**–**c** Distribution of task wrench projected to Fx-Fy, Fx-Fz, Ty-Tz subspace, respectively, where the task wrench is distributed mainly along −Fx, Fy and Fz directions; torque Tz is small so it is not reported here. **d**–**e** Two different hand configurations; **f** grasp quality Q versus scale k for the two hand configurations (Q1 and Q2 are quality measures for hand configuration in **d** and **e**)

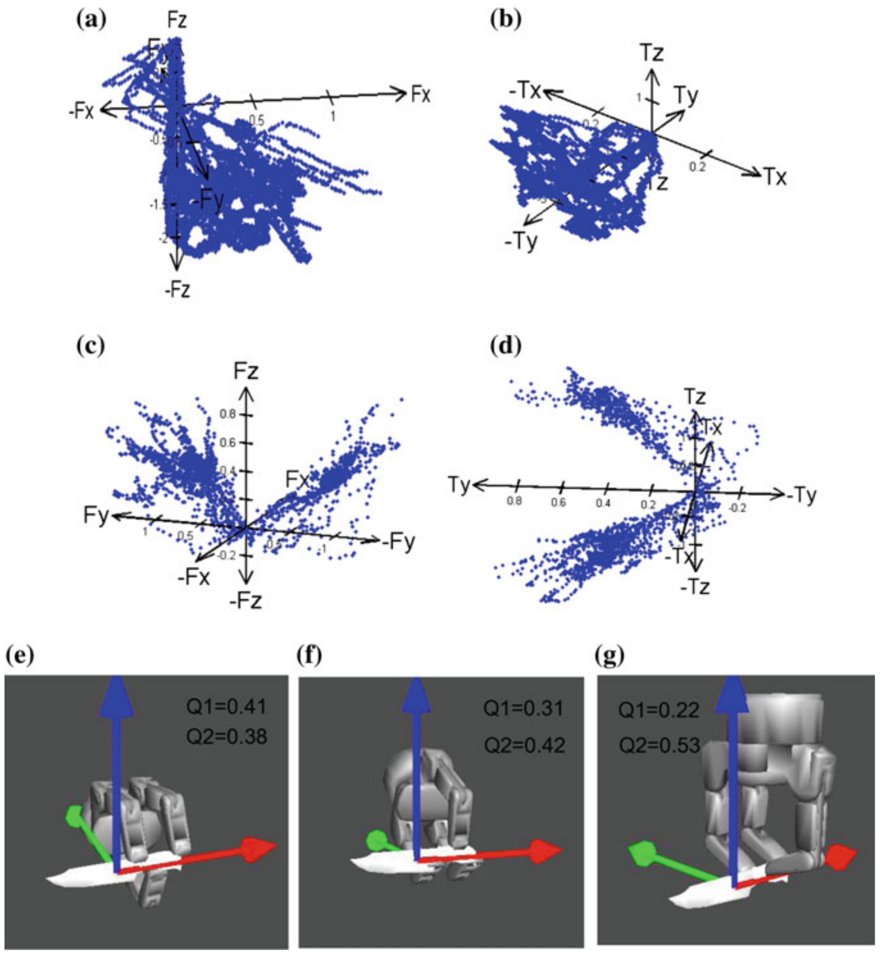


Fig. 8 Planning results for a cutting task and a butter spreading task with a knife. **a**–**b** Cutting task distribution of task wrenches projected to Fx-Fy-Fz and Tx-Ty-Tz subspaces respectively, where the task wrenches are distributed mainly in −Fz and Fx; **c**–**d** the corresponding task wrench distribution for butter spreading task, where the task wrenches are distributed primarily in +Fy, −Fy, +Fz, +Tz, −Tz; **e**–**g** three different hand configurations. Q1 is quality measure for the first task, and Q2 is the quality measure for the second task. Scale k is set to be 8.04 and 3.25 of the two tasks for a precision grasp planning

directions than that in Fig. 7e. The quality measures $Q1$ and $Q2$ changing with scale k for the two grasps are compared in Fig. 7f.

In the second experiment, grasps for two tasks were compared for a knife. The user was asked to perform two tasks: a cutting motion along one direction (+x marked by red color in Fig. 8); and a butter spreading motion using both sides of the blade. The disturbance distributions for the two tasks are reported in Fig. 8a–d.

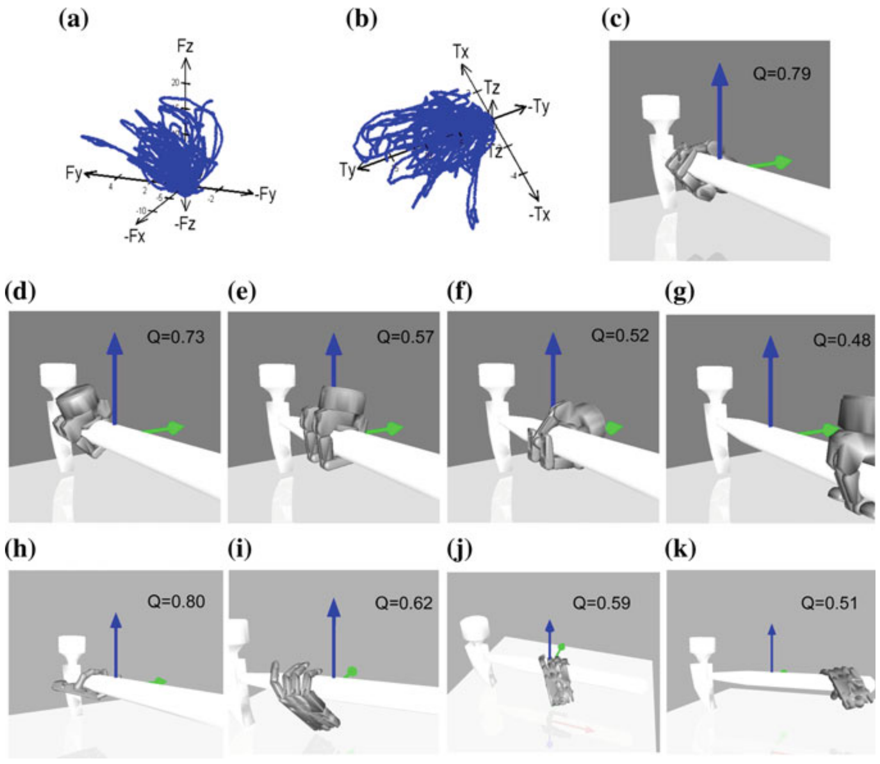


Fig. 9 Planning results for a hammer, where a power grasp is searched because a large power is needed. **a**, **b** Distribution of task wrenches projected to Fx-Fy-Fz and Tx-Ty-Tz subspace, respectively, where the task wrenches are distributed mainly in Fz and Ty; **c**–**g** five different hand configurations of the Barrett hand model; **h**–**k** four different hand configurations of the Barrett hand model. Scale k is set to be 20

As shown the cutting task in Fig. 8a, a grasp should be able to generate pressure along $-$z direction and friction mainly along $+$x direction to the blade. Torque generated along with the force is shown in Fig. 8b. While for the butter spreading task shown in Fig. 8c, d, the task wrenches cover partial area of two opposite friction cone, i.e., the grasp should be able to apply pressure along both $+$y and $-$y, and friction along $+$z. The thumb placement is constraint to the handle. Figure 8e–g contains evaluations of three grasps for the two tasks respectively ($Q1$ for cutting task and $Q2$ butter spreading task). For cutting task, where scale k is set to be 8.04, larger than $k = 3.25$ for butter spreading task. It can be seen that for cutting task, the hand configuration in Fig. 8e is better to apply force in $-$Fz, along with $-$Ty. The hand configuration in Fig. 8g has the worst quality measure of the three due to its deficient ability to apply force along z directions; Whereas for the butter spreading task, hand configuration in Fig. 8g is the best, and Fig. 8e is the worst.

In the third task, the user was asked to strike a plane with a hammer, and the grasp planning was performed to compare results of the Barrett hand model and the Shadow hand model. It can be imagined that the chosen grasp should be excellent for balancing the large pressure on the head of the hammer. As shown in Fig. 9a, b, the distribution covers almost the whole space of the friction cone, whose axis is along +z direction, and the pressure between the hammer and the environment along +z direction is as large as $20N$. The designated grasp type during grasp planning is a power grasp in order to perform powerful manipulation; the scale k of grasp wrench space is set to be 20 for the computation of quality measure. Figure 9 show the results of searching through the feasible area of thumb placement for the Barrett hand model (Fig. 9c–g), and for the Shadow hand model (Fig. 9h–k). It can be seen that the grasp closer to the head is better to counterbalance the forces that occur at the head. Note that the result of a hammer grasp is different from the intuitive grasping style of humans, who prefer to hold the handle on the other side away from the head, because humans desire to reach a large swing motion with a relatively small arm motion but to generate a large impact force. The grasp optimization considers only the ability to apply force other than the arm and wrist motions. It can be observed from the figure that similar results were obtained for the two hand models, because task distribution and thumb constraint are independent of hand mechanical structures.

Concluded from the experiments, the resulting grasp with a higher grasp quality criterion tends to be more efficient to apply frequently-occurring force, using the same magnitude of resultant force as the low quality grasp, thus improving the efficiency of power consumption.

4 Conclusion

For task-oriented grasp planning, manipulation tasks are known to be difficult to model. In this paper, a manipulation task was modeled by building non-parametric statistical distribution of disturbance from demonstration data. Instead of an evenly-distributed task wrench space, it is possible that disturbance wrenches in some directions occur more frequently than the other areas, even if they may be smaller than wrenches that occur less frequently. In favor of grasps that are able to apply frequently-occurring forces, this paper proposes a task-oriented grasp quality criterion based on the distribution of the task disturbance by computing the ratio of disturbance a grasp covers.

To reduce the computational complexity of the search in high-dimensional robotic hand configuration space, as well as to avoid a correspondence problem, the candidate grasp is computed from a set of given thumb placement and thumb direction. The experiment has been validated in simulation with a Barrett hand and a Shadow hand. Both the task model and the demonstration are independent of hand models, so they can be used for other robotic hands.

The hammer example in simulation implies that the resulting robotic grasps may be different from intuitive grasps of the humans, who consider a combination of hand

and arm motion as well as force required by a task. Therefore, including arm and hand motion factors in a grasp planning can be a direction of future work.

Another potential improvement is to measure task wrenches on the real object. Then demonstration can be performed on real objects rather than in simulation, so that the user can have more straightforward haptic feeling from the environment. In addition, the TWS can also be updated during the robot execution, which iteratively improves the grasp planning.

Although the current evaluation was conducted in simulation, where a simplified hard contact friction model was defined, the proposed task-oriented grasp quality metric can be extended to other friction models. In the future work, further evaluations will be carried out on real objects and robot platforms.

References

1. Lin, Y., Ren, S., Clevenger, M., Sun, Y.: Learning grasping force from demonstration. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1526–1531. IEEE (2012)
2. Kirkpatrick, D., Mishra, B., Yap, C.K.: Quantitative Steinitz's theorems with applications to multifingered grasping. Discrete Comput. Geom. **7**(1), 295–318 (1992)
3. Ferrari, C., Canny, J.: Planning optimal grasps. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 3, pp. 2290–2295 (1992)
4. Pollard, N.S.: Parallel methods for synthesizing whole-hand grasps from generalized prototypes. Ph.D Dissertation (1994)
5. Miller, A.T., Knoop, S., Christensen, H.I., Allen, P.K.: Automatic grasp planning using shape primitives. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 2, pp. 1824–1829 (2003)
6. Hsiao, K., Ciocarlie, M., Brook, P., Garage, W.: Bayesian grasp planning. Integrating Perception and Manipulation, In: Proceedings of the ICRA Workshop on Mobile Manipulation (2011)
7. Li, Z., Sastry, S.S.: Task-oriented optimal grasping by multifingered robot hands. IEEE J. Robot. Autom. **4**(1), 32–44 (1988)
8. Han, L., Trinkle, J.C., Li, Z.X.: Grasp analysis as linear matrix inequality problems. IEEE Trans. Robot. Autom. **16**(6), 663–674 (2000)
9. Borst, C., Fischer, M., Hirzinger, G.: Grasp planning: how to choose a suitable task wrench space. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 1, pp. 319–325 (2004)
10. Haschke, R., Steil, J.J, Steuwer, I., Ritter, H.: Task-oriented quality measures for dextrous grasping. In: Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, pp. 689–694, June 2005
11. Sahbani, A., El-Khoury, S., Bidaud, P.: An overview of 3d object grasp synthesis algorithms. Robot. Autonom. Syst. **60**(3), 326–336 (2012)
12. Ren, S., Sun, Y.: Human-object-object-interaction affordance. In: Proceedings of the IEEE Workshop in Robot, pp. 1–6. IEEE (2013)
13. Sun, Y., Ren, S., Lin, Y.: Human-object-object-interaction affordance. In: Proceedings of the Robotics and Autonomous System, pp. 1–10 (in press)
14. Lin, Y., Sun, Y.: Grasp mapping using locality preserving projections and knn regression. In: Proceedings of the International Conference on Robotics and Automation, vol. 3, May 2013
15. Billard, A., Calinon, S., Dillmann, R., Schaal, S.: Robot programming by demonstration. Handbook of Robotics. MIT Press, Cambridge (2008)

16. Romero, J., Kjellstrm, H., Kragic, D.: Human-to-robot mapping of grasps. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, WS on Grasp and Task Learning by Imitation (2008)
17. Hueser, M., Zhang, J.: Visual and contact-free imitation learning of demonstrated grasping skills with adaptive environment modelling. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, WS on Grasp and Task Learning by Imitation (2008)
18. Aleotti, J., Caselli, S.: Interactive teaching of task-oriented robot grasps. Robot. Autonom. Syst. **58**(5), 539–550 (2010)
19. Pollard, N.S.: Closure and quality equivalence for efficient synthesis of grasps from examples. Int. J. Robot. Res. **23**(6), 595–613 (2004)
20. Nguyen, V.D.: Constructing force-closure grasps. Int. J. Robot. Res. **7**(3), 3–16 (1988)
21. Ponce, J., Faverjon, B.: On computing three-finger force-closure grasps of polygonal objects. IEEE Trans. Robot. Autom. **11**(6), 868–881 (1995)
22. Zhu, X., Wang, J.: Synthesis of force-closure grasps on 3-d objects based on the q distance. IEEE Trans. Robot. Autom. **19**(4), 669–679 (2003)
23. Liu, J., Xu, G., Wang, X., Li, Z.: On quality functions for grasp synthesis, fixture planning, and coordinated manipulation. IEEE Trans. Autom. Sci. Eng. **1**(2), 146–162 (2004)
24. Roa, M.A., Suárez, R.: Finding locally optimum force-closure grasps. Robot. Comput.-Integr. Manuf. **25**(3), 536–544 (2009)
25. Rosales, C., Ros, L., Porta, J.M., Suárez, R.: Synthesizing grasp configurations with specified contact regions. Int. J. Robot. Res. **30**(4), 431–443 (2011)
26. Prattichizzo, D., Trinkle, J.C.: Grasping. Springer Handbook of Robotics, pp. 671–700. Springer, Berlin (2008)
27. Conti, F., Morris, D., Barbagli, F., Sewell, C.: Chai 3d. http://www.chai3d.org (2006)
28. Barber, C.B., Huhdanpaa, H.: Qhull. The geometry center, University of Minnesota. http://www.geom.umn.edu/software/qhull (1995)
29. Dai, W., Sun, Y., Qian, X.: Functional analysis of grasping motion. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (2013)

Towards Planning in Generalized Belief Space

Vadim Indelman, Luca Carlone and Frank Dellaert

Abstract We investigate the problem of planning under uncertainty, which is of interest in several robotic applications, ranging from autonomous navigation to manipulation. Recent effort from the research community has been devoted to design planning approaches working in a continuous domain, relaxing the assumption that the controls belong to a finite set. In this case robot policy is computed from the current robot belief (*planning in belief space*), while the environment in which the robot moves is usually assumed to be known or partially known. We contribute to this branch of the literature by relaxing the assumption of known environment; for this purpose we introduce the concept of *generalized belief space* (GBS), in which the robot maintains a joint belief over its state and the state of the environment. We use GBS within a Model Predictive Control (MPC) scheme; our formulation is valid for general cost functions and incorporates a dual-layer optimization: the outer layer computes the best control action, while the inner layer computes the generalized belief given the action. The resulting approach does not require prior knowledge of the environment and does not assume maximum likelihood observations. We also present an application to a specific family of cost functions and we elucidate on the theoretical derivation with numerical examples.

1 Introduction

Planning is an important component of robot navigation and manipulation, and it is crucial in application endeavours in which the robot operates in full or partial autonomy, e.g., multi-robot exploration, autonomous surveillance, and robotic surgery. The planning problem consists in establishing a map between the state space and the

V. Indelman (✉) · L. Carlone · F. Dellaert
College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA
e-mail: indelman@cc.gatech.edu

L. Carlone
e-mail: luca.carlone@gatech.edu

F. Dellaert
e-mail: frank@cc.gatech.edu

control space, such that the robot can autonomously determine a suitable action (e.g., a motion command), depending on its current state (e.g., current robot pose). The complexity of the problem stems from the fact that robot dynamics are stochastic; in most practical applications, the state of the robot is not directly observable, and can only be inferred from observations. Therefore, the robot maintains a probability distribution, the *belief*, over possible states, and computes a control policy using the current belief. The corresponding problem falls in the general framework of *Partially Observable Markov Decision Process* (POMDP).

The literature on planning under uncertainty can be sectioned in three main categories: *simulation-based approaches*, *infinite-horizon strategies*, and *receding horizon strategies*. Simulation-based approaches generate a few potential plans and select the best policy according to a given metric (e.g., *information gain*). They are referred to as *simulation-based* approaches, since they simulate the evolution of the belief for each potential plan, in order to quantify its quality. Examples of this approach are the work of Stachniss et al. [21, 22], Blanco et al. [3], and Du et al. [6], in which particle filters are used as inference engine. Martinez-Cantin et al. [15, 16] and Bryson and Sukkarieh [4] investigate simulation-based approaches in conjunction with the use of EKF as inference engine. Carrillo et al. [5] provide an analysis of the uncertainty metrics used in EKF-based planning. Other examples of simulation-based approaches are [12, 23, 24] in which the belief is assumed to be a Gaussian over current and past poses of the robot. These works assume *maximum likelihood observations*: since future observations are not given at planning time, the robot assumes that it will acquire the measurements that are most likely given the simulated belief. We notice that, while all the previous examples are applied to mobile robot navigation problems (the corresponding problem is usually referred to as *active Simultaneous Localization and Mapping*), similar strategies can be found with application to manipulation and computer vision (e.g., *next best view* problem [18]).

In the second category, infinite-horizon strategies, the search space is usually discretized (e.g., robot can only move between nodes of a uniformly spaced grid) and the plan may be subject to given budget constraints. The corresponding problem is also referred to as *informative path planning*. These problems are characterized by a combinatorial complexity [7], which increases with the available budget. A greedy strategy for informative path planning is proposed by Singh et al. [20] while a branch and bound approach is proposed by Binney et al. in [2]. More recently, Hollinger et. al [7] propose more efficient algorithms, based on rapidly-exploring random tree and probabilistic roadmap. The approaches falling in these second category usually assumes that the robot moves in a known environment; a remarkable property of these techniques is that they approach optimality when increasing the runtime (which is exponential in the size of the problem). A recent example of infinite-horizon planning is the work [1], in which Bai et al. apply a Monte Carlo sampling technique to update an initial policy, assuming maximum likelihood observations.

Finally, receding horizon strategies compute a policy over the next L control actions, where L is a given horizon. Huang et al. [8] propose a model predictive control (MPC) strategy, associated with EKF-SLAM. Leung et al. [14] propose an approach in which the MPC strategy is associated with a heuristic based on global attractors.

Sim and Roy [19] propose A-optimal strategies for solving the active SLAM problem. While these approaches are based on a discretization of the state space [19], or of the space of possible controls [14], recent efforts of the research community are pushing towards the use of continuous-domain models in which controls belong to a continuous set. Continuous models appear as more natural representations for real problems, in which robot states (e.g., poses) and controls (e.g., steering angles) are not constrained to few discrete values. These approaches are usually referred to as *planning in the belief space* (BS). Platt et al. [17] assume maximum likelihood observations and apply linear quadratic regulation (LQR) to compute locally optimal policies. Kontitsis et al. [13] recently propose a sampling based approach for solving a constrained optimization problem in which the constraints correspond to state dynamics, while the objective function to optimize includes uncertainty and robot goal. A hierarchical goal regression for mobile manipulation is proposed by Kaelbling et al. in [9–11]. While this branch of the literature has already produced excellent results in real problem instances, it still relies on two basic assumptions: (i) future observations are assumed to reflect current robot belief (*maximum likelihood observations*), and (ii) the environment in which the robot moves is partially or completely known. Van den Berg et al. [25] deal with the former issue and propose a general planning strategy in which maximum likelihood assumption is relaxed: future observations are treated as random variables and the future (predicted) belief preserves the dependence on these random variables. In the present work, instead, we deal with the second issue, as we assume no prior knowledge of the environment.

Our contribution belongs to the last category, as we use a receding horizon strategy. We introduce the concept of *generalized belief space* (GBS): the robot keeps a joint belief over *both* the state of the robot and the state of the surrounding environment. This allows relaxing the assumption that the environment is known or partially known, and enables applications in completely unknown and unstructured scenarios. Planning in GBS, similarly to planning in BS, is done in a continuous domain and avoids the maximum likelihood assumption that characterize earlier works. Our planning strategy, described in Sect. 2, comprises two layers: an inner layer that performs inference in the GBS, and an outer layer that computes a locally-optimal control action. In Sect. 3, we also present an application to a specific family of cost functions and we elucidate on the theoretical derivation with a numerical example in which a robot has to reach a goal while satisfying a soft bound on the admissible position estimation uncertainty. Conclusions are drawn in Sect. 4.

2 Planning in Generalized Belief Space (GBS)

2.1 Notation and Probabilistic Formulation

Let x_i and $\mathscr{W}_i$ denote the *robot state* and the *world state* at time t_i. For instance, in mobile robots navigation, x_i may describe robot pose at time t_i and $\mathscr{W}_i$ may describe the positions of landmarks in the environment observed by the robot by time t_i. In a manipulation problem, instead, x_i may represent the pose of the end effector of the manipulator, and $\mathscr{W}_i$ may describe the pose of an object to be grasped. The world state $\mathscr{W}_i$ is time-dependent in general (e.g., to account for possible variations in the environment, or to model the fact that the robot may only have observed a subset of the environment by time t_i) and for this reason we keep the index i in $\mathscr{W}_i$. Let z_i denote the available observations at time t_i and u_i the control action applied at time t_i. We define the joint state at time t_k as $X_k \doteq \{x_0, \ldots, x_k, \mathscr{W}_k\}$, and we write the probability distribution function (pdf) over the joint state as:

$$p\left(X_k | \mathscr{Z}_k, \mathscr{U}_{k-1}\right), \tag{1}$$

where $\mathscr{Z}_k \doteq \{z_0, \ldots, z_k\}$ represent all the available observations until time t_k, and $\mathscr{U}_{k-1} \doteq \{u_0, \ldots, u_{k-1}\}$ denotes all past controls. The probabilistic motion model given the control u_i and the state robot x_i is

$$p\left(x_{i+1} | x_i, u_i\right). \tag{2}$$

We consider a general observation model that involves at time t_i a subset of joint states $X_i^o \subseteq X_i$:

$$p\left(z_i | X_i^o\right). \tag{3}$$

The basic observation model, commonly used in motion planning, e.g., [25], involves only the current robot state x_i at each time t_i and is a particular case of the above general model (3).

The joint pdf (1) at the current time t_k can be written according to the motion and observation models (2) and (3) as

$$p\left(X_k | \mathscr{Z}_k, \mathscr{U}_{k-1}\right) = priors \cdot \prod_{i=1}^{k} p\left(x_i | x_{i-1}, u_{i-1}\right) p\left(z_i | X_i^o\right). \tag{4}$$

The *priors* term includes $p\left(x_0\right)$ and any other available prior information.

2.2 Approach Overview

The aim of this paper is to present a general strategy that allows a robot (autonomous vehicle, UAV, etc.) to plan a suitable control strategy to accomplish a given *task*. Task accomplishment is modelled through an objective function to be optimized; for instance the objective can penalize the distance to a goal position (*path planning*), the uncertainty in the state estimate (*active sensing*), or can model the necessity to visit new areas (*exploration*). We adopt a standard *model predictive control* (MPC) strategy in which the robot has to plan an optimal sequence of controls

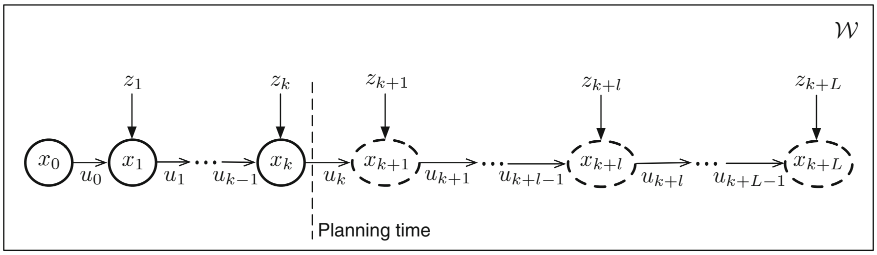


Fig. 1 Illustration of the L look-ahead steps planning problem. Past observations $\mathcal{Z}_k = \{z_0, \ldots, z_k\}$ and past controls $\mathcal{U}_{k-1} = \{u_0, \ldots, u_{k-1}\}$ are known at the planning time t_k. Future observations $z_{k+1:k+L}$ are instead unknown and treated as random variables. The objective is to compute a suitable control strategy $u_{k:k+L-1} = \{u_k, \ldots, u_{k+L-1}\}$ for L look-ahead steps. The figure only illustrates the temporal evolution of the system and we note that each observation may involve generic subset of the states (robot states and world state $\mathcal{W}$) according to the observation model (3)

$u_{k:k+L-1} = \{u_k, \ldots, u_{k+L-1}\}$ for L look-ahead steps, so that a given objective function is minimized over the time horizon (see Fig. 1). The presentation of this section is general and does not assume a specific cost function, while in Sect. 3 we discuss practical choices of the cost function.

At planning time t_k, the optimal control minimizes an objective function J_k $(u_{k:k+L-1})$ for L look-ahead steps. The objective function involves L immediate costs, one for each look-ahead step. We consider a *general* immediate cost c_{k+l} that may involve any subset of states $X^c_{k+l} \subseteq X_{k+l}$, where $l \in \{0, \ldots, L-1\}$ is the lth look-ahead step. The immediate cost c_{k+l} can be therefore written as

$$c_{k+l}\left(p\left(X^c_{k+l}|\mathcal{Z}_k, \mathcal{U}_{k-1}, z_{k+1:k+l}, u_{k:k+l-1}\right), u_{k+l}\right), \tag{5}$$

where the notation $a_{i:j} \doteq \left\{a_i, \ldots, a_j\right\}$ is used. As seen in Eq. (5), the immediate cost function c_{k+l} directly involves a distribution over the subset of states X^c_{k+l}. This distribution is conditioned on past measurements and controls $\mathcal{Z}_k, \mathcal{U}_{k-1}$ (that are known at planning time), as well as on future controls and observations $z_{k+1:k+l}, u_{k:k+l-1}$ while the actual observations $z_{k+1:k+l}$ are not given at planning time, the corresponding observation model is known (Eq. (3)) and involves additional subsets of states X^o_{k+j} with $j = [0, \ldots, l]$. Therefore, calculating the pdf $p\left(X^c_{k+l}|\mathcal{Z}_k, \mathcal{U}_{k-1}, z_{k+1:k+l}, u_{k:k+l-1}\right)$ involves an extended subset of the joint state X_{k+l}. For clarity of presentation, however, we will proceed with the *entire* joint state X_{k+l} which contains X^c_{k+l}.[1]

We thus define the *generalized belief space* (GBS) at the lth planning step as

$$gb\left(X_{k+l}\right) \doteq p\left(X_{k+l}|\mathcal{Z}_k, \mathcal{U}_{k-1}, z_{k+1:k+l}, u_{k:k+l-1}\right). \tag{6}$$

[1]In principle, for planning it is only necessary to maintain a distribution over the states X^c_{k+l} while marginalizing out the remaining states. This would avoid performing computation over a large state space, hence resulting in a computational advantage. We leave the investigation of this aspect as an avenue for future research.

The objective function $J_k\left(u_{k:k+l-1}\right)$ can now be defined as

$$J_k\left(u_{k:k+L-1}\right) \doteq \underset{z_{k+1:k+L}}{\mathbb{E}}\left\{\sum_{l=0}^{L-1} c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right) + c_L\left(gb\left(X_{k+L}\right)\right)\right\} \tag{7}$$

where the expectation is taken to account for all the possible observations during the planning lag, since these are not given at planning time and are stochastic in nature. Since the expectation is a linear operator we rewrite the objective function (7) as:

$$J_k\left(u_{k:k+L-1}\right) \doteq \sum_{l=0}^{L-1} \underset{z_{k+1:k+l}}{\mathbb{E}}\left[c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right)\right] + \underset{z_{k+1:k+L}}{\mathbb{E}}\left[c_L\left(gb\left(X_{k+L}\right)\right)\right]. \tag{8}$$

The optimal control $u^*_{k:k+L-1} \doteq \left\{u^*_k, \ldots, u^*_{k+L-1}\right\}$ is the control policy π:

$$u^*_{k:k+L-1} = \pi\left(gb\left(X_k\right)\right) = \underset{u_{k:k+L-1}}{\arg\min}\; J_k\left(u_{k:k+L-1}\right). \tag{9}$$

Calculating the optimal control policy (9) involves the optimization of the objective function $J_k\left(u_{k:k+L-1}\right)$. According to (8), the objective depends on the (known) GBS at planning time t_k, on the predicted GBS at time $t_{k+1}, \ldots, t_{k+L}$, and on the future controls $u_{k:k+L-1}$. Since in general the immediate costs $c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right)$ are non-linear functions, $\underset{z_{k+1:k+l}}{\mathbb{E}}\left[c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right)\right] \neq c_l\left(\underset{z_{k+1:k+l}}{\mathbb{E}}\left[gb\left(X_{k+l}\right)\right], u_{k+l}\right)$, and we have to preserve the dependence of the belief $gb\left(X_{k+l}\right)$ on the observations $z_{k+1:k+l}$. The latter are treated as random variables. Therefore, the belief at the lth look-ahead step depends on $u_{k:k+l-1}$ (which is our optimization variable) and $z_{k+1:k+l}$ (which is a random variable).

In order to optimize the objective function (8) we resort to an iterative optimization approach, starting from a known initial guess on the controls. The overall approach can be described as a *dual-layer inference*: the inner layer performs inference to calculate the GBS at each of the look-ahead steps, for a given $u_{k:k+L-1}$. The outer layer performs inference over the control $u_{k:k+L-1}$, minimizing the objective function (8). A schematic representation of the approach is provided in Fig. 2, while in the next sections we describe in detail each of these two inference processes, starting from the outer layer: inference over the control.

2.3 Outer Layer: Inference over the Control

Finding a locally-optimal control policy $u^*_{k:k+L-1}$ corresponds to minimizing the general objective function (8). The *outer* layer is an iterative optimization over the non-linear function $J_k\left(u_{k:k+L-1}\right)$. In each iteration of this layer we are looking for the delta vector $\Delta u_{k:k+L-1}$ that is used to update the current values of the controls:

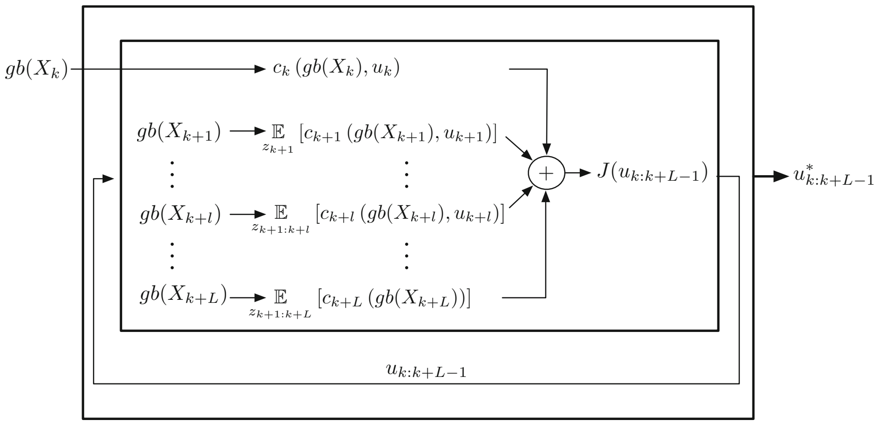

$gb(X_k)$
$c_k\,(gb(X_k), u_k)$
$gb(X_{k+1})$
$\underset{z_{k+1}}{\mathbb{E}}\,[c_{k+1}\,(gb(X_{k+1}), u_{k+1})]$
$gb(X_{k+l})$
$\underset{z_{k+1:k+l}}{\mathbb{E}}\,[c_{k+l}\,(gb(X_{k+l}), u_{k+l})]$
$gb(X_{k+L})$
$\underset{z_{k+1:k+L}}{\mathbb{E}}\,[c_{k+L}\,(gb(X_{k+L}))]$
$+$
$J(u_{k:k+L-1})$
$u^*_{k:k+L-1}$
$u_{k:k+L-1}$

Fig. 2 Illustration of the dual-layer inference planning approach. The algorithm takes as an input the GBS at the current time t_k, $gb\,(X_k)$, and produces as output a locally-optimal control $u^*_{k:k+L-1}$. The outer layer performs inference over the control $u_{k:k+L-1}$, while the inner layer evaluates the GBS for a given value of $u_{k:k+L-1}$. Note that the GBS at the lth look-ahead step is a function of controls $u_{k:k+l-1}$ as well as of the random observations $z_{k+1:k+l}$: $gb\,(X_{k+l}) = p\,(X_{k+l} | \mathscr{Z}_k, \mathscr{U}_{k-1}, z_{k+1:k+l}, u_{k:k+l-1})$

$$u^{(i+1)}_{k:k+L-1} \leftarrow u^{(i)}_{k:k+L-1} + \Delta u_{k:k+L-1}, \tag{10}$$

where i denotes the iteration number. Calculating $\Delta u_{k:k+L-1}$ involves computing the GBS of all the L look-ahead steps based on the current value of the controls $u^{(i)}_{k:k+L-1}$. This process of calculating the GBS is by itself a non-linear optimization and represents the *inner* layer inference in our approach. We describe this inference in Sect. 2.4. The GBS $gb\,(X_{k+l})$, given the current values of the controls $u_{k:k+L-1}$, is represented by the mean $\hat{X}^*_{k+l}\,(z_{k+1:k+l})$ and the information matrix I_{k+l}. The mean is a linear function in the unknown observations $z_{k+1:k+l}$. The immediate cost function, in the general case, may involve both the mean and the information matrix, and is therefore also a function of $z_{k+1:k+l}$. Taking the expectation over these random variables produces the expected cost that is only a function of $u_{k:k+L-1}$ and captures the effect of the current controls on the lth look-ahead step.

We conclude this section by noting that the control update (10) is performed in a continuous domain and can be realized using different optimization techniques (e.g., dynamic programming, gradient descent, Gauss–Newton).

2.4 Inner Layer: Inference in GBS

In this section we focus on calculating the GBS at the lth look-ahead step $gb\,(X_{k+l}) \equiv p\,(X_{k+l} | \mathscr{Z}_k, \mathscr{U}_{k-1}, z_{k+1:k+l}, u_{k:k+l-1})$, with $l \in [1, L]$. As this inference is performed as part of the higher-level optimization over the control (see Sect. 2.2), the current values for $u_{k:k+l-1}$ are given in the inner inference layer.

The GBS $gb\left(X_{k+l}\right)$ can be expressed in terms of the joint pdf at the planning time t_k and the individual motion and observation models applied since then:

$$gb\left(X_{k+l}\right)=p\left(X_k|\mathscr{Z}_k,\mathscr{U}_{k-1}\right)\prod_{j=1}^{l}p\left(x_{k+j}|x_{k+j-1},u_{k+j-1}\right)p\left(z_{k+j}|X^o_{k+j}\right). \tag{11}$$

We consider the case of motion and observation models with additive Gaussian noise:

$$x_{i+1}=f\left(x_i,u_i\right)+w_i \quad , \quad w_i\sim N\left(0,\Omega_w\right) \tag{12}$$
$$z_i=h\left(X^o_i\right)+v_i \quad , \quad v_i\sim N\left(0,\Omega_v\right), \tag{13}$$

where for notational convenience, we use $\varepsilon\sim N(\mu,\Omega)$ to denote a Gaussian random variable ε with mean μ and information matrix Ω (inverse of the covariance matrix). Then the distribution $p\left(X_{k+l}|\mathscr{Z}_k,\mathscr{U}_{k-1},z_{k+1:k+l},u_{k:k+l-1}\right)$ can also be modeled as a Gaussian and can be expressed as

$$p\left(X_{k+l}|\mathscr{Z}_k,\mathscr{U}_{k-1},z_{k+1:k+l},u_{k:k+l-1}\right)\sim N\left(\hat{X}^*_{k+l},I_{k+l}\right). \tag{14}$$

Our goal is then to calculate the mean $\hat{X}^*_{k+l}$ and the information matrix I_{k+l} describing the GBS at the lth look ahead step, bearing in mind that the observations $z_{k+1:k+l}$ are unknown at planning time t_k.

At this point, it is convenient to assume that the joint pdf at planning time t_k can be parametrized by a Gaussian distribution

$$p\left(X_k|\mathscr{Z}_k,\mathscr{U}_{k-1}\right)\sim N\left(\hat{X}_k,I_k\right), \tag{15}$$

with known $\hat{X}_k, I_k$. Taking the negative log of $p\left(X_{k+l}|\mathscr{Z}_k,\mathscr{U}_{k-1},z_{k+1:k+l},u_{k:k+l-1}\right)$ from Eq. (11) results in

$$\begin{aligned}&-\log p\left(X_{k+l}|\mathscr{Z}_k,\mathscr{U}_{k-1},z_{k+1:k+l},u_{k:k+l-1}\right)\\&=\left\|X_k-\hat{X}_k\right\|^2_{I_k}+\sum_{i=1}^{l}\left[\left\|x_{k+i}-f\left(x_{k+i-1},u_{k+i-1}\right)\right\|^2_{\Omega_w}+\left\|z_{k+i}-h\left(X^o_{k+i}\right)\right\|^2_{\Omega_v}\right],\end{aligned} \tag{16}$$

where we use the standard notation $\|y-\mu\|^2_{\Omega}=(y-\mu)^T\,\Omega\,(y-\mu)$ for the Mahalanobis norm. The maximum a posteriori (MAP) estimate of X_{k+l} is then given by

$$\hat{X}^*_{k+l}=\underset{X_{k+l}}{\arg\min}\ -\log p\left(X_{k+l}|\mathscr{Z}_k,\mathscr{U}_{k-1},z_{k+1:k+l},u_{k:k+l-1}\right). \tag{17}$$

This optimization problem lies at the core of the inner inference layer of our planning approach. In principle, solving (17) involves iterative nonlinear optimization.

A standard way to solve the minimization problem is the Gauss–Newton method, where a single iteration involves linearizing the above equation about the current estimate $\bar{X}_{l+l}$, calculating the delta vector ΔX_{k+l} and updating the estimate $\bar{X}_{l+l} \leftarrow \bar{X}_{l+l} + \Delta X_{k+l}$. This process should be repeated until convergence. While this is standard practice in information fusion, what makes it interesting in the context of planning is that the observations $z_{k+1:k+l}$ are unknown and considered instead as random variables.

In order to perform a Gauss–Newton iteration on (17), we linearize the motion and observation models in Eq. (16) about the linearization point $\bar{X}_{k+l}\left(u_{k:k+l-1}\right)$. The linearization point for the existing states at planning time is set to $\hat{X}_k$, while the future states are predicted via the motion model (12) using the current values of the controls $u_{k:k+l-1}$:

$$\bar{X}_{k+l}\left(u_{k:k+l-1}\right) \equiv \begin{pmatrix} \bar{X}_k \\ \bar{x}_{k+1} \\ \bar{x}_{k+2} \\ \vdots \\ \bar{x}_{k+l} \end{pmatrix} \doteq \begin{pmatrix} \hat{X}_k \\ f\left(\hat{x}_{k|k}, u_k\right) \\ f\left(\bar{x}_{k+1}, u_{k+1}\right) \\ \vdots \\ f\left(\bar{x}_{k+l-1}, u_{k+l-1}\right) \end{pmatrix}. \tag{18}$$

Using this linearization point, Eq. (16) turns into:

$$-\log p\left(X_{k+l} | \mathcal{Z}_k, \mathcal{U}_{k-1}, z_{k+1:k+l}, u_{k:k+l-1}\right) = \|\Delta X_k\|^2_{I_k} + \sum_{i=1}^{l}\left[\left\|\Delta x_{k+i} - F_i \Delta x_{k+i-1} - b_i^f\right\|^2_{\Omega_w} + \left\|H_i \Delta X^o_{k+i} - b_i^h\right\|^2_{\Omega_v}\right], \tag{19}$$

where the Jacobian matrices $F_i \doteq \nabla_x f$ and $H_i \doteq \nabla_x h$ are evaluated about $\bar{X}_{k+l}$ $\left(u_{k:k+l-1}\right)$. The right hand side vectors b_i^f and b_i^h are defined as

$$b_i^f \doteq f\left(\bar{x}_{k+i-1}, u_{k+i-1}\right) - \bar{x}_{k+i}, \quad b_i^h\left(z_{k+i}\right) \doteq z_{k+i} - h\left(\bar{X}^o_{k+i}\right) \tag{20}$$

Note that b_i^h is a function of the random variable z_{k+i}. Also note that under the maximum-likelihood assumption this terms would be nullified: assuming maximum likelihood measurements essentially means assuming zero *innovation*, and b_i^h is exactly the innovation for measurement z_{k+i}. We instead keep, for now, the observation z_{k+i} as a variable and we will compute the expectation over this random variable only when evaluating the objective function (8). In order to calculate the update vectors ΔX_k and $\Delta x_{k+1}, \ldots, \Delta x_{k+l}$, it is convenient to write Eq. (19) in a matrix formulation, which can be compactly represented as:

$$\left\|\mathscr{A}_{k+l}\left(u_{k:k+l-1}\right) \Delta X_{k+l} - \breve{b}_{k+l}\left(u_{k:k+l-1}, z_{k+1:k+l}\right)\right\|^2, \tag{21}$$

where we used the relation $\|a\|^2_{\Omega} \equiv \left\|\Omega^{1/2}a\right\|^2$, and $\mathscr{A}_{k+l}$ and $\breve{b}_{k+l}$ are of the following form:

$$\mathscr{A}_{k+l} \doteq \begin{bmatrix} \begin{bmatrix} I_k^{1/2} & 0 \end{bmatrix} \\ \mathscr{F}_{k+l} \\ \mathscr{H}_{k+l} \end{bmatrix} \quad , \quad \breve{b}_{k+l} = \begin{pmatrix} 0 \\ \Omega_w^{1/2}\breve{b}^f_{k+l} \\ \Omega_v^{1/2}\breve{b}^h_{k+l} \end{pmatrix}. \tag{22}$$

Here, $\mathscr{F}_{k+l}$ and $\mathscr{H}_{k+l}$ include all the Jacobian-related entries $\Omega_w^{1/2}F_i$ and $\Omega_v^{1/2}H_i$ (for all $i \in [1, l]$), respectively, and zeros in appropriate locations. Likewise, the vectors $\breve{b}^f_{k+l}$ and $\breve{b}^h_{k+l}$ respectively collect the terms b^f_i and $b^h_i\,(z_{k+i})$. The term $\begin{bmatrix} I_k^{1/2} & 0 \end{bmatrix}$ includes a matrix of zeros of appropriate size for padding.

The information matrix I_{k+l} can now be calculated from Eq. (21) as

$$I_{k+l}\,(u_{k:k+l-1}) \doteq \mathscr{A}^T_{k+l}\mathscr{A}_{k+l}, \tag{23}$$

and the update vector ΔX_{k+l}, that minimizes (21), is given by

$$\Delta X_{k+l}\,(u_{k:k+l-1}, z_{k+1:k+l}) \doteq \left(\mathscr{A}^T_{k+l}\mathscr{A}_{k+l}\right)^{-1} \mathscr{A}^T_{k+l}\breve{b}_{k+l} = I^{-1}_{k+l}\mathscr{A}^T_{k+l}\breve{b}_{k+l}. \tag{24}$$

Noting that the right hand side vectors b^f_i are zero for the linearization point (18), the only non-zero entries in the vector $\breve{b}_{k+l}$ are the right hand side vectors b^h_i, which depend *linearly* on $z_{k+1}, \ldots, z_{k+L}$. Using the definitions (22), Eq. (24) can hence be written as $\Delta X_{k+l}\,(u_{k:k+l-1}, z_{k+1:k+l}) = I^{-1}_{k+l}\mathscr{H}^T_{k+l}\breve{\Omega}_v\breve{b}^h_{k+l}$, where $\breve{\Omega}_v$ is an appropriate block diagonal matrix with Ω_v elements. The updated estimate is then calculated as

$$\hat{X}_{k+l}\,(u_{k:k+l-1}, z_{k+1:k+l}) = \bar{X}_{k+l} + \Delta X_{k+l} = \bar{X}_{k+l} + I^{-1}_{k+l}\mathscr{H}^T_{k+l}\breve{\Omega}_v\breve{b}^h_{k+l}. \tag{25}$$

The estimate $\hat{X}_{k+l}\,(u_{k:k+l-1}, z_{k+1:k+l})$ is the outcome of a single iteration of the nonlinear optimization (17). We remark that for a single iteration, the information matrix $I_{k+l}\,(u_{k:k+l-1})$ does not depend on $z_{k+1:k+l}$ and the mean depends on $z_{k+1:k+l}$ *linearly*. This fact greatly helps when taking the expectation over $z_{k+1:k+l}$ of the immediate cost function (8). Considering more iterations would better capture the dependence of the estimate on the measurements; however, more iterations would make $\hat{X}_{k+l}\,(u_{k:k+l-1}, z_{k+1:k+l})$ a nonlinear function of $z_{k+1}, \ldots, z_{k+L}$. We currently assume a single iteration sufficiently captures the effect of the measurements for a certain control action on the GBS. Therefore,

$$\hat{X}^*_{k+l}\,(u_{k:k+l-1}, z_{k+1:k+l}) = \hat{X}_{k+l}\,(u_{k:k+l-1}, z_{k+1:k+l})\,.$$

The difference with the maximum-likelihood observations assumption is evident from Eq. (25): in that case only the first term would appear in the above equation. Lastly, we notice that the same linearization point (18) will be used also for the next look-ahead step $(l + 1)$, in which only a few additional terms will be added to Eq. (19); this allows large re-use of calculations. Moreover we notice the matrices appearing in Eqs. (23) and (24) are sparse. We leave the investigation of these computational aspects to future research.

3 Application to a Specific Family of Cost Functions

3.1 Choice of the Cost Functions

The exposition of the approach thus far has been given for general immediate cost functions. To demonstrate the effectiveness of our approach we will now focus on a specific family of cost functions. We define:

$$c_l\left(gb\left(X_{k+l}\right), u_{k+l}\right) \doteq \left\| E^G_{k+l}\hat{X}^*_{k+l} - X^G \right\|_{M_x} + tr\left(M_\Sigma I^{-1}_{k+l} M^T_\Sigma\right) + \left\|\zeta\left(u_{k+l}\right)\right\|_{M_u} \tag{26}$$

$$c_L\left(gb\left(X_{k+L}\right)\right) \doteq \left\| E^G_{k+L}\hat{X}^*_{k+L} - X^G \right\|_{M_x} + tr\left(M_\Sigma I^{-1}_{k+L} M^T_\Sigma\right). \tag{27}$$

Here M_Σ, M_u and M_x are given weight matrices, and $\zeta\left(u\right)$ is some known function that, depending on the application, quantifies the usage of control u. X^G is the goal for some subset of states (e.g., the last pose), and E^G_{k+l} is a selection matrix, such that the matrix $E^G_{k+l}\hat{X}^*_{k+l}$ contains a subset of states for which we want to impose a goal. Similarly, the matrix M_Σ may also be used to choose the covariance of some subset of states from the joint covariance I^{-1}_{k+l} (e.g., consider only uncertainty of the landmarks in the environment).

Plugging Eqs. (26) and (27) into Eq. (8), taking the expectation, and rearranging the terms, we get

$$J_k\left(u_{k:k+L-1}\right) \doteq \sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|_{M_u} + \sum_{l=0}^{L} tr\left(M_\Sigma I^{-1}_{k+l} M^T_\Sigma\right) + \sum_{l=0}^{L} \mathop{\mathbb{E}}_{z_{k+1:k+l}}\left[\left\| E^G_{k+l}\hat{X}^*_{k+l} - X^G \right\|_{M_x}\right].$$

We recall that the posterior $\hat{X}^*_{k+l}$ at a generic step l is a function of the observations $z_{k+1:k+l}$, which are random variables. In order to obtain the final expression of the objective function we have to compute the expectation in the last summand in the above equation. We omit the complete derivation for space reasons; in this article we report the final result after taking the expectation:

$$\begin{aligned} J_k\left(u_{k:k+L-1}\right) \doteq & \sum_{l=0}^{L-1} \left\|\zeta\left(u_{k+l}\right)\right\|_{M_u} + \sum_{l=0}^{L} tr\left(M_\Sigma I^{-1}_{k+l} M^T_\Sigma\right) \\ & + \underbrace{\sum_{l=0}^{L}\left[\left\| E^G_{k+l}\bar{X}_{k+l} - X^G \right\|_{M_x} + tr\left(Q_{k+l}\left(\mathscr{H}^T_{k+l}\bar{I}^{-1}_{k+l}\mathscr{H}^T_{k+l} + \breve{\Omega}^{-1}_v\right)\right)\right]}_{(a)}, \end{aligned} \tag{28}$$

where $\bar{X}_{k+l}$ is the nominal belief (18), $\bar{I}_{k+l}$ is the information matrix of the nominal belief $\bar{X}_{k+l}$ and $Q_{k+l} = \left(E^G_{k+l} I^{-1}_{k+l} \mathscr{H}^T_{k+l} \breve{\Omega}_v\right)^T M_x \left(E^G_{k+l} I^{-1}_{k+l} \mathscr{H}^T_{k+l} \breve{\Omega}_v\right)$. The first sum contains the terms penalizing the control actions; the second sum contains terms penalizing uncertainty (captured by the information matrix I_{k+l} of the belief); the last term (a) was derived from $\underset{z_{k+1:k+l}}{\mathbb{E}}\left[\left\| E^G_{k+l} \hat{X}^*_{k+l} - X^G \right\|_{M_x}\right]$ and represents the expected incentive in reaching the goal. We notice that the term (a), thus being connected to goal achievement, also contains a term, $tr\left(Q_{k+l}\left(\mathscr{H}^T_{k+l} \bar{I}^{-1}_{k+l} \mathscr{H}^T_{k+l} + \breve{\Omega}^{-1}_v\right)\right)$, that depends on the uncertainty. This term appears because we did not assume maximum likelihood observations, therefore the random nature of the estimate $\hat{X}^*_{k+l}$ (as a function of the random variables $z_{k+1:k+l}$) is preserved.

3.2 Choice of the Weight Matrices

In this section we discuss how to properly choose the weight matrices M_u, M_Σ, and M_x. Most related work assume these matrices are given, while in practice their choice can be scenario dependent and can largely influence the control policy. The matrix M_u, appearing in the summand (a) of (28) has a very intuitive function: a larger M_u induces conservative policies that will penalize large controls (or large variations in the controls, depending on the definition of the function $\zeta(u)$). Consequently, M_u can be tuned to have smoother trajectories or when it is important to keep the controls small (e.g., in presence of strict fuel/power constraints).

The choice of the matrices M_x and M_Σ is instead less intuitive. A balance between these two matrices is crucial for letting the robot satisfy the concurrent tasks of reaching a goal and minimizing the estimation uncertainty. In this section, we propose a grounded way to select these matrices. For simplicity we first assume that the two matrices can be written as $M_x = \alpha_x \bar{M}_x$ and $M_\Sigma = \sqrt{\alpha_\Sigma}\bar{M}_\Sigma$ for some constant and known matrices $\bar{M}_x$ and $\bar{M}_\Sigma$. For instance, $\bar{M}_x$ can simply be a selection matrix that "extract" the subset of states for which we want to set a goal; similarly $\bar{M}_\Sigma$ can be a selection matrix extracting from I^{-1}_{k+l} the marginal covariance that we want to minimize at planning time. Under these assumption the objective function becomes:

$$
\begin{aligned}
J_k\left(u_{k:k+L-1}\right) = \sum_{l=0}^{L-1} \|\zeta\left(u_{k+l}\right)\|_{M_u} + \alpha_\Sigma \sum_{l=0}^{L} tr\left(\bar{M}_\Sigma I^{-1}_{k+l} \bar{M}^T_\Sigma\right) \\
+ \alpha_x \left[\sum_{l=0}^{L}\left[\left\| E^G_{k+l}\bar{X}_{k+l} - X^G \right\|_{\bar{M}_x}\right.\right. \\
\left.\left. + tr\left(\bar{Q}_{k+l}\left(\mathscr{H}^T_{k+l} \bar{I}^{-1}_{k+l} \mathscr{H}^T_{k+l} + \breve{\Omega}^{-1}_v\right)\right)\right]\right].
\end{aligned}
$$

with $\bar{Q}_{k+l} = \left(E^G_{k+l} I^{-1}_{k+l} \mathscr{H}^T_{k+l} \breve{\Omega}_v\right)^T \bar{M}_x \left(E^G_{k+l} I^{-1}_{k+l} \mathscr{H}^T_{k+l} \breve{\Omega}_v\right)$. The scalar α_x controls the "attraction" towards the goal; similarly α_Σ represents the "importance" of minimizing the uncertainty of the selected states. In order to determine suitable α_x and α_Σ we notice that we can divide the cost function by a constant term, without altering the solution of the optimization problem. Therefore, we divide the cost by $(\alpha_x + \alpha_\Sigma)$, obtaining:

$$\begin{aligned} J_k\left(u_{k:k+L-1}\right) = \sum_{l=0}^{L-1} \|\zeta\left(u_{k+l}\right)\|_{\bar{M}_u} + (\alpha) \sum_{l=0}^{L} tr\left(\bar{M}_\Sigma I^{-1}_{k+l} \bar{M}^T_\Sigma\right) \\ + (1-\alpha) \left[\sum_{l=0}^{L} \left[\left\| E^G_{k+l} \bar{X}_{k+l} - X^G \right\|_{\bar{M}_x} \right.\right. \\ \left.\left. + tr\left(\bar{Q}_{k+l} \left(\mathscr{H}^T_{k+l} \bar{I}^{-1}_{k+l} \mathscr{H}^T_{k+l} + \breve{\Omega}^{-1}_v \right)\right)\right]\right]. \end{aligned} \quad (29)$$

where $\alpha = \frac{\alpha_\Sigma}{\alpha_x + \alpha_\Sigma}$ and $\bar{M}_u = \frac{1}{\alpha_x + \alpha_\Sigma} M_u$. The previous expression highlights the trade-off between the last two terms in the cost function (uncertainty reduction VS goal achievement). In order to set α we assume the robot is given an upper bound β on $tr\left(\bar{M}_x I^{-1}_{k+L} \bar{M}^T_x\right)$ (which represents the uncertainty of a selected set of states at the end of the horizon) and we want to compute α so that this upper bound is satisfied. Therefore, we set $\alpha = \frac{tr\left(\bar{M}_x I^{-1}_{k+L} \bar{M}^T_x\right)}{\beta}$ such that for values of $tr\left(\bar{M}_x I^{-1}_{k+L} \bar{M}^T_x\right)$ close to the bound β, the ratio α is closer to 1 and the robot will give more importance to the second summand in (29) (i.e., it will prefer minimizing the uncertainty). Conversely, when the uncertainty is far from the upper bound, the term $(1-\alpha)$ will be large and the robot will prefer reaching the goal. We notice that the quantity $tr\left(\bar{M}_x I^{-1}_{k+L} \bar{M}^T_x\right)$ can eventually become larger than β, as we are not imposing a hard constraint on this term, and for this reason we set $\alpha = \min\left(\frac{tr\left(\bar{M}_x I^{-1}_{k+L} \bar{M}^T_x\right)}{\beta}, 1\right)$.

3.3 Simulation Results

We demonstrate our approach in simulated scenarios in which the robot has to navigate to different goals while operating in an unknown environment, comprising randomly scattered landmarks. The control is found by minimizing the objective function (29), according to our dual-layer inference approach. We use a gradient descent method for optimizing the outer layer and Gauss–Newton for calculating inference in the inner layer. The number of look-ahead steps (L) is set to 5.

For simplicity we assume the robot can only control its heading angle while keeping the velocity constant. The control effort $\zeta(u)$ in Eq. (29) is therefore defined as the change in the heading angle. We assume on-board camera and range sensors

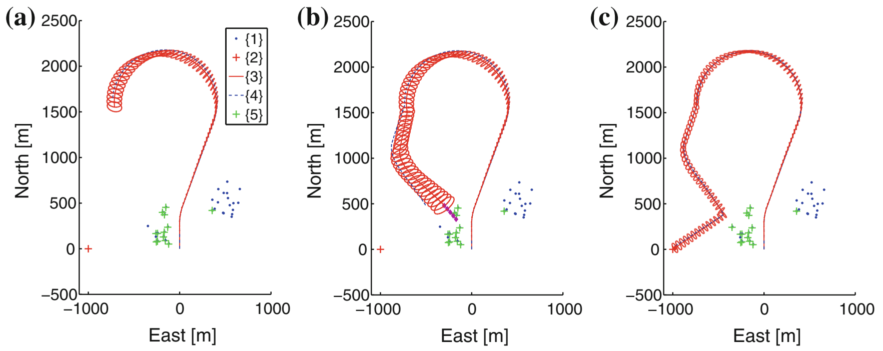


Fig. 3 A mobile robot starts from the configuration **a** and has to plan a motion strategy to reach a goal position, while reducing estimation uncertainty. **Legend**: {*1*} unknown environment $\mathscr{W}$; {*2*} Goal; {*3–4*} Actual and estimated trajectories of the robot with 1σ uncertainty bounds; {*5*} mapped environment $\mathscr{W}_k$. **a** Robot trajectory before planning begins. **b** Trajectory before loop closure. Planned motion (for $L = 5$ look ahead steps) is shown by *diamond* marks. **c** Final trajectory after reaching the goal

with measurements corrupted by Gaussian noise with standard deviation of 0.5 pixels and 1 meters, respectively. Using these sensors the robot can detect and measure the relative positions of nearby landmarks. The corresponding measurements can be described by the observation model (3), where the subset of states X^o comprises the robot's pose and the observed landmark. The motion model is represented by a zero-mean Gaussian with standard deviation of 0.05 meters in position and 0.5 degrees in orientation. The matrices $\bar{M}_x$ and $\bar{M}_\Sigma$ are set to extract the current robot state and covariance at each of the look-ahead steps; M_u is chosen to be the identity matrix.

We first present a basic scenario where the robot needs to navigate to a single goal (Fig. 3). For explanation purposes, in this first example we consider the planning phase starts from the configuration shown in Fig. 3a. During the first planning steps the distance to the goal is the dominant component in the objective function (28) and the calculated control guides the robot towards the goal. However, as the robot uncertainty increases, the parameter α increases causing more weight to be placed on uncertainty reduction. The planner then guides the robot towards previously observed landmarks in the environment (Fig. 3b). After the robot makes observations of these landmarks (*loop closure*), the uncertainty is reduced, hence the parameter α drops to lower values and more weight is put on guiding the robot to the goal (Fig. 3c).

We now consider a more complicated scenario, where the planning is carried out from the beginning and the robot has to navigate to a series of goals. The resulting robot trajectory using our approach is shown in Fig. 4a. As seen, robot uncertainty exceeds the threshold β twice (on the way to goals 4 and 7), see also Fig. 4b, and our planning strategy leads it to re-visiting previously observed landmarks. For comparison, Fig. 4c shows the trajectory when the objective function does not account for the uncertainty [i.e., without second and last terms in Eq. (29)]. Neglecting the uncertainty during planning leads to much larger covariances, as shown in Fig. 4c.

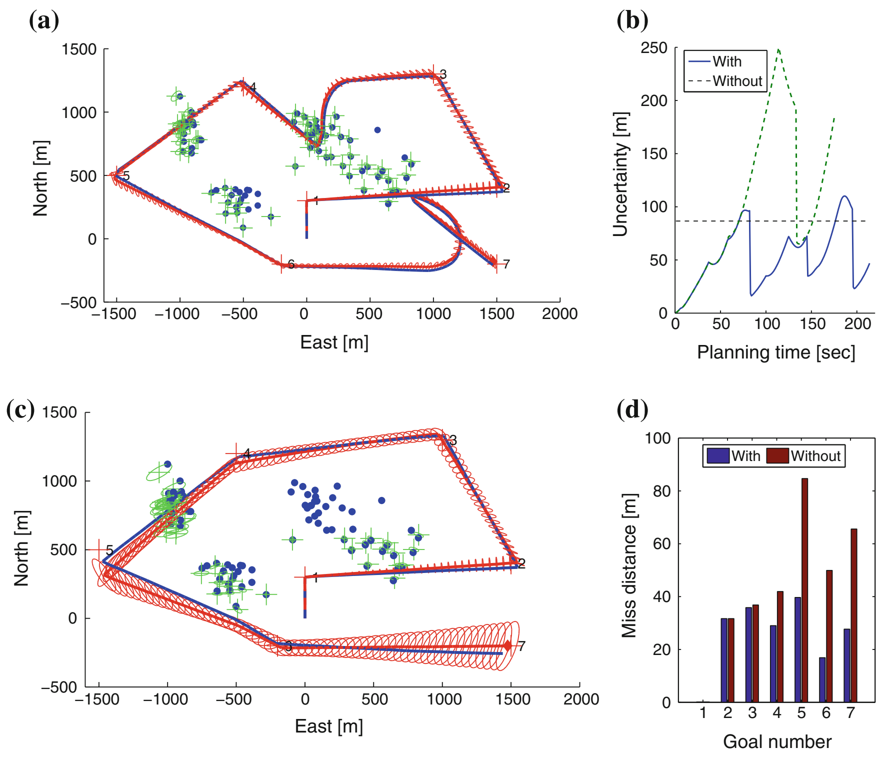


Fig. 4 Multi-goal planning example. The same notations as in Fig. 3 are used. Resulting trajectories when planning with and without uncertainty terms in the objective function are shown in **a** and **c**, respectively. **b** Covariance evolution in the two cases, with the covariance threshold β indicated by a *dashed-dotted line*. The drops in the covariance values correspond to loop closure events. **d** Miss distance at each of the goals

Moreover, this results in higher estimation errors, which leads to larger *miss distances*[2] with respect to the goals (Fig. 4d).

4 Conclusion

This work presents an approach for planning in the belief space assuming no prior knowledge of the environment in which the robot operates. In order to deal with the uncertainty about the surrounding environment and its state, the robot maintains a joint belief over its own state and the state of the world; this joint belief is used in the

[2]The robot considers a goal as achieved when its estimated position coincides with the goal; therefore, the miss distance is defined as the mismatch between the goal and the actual position at that time.

computation of a suitable control policy, leading to the concept of *generalized belief space* (GBS) planning. Our approach for planning in the GBS includes two layers of inference: the inner layer performs inference to calculate the belief at each of the look-ahead steps, for a given control action; the outer layer performs inference over the control, minimizing a suitable objective function. The approach does not assume maximum likelihood observations and allows planning in a continuous domain (i.e., without assuming a finite set of possible control actions). We elucidate on the theoretical derivation by presenting an application to a specific family of cost functions and discussing the policies computed in simulated examples.

References

1. Bai, H., David, H., Lee, W.S.: Integrated perception and planning in the continuous space: a POMDP approach. In: Robotics: Science and Systems (RSS) (2013)
2. Binney, J., Sukhatme, G.S.: Branch and bound for informative path planning. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2147–2154 (2012)
3. Blanco, J., Fernandez-Madrigal, J., Gonzalez, J.: A novel measure of uncertainty for mobile robot SLAM with rao-blackwellized particle filters. Int. J. Robot. Res. **27**(1), 73–89 (2008)
4. Bryson, M., Sukkarieh, S.: Observability analysis and active control for airborne SLAM. IEEE Trans. Aerosp. Electron. Syst. **44**, 261–280 (2008)
5. Carrillo, H., Reid, I., Castellanos, J.A.: On the comparison of uncertainty criteria for active SLAM. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2080–2087 (2012)
6. Du, J., Carlone, L., Kaouk Ng, M., Bona, B., Indri, M.: A comparative study on active SLAM and autonomous exploration with particle filters. In: IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), pp. 916–923 (2011)
7. Hollinger, G., Sukhatme, G.: Stochastic motion planning for robotic information gathering. In: Robotics: Science and Systems (RSS) (2013)
8. Huang, S., Kwok, N., Dissanayake, G., Ha, Q., Fang, G.: Multi-step look-ahead trajectory planning in SLAM: possibility and necessity. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 1091–1096 (2005)
9. Kaelbling, L.P., Lozano-Pérez, T.: Pre-image backchaining in belief space for mobile manipulation. In: Proceedings of the International Symposium of Robotics Research (ISRR) (2011)
10. Kaelbling, L.P., Lozano-Pérez, T.: Unifying perception, estimation and action for mobile manipulation via belief space planning. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2952–2959 (2012)
11. Kaelbling, L.P., Lozano-Pérez, T.: Integrated task and motion planning in belief space. Int. J. Robot. Res. **32**, 1194–1227 (2013)
12. Kim, A., Eustice, R.M.: Perception-driven navigation: active visual SLAM for robotic area coverage. In: IEEE International Conference on Robotics and Automation (ICRA) (2012)
13. Kontitsis, M., Theodorou, E.A., Todorov, E.: Multi-robot active SLAM with relative entropy optimization. In: American Control Conference (2013)
14. Leung, C., Huang, S., Dissanayake, G.: Active SLAM using model predictive control and attractor based exploration. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 5026–5031 (2006)
15. Martinez-Cantin, R., De Freitas, N., Doucet, A., Castellanos, J.A.: Active policy learning for robot planning and exploration under uncertainty. In: Robotics: Science and Systems (RSS) (2007)

16. Martinez-Cantin, R., de Freitas, N., Brochu, E., Castellanos, J.A., Doucet, A.: A Bayesian exploration–exploitation approach for optimal online sensing and planning with a visually guided mobile robot. Auton. Robots **27**(2), 93–103 (2009)
17. Platt Jr, R., Tedrake, R., Kaelbling, L.P., Lozano-Pérez, T.: Belief space planning assuming maximum likelihood observations. In: Robotics: Science and Systems (RSS), pp. 587–593 (2010)
18. Potthast, C., Sukhatme, G.: Next best view estimation with eye in hand camera. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2011)
19. Sim, R., Roy, N.: Global a-optimal robot exploration in SLAM. pp. 661–666 (2005)
20. Singh, A., Krause, A., Guestrin, C., Kaiser, W.J.: Efficient informative sensing using multiple robots. J. Artif. Intell. Res. **34**, 707–755 (2009)
21. Stachniss, C., Haehnel, D., Burgard, W.: Exploration with active loop-closing for FastSLAM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2004)
22. Stachniss, C., Grisetti, G., Burgard, W.: Information gain-based exploration using rao-blackwellized particle filters. In: Robotics: Science and Systems (RSS), pp. 65–72 (2005)
23. Valencia, R., Valls Miró, J., Dissanayake, G., Andrade-Cetto, J.: Active pose SLAM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1885–1891 (2011)
24. Valencia, R., Morta, M., Andrade-Cetto, J., Porta, J.M.: Planning reliable paths with pose SLAM. IEEE Trans. Robot. **29**, 1050–1059 (2013)
25. Van Den Berg, J., Patil, S., Alterovitz, R.: Motion planning under uncertainty using iterative local optimization in belief space. Int. J. Robot. Res. **31**(11), 1263–1278 (2012)

An Online POMDP Solver for Uncertainty Planning in Dynamic Environment

Hanna Kurniawati and Vinay Yadav

Abstract Motion planning under uncertainty is important for reliable robot operations in uncertain and dynamic environments. Partially Observable Markov Decision Process (POMDP) is a general and systematic framework for motion planning under uncertainty. To cope with dynamic environment well, we often need to modify the POMDP model during runtime. However, despite recent tremendous advances in POMDP planning, most solvers are not fast enough to generate a good solution when the POMDP model changes during runtime. Recent progress in online POMDP solvers have shown promising results. However, most online solvers are based on replanning, which recompute a solution from scratch at each step, discarding any solution that has been computed so far, and hence wasting valuable computational resources. In this paper, we propose a new online POMDP solver, called Adaptive Belief Tree (ABT), that can reuse and improve existing solution, and update the solution as needed whenever the POMDP model changes. Given enough time, ABT converges to the optimal solution of the current POMDP model in probability. Preliminary results on three distinct robotics tasks in dynamic environments are promising. In all test scenarios, ABT generates similar or better solutions faster than the fastest online POMDP solver today; using an average of less than 50 ms of computation time per step.

Vinay Yadav—All work were done while the author was an internship student at University of Queensland.

H. Kurniawati (✉)
School of Information Technology
and Electrical Engineering, University of Queensland, Brisbane, Australia
e-mail: hannakur@uq.edu.au

V. Yadav
Department of Electrical Engineering, Indian Institute of Technology, Kharagpur, India
e-mail: vinayyadav.iitkgp@gmail.com

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_35

1 Introduction

Motion planning under uncertainty is important for reliable robot operation in imperfectly known and dynamic environments. Partially Observable Markov Decision Process (POMDP) is a general and systematic framework for planning under uncertainty. Motion planning under uncertainty problems can be modelled as POMDPs quite naturally. However, solving a POMDP problem exactly is computationally intractable [14]. A lot of effort and tremendous progress have been made in developing efficient approximate POMDP solvers [5, 6, 11, 15, 16, 19, 21, 24], such that today, we have POMDP solvers that can solve simple to moderately difficult motion planning problems within seconds to minutes [7, 8, 12]. However, many of these solvers are not efficient enough to recompute or update its solution when the POMDP model changes during runtime. Such changes in the POMDP model are often required when a robot operates in dynamic environment. This paper proposes a new approximate POMDP solver that improves and updates its solution online, following changes in the environment.

In imperfectly known and dynamic environment, a robot rarely knows its exact state due to errors in control and sensing. POMDP provides a systematic way to reason about the best action to perform when perfect state information is unavailable. It finds the best action with respect to the set of states that are consistent with the available information so far. The set of states is represented as a probability distribution, called a belief b, and the set of all possible beliefs is called the belief space B. A POMDP solver calculates an optimal policy $\pi^*: B \rightarrow A$ that maps a belief in B to an action in the set A of all possible actions the robot can perform, so as to maximize a given objective function. An offline POMDP solver computes this mapping prior to execution, while an online POMDP solver computes the mapping during runtime.

Methods that use POMDP framework to solve planning in imperfectly known and dynamic environment can be classified into two approaches. The first approach embeds all possible environments and their dynamics as part of the POMDP model. It uses an offline POMDP solver to find a good policy, prior to execution. When the environment and its dynamics are largely unknown, this approach constructs POMDP models too huge to be solved by even the best offline solver today.

The second approach models only the known part of the environment and its dynamics (both stochastic and deterministic), and allows the model to change during execution when more information about the environment becomes available. Key to the success of this approach is an efficient online POMDP solver that can compute a good policy during runtime, following changes in the POMDP model.

Online POMDP solvers have advanced significantly over the past few years [5, 6, 18, 19]. However, most of these solvers [5, 6, 18] are based on replanning, which recompute the best action to perform from scratch at each step, discarding any policy that has been computed so far. As a result, these solvers often waste significant computational resources when changes happen gradually or only to some part of the environment, which are often the case in robotics tasks.

This paper proposes a new online POMDP solver, called Adaptive Belief Tree (ABT), that reuses and improves existing policy at each time step, and update the policy as needed whenever the POMDP model changes. To enable fast policy update, ABT uses the following two observations. First, a change in the POMDP model is directly reflected as a change in the robot's behaviour at a particular set of states. Second, a change in one optimal mapping $\pi^*(b)$ from a belief b, may affect the optimal policy $\pi^*(.)$ at other beliefs that can reach b. Using insight from these observations, ABT represents the policy as pairs of belief and action, and explicitly represents the relation between beliefs, states, and their reachability information, so that it can quickly identify subset of the policy affected by changes in the POMDP model and update the policy fast whenever necessary. To quickly generate a good policy, ABT plans with respect to only a set of representative sampled beliefs. It represents each belief as a set of state particles, and samples a belief b by sampling a set of state trajectories from a particle of the given initial belief b_0. An effective strategy for sampling state trajectories enables ABT to converge to an optimal policy in probability, and quickly generate a good policy. Preliminary results on three distinct robotics tasks in dynamic environment indicate that ABT can generate similar or better motion strategies faster than the best online POMDP solver today [19]. In all three test scenarios, ABT requires an average of less than 400 ms of preprocessing time, and an average of less than 50 ms of online computation time per step.

Furthermore, ABT is designed for POMDP problems with continuous state space and uses a generative model. A generative model is a black box simulator that enables us to generate experiences about the system dynamic and behaviour at various different states. By using a generative model, ABT does not need an explicit model on control error, observation error, and uncertainty about the system dynamics, which are often difficult to obtain in complex robotics tasks.

2 Related Work

2.1 POMDP Background

A POMDP is defined as a tuple $\langle S, A, O, T, Z, R, b_0, \gamma\rangle$, where S is the set of states, A is the set of actions, and O is the set of observations. At each step, the agent is in a state $s \in S$, takes an action $a \in A$, moves from s to an end state $s' \in S$, and perceives an observation $o \in O$. Due to action uncertainty, the system dynamic from s to s' is represented as a conditional probability function $T(s, a, s') = f(s'|s, a)$. Furthermore, due to sensing uncertainty, after performing action a and ends at state s', the observation that may be perceived by the agent is represented as a conditional probability function $Z(s', a, o) = f(o|s', a)$. At each step, the agent receives a reward $R(s, a)$, if it takes action a from state s. The agent's goal is to choose a suitable sequence of actions that will maximize its expected total reward, while the agent's initial belief is denoted as b_0. When the sequence of actions has infinite length, we specify a discount factor $\gamma \in (0, 1)$ so that the total reward is finite and the problem is well defined.

In many problems with large state space, explicit representation of the conditional probability functions T and Z may not be available. However, one can use a *generative model*, which is a black box simulator that outputs an observation perceived, reward received, and next state visited when the agent performs the input action from the input state.

A POMDP planner computes an *optimal policy* that maximizes the agent's expected total reward. A POMDP policy $\pi : B \rightarrow A$ assigns an action a to each belief $b \in B$. A policy π induces a value function $V(b, \pi)$ which specifies the expected total reward of executing policy π from belief b, and is computed as $V(b, \pi) = E[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)|b, \pi]$. A policy can be represented by various representations, e.g., policy-graph [3] and pairs of belief and action [23]. However, most online solvers do not maintain an explicit representation of the policy. Instead, they calculate the mapping from beliefs to actions on the fly. In contrast, our proposed online solver maintains an explicit representation of a subset of the policy, and improves and updates the policy on the fly.

To execute a policy π, an agent executes action selection and belief update repeatedly. For example, if the agent's current belief is b, it selects the action referred to by $a = \pi(b)$. After the agent performs action a and receives an observation o according to the observation function Z, it updates b to a new belief b' given by $b'(s') = \tau(b, a, o) = \eta Z(s', a, o) \int_{s \in S} T(s, a, s') ds$ where η is a normalization constant. We use generative model and represents each belief with a set of particles. The belief update is approximated using particle filter.

2.2 Related POMDP Solvers

POMDP is a systematic and general approach for planning under uncertainty. Although solving a POMDP exactly is computationally intractable [14], the past few years have seen tremendous increase in the capability of both offline and online POMDP solvers [11, 19, 21], such that POMDP approach is now practical for solving simple to moderately difficult motion planning problems.

The fastest offline POMDP solvers today are based on point-based approach [11, 15, 20, 21]. This approach reduces the complexity of planning in the belief space B by representing B as a set of sampled beliefs and planning with respect to this set only. To generate a policy, most point-based POMDPs use value iteration, utilizing the fact that the optimal value function satisfies Bellman equation. They start from an initial policy, represented as a value function V. And iteratively perform Bellman backup on V at the sampled beliefs, i.e., $V(b) = \max_{a \in A} \left(R(b, a) + \gamma \sum_{o \in O} \tau(b, a, o) \hat{V}^*(\tau(b, a, o)) \right)$ where $\hat{V}^*(b')$ is the current best value of b'. The iteration is performed until it converges. Over the past few years, impressive progress have been gained by improving the strategy for sampling

B [11, 21] and utilizing problem structures [13]. Different approaches have also been proposed for restricted types of uncertainty, e.g., [17, 24] for Gaussian beliefs.

Many online solvers have been proposed too [18]. One of the fastest general online POMDP solvers today is POMCP [19]. Starting from the current belief b, POMCP performs best first search in the belief space. It samples action sequences to quickly focuses on parts of the belief space that is most promising for generating the optimal policy from b. As any sampling based method, the sampling strategy is crucial. POMCP frames the problem of sampling the most promising action sequences as a problem of balancing exploration and exploitation, often called multi-armed bandit problem [22], and uses the Upper Confidence Bounds1 (UCB1) algorithm [1] to select the actions. After POMCP finishes the search, it performs the best action, updates the robot's belief, and repeats the procedure until the goal is reached.

Aside from POMCP, various approaches have been proposed for online POMDP solvers. PUMA [6] and RBSR [5] perform best first search in the belief space, just as POMCP does. However, instead of solving a multi arm bandit problem, PUMA and RBSR sample action sequences using heuristics in the state space, assuming that states are fully observed after an action is performed. The work in [16] plans with respect to only the most likely observation and then replan at each step. Recent work, LQG-Obstacles [25] is very fast, but restricts the belief to be Gaussian and is designed specifically for collision avoidance problems. Our new method is designed as a general online POMDP solver and the beliefs can be any type of distribution.

When the POMDP model changes, in general, the above offline and online solvers recompute the policy from scratch, wasting all computational effort that have been performed so far. In contrast, our new solver ABT can reuse and improve the policy, as well as update it as needed during runtime.

Recent work [10] has proposed a point-based method to modify a pre-computed policy. However, the time it needs to update a policy is too slow to be practical for dynamic environment, and the types of model changes that can be handled are limited to changes in transition, observation, and reward functions. In contrast, our new method can handle any types of changes in the model, and is fast enough to update a policy online, as presented in Sect. 5.3.

3 Policy Representation

ABT's policy representation must support fast identification on which parts of the policy are affected by changes in the POMDP model, and must support fast update of the policy. To this end, we note two observations. First, a change in any element of the POMDP model can be identified from changes in the robot's behaviour at a particular set of states. By definition, any change in the state space can be identified as addition, reduction, or changes in the types of a set of states. When the transition, observation, or reward function changes, then in the long run, the results of performing an action, the observations perceived, or the reward the robot received at a set of states would change. Therefore, changes in these functions can be identified from changes in the robot's behaviour at a particular set of states, too. Changes in action and observation spaces will affect the transition and observation functions of a set of states, and therefore these changes can be identified from the changes in the robot's behaviour at a particular set of states too. The second observation is a change in an optimal policy $\pi^*(b)$ from a belief $b \in B$ may change the optimal policy $\pi^*(.)$ from other belief(s) that can reach b.

Utilizing the above observations, ABT represents the policy as pairs of belief and action, and explicitly represents the relation between beliefs, states, and their reachability information. This representation helps to quickly identify a subset of the policy that needs to be updated, and to quickly update it. To maintain the relation, ABT represents each belief as a set of state particles, and associates each belief b with state trajectories that reach a particle of b from a particle of the given initial belief b_0. The details of the representation are below.

ABT maintains a set H of sampled episodes. An episode $h \in H$ is a sequence of quadruples (s, a, o, r) of state $s \in S$, action $a \in A$, observation $o \in O$, and immediate reward $r = R(s, a)$. To sample an episode h, ABT samples an initial state $s_0 \in S$ from a given initial belief b_0 and selects an action $a_0 \in A$. The details of action selection are discussed in Sect. 4.1. After an action a_0 is selected, ABT calls the generative model to sample an observation $o_0 \in O$, an immediate reward r_0, and a next state s_1 when the agent performs a_0 at s_0. ABT inserts the quadruple (s_0, a_0, o_0, r_0) as the first element of h, and iteratively repeats the above steps starting from s_1. The iteration stops after either a terminal state is reached or h has exceeded a certain length. As a last step, ABT inserts $(s,–,–, r)$ as the last element of h, where s is the next state sampled by the last call to the generative model and $r = R(s)$ is the reward of being at state s.

To maintain an explicit relation between beliefs, states, and their reachability information efficiently, ABT maintains a belief tree, denoted as $\mathcal{T}$, and associates it with the set of episodes in H. Each node in $\mathcal{T}$ represents a belief. For writing compactness, we refer to the node and the belief it represents interchangeably. The root of $\mathcal{T}$ represents the initial belief b_0. Each edge $\overline{bb'}$ in $\mathcal{T}$ is labelled by a pair of action and observation $a - -o$. An edge $\overline{bb'}$ with label $a - -o$ means that when a robot at belief b performs action a and perceives observation o, its next belief would be b', i.e., $b' = \tau(b, a, o)$ where $b, b' \in B$, $a \in A$, and $o \in O$.

The paths in the belief tree $\mathcal{T}$ are associated with the episodes in H. Suppose ϕ is a path in $\mathcal{T}$ and $\phi = \langle b_0, a_0, o_0, \ldots, a_n, o_n, b_{n+1}\rangle$, where $b_i, b_{n+1} \in B$, $a_i \in A$, and $o_i \in O$ for $i \in [0, n]$. Then, ϕ is associated with the set of episodes $H_\phi \subseteq H$ which consists of all episodes in H that contains $\langle(s_0, a_0, o_0, *), \ldots, (*, a_n, o_n, *), (*,–,–, *)\rangle$, where s_0 is any state sampled from b_0, a_i and o_i ($i \in [0, n]$) are the corresponding actions and observations in ϕ, and $*$ means any relevant value. Figure 1 illustrates the relation between an episode in H and a path in the belief tree $\mathcal{T}$. *Each episode in H corresponds to exactly one path of $\mathcal{T}$, but a path of $\mathcal{T}$ may be associated with many episodes.*

Each belief in $\mathcal{T}$ is represented by a set of particles, which comprises the states in the corresponding quadruples of the corresponding episodes. Suppose b is a node at level-l of $\mathcal{T}$ (the root has level 0). Suppose $\Phi(b)$ is the set of all paths in $\mathcal{T}$

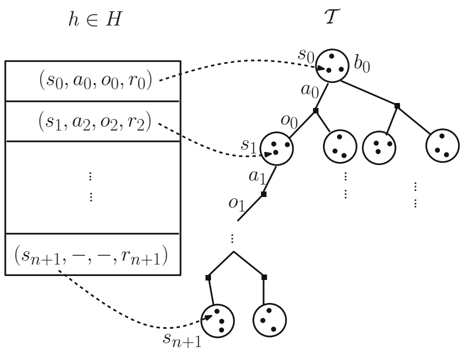


Fig. 1 Illustration of an association between an episode $h \in H$ and path in the belief tree $\mathcal{T}$

that starts from the root and contains the node b, and $H_b = \bigcup_{\phi \in \Phi(b)} H_\phi$. Then, b is approximated with the set of particles $\{h_l.s \mid h \in H_b\}$, which comprises the state in the lth quadruple of each episode in H_b (the quadruples are indexed from 0).

The policy π of ABT is embedded in the belief tree $\mathcal{T}$, with

$$\pi(b) = \arg\max_{a \in A(E,b)} \hat{Q}(b, a) \tag{1}$$

$$\text{and value} \quad V(b, \pi) = \max_{a \in A(E,b)} \hat{Q}(b, a) \tag{2}$$

where $b \in B$, E is the set of edges in $\mathcal{T}$, and $A(E, b) \subseteq A$ is the set of actions that have been used to expand b, i.e., the actions that labelled the out-edges of b in $\mathcal{T}$. The value $\hat{Q}(b, a)$ denotes the estimated Q-value. Q-value $Q(b, a)$ is the value of performing action a from belief b and continuing optimally afterwards, i.e., $Q(b, a) = R(b, a) + \gamma \sum_{o \in O} \tau(b, a, o) V^*(\tau(b, a, o))$. ABT estimates $Q(b, a)$ as

$$\hat{Q}(b, a) = \frac{1}{\left|H_{(b,a)}\right|} \sum_{h \in H_{(b,a)}} V(h, l) \tag{3}$$

where $H_{(b,a)} \subseteq H$ is the set of all episodes associated with all paths in $\mathcal{T}$ that start from b_0 and contains the sequence (b, a), l is the depth level of b in $\mathcal{T}$, and $V(h, l)$ is the value of an episode h starting from the lth element. $V(h, l)$ is computed as

$$V(h, l) = \sum_{i=l}^{|h|} \gamma^{i-l} R(h_i.s, h_i.a) \tag{4}$$

where γ is the discount factor and R is the reward function. Note that each state in the lth quadruple of each episode in $H_{(b,a)}$ is a particle of b and the action in that quadruple is the action a. Therefore, it is clear that $\hat{Q}(b, a)$ approximates the first component of $Q(b, a)$ well. However, it may seem odd that Eqs. (3) and (4) can approximate the second component of $Q(b, a)$, which is $\sum_{o \in O} \tau(b, a, o) V^*(\tau(b, a, o))$, as $V(h, l+1)$ for different h may correspond to different policy. It turns out by using an appropriate action selection strategy when sampling the episodes, one can ensure that as the number of episodes in $H_{(b,a)}$ increases, $V(h, l+1)$ converges to $\sum_{o \in O} \tau(b, a, o) V^*(\tau(b, a, o))$ in probability. This convergence result is based on the convergence result of POMCP [19]. The action selection strategy is discussed in Sect. 4.1 while the convergence result is discussed in Sect. 4.3.

The above policy representation and value calculation enable ABT to quickly identify and update the policy following changes in the POMDP model. To identify which parts of the policy need to be updated, ABT only needs to find the episodes in H that contain states that are affected by the changes in the POMDP model. To update the policy, ABT disconnects the association between each affected episode h and its corresponding nodes in $\mathcal{T}$, revises h according to the new POMDP model, and associates it back with the nodes of $\mathcal{T}$ (which may be different than the previously associated path). Then, ABT updates the values and Q-values of beliefs that have new association or disassociation with h. Using Eqs. (2)–(4), the value and Q-value revisions require only simple arithmetic calculation. With proper data structure, these values can be updated incrementally, and finding the affected episodes and the process of association and disassociation can be done fast. Details on the identification and policy update process are presented in Sect. 4.2.

4 Offline and Online Policy Computation and Update

ABT starts by computing a good approximation to the optimal policy for the a priori POMDP model, offline. During runtime, if the environment and hence the POMDP model changes, ABT identifies subset of the policy that needs to be updated and updates it. Otherwise, ABT improves its current policy. Algorithm 1 presents an overview of ABT.

4.1 Sampling the Episodes

The key strategy in generating an initial policy (GENERATE-POLICY function) and improving a policy (IMPROVE-POLICY function) are the same, which is in sampling the episodes. The overall sampling strategy of ABT is in Algorithm 2.

To sample a new episode, ABT starts by sampling a particle state $s \in S$ from a given starting belief $b_{start} \in B$. For the offline policy generation GENERATE-POLICY function, the starting belief is always the given initial belief, while for IMPROVE-POLICY, the starting belief is the belief at the current time. After a state is sampled, ABT selects an action and uses the generative model to sample an observation, reward, and next state (line 7–15, 20–30).

To select an action from state $s \in S$ that corresponds to node b in $\mathcal{T}$, ABT uses two strategies. First is the UCB strategy [1, 19], which is used when all actions in

Algorithm 1 Adaptive Belief Tree (b_0)

```
PREPROCESS (OFFLINE)
$(H, \mathcal{T})$ = GENERATE-POLICY($P_0$, $b_0$). {$P_i$ is the POMDP model at time-$i$.}
Let $S'$ be the set of all sampled states in $H$, i.e., $S' = \{h_i.s \mid i \in [0, |h|], h \in H\}$
Let $\mathcal{R}$ be a range tree representation of $S'$.
$b = b_0$.

RUNTIME (ONLINE)
while running do
  if $P_t \neq P_{t-1}$ then
    $H'$ = IDENTIFY-AFFECTED-EPISODES($P_{t-1}$, $P_t$, $H$, $\mathcal{R}$, $\mathcal{T}$).
    REVISE-EPISODES($P_t$, $\mathcal{T}$, $b$, $H'$).
    UPDATE-VALUES($\mathcal{T}$, $b$, $H'$).
  while there is still time do
    IMPROVE-POLICY($P_t$, $H$, $\mathcal{R}$, $\mathcal{T}$, $b$).
  $a$ = Get best action in $\mathcal{T}$ from $b$.
  Perform action $a$.
  $o$ = Get observation.
  $b = \tau(b, a, o)$.
  $t = t + 1$.
```

A have been used to expand b at least once (line 8–9). UCB strategy frames the action selection problem from each node as a multi-arm bandit problem. Multi-arm bandit problem is a reinforcement learning problem to select a sequence of actions, so as to maximize the total reward when the rewards for selecting the actions are not known in advance. This problem is essentially a problem of balancing exploration and exploitation, i.e., should one uses the action that has shown good performance so far even though it may not be the best action (exploitation) or should one tries other actions that have not shown good performance but may actually be the best action (exploration). To select an action using UCB strategy, ABT uses UCB1 algorithm [1], which selects an action according to

$$a = \arg\max_{a \in A} \left(\hat{Q}(b, a) + c \sqrt{\frac{\log(|H_b|)}{|H_{(b,a)}|}} \right) \tag{5}$$

where H_b is the set of episodes in H that has been associated with b, $H_{(b,a)}$ is the set of episodes in H that corresponds to sequence (b, a), $|.|$ is size of a set, and c is a scalar factor that determines the ratio between exploration and exploitation. UCB1 algorithm is one of the best multi-arm bandit solutions when the reward of performing an action follows a stationary distribution, which may not be known in advance. UCB1 algorithm has also been used for action selection by the fastest online POMDP solver today [19], and has been shown to enable convergence to the optimal policy [9, 19].

Algorithm 2 SAMPLING-AN-EPISODE(P, $\mathcal{T}$, b_{start}, H, ε)

```
1: $b = b_{start}$
2: Let $l$ be the depth level of node $b$ in $\mathcal{T}$.
3: Let $s$ be a state sampled from $b$.
   The sampled state $s$ is essentially the state at the $l^{th}$ quadruple of an episode $h' \in H$.
4: Initialize $h$ with the first $l$ elements of $h'$.
5: Initialize doneMode as UCB.
6: Let $A$ be the action space of POMDP model $P$.
7: while $\gamma^l > \varepsilon$ AND doneMode == UCB do
8:    Let $A'$ be the set of actions that labelled the edges from $b$ in $\mathcal{T}$.
9:    if $|A'| == |A|$ then
10:      $a$ = UCB-ACTION-SELECTION($\mathcal{T}$, $b$).
11:   else
12:      $a$ = an action sampled from $A \setminus A'$ uniformly at random.
13:      doneMode = Rollout.
14:   $(o, r, s')$ = GenerativeModel($P$, $s$, $a$).
15:   Insert $(s, a, o, r)$ to $h$.
16:   Add $h_l.s$ to the set of particles that represent belief node $b$ and associate $b$ with $h_l$.
17:   $s = s'$
18:   $b$ = child node of $b$ via an edge labelled $a$-$o$. If no such child exist, create the child.
19:   $l = l + 1$.
20: if doneMode == Rollout then
21:   Let $p$ be a number sampled uniformly at random from [0, 1].
22:   if $p < p_{policy}$ then
23:      $r$ = ROLLOUT-POLICY($\mathcal{T}$, $s$, $b$).
24:      rolloutUsed = policy.
25:   else
26:      $r$ = ROLLOUT-DET($P$, $s$).
27:      rolloutUsed = deterministic.
28: else
29:   $r$ = GenerativeModel($P$, $s$).
30: Insert $(s, -, -, r)$ to $h$.
31: Add $h_l.s$ to the set of particles that represent belief node $b$ and associate $b$ with $h_l$.
32: valueImprovement = UPDATE-VALUES($\mathcal{T}$, $h$)
33: $p_{policy}$ = UPDATE-ROLLOUT-PROB(rolloutUsed, valueImprovement).
34: Insert $h$ to $H$.
```

When the condition for using UCB is not satisfied, ABT selects an action towards satisfying the condition of UCB using rollout strategy. To select an action from state $s \in S$ that corresponds to node b of $\mathcal{T}$, rollout strategy samples an action a uniformly at random from the set of actions that has not been used to expand b (line 12).

Furthermore, to generate a good policy fast, ABT also tries to compute a good estimate of the Q-value $Q(b, a)$ in its rollout strategy (line 21–27). A good estimate of the Q-value will help the UCB strategy to converge faster to the optimal action once the condition to run UCB strategy has been satisfied. If the time to generate or improve the policy has run out before the condition to run UCB is satisfied, a good estimate of the Q-value helps ABT choose a good action.

To estimate the Q-value during rollout, ABT uses two heuristics. Suppose rollout strategy selects action $a \in A$ to be performed from state $s \in S$ that corresponds to node b of $\mathcal{T}$. *The first heuristic to estimate* $Q(b, a)$ *assumes the problem is deterministic* (line 26). ABT uses methods from deterministic motion planning to find a good solution. It calculates the total reward if the robot starts from state s, performs action a, and continues optimally, assuming the system is deterministic. This total reward is the estimated Q-value of this heuristic and the output of ROLLOUT-DET in line 26. *The second heuristic is based on existing policy* (line 23). For this purpose, ABT first uses the generative model to sample an observation $o \in O$, computes the belief $b' = \tau(b, a, o)$ using particle filter, and finds the node in $\mathcal{T}$ nearest to b'. Any distance metric for distributions can be used. ABT assumes that the state space is a metric space, which is mostly the case for robotics tasks, and defines the distance between two beliefs as the expected state space distance assuming the two beliefs are independent. Suppose the nearest node to b' is $\widehat{b'}$. If the distance between b' and $\widehat{b'}$ is more than a given threshold, ABT uses ROLLOUT-DET to estimate the Q-value. Otherwise, ABT assumes that b' is equal to $\widehat{b'}$. It simulates the robot's movement according to the policy embedded in $\mathcal{T}$ starting from $\widehat{b'}$, until a leaf node of $\mathcal{T}$ is reached. The total discounted reward gathered during this simulation becomes this heuristic's estimate of the Q-value $Q(b, a)$ and the output of ROLLOUT-POLICY in line 23.

Now, the question is which heuristic should be used at a particular rollout operation. This problem is similar to the action selection problem discussed earlier in this section, and similarly ABT frames the problem of choosing which heuristic to use as a multi-arm bandit problem. However for simplicity, this selection is valid globally instead of per belief node as in the case with action selection. Due to this simplification, we cannot use UCB1, as it assumes that the rewards follow a stationary distribution. Instead, we use Exp3 [2], one of the best multi-arm bandit solutions when no statistical assumptions are made about the underlying reward function. Furthermore, Exp3 has been shown to be competitive to the strategy that uses the best action at each step. Using Exp3, ABT assigns a probability to each heuristic strategy, and selects which heuristic to use based on this probability (line 22). The probability is adapted based on how much the heuristic improves the value of the starting belief (line 33), as follows

$$p_i(t+1) = (1 - c_r)\frac{w_i(t+1)}{w_1(t+1) + w_2(t+1)} + \frac{c_r}{2}$$
$$\text{where} \quad w_i(t+1) = w_i(t)\exp\left(\frac{c_r\,(\max(0, V_t(b_{start}) - V_{t-1}(b_{start})))}{2p_i(t)}\right) \quad (6)$$

where $i \in [1, 2]$ indicates the different heuristics, t is the current time step, and $c_r \in (0, 1)$ is the ratio between exploration and exploitation.

4.2 Handling Changes in the POMDP Model

When the POMDP model changes, ABT identifies a subset of the policy that is affected by the changes (IDENTIFY-AFFECTED-EPISODES function) and updates it (REVISE-EPISODES and UPDATE-VALUES functions).

To identify a subset of the policy that are affected by changes in the POMDP model, ABT needs to identify the set $S_{ch} \subseteq S$ of states affected by the changes, i.e., all states $s \in S$ where the robot's behaviour in s changes. In this paper, we do not focus on how to identify changes in the POMDP model. Instead, we assume that either changes in the POMDP model can be identified easily by identifying changes in the environment map, or the user provides information on the set of affected states S_{ch}.

Once the set S_{ch} of affected states is known, ABT finds the set of episodes affected by the changes. An episode $h \in H$ is affected by the changes in the POMDP model if at least one of its state element is affected by the changes, i.e., $\exists\, i \in [0, |h|]\ \ h_i.s \in\ S_{ch}$.

To facilitate fast identification of affected episodes, ABT structures the set $S' \subseteq S$ of all sampled states, i.e., the set of all states in each sampled episode $S' = \{h_i.s \mid i \in [0, |h|], h \in H\}$, in a range tree denoted as $\mathcal{R}$. Since multiple episodes may contain the same state, ABT labels each state $s \in S'$ with a set of two-tuple (h, idx) indicating which episodes h of H and which index idx element of h contain s.

To identify episodes in H that are affected by the changes in the POMDP model, ABT finds the intersection between the set S_{ch} of affected states and the set S' of sampled states. For this purpose, ABT constructs a bounding rectangle for each connected component of S_{ch}. Here, rectangle is used in a general sense, referring to hyper-rectangle when the dimension of S is more than two. Then, ABT solves a rectangular query on the range tree $\mathcal{R}$ for each bounding rectangle and checks if the states resulting from the rectangular queries are indeed in S_{ch}. The results of the rectangular queries that lie in S_{ch} are sampled states that are affected by the changes in the POMDP model. The two-tuple labels associated with these sampled states indicate the episodes that are affected by the changes in the POMDP model.

Assuming the range tree has been constructed, the above identification procedure takes $O\left(\log^{dim(S)} |S'| + k + |H'|\right)$, where $dim(S)$ is the dimension of the state space S and k is the total number of states outputted by all rectangular range queries [4]. The first construction of the range tree, which happens offline, takes $O(|S'_0| \log^{dim(S)-1} |S'_0|)$, where S'_0 is the set of all sampled states right after the offline policy generation. During runtime, the time to insert the state of the newly sampled quadruple to the range tree is $O(\log^{dim(S)-1} |S'_t|)$, where S'_t is the set of all sampled states at time t.

Once the set $H' \subseteq H$ of affected episodes are identified, ABT revises affected elements of all episodes in H' according to the new POMDP model. Suppose $h \in H'$

is an affected episode to be revised. First, ABT finds the lowest element index $\underline{idx}$ of h where the state is affected by model changes. Then, ABT revises the episode starting from element index $idx_u = \max(0, \underline{idx} - 1)$ of h until the last element. When $idx_u = 0$, ABT erases the episode h from H, because in this case the entire episode needs to be revised, which means the episode is not reusable and the results would be similar as if ABT samples an entirely new episode. If $idx_u > 0$, ABT uses the generative model to re-sample the sequence of observations perceived, rewards received, and next states visited, when the sequence of actions from element idx_u until the last element of h is performed, starting from the state in element idx_u of h. When this sequence of actions is obviously sub-optimal, e.g., when it causes the robot to collide with a newly added obstacle, ABT uses heuristics to modify the sequence of actions. The heuristics is the same as that used in ROLLOUT-DET (line 26 of Algorithm 2). It assumes the problem is deterministic and uses methods from deterministic motion planning to find a good sequence of actions. Finally, ABT replaces the content of the quadruples of h with the re-sampled sequence of observations, rewards, and next states, at the respective indices. This revision of $h \in H'$ triggers re-computation of the values and Q-values of all beliefs in $\mathcal{T}$ that correspond to h, and hence update the policy embedded in $\mathcal{T}$.

4.3 Convergence to an Optimal Policy

ABT converges in probability to the optimal policy from the current belief under the current POMDP model.

First, let us discuss the case when the POMDP model does not change. Key to ABT's convergence is the strategy for sampling the episodes H (Sect. 4.1), in particular the action selection strategy. ABT frames the problem of selecting an action when sampling an episode, as a multi-arm bandit problem and uses UCB1 algorithm. This action selection strategy has been proven to enable convergence to the optimal policy in probability, regardless of the rollout strategy being used [19]. In fact, [19] has shown that $\hat{Q}(b, a)$ will converge to the optimal Q-value with a rate of $O(\log(|H_{\Gamma(b)}|)/|H_{\Phi(b)}|)$, where $H_{\Phi(b)} \subseteq H$ is the set of episodes that correspond to paths in $\mathcal{T}$ that starts from b_0 and contains node b, and $|.|$ is the size of a set. When $\hat{Q}(b, a)$ converges to the optimal Q-value, the value $\hat{V}(b)$ converges to the optimal value function and the corresponding $\pi(b)$ converges to the optimal policy.

When the POMDP model changes, the existing policy that has been revised can be considered as an initial policy. When enough time is given to improve the policy, the number of sampled episodes keeps increasing, such that the quality of the policy will eventually be dominated by the results of the episode sampling strategy. Therefore, the above results remain valid when the POMDP model changes.

5 Experiments

5.1 Robotics Tasks

We have tested ABT on three robotics tasks that require the POMDP model to be modified several times during runtime. To ensure that changes in the environment affect the solution to the problem regardless of how fast or slow a solver is, we define changes in terms of time steps. The three test scenarios are as follows.

Underwater navigation (Fig. 2). An Autonomous Underwater Vehicle (AUV) navigates in an environment populated by obstacles and vortices that are not known a priori. In the beginning, we only know the start and goal regions, and the positions of underwater beacons where the AUV can localize perfectly (labelled 'O'). During run time, at time step 10, the obstacles (dark grey) become known. These obstacles possess unique features that can be used by the AUV to localize itself, but at the same time obstruct some of the underwater beacons. To reflect these new information, the POMDP model is modified; the number of states decreases while the number of observations increases. At time step 20, the vortex (light grey with cross mark) becomes known. The POMDP model again changes to reflect the vortex.

The AUV may start from one of the two possible regions labelled 'S' and needs to reach the goal region labelled 'G' while avoiding obstacles and being dragged in a vortex region. The environment is represented as a uniform grid of size 51×52. At each step, the AUV can move one cell in 5 directions, i.e., East, North, South, Northeast, and Southeast. Due to underwater current, the AUV movement is accurate only 80 % of the time. The rest of the time, it reaches the left or right of the intended destination with equal probability. The AUV does not have a GPS, but can localize perfectly at cells marked by 'O'. At other cells, no observations are perceived.

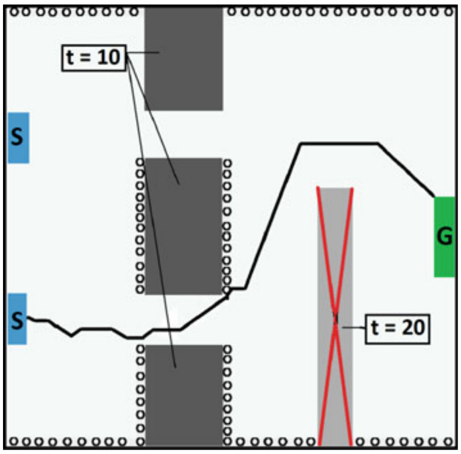


Fig. 2 Underwater navigation. $|S_{t=0...9}| = 2,652$, $|S_{t\geq 10}| = 2,274$, $|A| = 5$, $|O_{t=0...9}| = 104$, $|O_{t=10...19}| = 142$, $|O_{t\geq 20}| = 138$. *Black line* is a path the robot follows when using ABT

The AUV receives a high reward for reaching the goal, a small penalty for every action taken, and a high penalty for being in the vortex region.

Homecare (Fig. 3). A robot is deployed for caretaking purposes in a home environment. The robot needs to find and attend to an elderly whenever she needs assistance. The elderly moves around in the house, starting from one of the regions labelled ‘T’. Her motion is non-deterministic: At each step, she may pause or continue following one of several paths (marked by dashed line). She is likely to stop for longer time in places marked ‘W’, which represents washroom, regions near dining table, TV, or refrigerator. At time step 30, 60, 90, and 120, new furnitures and appliances where the elderly may pause longer are added (labelled ‘W’). To reflect these changes, the transition function of the POMDP model is modified accordingly.

The environment is populated by obstacles (dark grey) and is represented as a uniform grid of size 50 × 50. The robot starts from a region marked by ‘S’. At each step, the robot may stay or move one cell in one of the 8 wind directions. Its motion is accurate only 80 % of the time. The rest of the time, it reaches the left or right of the intended destination with equal probability. The elderly calls the robot whenever assistance is needed and turns off the call when assistance is no longer needed. When a call is made, the elderly pause at her current location until the robot comes or until assistance is no longer needed. The robot receives a high reward whenever it reaches the elderly when assistance is still needed. A small penalty is imposed for every move the robot takes to discourage it from wasting energy. The robot has access to four types of observations. First is a GPS. Second is a visibility sensor that enables the robot to localize the elderly exactly if she is within 1 cell away from the robot. Third is from four sensors mounted on the ceiling. These sensors divide the home into four equal regions and can identify which region the elderly is in with 90 % accuracy. The rest of the time, the sensor wrongly identifies the region where the elderly is in with equal probability. Last is observation on whether the elderly requires assistance, which is 100 % accurate.

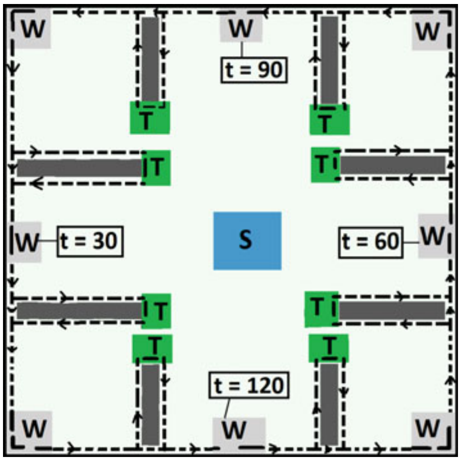


Fig. 3 Homecare. $|S| = 1,926,912$, $|A| = 9$, $|O| = 200,000$

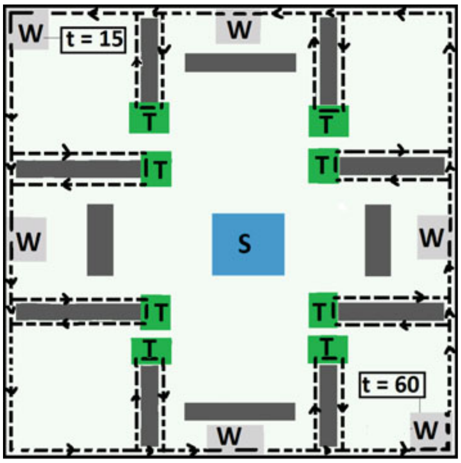


Fig. 4 Target finding. $|S| = 923,520$, $|A| = 9$, $|O| = 100,000$.

Target finding (Fig. 4). This problem is similar to homecare, but here the goal is for the robot to quickly find the elderly, while she is moving in the house. The environment is slightly more complex than homecare, due to additional obstacles. Similar to homecare, the elderly behaviour changes with the addition of new furnitures and appliances. The changes (time step 30 and 60) are reflected in the transition function. The robot's dynamics are the same as in homecare. The robot's observations are also the same as homecare, but without observation on whether the elderly needs assistance.

5.2 Experimental Setup

We implement ABT in C++ and test it on the above tasks. To calculate the quality of the motion strategies generated by ABT, we estimate the expected total reward of using ABT to solve each task. To this end, for each task, we first ran a few trial runs to determine the best parameters for ABT to use. Then, we use the best parameters for each task to generate 30 different offline policies. Finally, for each task and each policy, we run 100 simulation runs and compute the total reward of each simulation. The average of these 3,000 simulation runs is the estimated expected total reward.

As a comparator, we also apply POMCP [19], the fastest online POMDP solver today, on the above tasks. For POMCP, we use the software released by the original author, which is written in C++. Similar to ABT, for each task, we first ran a few trial runs to determine the best parameters for POMCP to use. These parameters include the use of additional heuristic to help POMCP's rollout function performs better (knowledge option in POMCP software). We use the best parameters to run 500 simulation runs for each task. The average of these simulation runs is the estimated expected total reward generated by POMCP for solving the task.

All experiments are conducted in a PC with Intel Xeon E5-1620 3.6 GHz processor and 16GB RAM.

5.3 Results

The results of ABT and POMCP are in Table 1. All values in Table 1 are in the form of average ±0.95 confidence interval. POMCP recomputes the solution at each step using 1,024 particles (note: The POMCP s/w always recomputes from scratch). ABT improves the solution using an additional 1,000 or 2,500 unweighted particles per step. The first column shows the expected total discounted reward. The second column shows the time ABT uses for preprocessing, to generate an initial policy for the a priori POMDP model. POMCP does not perform any preprocessing. The last column shows the average online computation time ABT and POMCP use at each step. The average time per step for ABT includes the time to improve existing policy and to update the policy when the POMDP model changes. The average time to perform one policy update for underwater navigation is (87.42 ± 1.13) ms, while the average policy update of all ABT runs for homecare and target finding are less than 1.5 ms, which is below the timer accuracy of our computer system (4 ms).

The results show that in all three scenarios, ABT significantly outperforms POMCP. It can generate similar or better motion strategies up to 120× faster than POMCP. By reusing existing policy, ABT can focus its search faster on parts of the

Table 1 Performance comparison

	Total discounted reward	Offline computation time (ms)	Online computation time Average time/step (ms)
Underwater navigation			
POMCP[a]	138.98 ± 47.77	–	754.00 ± 11.23
ABT[b]	185.50 ± 38.23	364.00 ± 45.96	42.90 ± 0.25
Homecare			
POMCP[a]	2,251.04 ± 275.98	–	1,933.75 ± 13.80
ABT[b]	2,297.34 ± 102.91	63.33 ± 5.82	16.12 ± 0.11
ABT[c]	2,509.38 ± 104.53	63.33 ± 5.82	39.98 ± 0.15
Target finding			
POMCP[a]	2,237.12 ± 142.32	–	1,221.95 ± 6.59
ABT[b]	2,284.69 ± 60.28	63.00 ± 5.65	15.12 ± 0.16
ABT[c]	2,594.35 ± 60.28	63.00 ± 5.65	44.61 ± 0.33

[a]Use 1,024 particles/step
[b]Improve with 1,000 particles/step
[c]Improve with 2,500 particles/step

belief space that are most promising for generating the best action strategy from the current belief under the current POMDP model.

In underwater navigation, ABT requires more offline and online computation time, compared to homecare and target finding, even though the size of state, action, and observation spaces are smaller. The reason is underwater navigation requires longer planning horizon and has more complex geometry, compared to the other two problems. As a result, it takes more time to compute the deterministic motion planning heuristic, one of the heuristics used to estimate the Q-value (step 26 of Algorithm 2). This computation is also the reason why policy update time for underwater navigation takes much longer than the other two problems. A more efficient implementation of the deterministic motion planner will reduce the offline and online computation time of underwater navigation.

6 Summary

This paper proposes a new online POMDP solver, called ABT, that reuses and improves existing policy, and updates the policy as needed whenever the POMDP model changes. It is designed for POMDP problems with continuous state space and uses a generative model. We have successfully tested ABT on three different robotics tasks in dynamic environment, where each task requires the POMDP model to change several times during runtime, so as to reflect the environment correctly. Simulation results on these test scenarios show that ABT generates similar or better motion strategies faster than the fastest online POMDP solver today. In all test scenarios, ABT requires an average of less than 400 ms of preprocessing time, and an average of less than 50 ms of online computation time at each step. These results suggest that ABT brings POMDP a step closer to become practical for non-trivial robotics tasks in uncertain and dynamic environments, even when the environment dynamics are unknown in advance, a class of robotics tasks often deemed too difficult to be solved using POMDP approach.

References

1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Mach. Learn. **47**(2–3), 235–256 (2002)
2. Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: The non-stochastic multi-armed bandit problem. SIAM J. Comput. **32**(1), 48–77 (2003)
3. Bai, H., Hsu, D., Lee, W.S., Ngo, A.V.: Monte Carlo value iteration for continuous-state POMDPs. In: Proceedings of the WAFR (2010)
4. de Berg, M., Cheong, O., Kreveld, M.V., Overmars, M.: Computational Geometry: Algorithms and Applications. Springer, Berlin (2000)
5. Hauser, K.: Randomized belief-space replanning in partially-observable continuous spaces. In: Proceedings of the WAFR (2010)

6. He, R., Brunskill, E., Roy, N.: PUMA: planning under uncertainty with macro-actions. In: Proceedings of the AAAI (2010)
7. Horowitz, M., Burdick, J.: Interactive non-prehensile manipulation for grasping via POMDPs. In: Proceedings of the ICRA (2013)
8. Hsiao, K., Kaelbling, L.P., Lozano-Perez, T.: Grasping POMDPs. In: Proceedings of the ICRA, pp. 4685–4692 (2007)
9. Kocsis, L., Szepesvri, C.: Bandit based monte-carlo planning. In: ECML-06. LNCS, vol. 4212, pp. 282–293. Springer, Berlin (2006)
10. Kurniawati, H., Patrikalakis, N.M.: Point-based policy transformation: adapting policy to changing POMDP models. In: Proceedings of the WAFR (2012)
11. Kurniawati, H., Hsu, D., Lee, W.S.: SARSOP: efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: Proceedings of the RSS (2008)
12. Kurniawati, H., Du, Y., Hsu, D., Lee, W.S.: Motion planning under uncertainty for robotic tasks with long time horizons. IJRR **30**(3), 308–323 (2011)
13. Ong, S.C.W., Png, S.W., Hsu, D., Lee, W.S.: Planning under uncertainty for robotic tasks with mixed observability. IJRR **29**(8), 1053–1068 (2010)
14. Papadimitriou, C.H., Tsitsiklis, J.N.: The complexity of Markov decision processes. Math. Oper. Res. **12**(3), 441–450 (1987)
15. Pineau, J., Gordon, G., Thrun, S.: Point-based value iteration: an anytime algorithm for POMDPs. In: Proceedings of the IJCAI, pp. 1025–1032 (2003)
16. Platt, R., Tedrake, R., Lozano-Perez, T., Kaelbling, L.P.: Belief space planning assuming maximum likelihood observations. In: Proceedings of the RSS (2010)
17. Prentice, S., Roy, N.: The belief roadmap: efficient planning in linear POMDPs by factoring the covariance. In: Proceedings of the ISRR (2007)
18. Ross, S., Pineau, J., Paquet, S., Chaib-draa, B.: Online planning algorithms for POMDPs. JAIR **32**, 663–704 (2008)
19. Silver, D., Veness, J.: Monte-Carlo planning in large POMDPs. In: Proceedings of the NIPS (2010)
20. Smith, T., Simmons, R.: Heuristic search value iteration for POMDPs. In: Proceedings of the UAI (2004)
21. Smith, T., Simmons, R.: Point-based POMDP algorithms: improved analysis and implementation. In: Proceedings of the UAI, July 2005
22. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (2012)
23. Thrun, S.: Monte carlo POMDPs. In: Proceedings of the NIPS, pp. 1064–1070 (2000)
24. van den Berg, J., Abbeel, P., Goldberg, K.: LQG-MP: optimized path planning for robots with motion uncertainty and imperfect state information. In: Proceedings of the RSS (2010)
25. van den Berg, J., Wilkie, D., Guy, S.J., Niethammer, M., Manocha, D.: LQG-Obstacles: feedback control with collision avoidance for mobile robots with motion and sensing uncertainty. In: Proceedings of the ICRA (2012)

An Enzyme-Inspired Approach to Stochastic Allocation of Robotic Swarms Around Boundaries

Theodore P. Pavlic, Sean Wilson, Ganesh P. Kumar and Spring Berman

Abstract This work presents a novel control approach for allocating a robotic swarm among boundaries. It represents the first step toward developing a methodology for encounter-based swarm allocation that incorporates rigorously characterized spatial effects in the system without requiring analytical expressions for encounter rates. Our approach utilizes a macroscopic model of the swarm population dynamics to design stochastic robot control policies that result in target allocations of robots to the boundaries of regions of different types. The control policies use only local information and have provable guarantees on the collective swarm behavior. We analytically derive the relationship between the stochastic control policies and target allocations for a scenario in which circular robots avoid collisions with each other, bind to boundaries of disk-shaped regions, and command bound robots to unbind. We validate this relationship in simulation and show that it is robust to environmental changes, such as a change in the number or size of robots and disks.

Keywords Distributed robotic systems · Stochastic robotics · Bio-inspiration · Chemical reaction networks · Attachment–detachment

T.P. Pavlic (✉)
School of Computing, Informatics, and Decision Systems Engineering / School of Sustainability, Arizona State University, Tempe, AZ, USA
e-mail: tpavlic@asu.edu

S. Wilson · S. Berman
School for Engineering of Matter, Transport, and Energy,
Arizona State University, Tempe, AZ, USA
e-mail: Sean.T.Wilson@asu.edu

S. Berman
e-mail: Spring.Berman@asu.edu

G.P. Kumar
School for Computing, Informatics and Decision Systems Engineering,
Arizona State University, Tempe, AZ, USA
e-mail: Ganesh.P.Kumar@asu.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_36

1 Introduction

In recent years, there has been an increasing focus on the development of *robotic swarms* [5] for performing tasks that require high degrees of parallelism, redundancy in system components and behaviors, and adaptability to changes in environmental conditions and failures. These systems would be composed of hundreds or thousands of relatively expendable, resource-constrained robots that operate with little-to-no human supervision. Advances in computing, sensing, actuation, power, communication, and control technologies are currently enabling the production of affordable robots that are designed to act in collectives, both in research and education [4, 20]. In the past few years, the miniaturization of these technologies has led to a plethora of novel platforms for swarm applications, including micro quadrotors [15] and flapping-wing micro aerial vehicles (MAVs) [27]. At even smaller scales, advances in MEMS, low-power VLSI, and nanotechnology are facilitating the development of sub-millimeter self-powered robots [25].

Many potential swarm applications will require the self-organization of robots into groups of different sizes around various regions or objects in their environment (see Fig. 1). For instance, a swarm may be tasked to transport multiple payloads that are each too heavy for a single robot to retrieve, which would necessitate that enough robots aggregate around each load to move it to a target destination, possibly at a desired speed. However, simply allocating robots to saturation on each payload may be inefficient. Similarly, surveillance tasks may require a swarm to surround different types of regions, such as structure perimeters, to achieve particular degrees of sensor coverage. Other possible applications include environmental monitoring and mapping, automated construction and manufacturing, and disaster response tasks such as cordoning off a hazardous area or extinguishing a fire. At the micro- and nano-scale, applications include micro-object manipulation, microfactories and nanofactories, and medical monitoring, diagnosis, and treatment. For example, nano-scale robots

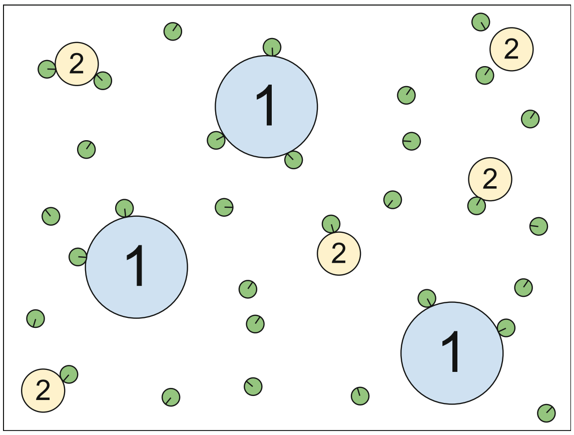


Fig. 1 Example scenario with two types of *disk-shaped* regions, labeled 1 and 2. The unlabeled *circles* are robots that are allocating themselves to the region boundaries

could collect in desired proportions around objects that are transparent to macroscopic sensing technologies. If the proportions of nano-scale robots were detectable, then the presence of the objects could be inferred.

In order for robotic swarms to reliably carry out these tasks, a rigorous methodology is needed for synthesizing individual robot behaviors that provably result in target robot allocations around boundaries. The swarm control framework must be *scalable* to arbitrary robot population sizes and accommodate possibly extreme *limitations* on each robot's sensing, communication, and computation abilities. It must also account for *stochasticity* arising from noise due to sensor and actuator errors; inherent randomness in robot encounters with each other and with environmental features; and, for nanorobots, the effects of Brownian motion and chemical interactions at scales below tens of micrometers [9].

In this work, we present our first efforts toward developing a control framework with the aforementioned properties for the problem of allocating a robotic swarm in target group sizes around the boundaries of disjoint, stationary regions of different types. For simplicity, we consider only disk-shaped regions here, which we refer to as *disks*, but this work can be extended to other region shapes. The robots have no prior information about the disks and they use only local sensing and local communication, encountering the disks during the course of random walks. Disk *types* may be categorized according to physical or subjective properties; for instance, size or weight if the disks are payloads to be transported, or relative surveillance value if they are areas to be monitored. Figure 1 depicts an example scenario in which the objective is to attain an average allocation of three robots per type-1 disk and one robot per type-2 disk. Stochastic binding and unbinding behaviors of the robots will result in fluctuations around these target allocations, as illustrated by the variation in number of robots bound to each disk type.

We employ a top–down approach to synthesizing robot control policies that produce target allocations among the disks with probabilistic guarantees on performance. We represent the stochastic robot interactions with disks and with each other as a well-mixed chemical reaction network (CRN). The robot-to-robot interactions consist of an enzyme-inspired behavior, implemented at the disk boundaries, that greatly reduces the dependence of the allocation strategy on the *encounter rates*, the difficult-to-characterize probabilities per unit time that a robot encounters an occupied or unoccupied section of a disk boundary. This behavior also decouples allocation tasks that may be occurring in parallel. The CRN formulation allows us to abstract the system to a *macroscopic* population model, a set of ordinary differential equations (ODEs) that is amenable to analysis and control. Our approach can be implemented by using a supervisory agent to design the model parameters for a particular global objective and broadcast them to the robots. The robots use these parameters to define their stochastic decision-making policies, and the resulting collective behavior follows the macroscopic prediction in expectation.

Through agent-based NetLogo [26] simulations[1] of a *microscopic* model of robots interacting with disks and other robots, we have validated that the simulated system retains all of the qualitative features of the predictions of the macroscopic model and is robust to environmental parameter variations. That is, a single control strategy can be implemented based on the geometric properties of a single robot and a single disk, and the equilibrium occupancy levels of robots around disks will be invariant to changes in the total numbers of robots, disks, and disk types, as well as robust to changes in robot speed (e.g., due to battery decay). Hence, the control strategy need not be re-tuned if the environment around the robot changes over time.

1.1 Related Work

Similar to our swarm allocation strategy, much existing work on understanding and controlling robotic swarm behaviors relies on developing an accurate macroscopic model of the population dynamics. The model's dimensionality is independent of the swarm size, which facilitates quick simulation and a scalable control approach. Previous work has addressed stochastic approaches to swarm robotic task allocation, in which the robot task-switching rates are optimized using non-spatial macroscopic models that describe the time evolution of the robot population in each state [1, 6, 16, 18, 23]. Non-spatial swarm models have also been used to optimize stochastic robot behaviors in problems of robotic assembly of parts into products and self-assembly via binding through random collisions [10, 14, 19, 22]. Spatial macroscopic models of swarms that describe robot deterministic and random motion in addition to stochastic task switching have also been developed recently [8, 12, 24].

The utility of the macroscopic model in these works hinges on the ability to accurately determine the non-tunable components of the model parameters. Specifically, in applications where the model captures random interactions between entities in the system, these components are the corresponding encounter rates. However, encounter rates can often be determined only through simulation [11, 13] because the robot motion pattern and sensor footprint and the environment configuration can induce unpredictable spatially dependent effects, or simply *spatial effects*, on the frequency of robot encounters with other system entities. Encounter-rate formulas based on geometric parameters have been used in previous work on macroscopic swarm modeling of systems in which robots encounter objects that are small [17] or large (and elongated) [7] relative to them. However, these formulas can be applied only for environments with low densities of robots and objects in which robots are uniformly spatially distributed at all times, which is ensured when robots execute random walks and the objects do not bias the robots' movements. Our scenario violates these assumptions in that the encountered objects are adjacent to one another and thus not distributed at low density. The implausibility of deriving analytical solu-

[1]To obtain the code for the simulations presented in this paper, contact Dr. Theodore Pavlic (email tpavlic@asu.edu).

tions of encounter rates for our scenario motivated us to find a way of controlling the swarm without knowledge of these rates while still using a macroscopic model.

2 Robot Controller

We assume that each robot has a small sensing and communication radius. Even when communication is possible, it may be difficult for a robot to identify the location of the source of a message. Consequently, we propose a control strategy that achieves a desired average allocation around each type of disk using only local robot–robot and robot–disk interactions. The controller for each robot, shown in Fig. 2, incorporates:

Robot movement: Robots move according to a *correlated random walk* (CRW) in order to achieve approximately uniform distributions throughout empty space. That is, each robot moves straight ahead in a short segment and then turns to a random angle before repeating. If this assumption of spatial homogeneity is violated, then different regions of space may approach equilibrium at faster rates than others. However, the limiting average allocations will be robust to inhomogeneity of robot density.

Robot–disk interactions: Each robot can identify the type of disk that it encounters. The robot then chooses to bind to that disk with probability p_b that depends on disk type; otherwise, the robot ignores the encounter and continues its CRW.

Robot–robot interactions: Upon encountering another robot, a robot can identify whether it is an unbound robot or a robot that is bound to a disk. If it encounters an unbound robot, the robot executes a collision avoidance maneuver. If it encounters a bound robot, the robot chooses to command that robot to unbind based on a probability p_u that depends on the disk type; otherwise, the robot ignores the encounter and continues its CRW.

We do not specify a behavior in which robots unbind spontaneously at a certain probability rate. Robots either probabilistically choose to bind to encountered disks

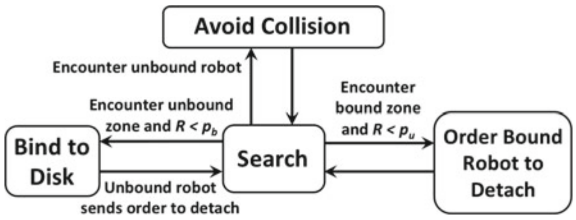

Avoid Collision
Encounter unbound robot
Encounter unbound zone and $R < p_b$
Encounter bound zone and $R < p_u$
Bind to Disk
Search
Order Bound Robot to Detach
Unbound robot sends order to detach

Fig. 2 Diagram of control flow. Robots randomly cycle through searching and encountering robots and unbound zones. On encountering those disk regions, they probabilistically choose to bind to unbound regions or tell bound robots to unbind. On encountering unbound robots, they avoid collisions. Probabilities can be implemented with a pseudo-random number $R \in \text{unif}(0, 1)$

or stochastically command other robots to unbind from disks. If a bound robot is never encountered by a free robot, it will never unbind from the disk.

3 Microscopic Model: Enzymatic Chemical Reaction Network

The robot–disk system described in Sect. 2 resembles a gas made up of species of *free robots* and disk zones that are either *bound* or *unbound*. For simplicity, we only consider one disk type here; however, we will show how these results naturally extend to an arbitrary number of disk types. The corresponding well-mixed chemical reaction network (CRN) for the single-type case is:

$$r + U \xrightarrow{p_b e_u} B \tag{1a}$$

$$r + B \xrightarrow{p_u e_b} U + 2r \tag{1b}$$

where r represents the free-robot species, B represents bound zones, and U represents unbound zones. The mass-action rate constants $p_b e_u$ and $p_u e_b$ include:

e_u: The probability per unit time that a single free robot will encounter a single unbound zone. This encounter rate is an environmental parameter.
e_b: The probability per unit time that a single free robot will encounter a single bound zone. This encounter rate is an environmental parameter.
p_b: The probability that a free robot will bind to an unbound zone given that it has just encountered it. This parameter is under the control of the designer.
p_u: The probability that a free robot will command a bound robot to unbind given that the free robot has just encountered the bound zone. This parameter is under the control of the designer.

Reverse reactions are necessary to stabilize unique non-trivial equilibria. Without a reverse reaction, the reaction in Eq. (1a) would cause the system to reach trivial saturation of bound zones. In other examples in stochastic robotics (e.g. [1–3, 14, 19, 21]), event-driven forward reactions like Eq. (1a) are accompanied with delay-driven reverse reactions—robots have a tendency to *decay* back into earlier behavioral modes. Instead of implementing the reverse reactions as decay processes, we implement the reverse direction with the event-driven enzymatic reaction in Eq. (1b). That is, the free robot that encounters the bound zone in Eq. (1b) is not consumed by the reaction; it is analogous to an enzyme which rapidly increases the decay rate of bound zones. As we will show, because both reactions are event driven, the expected equilibrium distributions will vary with the ratio e_b/e_u as opposed to the absolute encounter rates. In general, the absolute encounter rates e_b and e_u will change with robot density, total number of zones, and robot speed (which itself can change over time with battery fatigue). However, the ratio e_b/e_u, and thus the equilibrium distribution, will be invariant to these changes.

4 Macroscopic Model: Concentration Fields of Zones and Robots

From the theory of mass-action kinetics in well-mixed gases, a smooth concentration-field approximation of the CRN in Eq. (1) for large populations is the multi-affine system

$$\begin{cases} \dot{r} = p_u e_b r B - p_b e_u r U \\ \dot{U} = p_u e_b r B - p_b e_u r U \\ \dot{B} = p_b e_u r U - p_u e_b r B \end{cases} \tag{2}$$

where r, U, and B represent the number of free robots, unbound zones, and bound zones, respectively, for a given arena (i.e., concentrations in a fixed volume size). The system clearly has a continuum of trivial equilibria characterized by $r = 0$, which represents the total depletion of free robots. Moreover, because this system is continuous, the set $\{r : r > 0\}$ is positively invariant; if the initial concentration of free robots is positive, then the concentration will remain positive indefinitely. Thus, assuming non-zero mass-action rate constants, there is an additional equilibrium $(r, U, B) = (r^*, U^*, B^*)$ where $r^* > 0$ and

$$\frac{B^*}{U^*} = \frac{p_b e_u}{p_u e_b} \quad \text{or, equivalently,} \quad \frac{B^*}{B^* + U^*} = \frac{p_b e_u}{p_b e_u + p_u e_b} = \frac{\frac{p_b}{p_u}}{\frac{p_b}{p_u} + \frac{e_b}{e_u}}. \tag{3}$$

Let B_0, U_0, and r_0 represent the initial number of bound zones, unbound zones, and free robots, respectively. Noting that $\dot{U} = \dot{r}$ and $\dot{B} = -\dot{r}$, it must be the case that $B^* + U^* = B_0 + U_0$. Additionally, the third-order system in Eq. (2) can be re-written as a first-order differential equation

$$\begin{aligned} \dot{r} &= p_u e_b r(\overbrace{B_0 - (r - r_0)}^{B}) - p_b e_u r(\overbrace{U_0 + (r - r_0)}^{U}) \\ &= ((B_0 + r_0) p_u e_b - (U_0 - r_0) p_b e_u) r - (p_b e_u + p_u e_b) r^2 \end{aligned} \tag{4}$$

representing the dynamics of number of free robots. Under the assumption of non-zero mass-action rate constants, the non-trivial equilibrium at $r = r^* > 0$ is such that

$$r^* = (B_0 + r_0) \frac{p_u e_b}{p_b e_u + p_u e_b} - (U_0 - r_0) \frac{p_b e_u}{p_b e_u + p_u e_b}.$$

So, by Eq. (3) and because $B^* + U^* = B_0 + U_0$,

$$r^* = (B_0 + r_0) \left(1 - \frac{B^*}{B_0 + U_0}\right) - (U_0 - r_0) \frac{B^*}{B_0 + U_0}$$

which is positive and asymptotically stable so long as $B_0 + r_0 > B^*$. That is, so long as the total number of free and bound robots $r_0 + B_0$ is larger than the predicted equilibrium number of bound robots B^*, the (r^*, U^*, B^*) equilibrium will be asymptotically stable with $r^* > 0$. In other words, from Eq. (3) and the condition that $B_0 + r_0 > B^*$, the system in Eq. (1) has an asymptotically stable equilibrium described by

$$(r, B, U) = \begin{cases} (r^*, B^*, U^*) & \text{if } r_0 + B_0 > B^*, \\ (0, U_0 - r_0, B_0 + r_0) & \text{otherwise.} \end{cases} \tag{5}$$

So the system is driven by the imbalance between fluxes to and from bound and unbound zones; it comes to rest when enough free robots are converted into bound zones to restore flux balance or when the pool of free robots is totally depleted.

4.1 High-Population Linear Approximation

Due to its quadratic structure, Eq. (4) can be solved explicitly. For any $t > 0$,

$$r(t) = r^* \frac{r_0}{r_0 + (r^* - r_0)\exp(-r^*(p_b e_u + p_u e_b)t)},$$

which, for $r_0 \gg 0$, is essentially constant. That is, $r(t) \approx r^* \approx r_0$ for $r_0 \gg 0$. Consequently, for $r_0 \gg 0$, the multi-affine system in Eq. (2) that approximates the bimolecular CRN in Eq. (1) can be viewed as the linear system

$$\begin{cases} \dot{U} = p_u e_b r_0 B - p_b e_u r_0 U \\ \dot{B} = p_b e_u r_0 U - p_u e_b r_0 B \end{cases} \quad \text{that models the } \textit{unimolecular} \quad \begin{array}{c} U \xrightarrow{p_b e_u r_0} B \\ B \xrightarrow{p_u e_b r_0} U \end{array}. \tag{6}$$

In this $r_0 \gg 0$ regime, the number of free robots scales the per-zone encounter rates. Moreover, although the linear system in Eq. (6) has an equilibrium in Eq. (3) that is independent of r_0, the time constant of the system is

$$\frac{1}{(p_b e_u + p_u e_b) r_0}. \tag{7}$$

Thus, increasing r_0 increases the total speed of the system but has no impact on the equilibrium allocation of robots to disks.

4.2 Multiple Disk Types: Decoupled Analysis and Control

For any number of zones, there is some sufficiently large initial number of free robots r_0 that satisfies the condition that $B_0 + r_0 > B^*$ and thus guarantees the stability of a non-trivial equilibrium zone concentration described by Eq. (3), which is invariant to changes in r_0. So if the pool of free robots is sufficiently large, the equilibrium analysis of a system with multiple disk types can be performed independently for each disk type.

For example, if there are n disk types and the number of free robots r_0 is initially greater than the total number of unbound zones across all disk types, then the concentration of bound zones B^i and unbound zones U^i on disk type $i \in \{1, 2, \ldots, n\}$ at equilibrium is such that $B^i/U^i = (e^i_u/e^i_b)(p^i_b/p^i_u)$ where e^i_u, p^i_b, e^i_b, and p^i_u are the encounter rates and reaction probabilities specialized for type i. That is, the equilibrium analysis for any type is decoupled from the analysis of any other type.

Although the multiple types have a coupled effect on convergence rate and transient dynamics in general, the equilibrium allocations can be predicted in isolation. So for the remainder of this paper, we assume a sufficiently large pool of robots to meet subjective convergence time constraints for an arbitrary number of disk types. Moreover, we will only explicitly discuss design for a single disk type; it is implied that the process is identical for multiple coexisting types.

4.3 Corrections for Spatial Effects on Boundaries

In principle, robots can be organized around a disk to reach 100 % allocation (i.e., $B/(B + U) = 1$). However, in practice, it is likely that two robots interacting stochastically with a disk will bind with non-zero inter-robot space between them that is nevertheless too small for another robot to encounter. So although the amount of unbound space on a disk may be large, the actual number of unbound zones available for additional binding may be small. Thus, the maximum value of $B/(B + U)$ will be less than unity; even a $(p_b, p_u) = (1, 0)$ policy will saturate with free space remaining on disks. Moreover, even well before saturation, some amount of free space will be inaccessible for free robots to bind to because it will be too close to existing bound robots. Thus, a theory is needed to model the non-linear reduction in remaining unbound space as robots bind to disks.

In the following, assume that all linear distances are given in units of the arc length occupied by a robot when bound to a disk. That is, each robot binds to 1 unit of arc length, and so the theoretical maximum number of robots bound to a disk is equal to the disk's circumference. However, because robots are not equipped with the ability to cluster together, the actual maximum number of bound robots will be much lower. Consider:

- A disk with a single robot bound to it. An incoming unbound robot will not be able to discover disk space adjacent to a bound robot unless its center is at least 0.5 units away from the edge of the bound robot. So the bound robot effectively occupies both its own 1 unit of arc space plus an additional 1 unit of arc length adjacent to it. Additionally, if the incoming unbound robot maintains a distance a between itself and every other robot (e.g., to avoid collisions), then the additional space occupied by the bound robot increases to $\delta_{\max} \triangleq 1 + 2a$ units because there are $0.5 + a$ units of additional occupation on both sides of the bound robot.

- A disk with many robots bound to it. If two robots have less than $1 + 2a$ unbound arc length between them, an incoming unbound robot will not be able to discover it. However, these small distances can be no smaller than the avoidance distance $\delta_{\min} \triangleq a$.

With this in mind, we partition the space between robots into sections no longer than $\delta_{\max} \triangleq 1 + 2a$, as shown in Fig. 3. We then define the quantity δ to be the mean size of the partitioned inter-robot spaces. Thus, although truncation of an inter-robot space that is only slightly larger than $1 + 2a$ can create a truncated space smaller than a, the mean δ is bounded above and below such that

$$\delta_{\min} = a \leq \delta \leq 1 + 2a = \delta_{\max}.$$

The statistic δ is actually a function $\delta : [0, 1] \to [a, 1 + 2a]$ that maps an allocation ratio $B/(U + B)$ to the mean additional arc occupancy per bound zone $\delta(B/(U + B))$, which we abbreviate to δ here for convenience. At low allocation ratios, $\delta \approx 1 + 2a$ because the space between bound robots is large. Consequently, there will be more encounters with bound zones and fewer encounters with unbound zones than otherwise expected. Similarly, at high allocation ratios, $\delta \approx a$. So although bound zones are still magnified, this magnification decreases with added allocation ratio.

To model the effective increase in B by δB and the corresponding effective decrease in U by δB, we apply the substitution $B^* \mapsto (1 + \delta)B^*$ and $U^* \mapsto U^* - \delta B^*$ to the

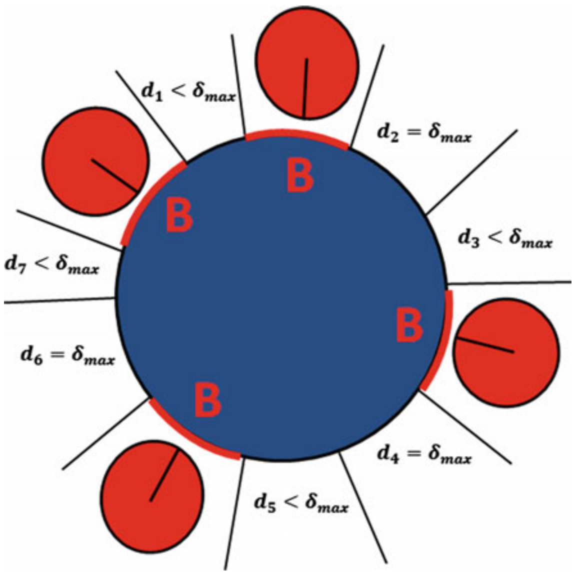


Fig. 3 Partitioning of space between robots. Here, four robots are shown connected to a single disk. Each robot with its corresponding disk sector, shown with a "B", constitutes a bound zone. The four spaces between each pair of robots have been partitioned into smaller spaces no larger than $\delta_{\max} = 1 + 2a$, which represents the maximum additional arc length that a bound robot can interfere with due to spatial effects

equilibrium condition in Eq. (3). Consequently, the actual (B^*, U^*) equilibrium will be such that

$$\overbrace{\frac{(1+\delta)}{1-\delta\frac{B^*}{U^*}}}^{\text{Correction factor}}\frac{B^*}{U^*}=\frac{e_u p_b}{e_b p_u} \tag{8}$$

where the overbraced expression is a correction factor for the spatial effects in a physical robot scenario. For comparison, the corrected allocation can be related to the idealized allocation by

$$\frac{B^*}{U^*+B^*}=\frac{B^*}{(U^*-\delta B^*)+(1+\delta)B^*}=\frac{\frac{B^*}{(1+\delta)B^*}}{\frac{U^*-\delta B^*}{(1+\delta)B^*}+1}=\frac{1}{1+\delta}\overbrace{\frac{\frac{p_b}{p_u}}{\frac{p_b}{p_u}+\frac{e_b}{e_u}}}^{\text{Idealized allocation}} \tag{9}$$

where the overbraced expression matches the idealized allocation ratio in Eq. (3). So the ideal and actual allocations are predicted to be related by a $1/(1+\delta)$ gain. For low allocations, this gain will be $1/(1+2a)$; for high allocations, this gain will be determined by the saturated value of $\delta > \delta_{\min} = a$.

4.3.1 Shape of δ Function

An important future direction is to develop theory to predict the precise shape of the δ function. However, as we will discuss later, we have experimental evidence that δ is only determined by the value of a. Moreover, δ appears to be a cosine of the form $\delta(r) = A\cos(2\pi(r/T)+c)$ that pierces $\delta(0)\approx 1+2a$ and $\delta(1/(1+a)) = a$ subject to the constraints $A \geq (1+a)/2$, $T \geq 2/(1+a)$, and $c \geq 0$.

5 Control of Equilibrium Allocations

From the equilibrium described by Eq. (8), a (p_b, p_u) control policy can be synthesized using the rule

$$\frac{p_b}{p_u}=\frac{e_b}{e_u}\frac{B^*}{U^*}\frac{(1+\delta)}{1-\delta\frac{B^*}{U^*}} \tag{10}$$

where B^*/U^* is the desired bound–unbound allocation ratio of zones at equilibrium. Equivalently, if the desired robot-to-boundary-space *allocation ratio* is $B^*/(B^*+U^*)$, then (p_b, p_u) should chosen according to Eq. (9). Thus, for any given allocation ratio, there is a continuum of (p_b, p_u) pairs that will achieve the desired equilibrium.

The control policy in Eq. (10) has one degree of freedom over which some feature of the system can be optimized. For example, by Eq. (7), the convergence rate of the system can be maximized by making the sum $p_b + p_u$ as large as possible. So, for fastest convergence for to a desired allocation (B^*, U^*), p_b and p_u can be chosen so that

$$(p_b, p_u) = \begin{cases} \left(\frac{e_b}{e_u} \frac{B^*}{U^*} \frac{(1+\delta)}{1-\delta B^*/U^*}, 1 \right) & \text{if } e_b B^*(1+\delta) < e_u U^*(1-\delta B^*/U^*), \\ \left(1, \frac{e_u}{e_b} \frac{U^*}{B^*} \frac{1-\delta B^*/U^*}{(1+\delta)} \right) & \text{otherwise.} \end{cases} \tag{11}$$

However, optimization criteria other than maximal convergence rate may suggest other choices of (p_b, p_u). For example, there will be fewer temporal variations in the number of robots bound to each disk if $p_b + p_u$ is reduced. Similarly, the variance in allocation across disks may be reduced for certain (p_b, p_u) combinations. Furthermore, if the e_b/e_u ratio can be artificially shifted (e.g., by asymmetrically changing the relative distance that sensors react to bound and unbound zones) or the avoidance distance a changed, it is possible to shift the p_b/p_u control policy for a desired B^*/U^* allocation ratio. Thus, there are mechanisms that can further adjust the $p_u + p_b$ sum without changing the equilibrium allocation ratio.

6 Model Validation in Simulation

To test our macroscopic model of stochastic allocation to circular boundaries, experimental trials were conducted using NetLogo [26]. The framework allowed for simulating hundreds of mobile robots randomly interacting with each other and with disks of different types.

6.1 Variations Due to Encounter-Rate Ratio

For many robot motion behaviors, the encounter-rate ratio e_b/e_u may be approximated, for example, by dividing the sum of the areas of a robot and an unbound zone sector by the area of an unbound zone sector alone, where zone sectors are slices of each disk with arcs that are the length of the interaction region with the robot. In general, it can be estimated from equilibrium allocation data. If, for example, it is incorrectly assumed that the e_b/e_u ratio is unity, the equilibrium allocation will shift in a predictable way based on the correct e_b/e_u ratio, as shown in Eq. (4). Consequently, if the e_b/e_u ratio is not well known, it can be estimated by measuring this curve during system testing. Inferring this encounter-rate *ratio* is empirically much simpler than inferring the actual encounter rates. Also shown in Fig. 4 is the effect of the inter-robot space $\delta(1.0)$ being both non-zero and yet smaller than required to fit any additional robots. Thus, disks saturate at a level less than full occupancy.

To validate these predictions, experimental trials were conducted using 500 simulated mobile robots moving along correlated random walks in a space with 6 disks with circumference capacity for 28.27 robots per disk. By increasing the so-called

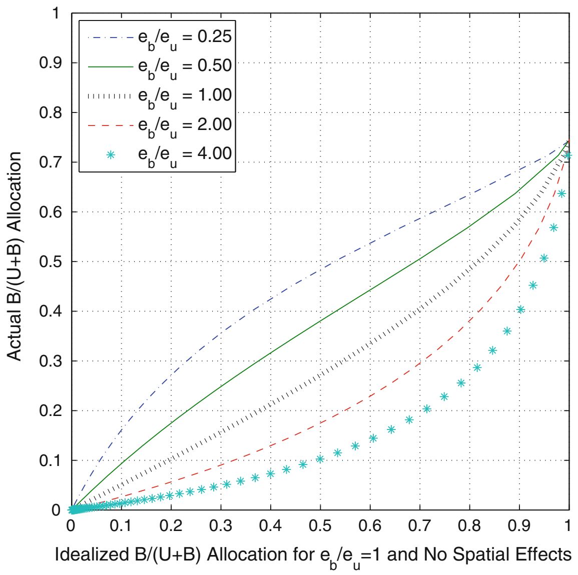


Fig. 4 Effect of encounter ratio. For each e_b/e_u ratio, a plot comparing the idealized allocation ratio to the actual allocation ratio is shown according to Eq. (9). Here, $\delta(r) = \cos(2\pi(r/4.6)+0.2)$, which is consistent with an $a = 0$ case. Allocations saturate near an actual ratio of 0.75 because the mean slack space $\delta(1)$ is both non-zero and too small to accommodate additional binding

turning angle of the CRW, the robot motion became less directional and more Brownian. Consequently, robots with higher turning angle are more likely to re-encounter a disk and re-bind immediately after being told to unbind. This decrease in unbinding efficacy decreases the effective e_b/e_u ratio, as shown in Fig. 5 which matches Fig. 4 for different e_b/e_u ratios.

6.1.1 Estimation of δ Function

The δ function used in Figs. 4 and 5 is based on an avoidance range of $a = 0$ and a circumference of 28.27 robot widths. As shown in Fig. 6, this δ fits mean data from simulated scenarios regardless of CRW parameters and effective encounter-rate ratio. The empty-occupancy $\delta(0) < 1 + 2a$ because the circumference does not divide evenly by $1 + 2a$. That is, the partitioned space of the empty disk includes a residual sub-unity partition, and so the mean across those partitions is less than 1.

6.2 *Robustness to Environmental Variations*

Empirical studies show that the relationship between idealized and actual allocation is not sensitive to environmental variations. For example, Fig. 7 shows statistics taken from simulation runs with several combinations of robot population, robot size, number of disks, and disk size. As shown, varying the size and number of disks

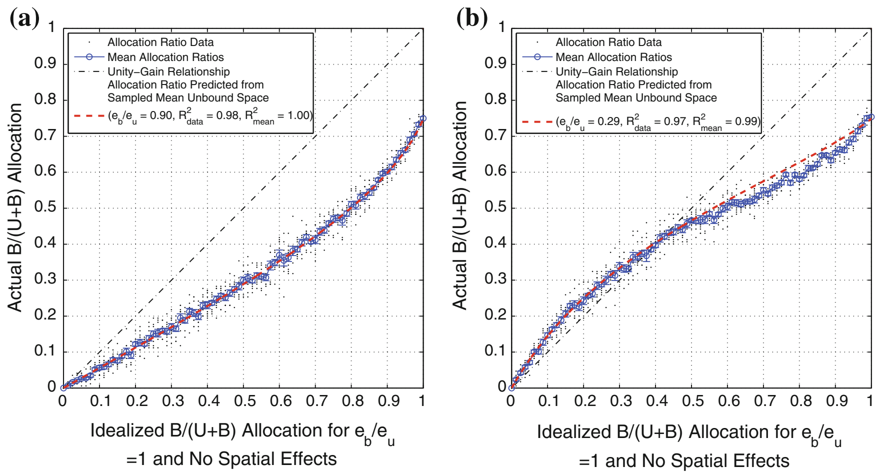


Fig. 5 Effect of encounter ratio in simulation. For each e_b/e_u ratio, a plot comparing the idealized allocation ratio to the actual allocation ratio is shown according to Eq. (9). The particular e_b/e_u ratios corresponding to the two motion primitives (i.e., low and high turning angle) were fit to the observed data. Additionally, $\delta(r) = \cos(2\pi r/4.6 + 0.2)$, which is consistent with predictions from an $a = 0$ case with a disk circumference of 28.27 robot widths. Allocations saturate near an actual ratio of 0.75 because the mean slack space $\delta(1)$ is both non-zero and too small to accommodate additional binding. The slight deviations from prediction in (b) can be improved with better understanding of the derivation of the δ function. Small *dots* show outcomes of individual simulation runs. Open *circles* show means across ten trials of each allocation ratio. Error bars show ± 1 standard error of the mean (SEM). Each trial uses 500 simulated robots and 6 disks. **a** Low CRW turning angle ($e_b/e_u \approx 0.9$). **b** High CRW turning angle ($e_b/e_u \approx 0.29$)

and robots does not change the actual allocation ratio. A single control strategy leads to the same equilibrium mean allocation ratio in every case.

7 Conclusions and Future Work

We have presented a rigorous methodology for designing control policies that distribute a swarm of robots among a set of boundaries. The control policies rely only on local information obtained by the robots and have probabilistic guarantees on steady-state performance. We investigated the effect of robot interactions on the actual steady-state allocations that were achieved with the use of control policies derived from a coarse-grained description of the swarm population. Our simulation results illustrate that the actual system behavior follows predictable and controllable trends when robot interactions at the boundaries and in the free space are introduced. In this way, we build a foundation for further work on encounter-based swarm robotic allocation that obviates an analytical characterization of encounter rates.

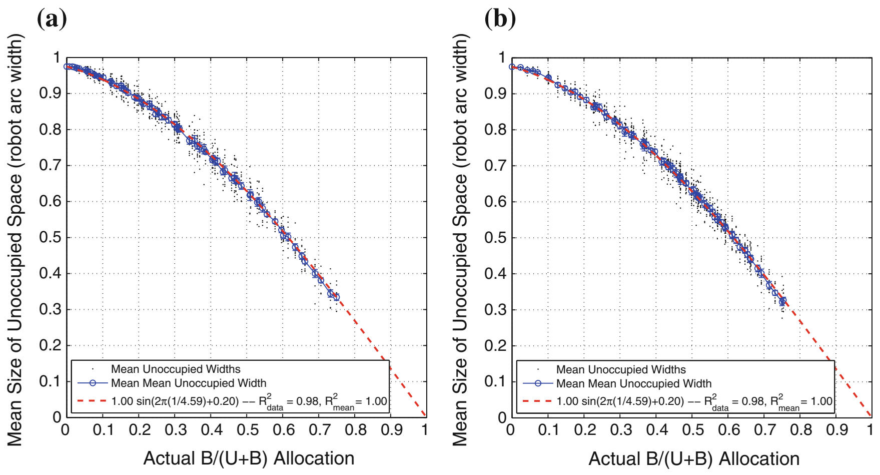


Fig. 6 Simulated effect of encounter ratio on δ. The single δ function from Fig. 4 accurately predicts mean inter-robot space across different encounter ratios and motion primitives. Each small *dot* shows a result from an individual simulation run. Open *circles* show means for different allocations. Error bars show SEM. Ten trials were run per allocation ratio. **a** Low CRW turning angle. **b** High CRW turning angle

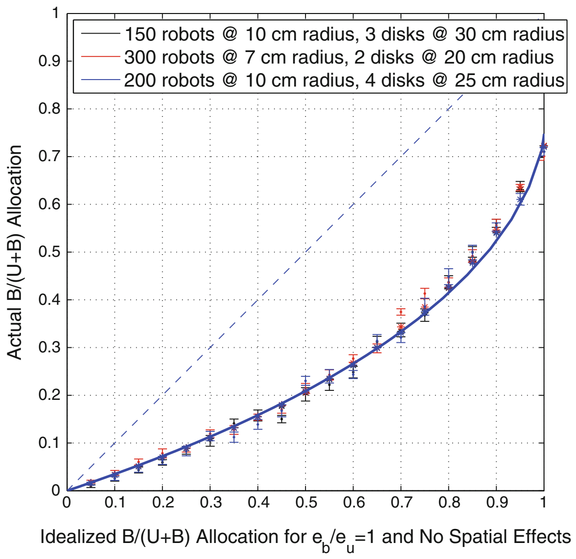


Fig. 7 Effect of varying environmental parameters. Ten trials were generated for each disk size, and the average across the trials are shown with error bars indicating ±1 standard error of the mean. A *dashed line* of unity slope is shown for reference. The *solid line* represents the predicted *curve* based on the avoidance distance *a*, which is non-zero for these cases

In future work, we will repeat these investigations for more arbitrary shapes. We also plan to gain a better understanding of the relationship between actual allocation ratio and the mean space between robots. Leveraging the several degrees of freedom in the enzymatic swarms (i.e., binding and unbinding probabilities, robot geometry, avoidance distances, and motion primitives that shape encounter-rate ratios), we will explore the design of optimal robot control policies to achieve convergence to desired allocations subject to other constraints, such as ensuring minimal allocation variance or settling within a specified time. We will extend our control approach to other scenarios in which robotic swarms must allocate among both static and dynamically moving regions, such as surveillance and target-tracking applications, and we will investigate how this approach can simplify other stochastic strategies such as swarm self assembly. Finally, we plan to experimentally validate our approach on a physical multi-robot testbed.

Acknowledgments This work was supported in part by the National Science Foundation (grant no. CCF-1012029).

References

1. Berman, S., Halász, Á., Hsieh, M.A., Kumar, V.: Optimized stochastic policies for task allocation in swarms of robots. IEEE Trans. Robot. **25**(4), 927–937 (2009). doi:10.1109/TRO.2009.2024997
2. Berman, S., Kumar, V., Nagpal, R.: Design of control policies for spatially inhomogeneous robot swarms with application to commercial pollination. In: Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China (2011)
3. Berman, S., Nagpal, R., Halász, Á.: Optimization of stochastic strategies for spatially inhomogeneous robot swarms: a case study in commercial pollination. In: Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Francisco, pp 3923–3930 (2011). doi:10.1109/IROS.2011.6094771
4. Bonani, M., Longchamp, V., Magnenat, S., Rétornaz, P., Burnier, D., Roulet, G., Vaussard, F., Bleuler, H., Mondada, F.: The marXbot, a miniature mobile robot opening new perspectives for the collective-robotic research. In: Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei, pp. 4187–4193 (2010). doi:10.1109/IROS.2010.5649153
5. Brambilla, M., Ferrante, E., Birattari, M., Dorigo, M.: Swarm robotics: a review from the swarm engineering perspective. Swarm Intell. **7**(1), 1–41 (2013). doi:10.1007/s11721-012-0075-2
6. Correll, N.: Parameter estimation and optimal control of swarm-robotic systems: a case study in distributed task allocation. In: Proceedings of the 2008 IEEE International Conference on Robotics and Automation, Pasadena, pp 3302–3307, (2008). doi:10.1109/ROBOT.2008.4543714
7. Correll, N., Martinoli, A.: Modeling and optimization of a swarm-intelligent inspection system. In: Proceedings of the Seventh International Symposium on Distributed Autonomous Robotics Systems (DARS 2004), Toulouse, France, pp. 369–378, (2004). doi:10.1007/978-4-431-35873-2_36
8. Dantu, K., Berman, S., Kate, B., Nagpal, R.: A comparison of deterministic and stochastic approaches for allocating spatially dependent tasks in micro-aerial vehicle collectives. In: Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal, pp. 793–800, (2012). doi:10.1109/IROS.2012.6386233
9. Diller, E., Sitti, M.: Micro-scale mobile robotics. Found. Trends Robot. **2**, 143–259 (2013)
10. Evans, W.C., Mermoud, G., Martinoli, A.: Comparing and modeling distributed control strategies for miniature self-assembling robots. In: Proceedings of the 2010 IEEE International Conference on Robotics and Automation, pp. 1438–1445. Anchorage, AK (2010)

11. Gurarie, E.: Models and analysis of animal movements: From individual tracks to mass dispersal. Ph.D. thesis, University of Washington (2008)
12. Hamann, H., Wörn, H.: A framework of space-time continuous models for algorithm design in swarm robotics. Swarm Intell. **2**(2–4), 209–239 (2008). doi:10.1007/s11721-008-0015-3
13. Hutchinson, J.M.C., Waser, P.M.: Use, misuse and extensions of "ideal gas" models of animal encounter. Biol. Rev. **82**(3), 335–359 (2007). doi:10.1111/j.1469-185X.2007.00014.x
14. Klavins, E., Burden, S., Napp, N.: Optimal rules for programmed stochastic self-assembly. In: Proceedings of Robotics: Science and Systems II, Philadelphia (2006)
15. Kushleyev, A., Kumar, V., Mellinger, D.: Towards a swarm of agile micro quadrotors. In: Proceedings of Robotics: Science and Systems VIII, Sydney, NSW, Australia (2012)
16. Liu, W., Winfield, A.F.T.: Modeling and optimization of adaptive foraging in swarm robotic systems. Int. J. Robot. Res. **29**(14), 1743–1760 (2010). doi:10.1177/0278364910375139
17. Martinoli, A., Easton, K., Agassounon, W.: Modeling swarm robotic systems: a case study in collaborative distributed manipulation. Int. J. Robot. Res. **23**(4–5), 415–436 (2004). doi:10.1177/0278364904042197
18. Mather, T.W., Hsieh, M.A.: Distributed robot ensemble control for deployment to multiple sites. In: Proceedings of Robotics: Science and Systems VII, Los Angeles, CA, USA (2011)
19. Matthey, L., Berman, S., Kumar, V.: Stochastic strategies for a swarm robotic assembly system. In: Proceedings of the 2009 IEEE International Conference on Robotics and Automation, pp. 1953–1958. Kobe, Japan (2009)
20. McLurkin, J., Rykowski, J., John, M., Kaseman, Q., Lynch, A.J.: Using multi-robot systems for engineering education: teaching and outreach with large numbers of an advanced, low-cost robot. IEEE Trans. Educ. **56**(1), 24–33 (2013). doi:10.1109/TE.2012.2222646
21. Napp, N., Klavins, E.: A compositional framework for programming stochastically interacting robots. Int. J. Robot. Res. [Special Issue Stochasticity in Robot Bio-Systems Part 2] **30**(6), pp. 713–729 (2011). doi:10.1177/0278364911403018
22. Napp, N., Burden, S., Klavins, E.: Setpoint regulation for stochastically interacting robots. In: Proceedings of Robotics: Science and Systems V, Seattle, WA, USA (2009)
23. Odhner, L.U., Asada, H.: Stochastic recruitment control of large ensemble systems with limited feedback. J. Dyn. Syst. Meas. Control **132**(4), 041008 (2010). doi:10.1115/1.4001706
24. Prorok, A., Correll, N., Martinoli, A.: Multi-level spatial modeling for stochastic distributed robotic systems. Int. J. Robot. Res. **30**(5), 574–589 (2011). doi:10.1177/0278364910399521
25. Robotics Virtual Organization (Robotics VO) (2013) A Roadmap for U.S. Robotics: From Internet to Robotics (2013) Edition. http://robotics-vo.us/sites/default/files/2013%20Robotics%20Roadmap-rs.pdf
26. Wilensky, U.: NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, USA, (1999). http://ccl.northwestern.edu/netlogo/
27. Wood, R.J., Finio, B., Karpelson, M., Ma, K., Pérez-Arancibia, N.O., Sreetharan, P.S., Tanaka, H., Whitney, J.P.: Progress on 'pico' air vehicles. Int. J. Robot. Res. **31**(11), 1292–1302 (2012). doi:10.1177/0278364912455073

Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments

Charles Richter, Adam Bry and Nicholas Roy

Abstract We explore the challenges of planning trajectories for quadrotors through cluttered indoor environments. We extend the existing work on polynomial trajectory generation by presenting a method of jointly optimizing polynomial path segments in an unconstrained quadratic program that is numerically stable for high-order polynomials and large numbers of segments, and is easily formulated for efficient sparse computation. We also present a technique for automatically selecting the amount of time allocated to each segment, and hence the quadrotor speeds along the path, as a function of a single parameter determining aggressiveness, subject to actuator constraints. The use of polynomial trajectories, coupled with the differentially flat representation of the quadrotor, eliminates the need for computationally intensive sampling and simulation in the high dimensional state space of the vehicle during motion planning. Our approach generates high-quality trajecrtories much faster than purely sampling-based optimal kinodynamic planning methods, but sacrifices the guarantee of asymptotic convergence to the global optimum that those methods provide. We demonstrate the performance of our algorithm by efficiently generating trajectories through challenging indoor spaces and successfully traversing them at speeds up to 8 m/s. A demonstration of our algorithm and flight performance is available at: http://groups.csail.mit.edu/rrg/quad_polynomial_trajectory_planning.

C. Richter (✉) · A. Bry · N. Roy
Massachusetts Institute of Technology, 77 Massachusetts Ave,
Cambridge, MA 02139, USA
e-mail: car@mit.edu

A. Bry
e-mail: abry@mit.edu

N. Roy
e-mail: nickroy@mit.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_37

1 Introduction

Recent advances in small unmanned aircraft have enabled highly dynamic, aerobatic flight maneuvers [1, 9, 10, 20]. Simultaneously, advances in fast, accurate state estimation methods have enabled these vehicles to fly through dense, cluttered spaces without the need for a motion capture system [4, 25]. However motion planning algorithms have not yet succeeded in joining these capabilities to enable quadrotors to navigate autonomously at high speeds using their full dynamic capabilities. This paper addresses that need and provides a planning algorithm that enables autonomous, aggressive, high-speed quadrotor flight through complex indoor environments.

While there exist advanced techniques for robotic navigation and trajectory optimization, there has yet to emerge a single algorithm that can both find *and* optimize a quadrotor trajectory through a complex real-world environment quickly enough to be useful for a deployable robotic system. While algorithms such as RRT* provably converge to the optimal solution in the limit of infinite samples, it is often impractical to rely on this limit to perform optimization for vehicles with nonlinear 12-DOF dynamics. These algorithms have been most successful for simple Dubins vehicle or double-integrator systems where analytical techniques can be used to steer between two points in state space [13]. For other systems, the search over dynamically feasible trajectories often requires iterative simulation of the equations of motion.

Nonlinear programming techniques for trajectory optimization, such as direct collocation and shooting methods, can also be used to find locally optimal paths for systems with general dynamics. However, these methods are also computationally intensive and may require accurate analytical representations of environmental constraints in order to efficiently compute cost gradients with respect to obstacles. These limitations make them impractical when constraints are represented in the form of an occupancy map.

Nevertheless, explicit optimization is useful for high-speed trajectories through cluttered environments. Minimum-snap polynomial splines have proven very effective as quadrotor trajectories, since the motor commands and attitude accelerations of the vehicle are proportional to the snap, or fourth derivative, of the path [19]. Minimizing the snap of a trajectory quantifies a notion of gracefulness that is desirable for maintaining the quality of onboard sensor measurements as well as avoiding abrupt or excessive control inputs.

The differentiability of polynomial trajectories makes them a natural choice for use in a *differentially flat* representation of the quadrotor dynamics. Differential flatness provides an analytical mapping from a path and its derivatives to the states and control inputs required to follow that path. This powerful property effectively guarantees feasibility of any differentiable trajectory, provided that its derivatives are sufficiently bounded to avoid input saturation, thus eliminating the need for iterative simulation in the search for trajectories.

Our contribution is to extend the work of Mellinger et al. [19], and show that their minimum-snap trajectory generation can be solved in a numerically stable unconstrained quadratic program (QP) for long-range trajectories composed of many segments. We show that this minimum-snap technique can be coupled with an appropriate kinematic planner to generate fast, graceful flight paths in cluttered environments, while accounting for collisions of the resulting polynomial trajectories. This combination of search and optimization significantly outperforms pure search-based planning methods in computational performance. Finally, we modify their strategy of allocating time along the trajectory to allow the planner to automatically adjust to widely varying size scales with a single user-set parameter on aggressiveness.

1.1 Problem Statement and Solution Outline

Given a 3D occupancy map of an environment, we wish to efficiently compute feasible, minimum-snap trajectories that follow the shortest collision-free path from start to goal utilizing the full dynamic capabilities of the quadrotor.

Our solution to this problem is to utilize the RRT* algorithm to find a collision-free path through the environment, initially considering only the kinematics of the vehicle and ignoring the dynamics. That path is pruned to a minimal set of waypoints, and a sequence of polynomial segments is jointly optimized to join those waypoints into a smooth minimum-snap trajectory from start to goal. Utilizing a differentially flat model of the quadrotor and the associated control techniques, we can follow these paths precisely.

The paper proceeds as follows. We first discuss the differentially flat quadrotor model and its implications for planning and polynomial trajectories. We then present a closed-form solution to the QP used to obtain the polynomial trajectory that is numerically stable for both high-order polynomials and large numbers of segments. For comparison with purely sampling-based approaches, we compare our process with an RRT* algorithm that uses polynomial segments to grow a tree of candidate trajectories (i.e., as its *steer function* to connect sampled points in state space). We show that our process returns superior paths in much shorter running time. Finally, we highlight the performance of our QP formulation and show the results of flight tests in real-world environments.

2 Quadrotor Dynamics and Control

In order to ensure that we can precisely follow the polynomial trajectories we intend to generate, we utilize the property of differential flatness for the standard quadrotor equations of motion:

$$m\ddot{\mathbf{r}} = mg\mathbf{z}_W - f\mathbf{z}_B \tag{1}$$

$$\dot{\omega} = J^{-1}\left[-\omega \times J\omega + M\right] \tag{2}$$

Differential flatness of this model was demonstrated in [19]. Here, $\mathbf{r}$ is the position vector of the vehicle in a global coordinate frame, ω is the angular velocity vector in the body-fixed coordinate frame and f and M are the net thrust and moments in the body-fixed coordinate frame. J and m are the inertial tensor and mass of the quadrotor. $\mathbf{z}_B$ is the unit vector aligned with the axis of the four rotors and indicates the direction of thrust, while $\mathbf{z}_W$ is the unit vector expressing the direction of gravity. There exists a simple mapping from f and M to the four desired motor speeds.

A polynomial trajectory segment consists of four polynomial functions of time specifying the independent evolution of the so-called *flat output* variables, x, y, z, and ψ (yaw angle) between two states in flat output space. The nonlinear controller employed to follow differentiable trajectories was developed in [18], and consists of independent calculations for thrust and moments:

$$f = (-k_x\mathbf{e}_x - k_v\mathbf{e}_v + mg\mathbf{z}_W + m\ddot{\mathbf{r}}_d) \cdot R\mathbf{z}_w \tag{3}$$

$$\begin{aligned} M = &-k_R\mathbf{e}_R - k_\omega\mathbf{e}_\omega + \omega \times J\omega \\ &- J(\hat{\omega}R^T R_d\omega_d - R^T R_d\dot{\omega}_d) \end{aligned} \tag{4}$$

where $\mathbf{e}_x$, $\mathbf{e}_v$, $\mathbf{e}_R$, and $\mathbf{e}_\omega$ are the error vectors in position, velocity, orientation and angular velocity, k_x, k_v, k_R, and k_ω are associated control gains, and R is the rotation matrix representing the orientation of the quadrotor.

Since the desired trajectory and its derivatives are sufficient to compute the states and control inputs at every point along the path in closed form (Eqs. 3 and 4), these quantities serve as a simulation of the vehicle's motion in the absence of disturbances. This is the powerful capability enabled by differential flatness that eliminates the need for iterated numerical integration of equations of motion, or a search over the space of inputs during each iteration of the planning algorithm.

3 Polynomial Trajectory Optimization

We now describe an analytical method for generating minimum-snap polynomial trajectories to be followed by a quadrotor using the control techniques outlined above. We assume that we have obtained a sequence of waypoints in 3D space representing the shortest piecewise-linear path through the environment, and we wish to generate a minimum-snap polynomial path passing through each of those waypoints. For this purpose, we use a simple RRT* algorithm to obtain the optimal straight-line path from start to goal, and then select waypoints from that optimal path according to a line-of-sight technique. Figure 1b shows the sequence of waypoints obtained by this method.

The choice of polynomial trajectories is natural for highly dynamic vehicles and robots since these trajectories can be obtained efficiently as the solution to a QP that minimizes a cost function of the path derivatives. This optimization framework

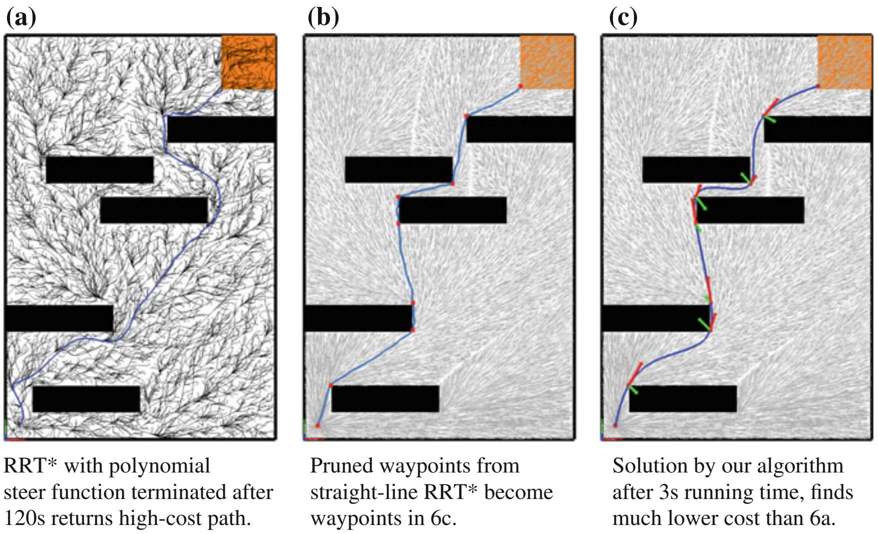

(a)
RRT* with polynomial steer function terminated after 120s returns high-cost path.
(b)
Pruned waypoints from straight-line RRT* become waypoints in 6c.
(c)
Solution by our algorithm after 3s running time, finds much lower cost than 6a.

Fig. 1 Using polynomial segments directly as a RRT* steer function (**a**) is computationally slow. Therefore, we run a straight-line RRT* and select waypoints from the optimal path (**b**). However, the straight-line RRT* ignores dynamics and returns a path that does not match our objective function. We therefore jointly optimize a set of polynomials through those waypoints to obtain a minimum-snap path (**c**)

allows the endpoints of path segments to be optionally fixed to desired values or left free, and the polynomials can be jointly optimized while maintaining continuity of the derivatives up to arbitrary order. Maintaining continuity of derivatives ensures smooth motions and can be used to generate trajectories that do not require step inputs to the vehicle's actuators.

Polynomial trajectories allow for a analytical solution via elimination as a constrained QP [2]. While this method is acceptable for joint optimization of a few segments, it involves the inversion of matrices that may be very close to singular, along with high sensitivity to coefficients on the order of 10^{-20} or smaller, leading to inaccurate results. We present this constrained QP solution next and then use it in a following section as the basis for an unconstrained QP reformulation, which is robust to numerical instability.

For the following derivations, we require that the vector of segment times is fixed. That is, we require an a priori selection of the amount of time required to traverse between one waypoint and the next. These times can be selected approximately based on a desired average speed of the vehicle, however in general an arbitrary selection of times will not yield the lowest-cost solution. Therefore, we relax this assumption in a subsequent section where we iteratively refine the vector of times.

3.1 Cost Function for Minimizing Derivatives

For quadrotors, a single trajectory segment between two points in flat output space is composed of independent polynomials, $P(t)$, for the flat output variables x, y, z and yaw angle. The cost function penalizing the squares of the derivatives of $P(t)$ can be written as:

$$J(T) = \int_0^T c_0 P(t)^2 + c_1 P'(t)^2 + c_2 P''(t)^2 + \cdots + c_N P^{(N)}(t)^2 dt = \mathbf{p}^T Q(T)\mathbf{p} \qquad (5)$$

In this expression, $\mathbf{p}$ is a vector of the N coefficients of a single polynomial. In order to minimize snap, all derivative penalties in the cost function except for c_4 would be set to zero. The construction of the Hessian matrix Q is omitted for brevity, but follows from differentiation of the square of the polynomial with respect to each of its coefficients. Since the cost of a given polynomial is a function of its duration T, we must fix T prior to optimization. M polynomial segments can be jointly optimized by concatenating their cost matrices in a block-diagonal fashion:

$$J_{total} = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T \begin{bmatrix} Q_1(T_1) & & \\ & \ddots & \\ & & Q_M(T_M) \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T \qquad (6)$$

3.2 Constraints

Constraints in the polynomial optimization are imposed on the endpoints of each segment. These constraints allow the endpoints to be pinned to known locations in space, or assigned specific values of velocity, acceleration, jerk or snap. Such constraints are useful to enforce, for example, that the quadrotor start from rest at the beginning of a trajectory. Constraints on the ith segment in a trajectory are formulated using a mapping matrix (A) between coefficients and endpoint derivatives of a polynomial:

$$A_i \mathbf{p}_i = \mathbf{d}_i, \quad A_i = \begin{bmatrix} A_0 \\ A_T \end{bmatrix}_i, \quad \mathbf{d}_i = \begin{bmatrix} d_0 \\ d_T \end{bmatrix}_i \qquad (7)$$

where $\mathbf{d}_i$ is a vector containing the derivative values for the beginning (d_0) and end (d_T) of the ith segment. If specific derivatives are not known, then continuity constraints must be imposed to ensure that the derivatives at the end of the ith segment match the derivatives at the beginning of the $(i+1)$th segment:

$$A_{T,i}\mathbf{p}_i = A_{0,i+1}\mathbf{p}_{i+1} \qquad (8)$$

These constraints can be compiled into a single set of linear equality constraints for the joint optimization problem:

$$A_{total} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix} = \begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_M \end{bmatrix} \tag{9}$$

Standard methods can be used to solve the resulting constrained QP.

3.3 Reformulation as an Unconstrained QP

While the method above works well for single segments and small joint optimization problems as in [19], this formulation becomes ill-conditioned for more than several segments, polynomials of high order, and when widely varying segment times are involved. Hence, it is only useful for short trajectories and must be improved to be practical for optimizing long range paths requiring many waypoints and segments.

We improve upon the solution above using a technique of substitution to convert the problem into an unconstrained QP, and solve directly for endpoint derivatives as decision variables, rather than solving for polynomial coefficients. In practice, our reformulation is substantially more stable than the method above, allowing the joint optimization of more than 50 polynomial segments in a single matrix operation without encountering numerical issues. Once the optimal waypoint derivatives are found, the minimum-order polynomial connecting each pair of waypoints can be obtained by inverting the appropriate constraint matrix.

We begin by substituting the constraints into the original cost function:

$$J = \begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_M \end{bmatrix}^T \begin{bmatrix} A_1 & & \\ & \ddots & \\ & & A_M \end{bmatrix}^{-T} \begin{bmatrix} Q_1 & & \\ & \ddots & \\ & & Q_M \end{bmatrix} \begin{bmatrix} A_1 & & \\ & \ddots & \\ & & A_M \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_M \end{bmatrix} \tag{10}$$

Now the decision variables in this new quadratic cost function are the endpoint derivatives of the segments. We re-order these variables such that fixed/specified derivatives are grouped together ($\mathbf{d}_F$) and the free/unspecified derivatives are grouped together ($\mathbf{d}_P$). A permutation matrix assembled of ones and zeros (C) is used to accomplish this re-ordering. Now we have:

$$J = \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix}^T \underbrace{CA^{-T}QA^{-1}C^T}_{R} \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix} = \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix}^T \begin{bmatrix} R_{FF} & R_{FP} \\ R_{PF} & R_{PP} \end{bmatrix} \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix} \tag{11}$$

where we have written the block-diagonal matrices as A and Q for simplicity of notation. We group the new augmented cost matrix into a single matrix R and partition it according to the indices of the fixed and free derivatives. Partitioning allows us to write out the expression for total cost as:

$$J = \mathbf{d}_F^T R_{FF} \mathbf{d}_F + \mathbf{d}_F^T R_{FP} \mathbf{d}_P + \mathbf{d}_P^T R_{PF} \mathbf{d}_F + \mathbf{d}_P^T R_{PP} \mathbf{d}_P \tag{12}$$

Differentiating J and equating to zero yields the vector of optimal values for the free derivatives in terms of the fixed/specified derivatives and the cost matrix:

$$\mathbf{d}_P^* = -R_{PP}^{-1} R_{FP}^T \mathbf{d}_F \tag{13}$$

The polynomials can now be recovered from individual evaluations of the appropriate constraint equations mapping derivatives back into the space of coefficients.

3.4 Time Allocation

Until this point in the optimization, we have fixed an arbitrary amount of time associated with each segment, since these times factor into the construction of the cost matrix. These segment times constrain the solution quality, but can be allowed to vary to improve the overall solution with respect to a cost function. We therefore begin with an initial guess of segment times and then iteratively refine those times using gradient descent. Several cost functions may be suitable candidates: [5] minimizes total time subject to constraints, while [19] fixes the total time by hand and minimizes snap (the original cost function) with the remaining degrees of freedom. In the planning context, we do not know the total trajectory time a priori, so we allow it to vary in the optimization to perform a trade-off between minimizing snap and total trajectory time. We attempt to minimize:

$$J_T = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T \begin{bmatrix} Q_1(T_1) & & \\ & \ddots & \\ & & Q_M(T_M) \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T + k_T \sum_{i=1}^{M} T_i \tag{14}$$

where k_T is a user-specified penalty on time. The first term in this cost function is simply the original cost function for polynomial optimization. When penalizing only acceleration, jerk or snap, this original cost can be driven arbitrarily close to zero by increasing the total time, but Eq. (14) has a definite minimum value that varies with k_T. Figure 2 shows several iterations of gradient descent in which the total trajectory time is decreased from a large initial guess (red) to smaller optimal value (blue), while the ratio of times between segments also shifts to minimize the modified cost.

Rather than selecting total times arbitrarily, this cost function allows our algorithm to automatically adjust for environments of widely varying scales, or where the vehicle must slow down to navigate tightly spaced obstacles without incurring excessive snap. Furthermore, our procedure produces trajectories of comparable aggressiveness

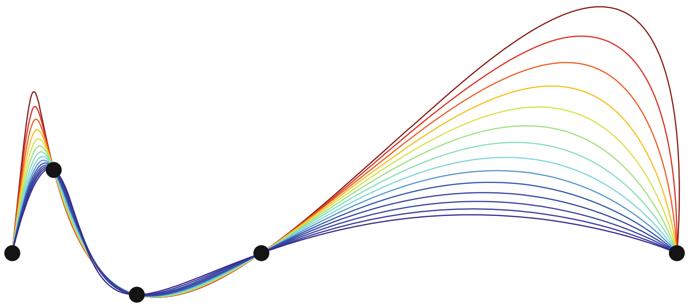


Fig. 2 Illustration of the iterative refinement of segment times, color-coded by total traversal time. The initial guess of total time is 10.5 s (*red*) and the final optimized total time is 7 s (*blue*)

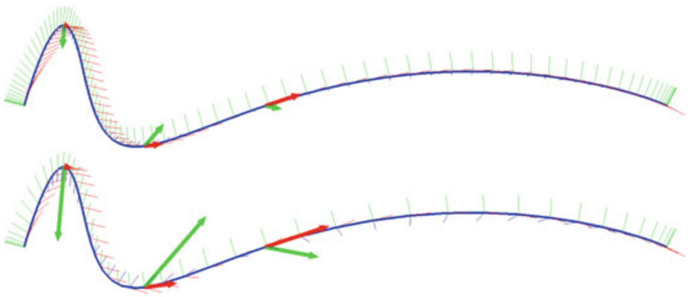
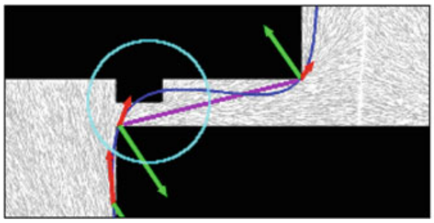


Fig. 3 Segment time optimization with the penalty on time k_T set at 500 (*top*) and 50,000 (*bottom*). The optimal total trajectory times are 9.1 and 5.1 s respectively. Vectors for waypoint velocity (*red*) and acceleration (*green*) are shown

in a wide range of scenarios for a given fixed value of the single scale-independent parameter, k_T.

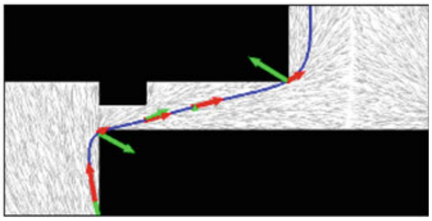
Figure 3 shows optimized trajectories for the same set of waypoints using two different k_T values. The red arrows indicate waypoint velocities while the green arrows indicate accelerations. These quantities are greater in the bottom trajectory due to the higher time penalty. The quadrotor axes are plotted at 0.1 s increments along the path. One emergent property resulting from time allocation is that the quadrotor moves very slowly around the sharp corner and then smoothly accelerates up to a higher speed in the straightaway where it does not incur a severe penalty on snap. Furthermore, the geometric shape of the optimal trajectory remains the same regardless of the value of k_T, indicating that the minimum-snap *ratios* of segment times are independent of k_T.

(a)



Polynomial trajectory (blue) intersects an obstacle even though the underlying straight line between waypoints is collision-free.

(b)



After bisecting the underlying straight line twice with two additional waypoints, the polynomial trajectory is collision-free.

Fig. 4 **a** The polynomial (*blue*) intersects an obstacle even though the line between waypoints is collision free (*magenta*). These scenarios are resolved by iteratively adding waypoints along the collision-free path returned by the search algorithm (**b**)

3.5 Ensuring the Trajectory Is Collision-Free

If a particular trajectory segment is found to intersect an obstacle after optimization, an additional waypoint is simply added halfway between its two ends, splitting this segment into two. This midpoint is known to be collision-free because it lies on the optimal piecewise-linear path returned by the search algorithm. The polynomial is re-optimized with the additional waypoint, and the process is repeated if necessary until the polynomial trajectory is collision free. A similar technique is used in [23]. Figure 4 illustrates this process successfully resolving a collision.

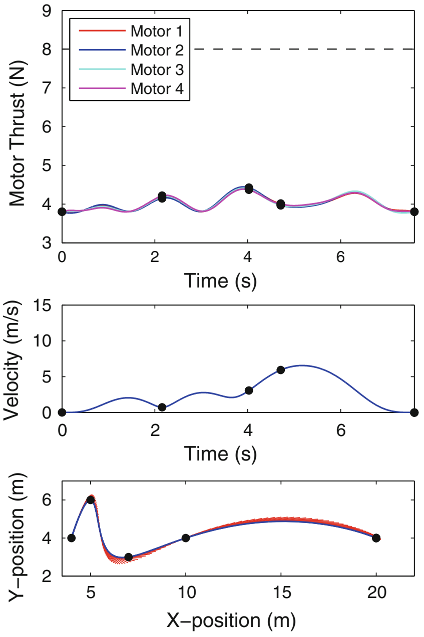
In very dense environments, trajectories may need many additional waypoints to repair collisions, thus requiring the optimization problem to be re-solved many times to find a feasible solution. Furthermore, additional waypoints increase the computational complexity of the QP being solved in each iteration. However, in our experience with indoor environments, the number of additional waypoints required to repair collisions was usually less than half of the original number of waypoints in the trajectory, representing only a modest increase in computational complexity.

3.6 Actuator Constraints

The second major factor contributing to feasibility is to ensure that the input constraints of the quadrotor are satisfied such that no portion of the commanded trajectory requires a thrust outside the range that the motors are capable of providing. Formally, solving a trajectory optimization problem in the flat output space of a differentially-flat model requires mapping the constraints into the flat output space as well as the dynamics. Some work has focused on computationally estimating the feasible set in flat output space [6], however this set is generally a non-convex function of nonlinear inequalities and is a hard optimization problem unto itself.

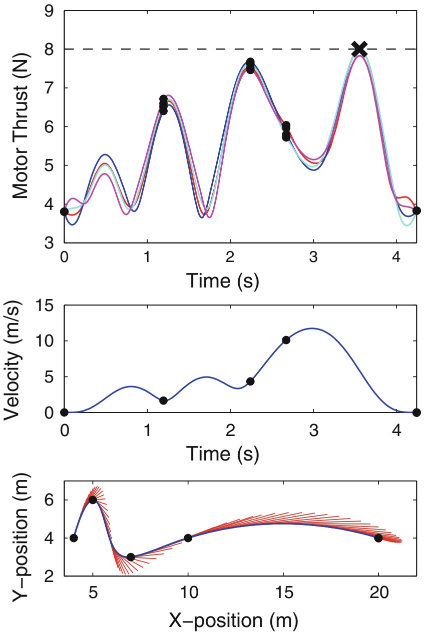
Instead, we address this challenge during the time-allocation step of trajectory optimization, since the distribution of time along the trajectory largely determines the required accelerations and therefore the peaks in required thrust. First, we observe that in the limit as $T \to \infty$, the quadrotor states along the trajectory converge to hover, which is known to be feasible. Therefore, we initialize our time-allocation optimization step with a conservatively large guess for initial segment times. Then, as the modified cost function is minimized, we compute the actuator commands algebraically during each iteration to verify that we remain within the feasible set. Optimization is terminated when either a local minimum is obtained or an actuator constraint becomes active. Figure 5 illustrates two different time allocations during the optimization of a sample trajectory. One of these time allocations is safely within

(a)



Motor commands for a conservative time allocation, barely exceeding the 3.8 N per motor required for hover (top). Velocity along the trajectory is low (middle). Trajectory with velocity vectors is shown for reference, with black dots indicating waypoints (bottom).

(b)



Motor commands for aggressive time allocation, with one motor command reaching the maximum available thrust, indicated by the 'X' (top). Velocity along the trajectory is high (middle). Trajectory with larger velocity vectors is shown for reference(bottom).

Fig. 5 Comparison between two time allocations during the gradient descent procedure. The first time allocation (**a**) is conservative in that it is a slower trajectory than the second one (**b**), which reaches one of the actuator constraints during the final deceleration

the feasible set, since it commands thrusts barely above the nominal thrust required for hover, whereas the other time allocation is very aggressive and activates an actuator constraint.

Due to the non-convexity of the feasible set in flat output space, the optimization algorithm may encounter an actuator limit and terminate before converging to the optimal ratio of segment times (for example, one of the red or orange lines in Fig. 2). To avoid this scenario, one strategy is to first optimize the ratio of segment times via gradient descent while ignoring actuator constraints, taking advantage of the fact that the optimal ratio of times is invariant to the total time as noted in Sect. 3.4. Then once the optimal ratio of times is achieved, scale the *total* trajectory time in a separate univariate optimization, preserving the optimal ratio, until the modified cost function is minimized or an actuator constraint becomes active.

4 Results

We have tested our trajectory generation process in a variety of environments. Figures 1 and 6 show solutions to challenging 2D and 3D problems. The use of a minimal set of waypoints and the joint polynomial optimization described above yields paths that are typically composed of natural high-speed arcs in unconstrained regions of the environment while slowing in tight spaces to minimize snap around sharp corners. Our process sacrifices the guarantee of asymptotic convergence to a globally optimal solution provided by sampling-based approaches, but returns superior paths in much shorter running times than a purely sampling-based approach.

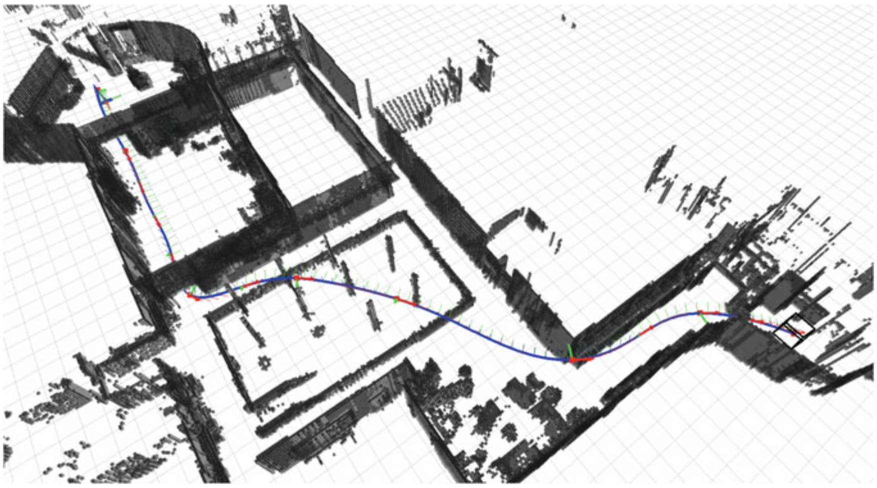


Fig. 6 Automatically generated 3D trajectory navigating a real-world environment with closely-spaced obstacles

4.1 Comparison with RRT* Using Polynomial Steer Function

For comparison to a strictly sampling-based planning approach, we implemented an RRT* algorithm using polynomial segments as the steer function to grow a search tree. Figure 1a shows the resulting solution. Sampling was performed in position and velocity space. We use the distance metric described by [11] of Euclidean distance divided by average velocity. One major difficulty with this approach is that segment times must be fixed when generating polynomials to extend the tree, however as discussed above, the selection of segment time can have a dramatic impact on the quality of a path, so an appropriate guess must be made a priori for each segment, or the segment time must be included in the sampling space. In our implementation, the segment times were chosen as the Euclidean distance between vertices divided by the desired average velocity along the segment.

Table 1 shows several statistics on the performance of the RRT* with a polynomial steer function compared to our algorithm. The RRT* runs much longer and fails to find a path as smooth or with a cost as low as our algorithm. When sampling in the full state space of the system, the RRT* with a polynomial steer function would converge to a globally optimal solution in the limit of infinite samples, however as shown here, the paths returned prior to convergence are of lower quality than those returned by our algorithm in a much shorter running time.

4.2 Performance of Polynomial Optimization

A key to the success of this trajectory planning process is the speed and numerical stability of the joint polynomial optimization method. We performed benchmark tests on an example problem consisting of four waypoints (3 polynomial segments) chosen to represent distance and time scales consistent with common environments for quadrotor flight. The results are given in Table 2 and reflect MATLAB as well as C++/Eigen implementations [7]. This computational efficiency makes it feasible to use this planning framework in online applications and to use iterative path refinement methods with polynomial optimization in the loop.

Table 1 Comparison of our method with RRT* using the polynomial steer function for the 2D problem in Fig. 1

Method	Runtime (s)	$J_{poly.}$	T_{path} (s)	L_{path} (m)
RRT* with polynomial steer function	120	5.72×10^8	21.94	40.35
Low-Dim. search + unconstrained QP optimization	3	1.07×10^5	19.66	35.51

Table 2 Comparison of polynomial optimization times

Benchmark problem: 3-segment joint optimization

Method	Solution time (ms)
MATLAB `quadprog.m`	9.5
MATLAB constrained	1.7
MATLAB unconstrained (dense)	2.7
C++/Eigen constrained	0.18
C++/Eigen unconstrained (dense)	0.34

Table 3 Numerical stability of optimization techniques for high-order polynomials and various numbers of segments

Success rates on randomized polynomial optimization problems

Formulation	Polynomial order	Number of segments	Success (%)
Constrained	9	3	100
	9	4	55
	9	≥ 5	0
Unconstrained	9	50+	100
	15	50+	100

While the unconstrained formulation is slightly slower than the constrained formulation, its primary benefit lies in its stability. The constrained formulation encounters matrices very close to singular for joint optimizations consisting of more than three 9th order polynomials, and therefore may return inaccurate results depending on the quality of the linear algebra solver. In contrast, the unconstrained formulation is robust to numerical issues, as shown in Table 3, which lists the results of 20 polynomial optimization problems in which the locations of intermediate waypoints and the segment times were randomly generated in the range $[1, 3]$. Clearly, the unconstrained optimization is much more robust to numerical instability, enabling this method to be used as a reliable, efficient long-range trajectory optimization tool for navigation outside of small motion-capture environments.

Finally, since A^{-1} and Q are sparse block-diagonal and C is sparse, these problems can be easily implemented using a sparse solver which is roughly an order of magnitude faster than the dense computation for 10-segment joint optimizations.

4.3 Experimental Flight Tests

We demonstrate the performance of our algorithm on a challenging real-world planning problem by generating and flying a trajectory through a complex indoor lab space in the Stata Center (MIT). The environment used for these tests was a lab space with

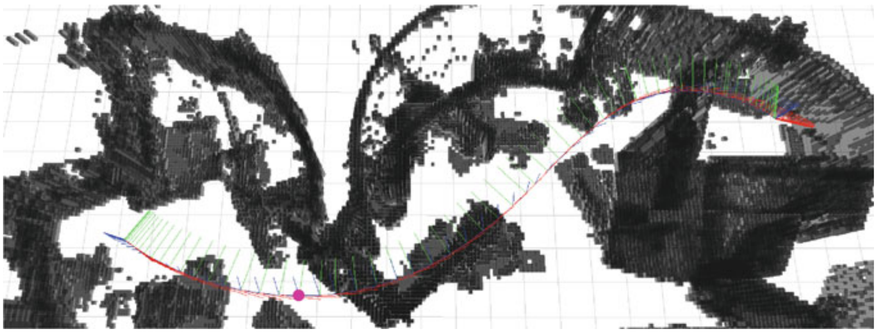


Fig. 7 Automatically generated trajectory through a map of a laboratory environment in the Stata Center, MIT

curved, non-vertical walls, interior columns and barriers aligned at oblique angles. An OctoMap representation of the lab was generated using a pair of planar laser range finders and each occupied cell was dilated with a radius of 0.65 m to leave room for the 0.35 m radius of the vehicle and a minimal allowance for error in estimation and control. Estimation and control were performed completely onboard the AscTec Pelican aircraft, using a Hokuyo LIDAR, a Microstrain IMU and an Intel Atom processor.

The trajectories returned by our algorithm are shown in Figs. 6 and 7, and were generated in several seconds. These trajectories exhibit roughly 2 m of altitude variation in order to fly through doorways and navigate over tall shelves and dividing walls. Figure 8 shows onboard video frames taken while executing these trajectories at speeds up to 8 m/s. Video of these trajectories and flights is available at: http://groups.csail.mit.edu/rrg/quad_polynomial_trajectory_planning.

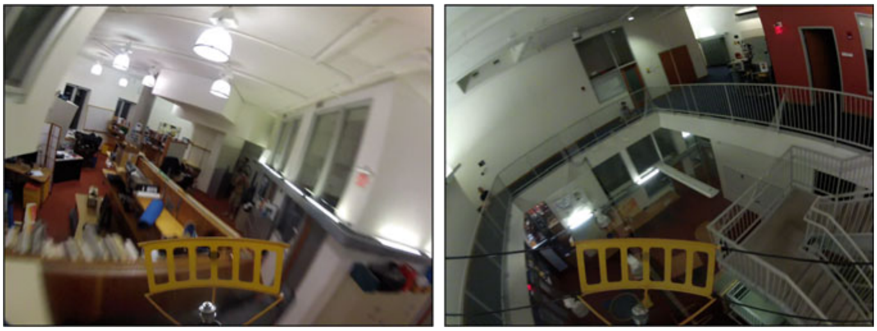


Fig. 8 Onboard video frames from aggressive quadrotor flight up to 8 m/s

5 Related Work

The literature on motion planning for robots and vehicles is extensive, considering both simple holonomic systems as well as those with differential constraints. Randomized algorithms such as PRM, RRT and RRT* have enjoyed success due to their simplicity and performance in high-dimensional spaces [12, 14, 16].

Sampling-based algorithms have also been demonstrated for motion planning under differential constraints, which often perform very well when there exist simple analytical techniques for obtaining a steer function from one vertex in state space to the next [13, 17]. However, for general dynamical systems, steering between two states may require iteratively simulating the vehicle dynamics at a significant computational cost [11]. Furthermore, the nearest vertex according to a Euclidean distance metric is not, in general, the vertex that will yield an optimal (or even feasible) path to a new sample in state space [26]. Nevertheless, sampling-based methods have proven successful in real-world applications to motion planning of vehicles with non-trivial dynamics [15].

Many methods exist for optimizing trajectories between two states of a dynamical system [3], and have been successfully applied to quadrotor control [24]. B-splines [23] and Legendre polynomials [21] have been used to avoid ill-conditioning in trajectory optimization problems, however these options preclude the efficient method presented here. Finally, our method is not limited to quadrotor control, as there exist simple differentially flat representations of fixed-wing aircraft [8] and cars [22] among many other systems.

6 Conclusion

We have presented an algorithm for generating trajectories for the differentially flat quadrotor model through complex real-world environments that is computationally much faster than solving the same problems using a pure sampling approach, though at the expense of global optimality. We observe that in this domain it is infeasible to rely on the limit of infinite sampling to perform optimization, and instead we perform low-dimensional search for route-finding followed by analytical optimization in which the shortest path is translated into a dynamically feasible polynomial trajectory. We then iteratively refine the polynomial trajectory by a time allocation procedure that trades off between time and snap of the path.

Acknowledgments The support of the ARO MAST CTA, the ONR under MURI N00014-09-1-1052, and the NDSEG fellowship is gratefully acknowledged.

References

1. Abbeel, P., Coates, A., Ng, A.: Autonomous helicopter aerobatics through apprenticeship learning. Int. J. Robot. Res. **29**(13), 1608–1639 (2010)
2. Bertsekas, D.P.: Nonlinear Programming. Athena Scientific, Belmont (1999)
3. Betts, J.T.: Survey of numerical methods for trajectory optimization. J. Guid. Control Dyn. **21**(2), 193–207 (1998)
4. Bry, A., Bachrach, A., Roy, N.: State estimation for aggressive flight in GPS-denied environments using onboard sensing. In: Proceedings of the International Conference on Robotics and Automation (2012)
5. Cutler, M., How, J.: Actuator constrained trajectory generation and control for variable-pitch quadrotors. In: Proceedings of the AIAA Guidance, Navigation, and Control Conference (2012)
6. Faiz, N., Agrawal, S., Murray, R.: Differentially flat systems with inequality constraints: an approach to real-time feasible trajectory generation. J. Guid. Control Dyn. **24**(2), 219–227 (2001)
7. Guennebaud, G., Jacob, B., et al.: Eigen v3 (2010). http://eigen.tuxfamily.org
8. Hauser, J., Hindman, R.: Aggressive flight maneuvers. In: Proceedings of Conference on Decision and Control (1997)
9. Hehn, M., D'Andrea, R.: Quadrocopter trajectory generation and control. In: Internatioanl Federation of Automatic Control, World Congress (2011)
10. How, J.P., Bethke, B., Frank, A., Dale, D., Vian, J.: Real-time indoor autonomous vehicle test environment. IEEE Control Syst. **28**(2), 51–64 (2008)
11. Jeon, J.H., Karaman, S., Frazzoli, E.: Anytime computation of time-optimal off-road vehicle maneuvers using the RRT*. In: Conference on Decision and Control (2011)
12. Karaman, S., Frazzoli, E.: Incremental sampling-based algorithms for optimal motion planning. In: Proceedings of Robotics Science and Systems (2010)
13. Karaman, S., Frazzoli, E.: Optimal kinodynamic motion planning using incremental sampling-based methods. In: Conference on Decision and Control (2010)
14. Kavraki, L.E., et al.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. **12**(4), 566–580 (1996)
15. Kuwata, Y., et al.: Real-time motion planning with applications to autonomous urban driving. IEEE Trans. Control Syst. Technol. **17**(5), 1105–1118 (2009)
16. LaValle, S.M., Kuffner, J.J.: Rapidly-exploring random trees: Progress and prospects. In: Workshop on Algorithmic Foundations of Robotics (2000)
17. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. Int. J. Robot. Res. **20**(5), 378–400 (2001)
18. Lee, T., Leoky, M., McClamroch, N.H.: Geometric tracking control of a quadrotor uav on se(3). In: Conference on Decision and Control (2010)
19. Mellinger, D., Kumar. V.: Minimum snap trajectory generation and control for quadrotors. In: Proceedings of International Conference on Robotics and Automation (2011)
20. Mellinger, D., Michael, N., Kumar, V.: Trajectory generation and control for precise aggressive maneuvers with quadrotors. In: Proceedings of International Symposium on Experimental Robotics (2010)
21. Mellinger, D., Kushleyev, A., Kumar, V.: Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In: Proceedings of International Conference on Robotics and Automation (2012)
22. Murray, R.M., Rathinam, M., Sluis, W.: Differential flatness of mechanical control systems: A catalog of prototype systems. In: Proceedings of ASME International Congress and Exposition (1995)
23. Pan, J., Zhang, L., Manocha, D.: Collision-free and smooth trajectory computation in cluttered environments. Int. J. Robot. Res. **31**(10), 1155–1175 (2012)
24. Ritz, R., Hehn, M., Lupashin, S., D'Andrea, R.: Quadrocopter performance benchmarking using optimal control. In: Proceedings of International Conference on Intelligent Robots and Systems (2011)

25. Shen, S., Mulgaonkar, Y., Michael, N., Kumar, V.: Vision-based state estimation and trajectory control towards aggressive flight with a quadrotor. In: Proceedings of Robotics Science and Systems (2013)
26. Shkolnik, A., Walter, M., Tedrake, R.: Reachability-guided sampling for planning under differential constraints. In: Proceedings of International Conference on Robotics and Automation (2009)

Fast Marching Trees: A Fast Marching Sampling-Based Method for Optimal Motion Planning in Many Dimensions

Lucas Janson and Marco Pavone

Abstract In this paper we present a novel probabilistic sampling-based motion planning algorithm called the Fast Marching Tree algorithm (FMT*). The algorithm is specifically aimed at solving complex motion planning problems in high-dimensional configuration spaces. This algorithm is proven to be asymptotically optimal and is shown to converge to an optimal solution faster than its state-of-the-art counterparts, chiefly PRM* and RRT*. An additional advantage of FMT* is that it builds and maintains paths in a tree-like structure (especially useful for planning under differential constraints). The FMT* algorithm essentially performs a "lazy" dynamic programming recursion on a set of probabilistically-drawn samples to grow a tree of paths, which moves steadily outward in cost-to-come space. As such, this algorithm combines features of both single-query algorithms (chiefly RRT) and multiple-query algorithms (chiefly PRM), and is conceptually related to the Fast Marching Method for the solution of eikonal equations. As a departure from previous analysis approaches that are based on the notion of almost sure convergence, the FMT* algorithm is analyzed under the notion of convergence in probability: the extra mathematical flexibility of this approach allows for significant algorithmic advantages and provides convergence rate bounds—a first in the field of optimal sampling-based motion planning. Numerical experiments over a range of dimensions and obstacle configurations confirm our theoretical and heuristic arguments by showing that FMT*, for a given execution time, returns substantially better solutions than either PRM* or RRT*, especially in high-dimensional configuration spaces and in scenarios where collision checking is expensive.

L. Janson
Department of Statistics, Stanford University, Stanford, CA 94305, USA
e-mail: ljanson@stanford.edu

M. Pavone (✉)
Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305, USA
e-mail: pavone@stanford.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_38

1 Introduction

Probabilistic sampling-based algorithms represent a particularly successful approach to robotic motion planning problems in high-dimensional configuration spaces, which naturally arise, e.g., when controlling the motion of high degree-of-freedom robots or planning under uncertainty [13, 20]. Accordingly, the design of rapidly converging sampling-based algorithms with sound performance guarantees has emerged as a central topic in robotic motion planning, and represents the main thrust of this paper.

Specifically, the key idea behind probabilistic sampling-based algorithms is to avoid the explicit construction of the configuration space (which is prohibitive in the high-dimensional case), and instead conduct a search that probabilistically probes the configuration space with a sampling scheme. This probing is enabled by a collision detection module, which the motion planning algorithm considers as a "black box" [13]. Probabilistic sampling-based algorithms can be divided between multiple-query and single-query. Multiple-query algorithms construct a topological graph called a roadmap, which allows a user to efficiently solve multiple initial-state/goal-state queries. This family of algorithms includes the probabilistic roadmap algorithm (PRM) [10] and its variants, e.g., Lazy-PRM [4], dynamic PRM [7], and PRM* [9]. On the contrary, in single-query algorithms, a single initial-state/goal-state pair is given, and the algorithm must search until it finds a solution (or it may report early failure). This family of algorithms includes the rapidly exploring random trees algorithm (RRT) [14], the rapidly exploring dense trees algorithm (RDT) [13], and their variants, e.g., RRT* [9]. Other notable sampling-based planners include expansive space trees (EST) [5, 17], sampling-based roadmap of trees (SRT) [18], rapidly-exploring roadmap (RRM) [2], and the "cross-entropy" planner in [11]. Analysis in terms of convergence to feasible or even optimal solutions for multiple-query and single-query algorithms is provided in [3, 5, 6, 9, 12]. A central result is that these algorithms provide *probabilistic completeness* guarantees in the sense that the probability that the planner fails to return a solution, if one exists, decays to zero as the number of samples approaches infinity [3]. Recently, it has been proven that both RRT* and PRM* are asymptotically optimal, i.e. the cost of the returned solution converges almost surely to the optimum [9]. Building upon the results in [9], the work in [15] presents an algorithm with provable "sub-optimality" guarantees, which trades "optimality" with faster computation.

Statement of Contributions: The objective of this paper is to propose and analyze a novel probabilistic motion planning algorithm that is asymptotically optimal and improves upon state-of-the-art asymptotically-optimal algorithms (namely RRT* and PRM*) in terms of the convergence rate to the optimal solution (convergence rate is interpreted with respect to execution time). The algorithm, named the Fast Marching Tree algorithm (FMT*), is designed to be particularly efficient in high-dimensional environments cluttered with obstacles. FMT* essentially combines some of the features of multiple-query algorithms with those of single-query algorithms, by performing a "lazy" dynamic programming recursion on a set of probabilistically-drawn samples in the configuration space. As such, this algorithm combines features of PRM and SRT (similar to RRM), and grows a tree of trajectories like RRT. Additionally, FMT* is *conceptually* similar to the Fast Marching Method, one of the main methods for the solution of stationary eikonal equations [19] (see [21] for a recent overview of path planning algorithms inspired by the Fast Marching Method). As in the Fast Marching Method, the main idea is to exploit a heapsort technique to systematically locate the proper sample point to update and to incrementally build the solution in an "outward" direction, so that one needs never backtrack over previously evaluated sample points. Such a Dijkstra-like one-pass property is what makes both the Fast Marching Method and FMT* particularly efficient.

The end product of the FMT* algorithm is a tree, which, together with the connection to the Fast Marching Method, gives the algorithm its name. An advantage of FMT* with respect to PRM* (in addition to a faster convergence rate) is the fact that FMT* builds and maintains paths in a tree-like structure, which is especially useful when planning under differential or integral constraints. Our simulations across 5 and 10 dimensions, both without obstacles and with 50 % obstacle coverage show that FMT* outperforms PRM* and RRT*. The speedups are particularly prominent in higher dimensions and in scenarios where collision checking is expensive, which is exactly the regime in which sampling-based algorithms are particularly useful.

It is important to note that in this paper we use a notion of asymptotic optimality (AO) different from the one used in [9]. In [9], AO is defined through the notion of convergence almost everywhere (a.e.). Explicitly, in [9], an algorithm is considered AO if the cost of the solution it returns converges a.e. to the optimal cost as the number of samples n approaches infinity. This definition is completely justified when the algorithm is sequential in n, such as RRT* [9], in the sense that it requires that with probability 1 the sequence of solutions converges to an optimal one, with the solution at $n + 1$ heavily related to that at n. However, for non-sequential algorithms such as PRM* and FMT*, there is no connection between the solutions at n and $n + 1$. Since these algorithms process all the nodes at once, the solution at $n + 1$ is based on $n + 1$ new nodes, sampled independently of those used in the solution at n. This motivates the definition of AO used in this paper, which is that the cost of the solution returned by an algorithm must converge *in probability* to the optimal cost. Although mathematically convergence in probability is a weaker notion than convergence a.e. (the latter implies the former), in practice there is no distinction when an algorithm is only run on a predetermined, fixed number of nodes. In this case, all that matters is that the probability that the cost of the solution returned by the algorithm is less than an ε fraction greater than the optimal cost goes to 1 as $n \to \infty$, for any $\varepsilon > 0$, which is exactly the statement of convergence in probability. Since this is a mathematically weaker, but practically identical condition, we sought to capitalize on the extra mathematical flexibility, and indeed find that our proof of AO for FMT* allows for an improved implementation of PRM* in [9]. Our proof of AO also gives a *convergence rate bound* both for FMT* and PRM*—a first in the field of optimal sampling-based motion planning. Hence, an additional important contribution of this paper is the analysis of AO under the notion of convergence in probability, which is of independent interest and could enable the design and analysis of other AO sampling-based algorithms.

Organization: This paper is structured as follows. In Sect. 2 we formally define the optimal path planning problem. In Sect. 3 we present FMT* and we discuss some basic properties, e.g., termination. In Sect. 4 we prove the asymptotic optimality of FMT*. In Sect. 5 we first conceptually discuss the advantages of FMT* and then we present results from numerical experiments supporting our statements. Finally, in Sect. 6, we draw some conclusions and we discuss directions for future work.

Notation: Consider the Euclidean space in d dimensions, i.e., $\mathbb{R}^d$. Given a point $x \in \mathbb{R}$, a ball of radius $r > 0$ centered at $\bar{x} \in \mathbb{R}^d$ is defined as $B(\bar{x};\ r) := \{x \in \mathbb{R}^d \mid \|x - \bar{x}\| < r\}$. Given a subset $\mathscr{X}$ of $\mathbb{R}^d$, its boundary is denoted by $\partial\mathscr{X}$. Given two points x and z in $\mathbb{R}^d$, the line connecting them is denoted by $\overline{xz}$. Let ζ_d denote the volume of the unit ball in the d-dimensional Euclidean space. The cardinality of a set S is written as card S. Given a set $\mathscr{X} \subseteq \mathbb{R}^d$, $\mu(\mathscr{X})$ denotes its d-dimensional Lebesgue measure. In this paper, we will interchangeably refer to points in $\mathscr{X}$ as nodes, samples, or vertices.

2 Problem Setup

The problem formulation follows closely the problem formulation in [9]. Let $\mathscr{X} = [0,\ 1]^d$ be the configuration space, where $d \in \mathbb{N}$, $d \geq 2$. Let $\mathscr{X}_{\text{obs}}$ be the obstacle region, such that $\mathscr{X} \backslash \mathscr{X}_{\text{obs}}$ is an open set (we consider $\partial\mathscr{X} \subset \mathscr{X}_{\text{obs}}$), and denote the obstacle-free space as $\mathscr{X}_{\text{free}} = \text{cl}(\mathscr{X} \backslash \mathscr{X}_{\text{obs}})$, where $\text{cl}(\cdot)$ denotes the closure of a set. The initial condition x_{init} is an element of $\mathscr{X}_{\text{free}}$, and the goal region $\mathscr{X}_{\text{goal}}$ is an open subset of $\mathscr{X}_{\text{free}}$. A path planning problem is denoted by a triplet $(\mathscr{X}_{\text{free}}, x_{\text{init}}, \mathscr{X}_{\text{goal}})$. A function of *bounded variation* $\sigma : [0, 1] \to \mathbb{R}^d$ is called a *path* if it is continuous. A path is said to be *collision-free* if $\sigma(\tau) \in \mathscr{X}_{\text{free}}$ for all $\tau \in [0,\ 1]$. A path is said to be a *feasible path* for the planning problem $(\mathscr{X}_{\text{free}}, x_{\text{init}}, \mathscr{X}_{\text{goal}})$ if it is collision-free, $\sigma(0) = x_{\text{init}}$, and $\sigma(1) \in cl(\mathscr{X}_{\text{goal}})$.

A goal region $\mathscr{X}_{\text{goal}}$ is said to be *regular* if there exists $\xi > 0$ such that $\forall y \in \partial\mathscr{X}_{\text{goal}}$, there exists $z \in \mathscr{X}_{\text{goal}}$ with $B(z; \xi) \subseteq \mathscr{X}_{\text{goal}}$ and $y \in \partial B(z; \xi)$. In other words, a regular goal region is a "well-behaved" set where the boundary has bounded curvature. We will say $\mathscr{X}_{\text{goal}}$ is ξ-regular if $\mathscr{X}_{\text{goal}}$ is regular for the parameter ξ. Let Σ be the set of all paths. A cost function for the planning problem $(\mathscr{X}_{\text{free}}, x_{\text{init}}, \mathscr{X}_{\text{goal}})$ is a function $c : \Sigma \to \mathbb{R}_{\geq 0}$ from the set of paths to the nonnegative real numbers; in this paper we will consider as cost functions $c(\sigma)$ the *arc length* of σ with respect to the Euclidean metric in $\mathscr{X}$ (recall that σ is, by definition, rectifiable). Extension to more general cost functions (possibly not satisfying the triangle inequality) are possible and are deferred to future work. The optimal path planning problem is then defined as follows:

Optimal path planning problem: Given a path planning problem $(\mathscr{X}_{\text{free}}, x_{\text{init}}, \mathscr{X}_{\text{goal}})$ with a regular goal region and an arc length function $c : \Sigma \to \mathbb{R}_{\geq 0}$, find a feasible path σ^* such that $c(\sigma^*) = \min\{c(\sigma) : \sigma \text{ is feasible}\}$. If no such path exists, report failure.

Finally, we introduce some definitions concerning the *clearance* of a path, i.e., its "distance" from $\mathscr{X}_{\text{obs}}$ [9]. For a given $\delta > 0$, the δ-interior of $\mathscr{X}_{\text{free}}$ is defined as the set of all states that are at least a distance δ away from any point in $\mathscr{X}_{\text{obs}}$. A collision-free path σ is said to have strong δ-clearance if it lies entirely inside the δ-interior of $\mathscr{X}_{\text{free}}$. A path planning problem with optimal path cost c^* is called δ-robustly feasible if there exists a strictly positive sequence $\delta_n \to 0$, and a sequence $\{\sigma_n\}_{i=1}^{n}$ of feasible paths such that $\lim_{n\to\infty} c(\sigma_n) = c^*$ and for all $n \in \mathbb{N}$, σ_n has strong δ_n-clearance, $\sigma_n(1) \in \partial\mathscr{X}_{\text{goal}}$, $\sigma_n(\tau) \notin \mathscr{X}_{\text{goal}}$ for all $\tau \in (0, 1)$, and $\sigma_n(0) = x_{\text{init}}$. Note this definition is slightly different mathematically than admitting a *robustly optimal solution* as in [9], but the two are nearly identical in practice. Briefly, the difference is necessitated by the definition of a homotopy class only involving pointwise limits, as opposed to limits in bounded variation, making the conditions of a *robustly optimal solution* potentially vacuously satisfied.

3 The Fast Marching Tree Algorithm (FMT*)

In this section we present the Fast Marching Tree algorithm, FMT*, described in pseudocode in Algorithm 1.

3.1 High-Level Description

At a high level, FMT* performs a forward dynamic programming recursion over a set of sampled vertices, and correspondingly generates a *tree* by moving steadily outward in cost-to-come space (see Fig. 1). The dynamic programming recursion performed by FMT* is characterized by two main adaptations (that make the algorithm "lazy"). First, two nodes are considered *neighbors* (hence connectable) if their distance is below a given *bound*, referred to as connectivity radius. Note that according to this definition two vertices are "lazily" considered neighbors even if the straight line joining them intersects an obstacle. The choice of the connectivity radius relies on a trade-off between computational complexity (roughly speaking, more neighbors lead to more computation) and quality of the computed path (roughly speaking, more neighbors lead to more paths one can optimize over), and is a central aspect for the analysis of FMT*. Second, for the evaluation of the immediate cost in the dynamic programming recursion, one "lazily" neglects the presence of obstacles, and when-

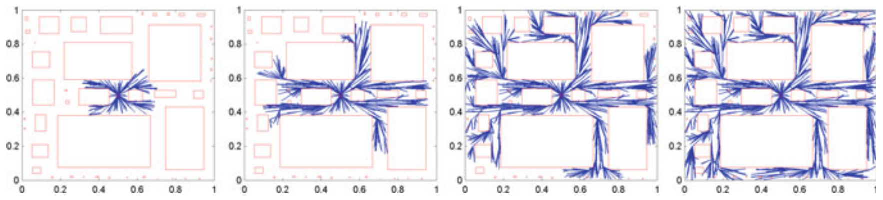


Fig. 1 The FMT* algorithm generates a tree by moving steadily outward in cost-to-come space. This figure portrays the growth of the tree in a 2D environment with 1000 nodes (not shown)

Algorithm 1 Fast Marching Tree Algorithm (FMT*)

```
1: $V \leftarrow \{x_{\text{init}}\} \cup \texttt{SampleFree}(n)$; $E \leftarrow \emptyset$
2: $W \leftarrow V \setminus \{x_{\text{init}}\}$; $H \leftarrow \{x_{\text{init}}\}$
3: $z \leftarrow x_{\text{init}}$
4: $N_z \leftarrow \texttt{Near}(V \setminus \{z\}, z, r_n)$
5: $\texttt{Save}(N_z, z)$
6: while $z \notin \mathcal{X}_{\text{goal}}$ do
7:   $H_{\text{new}} \leftarrow \emptyset$
8:   $X_{\text{near}} = \texttt{Intersect}(N_z, W)$
9:   for $x \in X_{\text{near}}$ do
10:     $N_x \leftarrow \texttt{Near}(V \setminus \{x\}, x, r_n)$
11:     $\texttt{Save}(N_x, x)$
12:     $Y_{\text{near}} \leftarrow \texttt{Intersect}(N_x, H)$
13:     $y_{\min} \leftarrow \arg\min_{y \in Y_{\text{near}}} \{\texttt{Cost}(y, T = (V, E)) + \texttt{Cost}(\overline{yx})\}$
14:     if $\texttt{CollisionFree}(y_{\min}, x)$ then
15:       $E \leftarrow \texttt{Merge}(E, \{(y_{\min}, x)\})$ {   // $\overline{y_{\min}x}$ is collision-free}
16:       $H_{\text{new}} \leftarrow \texttt{Merge}(H_{\text{new}}, \{x\})$
17:       $W \leftarrow W \setminus \{x\}$
18:     end if
19:   end for
20:   $H \leftarrow \texttt{Merge}(H, H_{\text{new}}) \setminus \{z\}$
21:   if $H = \emptyset$ then
22:     return Failure
23:   end if
24:   $z \leftarrow \arg\min_{y \in H} \{\texttt{Cost}(y, T = (V, E))\}$
25: end while
26: return $\texttt{Path}(z, T = (V, E))$
```

ever a locally-optimal (assuming no obstacles) connection to a new vertex intersects an obstacle, that vertex is simply skipped and left for later (as opposed to looking for other locally-optimal connections in the neighborhood), see line 14 in Algorithm 1. The reason this is possible, as we show in the proof of Theorem 1, is that the cases where an optimal connection under this strategy is not made become vanishingly rare as $n \to \infty$. This manifests itself into a key computational advantage. By only checking for collision on the locally-optimal (assuming no obstacles) connection, as opposed to every possible connection (essentially what is done in PRM*), FMT* saves a large (indeed unbounded as the number of vertices increases) number of costly collision-check computations.

As a more specific comparison to PRM*, when there are no obstacles and the cost is Euclidean distance, FMT* reports the exact same solution (or failure) as PRM*. This is because, without obstacles, FMT* is indeed using dynamic programming to build the minimum-cost spanning tree, with x_{init} as the root, of the set of nodes in the PRM* graph which are connected to x_{init}. The only difference is that by not starting with the entire PRM* graph itself, FMT* is able to find the solution faster (see Sect. 5). In the presence of obstacles, FMT* and PRM* no longer return the same solution in general, and this is due to how FMT* deals with obstructing obstacles. Looking at Algorithm 1, FMT* maintains dual sets H and W, where H keeps track of nodes which have already been added to the tree, although it drops nodes that are not near enough to the edge of the expanding tree to actually have any new connections made. The set W maintains the nodes which have not been added to the tree. When FMT* searches for a connection for a node v, it will leave v unconnected (and let it remain in W to be checked again in later iterations) if the node $x \in H$ whose connection (if obstacle-free) would produce the smallest cost-to-come for v is blocked from connecting to v by an obstacle. Assuming x_0 is the optimal parent of v with respect to the PRM* graph, v will never be connected to x_0 in FMT* only if when x_0 is the minimum-cost node in H, there is another node $x_1 \in H$ such that (a) x_1 has (necessarily, by the structure of H) greater cost-to-come than x_0, (b) x_1 is within a radius r_n of v, (c) x_1 is blocked from connecting to v by an obstacle, and (d) obstacle-free connection of v to x_1 would have lower cost-to-come than connection to x_0. If any of these conditions fail, then on some iteration (possibly not the first), v will be connected optimally with respect to PRM*. Note that the combination of conditions (a), (b), (c), and (d) ought to make such suboptimal connections quite rare (more specifically, vanishingly rare as $n \to \infty$).

3.2 Detailed Description

Let `SampleFree`(k) be a function that returns a set of $k \in \mathbb{N}$ points sampled independently and identically from the uniform distribution on $\mathcal{X}_{\text{free}}$. We use the uniform distribution in this paper for simplicity, but any distribution supported on $\mathcal{X}_{\text{free}}$ would yield identical theoretical results, in particular AO. Given a vertex $x \in \mathcal{X}$ and a set of vertices V, let `Save`(V, x) be a function that stores in memory a set of vertices V associated with a vertex x. Given a set of vertices V and a positive number r, let `Near`(V, x, r) be a function that returns the set of vertices $\{v \in V : \|v - x\| < r\}$. Given a graph $G = (V, E)$, where V is the vertex set and E is the edge set, and a vertex $x \in V$, let `Cost`(x, G) be a function that returns the cost of the shortest path in the graph G between the vertices x_{init} and x. With a slight abuse of notation, we define the function `Cost`$(\overline{vz})$ as the function that returns the cost of the line $\overline{vz}$ (note that `Cost`$(\overline{vz})$ is well defined regardless of $\overline{vz}$ being collision free). Given two vertices x and z in $\mathcal{X}_{\text{free}}$, let `CollisionFree`$(x, z)$ denote the Boolean function which is true if and only if $\overline{xz}$ does not intersect an obstacle. Given two sets of points $\mathcal{S}_1$ and $\mathcal{S}_2$ in $\mathcal{X}_{\text{free}}$, let `Intersect`$(\mathcal{S}_1, \mathcal{S}_2)$ be the function that returns the set of points that belong to both $\mathscr{S}_1$ and $\mathscr{S}_2$. Given two sets $\mathscr{S}_1$ and $\mathscr{S}_2$, let $\texttt{Merge}(\mathscr{S}_1, \mathscr{S}_2)$ be the function that returns the union of the two sets. Given a tree $T = (V, E)$, where V is the vertex set and E is the edge set, and a vertex $x \in T$, let $\texttt{Path}(x, T)$ be the function returning the unique path in the tree T from x_{init} to x. The algorithm is given in Algorithm 1.

In the extended version of this paper [8], we show that FMT* always terminates in at most n iterations. We also present fairly in-depth implementation details to ensure optimized algorithm speed. Finally, we show that the computational complexity of FMT* is $O(n \log(n))$, matching that of PRM* and RRT*. Proofs of these results are omitted here due to space limitations.

In the next section we discuss the (asymptotic) optimality of FMT*.

4 Asymptotic Optimality of FMT*

The following theorem presents the main result of this paper.

Theorem 1 (Asymptotic optimality of FMT*) *Let $(\mathscr{X}_{free}, x_{init}, \mathscr{X}_{goal})$ be a δ-robustly feasible path planning problem in d dimensions, with $\delta > 0$ and $\mathscr{X}_{goal}$ ξ-regular. Let c^* denote the arc length of an optimal path σ^*, and let c_n denote the cost of the path returned by FMT* (or ∞ if FMT* returns failure) with n vertices using the following radius,*

$$r_n = (1+\eta)\, 2 \left(1/d\right)^{1/d} \left(\mu(\mathscr{X}_{free})/\zeta_d\right)^{1/d} \left(\log(n)/n\right)^{1/d}, \tag{1}$$

for some $\eta > 0$. Then $\lim_{n\to\infty} \mathbb{P}\left(c_n > (1+\varepsilon)c^*\right) = 0$ for all $\varepsilon > 0$.

Proof Note that $c^* = 0$ implies $x_{\text{init}} \in cl(\mathscr{X}_{\text{goal}})$, and the result is trivial, therefore assume $c^* > 0$. Fix $\theta \in (0, 1/4)$ and define the sequence of paths σ_n such that $\lim_{n\to\infty} c(\sigma_n) = c^*$, $\sigma_n(1) \in \partial\mathscr{X}_{\text{goal}}$, $\sigma_n(\tau) \notin \mathscr{X}_{\text{goal}}$ for all $\tau \in (0, 1)$, $\sigma_n(0) = x_{\text{init}}$, and σ_n has strong δ_n-clearance, where $\delta_n = \min\left\{\delta, \frac{3+\theta}{2+\theta} r_n\right\}$. Such a sequence of paths must exist by the δ-robust feasibility of the path planning problem.

Let σ_n' be the concatenation of σ_n with the line that extends from $\sigma_n(1)$ in the direction perpendicular to the tangent hyperplane of $\partial\mathscr{X}_{\text{goal}}$ at $\sigma_n(1)$ of length $\min\left\{\xi, \frac{r_n}{2(2+\theta)}\right\}$. Note that this tangent hyperplane is well-defined, since the regularity assumption for $\mathscr{X}_{\text{goal}}$ ensures that its boundary is differentiable. Note that, trivially, $\lim_{n\to\infty} c(\sigma_n') = \lim_{n\to\infty} c(\sigma_n) = c^*$.

Fix $\varepsilon \in (0, 1)$, suppose $\alpha, \beta \in (0, \theta\varepsilon/8)$, and pick $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$ the following conditions hold: (1) $\frac{r_n}{2(2+\theta)} < \xi$, (2) $\frac{3+\theta}{2+\theta} r_n < \delta$, (3) $c(\sigma_n') < (1 + \frac{\varepsilon}{4})c^*$, and (4) $\frac{r_n}{2+\theta} < \frac{\varepsilon}{8} c^*$.

For the remainder of this proof, assume $n \geq n_0$. From conditions (1) and (2), σ_n' has strong $\frac{3+\theta}{2+\theta} r_n$-clearance. Letting $\kappa(\alpha, \beta, \theta) := 1 + (2\alpha + 2\beta)/\theta$, conditions (3) and (4) imply: $\kappa(\alpha, \beta, \theta)\, c(\sigma_n') + \frac{r_n}{2+\theta} \leq \kappa(\alpha, \beta, \theta)\left(1+\frac{\varepsilon}{4}\right) c^* + \frac{\varepsilon}{8} c^* \leq \left(\left(1+\frac{\varepsilon}{2}\right)\left(1+\frac{\varepsilon}{4}\right) + \frac{\varepsilon}{8}\right)c^* \leq (1+\varepsilon)c^*$. Therefore,

$$\mathbb{P}\left(c_n > (1+\varepsilon)c^*\right) = 1 - \mathbb{P}\left(c_n \leq (1+\varepsilon)c^*\right) \leq 1 - \mathbb{P}\left(c_n \leq \kappa(\alpha, \beta, \theta)\, c(\sigma_n') + \tfrac{r_n}{2+\theta}\right). \tag{2}$$

Define the sequence of balls $B_{n,1}, \ldots, B_{n,M_n} \subseteq \mathscr{X}_{\text{free}}$ parameterized by θ as follows. For $m = 1$ we define $B_{n,1} := B\Big(\sigma_n(\tau_{n,1});\ \frac{r_n}{2+\theta}\Big)$, with $\tau_{n,1} = 0$. For $m = 2, 3, \ldots,$ let $\varGamma_m = \Big\{\tau \in (\tau_{n,m-1}, 1) : \|\sigma_n(\tau) - \sigma_n(\tau_{n,m-1})\| = \frac{\theta r_n}{2+\theta}\Big\}$; if $\varGamma_m \neq \emptyset$ we define $B_{n,m} := B\Big(\sigma_n(\tau_{n,m});\ \frac{r_n}{2+\theta}\Big)$, with $\tau_{n,m} = \min_\tau\ \varGamma_m$. Let M_n be the first m such that $\varGamma_m = \emptyset$, then, $B_{n,M_n} := B\Big(\sigma_n'(1);\ \frac{r_n}{2(2+\theta)}\Big)$, and we stop the process, i.e., B_{n,M_n} is the last ball placed along the path σ_n (note that the center of the last ball is $\sigma_n'(1)$). Considering the construction of σ_n' and condition (1) above, we conclude that $B_{n,M_n} \subseteq \mathscr{X}_{\text{goal}}$. Recall that V is the set of nodes available to algorithm FMT* (see line 1 in Algorithm 1). We define the event $A_{n,\theta} := \bigcap_{m=1}^{M_n} \{B_{n,m} \cap V \neq \emptyset\}$; $A_{n,\theta}$ is the event that each ball contains at least one (not necessarily unique) node in V (for clarity, we made the event's dependence on θ, due to the dependence on θ of the balls, explicit). Further, for all $m \in \{1, \ldots, M_n - 1\}$, let $B_{n,m}^{\beta}$ be the ball with the same center as $B_{n,m}$ and radius $\frac{\beta r_n}{2+\theta}$, and let K_n^{β} be the number of smaller balls $B_{n,m}^{\beta}$ not containing any of the nodes in V, i.e., $K_n^{\beta} := \text{card}\{m \in \{1, \ldots, M_n - 1\} : B_{n,m}^{\beta} \cap V = \emptyset\}$. We now present three important lemmas, the first of which is proved in the Appendix, and the other two of which are proved in the extended version of this paper available on arXiv [8].

Lemma 1 *Under the assumptions of Theorem 1 and assuming $n \geq n_0$, the following inequality holds:*

$$\mathbb{P}\left(c_n \leq \kappa(\alpha, \beta, \theta)\, c(\sigma_n') + \tfrac{r_n}{2+\theta}\right) \geq 1 - \mathbb{P}(K_n^{\beta} \geq \alpha(M_n - 1)) - \mathbb{P}(A_{n,\theta}^c).$$

Lemma 2 *Under the assumptions of Theorem 1, for all $\alpha \in (0, 1)$ and $\beta \in (0, \theta/2)$, it holds that:* $\lim_{n\to\infty} \mathbb{P}(K_n^{\beta} \geq \alpha(M_n - 1)) = 0$.

Lemma 3 *Under the assumptions of Theorem 1, assume that $r_n = \gamma\ (\log n/n)^{1/d}$, where $\gamma = (1 + \eta) \cdot 2\,(1/d)^{\frac{1}{d}}\,(\mu(\mathscr{X}_{free})/\zeta_d)^{1/d}$ and $\eta > 0$. Then for all $\theta < 2\eta$,* $\lim_{n\to\infty} \mathbb{P}(A_{n,\theta}^c) = 0$.

Essentially, Lemma 1 provides a lower bound for the cost of the solution delivered by FMT* in terms of the probabilities that the "big" balls and "small" balls do not contain vertices in V. Lemma 2 states that the probability that the fraction of small balls not containing vertices in V is larger than an α fraction of the total number of balls is asymptotically zero. Finally, Lemma 3 states that the probability that at least one "big" ball does not contain any of the vertices in V is asymptotically zero. The asymptotic optimality claim of the theorem then follows easily. Let $\varepsilon \in (0, 1)$ and pick $\theta \in (0, \min\{2\eta, 1/4\})$ and $\alpha, \beta \in (0, \theta\varepsilon/8) \subset (0, \theta/2)$. From Eq. (2) and Lemma 1, one can write

$$\lim_{n\to\infty} \mathbb{P}\left(c_n > (1+\varepsilon)c^*\right) \le \lim_{n\to\infty} \mathbb{P}\left(K_n^{\beta} \ge \alpha(M_n - 1)\right) + \lim_{n\to\infty} \mathbb{P}\left(A_{n,\theta}^c\right).$$

The right hand-side of this equation equals zero by Lemmas 2 and 3, and the claim is proven. The case with general ε follows by monotonicity in ε of the above probability. □

Since the solution returned by FMT* is never better than the one returned by PRM*, the *same* result holds for PRM*. Note that this proof uses a γ which is a factor of $(d+1)^{1/d}$ *smaller* than that in [9]. We conclude this section by presenting a characterization of the convergence rate of FMT* (and thus also of PRM*), *assuming no obstacles*. As far as the authors are aware, this is the first such convergence rate result for an optimal sampling-based motion-planning algorithm, and is an important step towards understanding the behavior of this class of algorithms. We note that the bound converges to zero very slowly, but we have not studied how tight it is—this is a potential area for future work.

Theorem 2 (Convergence Rate of FMT*) *Let the configuration space be* $[0,1]^d$ *with no obstacles and the goal region be* $[0,1]^d \cap B(\mathbf{1};\xi)$. *Taking* x_{init} *to be the center of the configuration space, the shortest path has length* $c^* = \sqrt{d}/2 - \xi$ *and has clearance* $\delta = \xi\sqrt{(d-1)/d}$. *Denote by* c_n *the cost of the path returned by FMT* with* n *points sampled and by* ζ_d *the volume of the ball in* d *dimensions. Let* $\eta > 0$, $\varepsilon > 0$, $\theta \in (0, 2\eta \bigwedge \frac{1}{4})$, $\alpha, \beta \in (0, \frac{\theta}{8}(1 \bigwedge \varepsilon))$, $\nu \in (0,1)$. *Let* $H(a) = 1 + a(\log(a) - 1)$, $\gamma = 2(1+\eta)\,(\frac{1}{d\zeta_d})^{1/d}$, *and* $r_n = \gamma\,(\log(n)/n)^{1/d}$. *Letting* $n_0 > (\alpha/e^2)^{-\frac{(2+\theta)}{\nu\zeta_d\beta^d\gamma^d}}$ *and such that* $r_{n_0} < \min\left\{2\,\xi(2+\theta), \frac{2+\theta}{3+\theta}\delta, \frac{\varepsilon(2+\theta)}{8}c^*\right\}$, *then for* $n \ge n_0$,

$$\begin{aligned}
&\mathbb{P}(c_n > (1+\varepsilon)c^*) \\
&\quad < \frac{1}{1 - e^{-\nu n H\left(\frac{n+1}{\nu n}\right)}} e^{-\frac{\alpha}{2}\left\lfloor \frac{2+\theta}{\theta r_n} c^* \right\rfloor \left(\log\left(\alpha \left\lfloor \frac{2+\theta}{\theta r_n} c^* \right\rfloor\right) + \zeta_d \left(\frac{\beta r_n}{2+\theta}\right)^d \nu n\right)} \\
&\qquad + \left\lfloor \frac{2+\theta}{\theta r_n} c^* \right\rfloor \left(1 - \zeta_d\left(\frac{r_n}{2+\theta}\right)^d\right)^n + \left(1 - \zeta_d\left(\frac{r_n}{2(2+\theta)}\right)^d\right)^n.
\end{aligned}$$

Proof The proof essentially relies in collecting the bounds obtained for the proof of Theorem 1. The details are provided in the extended version of this paper [8]. □

5 Numerical Experiments and Discussion

In this section we discuss the advantages of FMT* over previous sampling-based motion planning algorithms. To our knowledge, the main other asymptotically optimal algorithms are PRM* and RRT* [9], so it is with these state-of-the-art methods that we draw comparison. We first present a brief conceptual comparison between FMT* and such algorithms, and then we present results from numerical experiments.

As compared to RRT*, we expect FMT* to show some improvement in solution quality per number of nodes placed. This is because for a given set of nodes, FMT* creates connections nearly optimally (exactly optimally when there are no obstacles) within the radius constraints, while RRT*, even with its rewiring step, is ultimately a greedy algorithm. However, it is hard to conceptually compare how long the algorithms might take to run on a given set of nodes, given how differently they generate paths. One advantage of FMT* over PRM*, besides the reduction in the number of collision checks (see Sect. 3.1), is that FMT* builds paths in a tree-like structure at all times, which has some potential advantages when differential or integral constraints are added to the paths. The extension of FMT* to the kinodynamic case is, however, left for future research.

All simulations were run in C++, using a Linux operating system with a 2.4 GHz processor and 7.5 GB of RAM. The implementation of RRT* was taken from the Open Motion Planning Library (OMPL). We adjusted the search radius to match the lower bound in [9] plus 10 %, and used a steering parameter of 60 % of the maximum extent of the configuration space. Additionally, we implemented an "ellipsoidal" node-rejection criterion [1, Sect. III.D], which is *guaranteed* to improve the performance (i.e., the convergence rate) of RRT*. To ensure a fair comparison, both PRM* and FMT* were also implemented in OMPL. This means that RRT*, PRM*, and FMT* used the *exact same primitive routines* (e.g., nearest neighbor search, collision checking, data handling, etc.). Specifically, for collision checking we used a hashing scheme [13, 16], where the partition resolution was optimized via numerical experiments. Similarly as for RRT*, for both PRM* and FMT* we used the radius suggested in [9] for PRM* plus 10 % (in other words, we used an r_n of 10 % over the lower bound given there, as opposed to the smaller r_n lower bound presented in this paper). The configuration space was the unit hypercube, with the initial state x_{init} set to be the center of the hypercube, and $\mathscr{X}_{\text{goal}}$ set to be the ball of radius $0.001^{1/d}$, centered at the 1-vector. The algorithms were run on *identical* sample sets; in case a sample set did not contain any nodes in $\mathscr{X}_{\text{goal}}$, the 1-vector (belonging, by construction, to $\mathscr{X}_{\text{goal}}$) was added at the end of that sample set. (The implementation of FMT* and the code used for algorithm comparison are available at: http://www.stanford.edu/~pavone/code/fmt.)

We performed two sets of simulations. In the first set of simulations a robot was modeled as a point mass. This scenario *strongly* favors PRM* against FMT*, as collision checking is computationally inexpensive. In the second set of simulations a robot was modeled as a d-dimensional *cuboid*. This scenario models the more realistic setting where collision checking is computationally expensive, and arguably represents the typical operating regime of sampling-based algorithms [13]. This is the scenario where FMT* is expected to prove its superior performance with respect to PRM* (see Sect. 3.1).

Figures 2 and 3 show the results of simulation runs for the first simulation scenario (i.e., the one with a point-mass robot) in 5 and 10 dimensions, respectively, with no obstacles and 50 % obstacle coverage (the obstacles are hyperrectangles). The points on the plots represent simulations of the three algorithms on node sets increasing in size to the right. The maximum number of nodes used varied slightly across graphs,

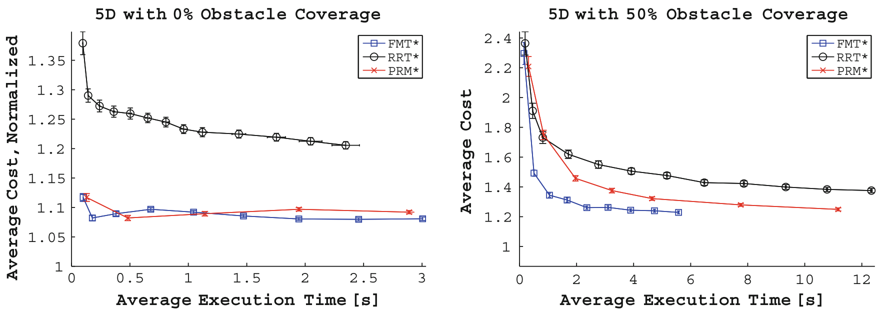


Fig. 2 Simulation results in 5 dimensions with and without obstacles for a *point-mass robot*. *Left figure* (normalized) cost versus time with 0 % obstacle coverage. *Right figure* cost versus time with 50 % obstacle coverage

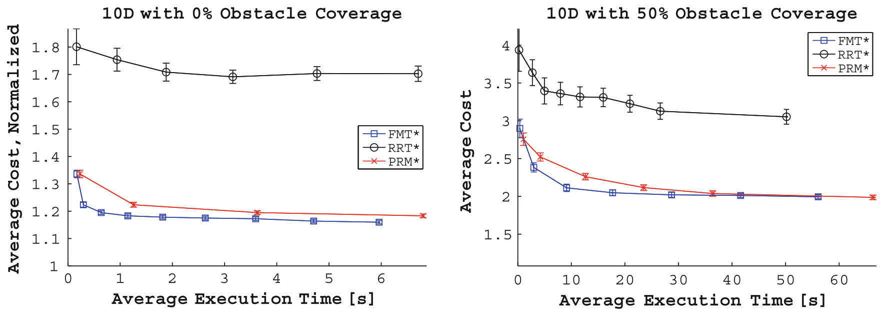


Fig. 3 Simulation results in 10 dimensions with and without obstacles for a *point mass robot*. *Left figure* (normalized) cost versus time with 0 % obstacle coverage. *Right figure* cost versus time with 50 % obstacle coverage

but FMT* always went up to 4000 nodes, PRM* went up to about 2000 nodes, and RRT* went up to about 4000 nodes in all obstacle-free graphs, and up to about 35000 in the graphs with 50 % obstacles. The error bars represent plus and minus one standard error of the mean at each node, reflecting that different graphs represent different numbers of simulations, 100 in 5D, and 50 in 10D. In all the figures, FMT* dominates the other two algorithms, in that the FMT* curve is below and to the left of the other two. Specifically, note that in Fig. 3, the solution of RRT* never dips below the solution of FMT* on the smallest node set, making it hard to estimate the speedup, but it is clearly multiple orders of magnitude. As compared to PRM*, FMT* also provides significant speedups (no less than a factor of three in any of the graphs, if one imagines drawing a horizontal line and comparing where it crosses the two curves). While both curves seem to plateau, FMT* reaches that plateau faster than PRM*. As noted above, this simulation scenario is characterized by inexpensive collision checking, so we argue that the speedup factors shown by Figs. 2 and 3 represent "minimum" speedups provided by FMT* (which is specifically designed to minimize the number of collision-check calls) over the other algorithms. We performed similar experiments in 2 and 7 dimensions: the general trend is essentially identical and the results are omitted in the interest of brevity. We also simulated RRT* by including *goal biasing*, which is implemented by attempting a connection to the goal region every 100 samples [13, p. 235]. Goal biasing somewhat improves the success rate (measured as the fraction of times the algorithm returns a feasible solution) of RRT*, but the general trend is essentially the same (the results are within 5 % of the results shown in Figs. 2 and 3 for RRT*) and, again, the results are omitted in the interest of brevity.

We also note that, although it is not exactly clear from these plots because the points are not labeled with the number of nodes, when all three algorithms are run on the same number of nodes, the curve for PRM* looks like that for FMT* but shifted to the right (similar solution quality, but slower), while that of RRT* looks like that of FMT* but shifted up and to the left (faster per node, but much worse solution quality). This agrees with our heuristic analysis. It is of some interest to note that in 10D with 50 % obstacle coverage, RRT* achieved a success rate ranging from 12 % for low sample counts (e.g., 100 nodes = 12 %) to approximately 95 % for high sample counts (35000 nodes = 96 %), while FMT* had a 100 % success rate for all but the smallest node set (100 nodes = 96 %). Finally, although the error bars depend on the number of simulations, each graph used the same number of simulations for each point, and thus it is noteworthy that RRT*'s error bars are uniformly larger than those of FMT*. This means that the solutions returned by FMT* are both higher quality and more consistent.

Figure 4 shows the results of simulation runs in 5 and 10 dimensions with 50 % obstacle coverage for the second simulation scenario (i.e., with a cuboid robot). In 5D, we use the same number of samples as in the point mass scenario. In 10D, FMT* was run up to 600 samples, PRM* was run up to 100 samples, and RRT* was run up to 4000 samples. Each curve starts at a point where the success rate is larger than 75 % (as in the point-mass scenario, the success rates for a given execution time is significantly larger for PRM* and FMT* than for RRT*). In both figures, FMT* outperforms the other two algorithms by orders of magnitude. As discussed, in this

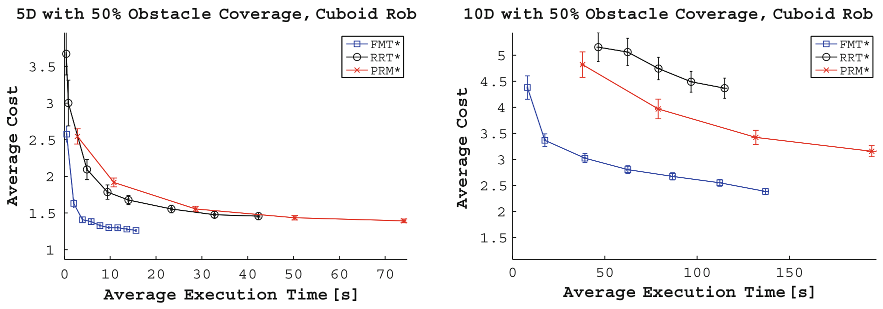


Fig. 4 Simulation results in 5 and 10 dimensions with obstacles for a *cuboid robot*. *Left figure* cost versus time in 5 dimensions with 50 % obstacle coverage. *Right figure* cost versus time in 10 dimensions with 50 % obstacle coverage

scenario collision-checking is more computationally expensive, which explains the significant move of the PRM* curve to the top-right corner of the plots (see also Sect. 3.1).

6 Conclusions

In this paper we have introduced and analyzed a novel probabilistic sampling-based motion planning algorithm called Fast Marching Tree algorithm (FMT*). This algorithm is asymptotically optimal and appears to converge *significantly* faster than its state-of-the-art counterparts. The speedups are particularly prominent in higher dimensions and in scenarios where collision checking is expensive, which is exactly the regime in which sampling-based algorithms are particularly useful. We used the weaker notion of convergence in probability, as opposed to convergence almost surely, and showed that the extra mathematical flexibility provides substantial theoretical and algorithmic benefits, including convergence rate bounds.

This paper leaves numerous important extensions open for further research. First, it is of interest to extend the FMT* algorithm to address problems with differential motion constraints and in non-metric spaces (relevant, e.g., for information-planning). Second, we plan to further explore the convergence rate bounds provided by the proof of AO given here. Third, we plan to use this algorithm as the backbone for scalable information-theoretic planning algorithms. Fourth, we plan to extend the FMT* algorithm for solving the eikonal equation. Finally, we plan to test the performance of FMT* on mobile ground robots and robotic manipulators operating in dynamic environments.

Acknowledgments The authors gratefully acknowledge the contributions of Ashley Clark, Edward Schmerling, Tim Wheeler, and Wolfgang Pointner to the implementation of FMT*. This work was supported by NASA under the Space Technology Research Grants Program, Grant NNX12AQ43G.

Appendix

Proof (*Proof of Lemma 1*) To start, note that $\mathbb{P}(K_n^\beta \geq \alpha(M_n - 1)) + \mathbb{P}(A_n^c) \geq \mathbb{P}(\{K_n^\beta \geq \alpha(M_n - 1)\} \cup A_n^c) = 1 - \mathbb{P}(\{K_n^\beta < \alpha(M_n - 1)\} \cap A_n)$, where the first inequality follows from the union bound and the second equality follows from De Morgan's laws. Note that the event $\{K_n^\beta < \alpha(M_n - 1)\} \cap A_n$ is the event that each $B_{n,m}$ contains at least one node, and more than a $1 - \alpha$ fraction of the $B_{n,m}^\beta$ balls also contains at least one node.

When two nodes x_i and x_{i+1}, $i \in \{1, \ldots, M_n - 2\}$, are contained in adjacent balls $B_{n,i}$ and $B_{n,i+1}$, respectively, their distance apart $\|x_{i+1} - x_i\|$ can be upper bounded by,

$$\begin{cases} \frac{\theta r_n}{2+\theta} + \frac{\beta r_n}{2+\theta} + \frac{\beta r_n}{2+\theta} : & \text{if } x_i \in B^{\beta}_{n,i} \text{ and } x_{i+1} \in B^{\beta}_{n,i+1} \\ \frac{\theta r_n}{2+\theta} + \frac{\beta r_n}{2+\theta} + \frac{r_n}{2+\theta} : & \text{if } x_i \in B^{\beta}_{n,i} \text{ or } x_{i+1} \in B^{\beta}_{n,i+1} \\ \frac{\theta r_n}{2+\theta} + \frac{r_n}{2+\theta} + \frac{r_n}{2+\theta} : & \text{otherwise,} \end{cases}$$

where the three bounds have been suggestively divided into a term for the distance between ball centers and a term each for the radii of the two balls containing the nodes. This bound also holds for $\|x_{M_n} - x_{M_n-1}\|$, although necessarily in one of the latter two bounds, since B^{β}_{n,M_n} being undefined precludes the possibility of the first bound. Thus we can rewrite the above bound, for $i \in \{1, \ldots, M_n - 1\}$, as $\|x_{i+1} - x_i\| \leq \bar{c}(x_i) + \bar{c}(x_{i+1})$, where

$$\bar{c}(x_k) := \begin{cases} \frac{\theta r_n}{2(2+\theta)} + \frac{\beta r_n}{2+\theta} : x_k \in B^{\beta}_{n,k}, \\ \frac{\theta r_n}{2(2+\theta)} + \frac{r_n}{2+\theta} : x_k \notin B^{\beta}_{n,k}. \end{cases} \tag{3}$$

Again, $\bar{c}(x_{M_n})$ is still well-defined, but always takes the second value in Eq. (3) above. Let $L_{n,\alpha,\beta}$ be the length of a path that sequentially connects a set of nodes $\{x_1 = x_{\text{init}}, x_2, \ldots, x_{M_n}\}$, such that $x_m \in B_{n,m}$ $\forall m \in \{1, \ldots, M_n\}$, and more than a $(1 - \alpha)$ fraction of the nodes $x_1, \ldots, x_{M_n-1}$ are also contained in their respective $B^{\beta}_{n,m}$ balls. The length $L_{n,\alpha,\beta}$ can then be upper bounded as follows

$$\begin{aligned} L_{n,\alpha,\beta} &= \sum_{k=1}^{M_n-1} \|x_{k+1} - x_k\| \leq \sum_{k=1}^{M_n-1} 2\bar{c}(x_k) - \bar{c}(x_1) + \bar{c}(x_{M_n}) \\ &\leq (M_n - 1)\frac{\theta r_n}{2+\theta} + \lceil (1-\alpha)(M_n - 1)\rceil \frac{2\beta r_n}{2+\theta} + \lfloor \alpha(M_n - 1)\rfloor \frac{2r_n}{2+\theta} + \frac{(1-\beta)r_n}{2+\theta} \\ &\leq (M_n - 1)\, r_n \frac{\theta + 2\alpha + 2(1-\alpha)\beta}{2+\theta} + \frac{(1-\beta)r_n}{2+\theta} \leq M_n\, r_n \frac{\theta + 2\alpha + 2\beta}{2+\theta} + \frac{r_n}{2+\theta}. \end{aligned} \tag{4}$$

In Eq. (4), $\lceil x \rceil$ denotes the smallest integer not less than x, while $\lfloor x \rfloor$ denotes the largest integer not greater than x. Furthermore, we can upper bound M_n as follows,

$$\begin{aligned} c(\sigma'_n) &\geq \sum_{k=1}^{M_n-2} \|\sigma_n(\tau_{k+1}) - \sigma_n(\tau_k)\| + \|\sigma'_n(1) - \sigma_n(\tau_{M_n-1})\| \geq (M_n - 2)\frac{\theta r_n}{2+\theta} + \frac{r_n}{2(2+\theta)} \\ &= M_n \frac{\theta r_n}{2+\theta} + \left(\frac{1}{2} - 2\theta\right)\frac{r_n}{2+\theta} \geq M_n \frac{\theta r_n}{2+\theta}, \end{aligned} \tag{5}$$

where the last inequality follows from the assumption that $\theta < 1/4$. Combining Eqs. (4) and (5) gives

$$L_{n,\alpha,\beta} \leq c(\sigma'_n)\left(1 + \frac{2\alpha + 2\beta}{\theta}\right) + \frac{r_n}{2+\theta} = \kappa(\alpha,\beta,\theta)\, c(\sigma'_n) + \frac{r_n}{2+\theta}. \tag{6}$$

We will now show that when A_n occurs, c_n is no more than the length of the path connecting any sequence of M_n vertices tracing through the balls $B_{n,1}, \ldots, B_{n,M_n}$ (this of course also implies $c_n < \infty$). Coupling this fact with Eq. (6), one can then conclude that the event $\{K_n^{\beta} < \alpha(M_n - 1)\} \cap A_n$ implies that $c_n \leq \kappa(\alpha, \beta, \theta)\, c(\sigma_n') + \frac{r_n}{2+\theta}$, which, in turn, would prove the lemma.

Let $x_1 = x_{\text{init}}$, $x_2 \in B_{n,2}, \ldots, x_{M_n} \in B_{n,M_n} \subseteq \mathcal{X}_{\text{goal}}$. Note that the x_i's need not all be distinct. The following property holds for all $m \in \{2, \ldots, M_n - 1\}$: $\|x_m - x_{m-1}\| \leq \|x_m - \sigma_n(\tau_m)\| + \|\sigma_n(\tau_m) - \sigma_n(\tau_{m-1})\| + \|\sigma_n(\tau_{m-1}) - x_{m-1}\| \leq \frac{r_n}{2+\theta} + \frac{\theta r_n}{2+\theta} + \frac{r_n}{2+\theta} = r_n$. Similarly, one can write $\|x_{M_n} - x_{M_n - 1}\| \leq \frac{r_n}{2+\theta} + \frac{(\theta+1/2) r_n}{2+\theta} + \frac{r_n}{2(2+\theta)} = r_n$. Furthermore, we can lower bound the distance to the nearest obstacle for $m \in \{2, \ldots, M_n - 1\}$ by $\inf_{w \in X_{\text{obs}}} \|x_m - w\| \geq \inf_{w \in X_{\text{obs}}} \|\sigma_n(\tau_m) - w\| - \|x_m - \sigma_n(\tau_m)\| \geq \frac{3+\theta}{2+\theta} r_n - \frac{r_n}{2+\theta} = r_n$, where the second inequality follows from the assumed δ_n-clearance of the path σ_n. Again, similarly, one can write $\inf_{w \in X_{\text{obs}}} \|x_{M_n} - w\| \geq \inf_{w \in X_{\text{obs}}} ||x_m - \sigma_n(1)\| - \|\sigma_n(1) - w\| \geq \frac{3+\theta}{2+\theta} r_n - \frac{r_n}{2+\theta} = r_n$. Together, these two properties imply that, for $m \in \{2, \ldots, M_n\}$, when a connection is attempted for x_m, x_{m-1} will be in the search radius and there will be no obstacles in that search radius. In particular, this implies that either the algorithm will return a feasible path before considering x_{M_n}, or it will consider x_{M_n} and connect it. Therefore, FMT* is guaranteed to return a feasible solution when the event A_n occurs. Since the remainder of this proof assumes that A_n occurs, we will also assume $c_n < \infty$.

Finally, assuming x_m is contained in an edge, let $c(x_m)$ denote the (unique) cost-to-come of x_m in the graph generated by FMT* at the end of the algorithm, just before the path is returned. If x_m is not contained in an edge, we set $c(x_m) = \infty$. Note that $c(\cdot)$ is well-defined, since if x_m is contained in any edge, it must be connected through a unique path to x_{init}. We claim that for all $m \in \{2, \ldots, M_n\}$, either $c_n \leq \sum_{k=1}^{m-1} \|x_{k+1} - x_k\|$, or $c(x_m) \leq \sum_{k=1}^{m-1} \|x_{k+1} - x_k\|$. In particular, taking $m = M_n$, this would imply that $c_n \leq \min\{c(x_{M_n}), \sum_{k=1}^{M_n - 1} \|x_{k+1} - x_k\|\} \leq \sum_{k=1}^{M_n - 1} \|x_{k+1} - x_k\|$, which, as argued before, would imply the claim.

The claim is proved by induction on m. The case of $m = 1$ is trivial, since the first step in the FMT* algorithm is to make every collision-free connection between $x_{\text{init}} = x_1$ and the nodes contained in $B(x_{\text{init}}; r_n)$, which will include x_2 and, thus, $c(x_2) = \|x_2 - x_1\|$. Now suppose the claim is true for $m - 1$. There are four exhaustive cases to consider:

1. $c_n \leq \sum_{k=1}^{m-2} \|x_{k+1} - x_k\|$,
2. $c(x_{m-1}) \leq \sum_{k=1}^{m-2} \|x_{k+1} - x_k\|$ and FMT* ends before considering x_m,
3. $c(x_{m-1}) \leq \sum_{k=1}^{m-2} \|x_{k+1} - x_k\|$ and $x_{m-1} \in H$ when x_m is first considered,
4. $c(x_{m-1}) \leq \sum_{k=1}^{m-2} \|x_{k+1} - x_k\|$ and $x_{m-1} \notin H$ when x_m is first considered.

Case 1: $c_n \leq \sum_{k=1}^{m-2} \|x_{k+1} - x_k\| \leq \sum_{k=1}^{m-1} \|x_{k+1} - x_k\|$, thus the claim is true for m. Without loss of generality, for cases 2–4 we assume that case 1 does not occur.

Case 2: $c(x_{m-1}) < \infty$ implies that x_{m-1} enters H at some point during FMT*. However, if x_{m-1} were ever the minimum-cost element of H, x_m would have been considered, and thus FMT* must have returned a feasible solution before x_{m-1} was ever the minimum-cost element of H. Since the end-node of the solution returned must have been the minimum-cost element of H, $c_n \leq c(x_{m-1}) \leq \sum_{k=1}^{m-2} \|x_{k+1} - x_k\| \leq \sum_{k=1}^{m-1} \|x_{k+1} - x_k\|$, thus the claim is true for m.

Case 3: $x_{m-1} \in H$ when x_m is first considered, $\|x_m - x_{m-1}\| \leq r_n$, and there are no obstacles in $B(x_m; r_n)$. Therefore, x_m must be connected to some parent when it is first considered, and $c(x_m) \leq c(x_{m-1}) + \|x_m - x_{m-1}\| \leq \sum_{k=1}^{m-1} \|x_{k+1} - x_k\|$, thus the claim is true for m.

Case 4: When x_m is first considered, there must exist $z \in B(x_m; r_n)$ such that z is the minimum-cost element of H, while x_{m-1} has not even entered H yet. Note that again, since $B(x_m; r_n)$ intersects no obstacles and contains at least one node in H, x_m must be connected to some parent when it is first considered. Since $c(x_{m-1}) < \infty$, there is a well-defined path $\mathscr{P} = \{v_1, \ldots, v_q\}$ from $x_{\text{init}} = v_1$ to $x_{m-1} = v_q$ for some $q \in \mathbb{N}$. Let $w = v_j$, where $j = \max_{i \in \{1,\ldots,q\}}\{i : v_i \in H$ when x_m is first considered$\}$. Then there are two subcases, either $w \in B(x_m; r_n)$ or $w \notin B(x_m; r_n)$. If $w \in B(x_m; r_n)$, then, $c(x_m) \leq c(w) + \|x_m - w\| \leq c(w) + \|x_{m-1} - w\| + \|x_m - x_{m-1}\| \leq c(x_{m-1}) + \|x_m - x_{m-1}\| \leq \sum_{k=1}^{m-1} \|x_{k+1} - x_k\|$, thus the claim is true for m (the second and third inequalities follow from the triangle inequality). If $w \notin B(x_m; r_n)$, then, $c(x_m) \leq c(z) + \|x_m - z\| \leq c(w) + r_n \leq c(x_{m-1}) + \|x_m - x_{m-1}\| \leq \sum_{k=1}^{m-1} \|x_{k+1} - x_k\|$, where the third inequality follows from the fact that $w \notin B(x_m, r_n)$, which means that any path through w to x_m, in particular the path $\mathscr{P} \cup x_m$, must traverse a distance of at least r_n between w and x_m. Thus, in the final subcase of the final case, the claim is true for m. Hence, we can conclude that $c_n \leq \sum_{k=1}^{M_n-1} \|x_{k+1} - x_k\|$. As argued before, coupling this fact with Eq. (6), one can conclude that the event $\{K_n^\beta < \alpha(M_n - 1)\} \cap A_n$ implies that $c_n \leq \kappa(\alpha, \beta, \theta)\, c(\sigma_n') + \frac{r_n}{2+\theta}$, and the claim follows. □

References

1. Akgun, B., Stilman, M.: Sampling heuristics for optimal motion planning in high dimensions. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2640–2645 (2011)
2. Alterovitz, R., Patil, S., Derbakovam, A.: Rapidly-exploring roadmaps: weighing exploration versus refinement in optimal motion planning. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 3706–3712 (2011)
3. Barraquand, J., Kavraki, L., Motwani, R., Latombe, J.-C., Li, T.-Y., Raghavan, P.: A random sampling scheme for path planning. In: International Journal of Robotics Research, pp. 249–264. Springer, New York (2000)
4. Bohlin, R., Kavraki, L.E.: Path planning using lazy PRM. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 521–528 (2000)
5. Hsu, D., Latombe, J.C., Motwani, R.: Path planning in expansive configuration spaces. Int. J. Comput. Geom. Appl. **9**, 495–512 (1999)
6. Hsu, D., Latombe, J.-C., Kurniawati, H.: On the probabilistic foundations of probabilistic roadmap planning. Int. J. Robot. Res. **25**(7), 627–643 (2006)
7. Jaillet, L., Siméon, T.: A PRM-based motion planner for dynamically changing environments. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 1606–1611 (2004)

8. Janson, L., Pavone, M.: Fast marching trees: a fast marching sampling-based method for optimal motion planning in many dimensions—extended version. http://arxiv.org/abs/1306.3532 (2013)
9. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. Int. J. Robot. Res. **30**(7), 846–894 (2011)
10. Kavraki, L.E., Svestka, P., Latombe, J.-C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. **12**(4), 566–580 (1996)
11. Kobilarov, M.: Cross-entropy motion planning. Int. J. Robot. Res. **31**(7), 855–871 (2012)
12. Ladd, A.M., Kavraki, L.E.: Measure theoretic analysis of probabilistic path planning. IEEE Trans. Robot. Autom. **20**(2), 229–242 (2004)
13. Lavalle, S.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
14. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. Int. J. Robot. Res. **20**(5), 378–400 (2001)
15. Marble, J.D., Bekris, K.E.: Towards small asymptotically near-optimal roadmaps. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 2557–2562 (2012)
16. Mirtich, B.: Efficient algorithms for two-phase collision detection. In: Practical Motion Planning in Robotics: Current Approaches and Future Directions, pp. 203–223. Wiley, New York (1997)
17. Phillips, J.M., Bedrossian, N., Kavraki, L.E.: Guided expansive spaces trees: a search strategy for motion- and cost-constrained state spaces. In: Proceedings of the IEEE Conference on Robotics and Automation, pp. 3968–3973 (2004)
18. Plaku, E., Bekris, K.E., Chen, B.Y., Ladd, A.M., Kavraki, L.E.: Sampling-based roadmap of trees for parallel motion planning. IEEE Trans. Robot. **21**(4), 597–608 (2005)
19. Sethian, J.A.: A fast marching level set method for monotonically advancing fronts. Proc. Natl. Acad. Sci. **93**(4), 1591–1595 (1996)
20. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. The MIT Press, Cambridge (2005)
21. Valero-Gomez, A., Gomez, J., Garrido, S., Moreno, L.: Fast marching methods in path planning. IEEE Robot. Autom. Mag., PP(99) (2013). To Appear

Safe Motion Planning for Imprecise Robotic Manipulators by Minimizing Probability of Collision

Wen Sun, Luis G. Torres, Jur van den Berg and Ron Alterovitz

Abstract Robotic manipulators designed for home assistance and new surgical procedures often have significant uncertainty in their actuation due to compliance requirements, cost constraints, and size limits. We introduce a new integrated motion planning and control algorithm for robotic manipulators that makes safety a priority by explicitly considering the probability of unwanted collisions. We first present a fast method for estimating the probability of collision of a motion plan for a robotic manipulator under the assumptions of Gaussian motion and sensing uncertainty. Our approach quickly computes distances to obstacles in the workspace and appropriately transforms this information into the configuration space using a Newton method to estimate the most relevant collision points in configuration space. We then present a sampling-based motion planner based on executing multiple independent rapidly exploring random trees that returns a plan that, under reasonable assumptions, asymptotically converges to a plan that minimizes the estimated collision probability. We demonstrate the speed and safety of our plans in simulation for (1) a 3-D manipulator with 6 DOF, and (2) a concentric tube robot, a tentacle-like robot designed for surgical applications.

W. Sun (✉) · L.G. Torres · R. Alterovitz
The University of North Carolina at Chapel Hill, Chapel Hill, USA
e-mail: wens@cs.unc.edu

L.G. Torres
e-mail: luis@cs.unc.edu

R. Alterovitz
e-mail: ron@cs.unc.edu

J. van den Berg
The University of Utah, Salt Lake City, USA
e-mail: berg@cs.utah.edu

M. Inaba and P. Corke (eds.), *Robotics Research*, Springer Tracts in Advanced Robotics 114, DOI 10.1007/978-3-319-28872-7_39

1 Introduction

As robotic manipulators enter emerging domains such as home assistance and enable new minimally-invasive surgeries, their designs are increasingly diverging from the designs of traditional manufacturing robots. Home assistance robotic manipulators must feature compliant joints for safety and must be lower cost to spur adoption, which results in decreased precision of sensors and actuators. Medical robots, such as tentacle-like and snake-like robots (e.g., [7, 9, 23, 32]), are becoming smaller and also are gaining larger numbers of degrees of freedom. These features are necessary to enable new, less invasive surgical procedures that require maneuvering around sensitive or impenetrable anatomical obstacles. These trends in robotic manipulator design and applications have an inevitable consequence: uncertainty in robot motion and state estimation. For robotic manipulators to operate with some level of autonomy in people's homes and inside people's bodies, uncertainty should explicitly be considered when planning motions to ensure both safety and task success.

In this paper, we introduce a new integrated motion planning and control algorithm for manipulators with uncertainty in actuation and sensing. Our objective is to compute plans and corresponding closed-loop controllers that a priori minimize the probability that any link of the robot will collide with an obstacle. To accomplish this objective, our algorithm incorporates two primary contributions. First, we introduce a fast and accurate method for assessing the quality of a manipulator motion plan by efficiently estimating the a priori probability of collision using fast numerical computations that do not require sampling in configuration space. Second, we introduce a sampling-based motion planner that, under reasonable assumptions, guarantees that the probability of finding a plan that minimizes the estimated probability of collision approaches 100 % as computation time is allowed to increase. We note that current planners such as RRT* [12] cannot guarantee asymptotic optimality for our problem since the optimal substructure property does not hold, i.e., the optimal plan from a particular state is not independent of the robot's prior history. Our approach is applicable to robotic manipulators for which uncertainty in actuation and sensing can be modeled using Gaussian distributions, a Kalman filter is used for state estimation, and an optimal linear controller (i.e., LQG control) is used to follow the plan. To the best of our knowledge, this is the first approach for computing a plan that minimizes the a priori probability of collision for general robotic manipulators.

For robots with motion and sensing uncertainty, collision detection during motion planning must be done in a probabilistic sense by considering the possibility of collision with respect to all possible states of the robot. Extensive prior work has investigated integrated motion planning and control under uncertainty for robots that can be approximated as points in the workspace (e.g., [4, 20, 26, 27]), but robotic manipulators raise new challenges. Whereas for point robots analytically estimating probability of collisions is facilitated by the fact that the robot's configuration space and workspace share parameters, for manipulators the configuration space and workspace are disjoint. Furthermore, manipulators (especially tentacle-like and snake-like medical robots) can have large numbers of degrees of freedom.

In our first contribution, we introduce a method to efficiently estimate the a priori probability of collision for a given plan for a robotic manipulator. Given the robot's uncertainty represented as a distribution in configuration space, efficiently estimating the probability of collision is challenging because the shapes of the obstacles are defined in the workspace and cannot be directly computed in configuration space in closed form [6]. The key insight of our method is that an appropriate minimization of distance from the robot to an obstacle in C-space corresponds to a minimum distance in the workspace. Hence, we propose a fast method using Newton's method to estimate the closest point in configuration space that will cause a collision, and use this information to estimate the probability of collision. We extend this formulation to multiple obstacles and then propagate these estimates over time (in a manner that considers dependencies across time steps) to estimate the a priori probability of collision for a plan.

In our second contribution, we present an asymptotically optimal motion planner that minimizes the estimated a priori probability of collision for a manipulator. Our motion planner is based on a simple idea: it generates a large number of plans, computes an optimal linear controller for each plan, then estimates the probability of collision for each plan using our approach above, and selects the best plan. By evaluating cost over entire plans, we properly handle the fact that the probability of collision from a configuration onwards depends on prior history. We show that, under reasonable assumptions, the computed plan will converge to a minimum estimated collision probability plan as computation time is allowed to increase.

We evaluate our method using simulated scenarios involving a 6-DOF manipulator and a tentacle-like robot designed for medical applications such as skull base surgery. Our results show that we can quickly estimate the a priori probability of collision of motion plans across highly distinct robotic manipulators and select plans that safely and robustly guide the robot's end effector to desired goals.

2 Related Work

Since uncertainty is inherent in many robotics applications, approaches for managing uncertainty have been investigated for a variety of settings. Our focus in this paper is on robots with uncertainty in their motion and state estimation; we do not consider uncertainty in sensing of obstacle locations (e.g., [11]) or grasping (e.g., [19]). Extensive prior work has investigated motion planning under uncertainty for mobile robots that can be approximated as points or spheres in the workspace, e.g., [2, 4, 8, 17, 18, 20, 21, 26, 28]. For point or spherical robots, computing an estimate of the probability of collision with obstacles can be done in the workspace since the geometry of the C-obstacles is low dimensional and can be directly computed. It is not trivial to directly extend these methods to robotic manipulators, which are typically articulated and are composed of more complex shapes. Our approach avoids complexity in the configuration space by computing distance only in the workspace, as has been done in other contexts [3]. Another approach to estimate the probability of collision is using Monte Carlo simulation in the space of uncertain parameters, but the computation time needed to run a sufficient number of simulations to achieve a desired accuracy can be prohibitive in some applications.

Our approach computes a plan and controller simultaneously to minimize probability of collision. Approaches that blend planning and control by defining a global control policy over the entire environment have been developed using Markov decision processes (MDPs) [2] and partially-observable MDPs (POMDPs) [14]. These approaches are difficult to scale, and computational costs may prohibit their application to robots with higher dimensional configuration spaces. Another class of approaches rely on sampling-based methods to compute a path and then compute an LQG feedback controller to follow that path [1, 4, 21, 26]. Other approaches compute a locally-optimal trajectory and an associated control policy [20, 27, 30]. Recent work has begun to investigate computing plans for manipulators that are robust to uncertainty using local optimization [15], but place restrictions on robot geometry and do not accurately estimate probability of collision.

For some applications, uncertainty in robot motion and sensing necessitates that the robot perform maneuvers purely to gain information. The general POMDP formulation enables such information gathering. However, this typically comes at the cost of additional computational complexity [14] or the ability to only compute locally optimal rather than globally optimal plans [27]. Although in this paper we address a broad class of problems, the use of sampling-based motion planning in our approach does place restrictions, e.g., the optimal plan must be goal-oriented (i.e., optimality is not guaranteed for problems that require returning to previously explored regions of the state space for information gathering). Because of these restrictions, our method does not address the general POMDP problem.

Sampling-based methods such as RRT* provide asymptotic optimality [12], but they are not suitable for finding the optimal plan for the cost metric of minimizing probability of collision because the required optimal substructure property does not hold. We show that our motion planner, under reasonable assumptions, is asymptotically optimal for goal-oriented problems when minimizing probability of collision.

3 Problem Definition and Overview

We consider an articulated robotic manipulator with l links operating in an environment with obstacles. Let $\mathcal{C}$ be the configuration space of the robot. Let $\mathbf{q} \in \mathcal{C}$ denote a configuration of the robot, which consists of the parameters over which the robot has control (e.g., joint angles). We assume we are given a description of the geometry $X(\mathbf{q})$ of the robot in the workspace for any given configuration $\mathbf{q} \in \mathcal{C}$ and a description of the geometry of the obstacles $\mathcal{O}$ in the workspace. The continuous time τ is discretized into periods with equal time duration Δ, and we define $\mathbf{q}(\tau)$ as the configuration at time τ. For simplicity, we define $\mathbf{q}_t = \mathbf{q}(t\Delta)$ for time step

$t \in \mathbb{N}$. Let $\mathbf{u} \in \mathcal{U}$ denote the robot's control inputs, which are provided at discrete time steps. At time step t, the dynamics of the robot evolves as

$$\mathbf{q}_{t+1} = f(\mathbf{q}_t, \mathbf{u}_t, \mathbf{m}_t), \quad \mathbf{m}_t \sim \mathcal{N}(\mathbf{0}, M) \tag{1}$$

where $\mathbf{m}_t$ is the process noise with variance M. We assume noisy and partial observation $\mathbf{z_t}$ can be obtained by the sensing model

$$\mathbf{z_t} = h(\mathbf{q}_t, \mathbf{n}_t), \quad \mathbf{n}_t \sim \mathcal{N}(\mathbf{0}, N), \tag{2}$$

where $\mathbf{n}_t$ is the sensing noise with variance N.

Our objective is to enable the robotic manipulator to move from a start configuration $\mathbf{q}_0$ to a goal $\mathcal{G} \subseteq \mathcal{C}$ in a manner that minimizes the probability of collision with obstacles. We define a motion plan π as a sequence of nominal configurations and corresponding control inputs, $\pi = \{\mathbf{q}_0, \mathbf{u}_0, \mathbf{q}_1, \mathbf{u}_1, \ldots, \mathbf{q}_T, \mathbf{u}_T\}$, where $\mathbf{q}_T \in \mathcal{G}$ and $\mathbf{u}_T = \mathbf{0}$. When executing a plan, we assume the robot uses an optimal linear controller (a linear quadratic Gaussian (LQG) feedback controller) in combination with a Kalman Filter for state estimation to guide the robot along the nominal plan.

In Sect. 4, we introduce a method to efficiently estimate the a priori probability of collision for a given plan. In Sect. 5 we introduce an asymptotically optimal motion planner that computes a plan that reaches a goal and asymptotically minimizes the a priori probability of collision.

4 Estimating Probability of Collision

In this section, we present our approach to estimating the probability of collision for a robotic manipulator moving along a planned trajectory. The approach works for complex workspace geometry and configuration spaces of arbitrary dimension and shape (including high-DOF manipulators).

We begin by considering the probability of collision when the robot is at a particular configuration with a given uncertainty distribution. We assume we have access to a collision-checker (e.g., [25]) that can compute the (signed) distance $d(\mathbf{q})$ in the workspace between the geometry of the robot $X(\mathbf{q})$ configured at $\mathbf{q}$ and the geometry of the obstacles $\mathcal{O}$ (the distance is negative if the robot collides with an obstacle, in which case the penetration depth is returned). The goal is to approximate the probability that the robot is in collision, i.e., $p(X(\mathbf{q}) \cap \mathcal{O} \neq \emptyset)$, given a Gaussian distribution of the configuration $\mathbf{q} \in \mathcal{C}$ of the robot;

$$\mathbf{q} \sim \mathcal{N}(\hat{\mathbf{q}}, \Sigma), \tag{3}$$

with mean $\hat{\mathbf{q}}$ and variance Σ.

4.1 Approach

Our approach is as follows. Let us assume that the geometry of the configuration space obstacles is (locally) convex in the neighborhood of $\hat{\mathbf{q}}$ (we will alleviate this assumption below). Then, we can approximate the free part of the configuration space by a single linear inequality constraint ($\mathbf{a}^T\mathbf{q} < b$) that is tangent to the configuration space obstacles at the point on the C-space obstacles "closest" to $\hat{\mathbf{q}}$. Given a linear inequality constraint, there is a closed-form expression for the probability that the constraint is violated if $\mathbf{q}$ has a Gaussian distribution [26], which serves as a (conservative) approximation of the probability that the robot is in collision. The challenge is to find a "closest" point on the boundary of the C-space obstacles, since we only have access to the geometry of the obstacles in the workspace. In our approach, we make use of one key relation between workspace geometry and configuration space geometry that holds in general: *configuration* $\mathbf{q} \in \mathcal{C}$ *lies on the boundary of a configuration space obstacle if and only if the workspace distance* $d(\mathbf{q})$ *between the robot* $X(\mathbf{q})$ *configured at* $\mathbf{q}$ *and the workspace obstacles* $\mathcal{O}$ *is zero* (i.e., the robot touches an obstacle).

In order to define "closest", we use the distance metric $\sqrt{(\mathbf{q}-\hat{\mathbf{q}})^T\Sigma^{-1}(\mathbf{q}-\hat{\mathbf{q}})}$. This ensures that the constraint includes as much probability mass as possible. Hence, decomposing $\Sigma = LL^T$ and defining a transformed configuration space $\mathcal{C}' = L^{-1}\mathcal{C}$ allows us to use the standard Euclidean distance metric $\sqrt{(\mathbf{q}'-\hat{\mathbf{q}}')^T(\mathbf{q}'-\hat{\mathbf{q}}')}$, where $\hat{\mathbf{q}}' = L^{-1}\hat{\mathbf{q}}$ is the mean of the distribution in the transformed configuration space $\mathcal{C}'$. Also, let us define a workspace distance function that takes in configurations $\mathbf{q}' \in \mathcal{C}'$ from the transformed configuration space:

$$d'(\mathbf{q}') = d(L\mathbf{q}'). \tag{4}$$

We are looking for a transformed configuration $\mathbf{q}'$ for which $d'(\mathbf{q}') = 0$ (i.e., a configuration $\mathbf{q}'$ on the boundary of the transformed C-space obstacles) that is closest to the transformed mean $\hat{\mathbf{q}}'$. For this, we use a variant of Newton's root finding method with $\hat{\mathbf{q}}'$ as the initial "guess." Newton's method (with line search) finds a root close to the given initial "guess," but does not guarantee to find the actual closest root. (We note that computing the true minimum distance point requires solving an optimization problem subject to a constraint that the solution is on the surface of a C-obstacle, which cannot be efficiently computed). Newton's method iteratively performs the following update:

$$\mathbf{q}'_{i+1} = \mathbf{q}'_i - d'(\mathbf{q}'_i)\frac{\partial d'}{\partial \mathbf{q}'}[\mathbf{q}'_i] / \left(\frac{\partial d'}{\partial \mathbf{q}'}[\mathbf{q}'_i]^T \frac{\partial d'}{\partial \mathbf{q}'}[\mathbf{q}'_i]\right), \tag{5}$$

with $\mathbf{q}'_0 = \hat{\mathbf{q}}'$. Here, $\frac{\partial d'}{\partial \mathbf{q}'}[\mathbf{q}'_i]$ is the gradient vector of d' at configuration $\mathbf{q}'_i$. The gradient points in the direction of steepest ascent of d' and has a magnitude equal to the slope of d' in that direction. Hence, if the function d' would be linear along the gradient, the above equation gives a configuration $\mathbf{q}'_{i+1}$ for which d' is zero. Since this is in general not the case, the equation is iterated, which lets it approach a root $\mathbf{q}'_\star$ of d' with a second-order convergence rate [16]. In our implementation, we use line search in Newton's method to ensure that the (absolute value) of the distance strictly decreases with each iteration, i.e., $|d'(\mathbf{q}'_{i+1})| < |d'(\mathbf{q}'_i)|$. The gradient vector $\frac{\partial d'}{\partial \mathbf{q}'}[\mathbf{q}'_i]$ can be computed using numerical differentiation (recall that $d'(\mathbf{q}')$ can be evaluated for any $\mathbf{q}' \in \mathcal{C}'$ using Eq. (4) and the collision checker).

Figure 1 shows an example workspace and configuration space of a 2-D manipulator with mean $\hat{\mathbf{q}}'$ and the approximately closest point $\mathbf{q}'_\star$ on the boundary of the C-obstacles as found by Newton's method. As can be seen, the $\mathbf{q}'_\star$ is indeed close to the mean, but is not the exact closest point. To construct the linear constraint tangent to the transformed C-obstacles at $\mathbf{q}'_\star$, we need the vector $\mathbf{n}'$ normal to the surface of the transformed C-obstacle at $\mathbf{q}'_\star$. For this, we use $\mathbf{n}' = \text{sign}(d'(\mathbf{q}'_{\star-1}))(\mathbf{q}'_\star - \mathbf{q}'_{\star-1})$, where $\mathbf{q}'_{\star-1}$ is the before-last iterate of Newton's method. Given $\mathbf{n}'$, the linear constraint ($\mathbf{a}^T\mathbf{q} < b$) in the *original* configuration space $\mathcal{C}$ is given by:

$$\mathbf{a}^T = \mathbf{n}'^T L^{-1}, \quad b = \mathbf{n}'^T \mathbf{q}'_\star. \tag{6}$$

This equation, as well as the ones above, suggest that we need to decompose Σ into LL^T and compute the inverse of L, which are potentially costly operations and require Σ to be non-singular. However, this is not necessary. It can be shown that the configurations $\mathbf{q}_i = L\mathbf{q}'_i$ in the *untransformed* configuration space evolve in Newton's method as:

$$\mathbf{q}_{i+1} = \mathbf{q}_i - d(\mathbf{q}_i)\Sigma\frac{\partial d}{\partial \mathbf{q}}[\mathbf{q}_i] / \left(\frac{\partial d}{\partial \mathbf{q}}[\mathbf{q}_i]^T \Sigma \frac{\partial d}{\partial \mathbf{q}}[\mathbf{q}_i]\right), \tag{7}$$

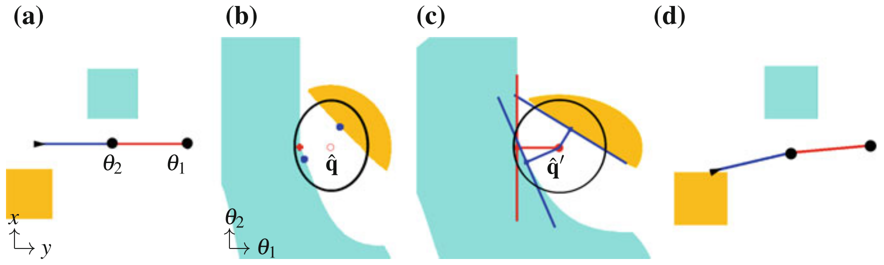


Fig. 1 Example of our method for estimating the closest collision point. The 2-D manipulator at configuration $\hat{\mathbf{q}}$ with 2 obstacles in the workspace (**a**). The configuration space of the manipulator defined by (θ_1, θ_2) zoomed in to the area around $\hat{\mathbf{q}}$ (**b**). The uncertainty ellipse is shown with greater uncertainty in θ_2 than in θ_1. We transform the C-space such that the ellipse is circular, enabling us to compute distances using a collision checker (**c**). Using Newton's method, we compute the closest point ($\mathbf{q}'_\star$) and corresponding constraint tangent for each link that collides with an obstacle. We illustrate in the workspace the $\mathbf{q}'_\star$ that collides with the orange obstacle (**d**). Using the tangents in **c**, we truncate the uncertainty distribution and propagate to next time steps, enabling us to estimate probability of collision along a trajectory

with $\mathbf{q}_0 = \hat{\mathbf{q}}$. The constraint ($\mathbf{a}^T\mathbf{q} < b$) upon convergence is then given by:

$$\mathbf{a} = -|d(\mathbf{q}_{\star-1})|\frac{\partial d}{\partial \mathbf{q}}[\mathbf{q}_{\star-1}], \quad b = \mathbf{a}^T\mathbf{q}_\star, \tag{8}$$

where the distance and gradient are available from the last step of Newton's method. From Eq. (7) we can see that in each iteration our method needs $O(n)$ distance queries and $O(n^2)$ operations, where n is the dimension of the configuration space.

4.2 Multiple Constraints

Above we made the assumption that the geometry of the configuration space obstacles is (locally) convex. This is in general not an accurate assumption, in particular if the geometry of the workspace and of the robot is highly non-convex (as is the case with a manipulator). Therefore, we take the following approach to create multiple constraints: we assume that the geometry of the workspace obstacles $\mathcal{O}$ is decomposed into n convex sets $\mathcal{O}_1, \dots, \mathcal{O}_n$ such that $\bigcup_i \mathcal{O}_i = \mathcal{O}$ and, similarly, that the workspace geometry $X(\mathbf{q})$ of the robot for a configuration $\mathbf{q}$ is decomposed into m convex sets $X_1(\mathbf{q}), \dots, X_m(\mathbf{q})$ such that $\bigcup_i X_i(\mathbf{q}) = X(\mathbf{q})$. For a manipulator robot for instance, one can imagine that each link of the manipulator forms a convex set.

We then apply the above approach for each pair of workspace obstacle $\mathcal{O}_i$ and robot subset $X_j(\mathbf{q})$, giving a set of nm approximately locally-closest configurations on the geometry of the C-space obstacles and their associated constraints. To avoid having unuseful constraints, we take the following pruning approach: we consider the locally-closest configurations in order of increasing distance from the mean $\hat{\mathbf{q}}$, and remove the configuration from the list if it does not obey one of the constraints associated with configurations that came before it in the list. The intersection of the remaining constraints form an approximate local convexification of the free configuration space around the distribution of the configuration of the robot. We note that this approach relies on an implicit assumption that for a convex (piece of the) robot and a convex obstacle, the corresponding C-space obstacle is (locally) convex. This is not the case in general, but it is reasonable to assume that the surfaces of such C-obstacles are "well-behaved" and that the constraint as found by Newton's method gives a reasonable local approximation. This is the case for the example in Fig. 1.

Given the set of constraints $\{\mathbf{a}_i^T\mathbf{q} < b_i\}$ thus computed, we can approximate the probability that any of the constraints is violated given the Gaussian distribution of $\mathbf{q}$ (see [18, 30]). In addition, we can truncate the Gaussian distribution to approximate the conditional distribution that the robot is collision free. This conditional distribution can then be propagated along a given motion plan for the robot (e.g., using LQG-MP [26]). Since arriving at a particular time step along a plan is conditioned on the previous time steps being collision free, we account for dependencies between successive time steps by repeating the above procedure for each time step along the plan to approximate the probability that the entire path of the robot is collision free (see, e.g., [18]).

5 Computing Motion Plans that Minimize Collision Probability

We next present a motion planner that guarantees that, as computation time is allowed to increase, the computed plan will approach a plan that minimizes probability of collision estimated using the method in Sect. 4. Guaranteeing asymptotic optimality for motion planning under uncertainty is challenging because it is necessary to not only explore the configuration space but also to consider the a priori uncertainty distribution at each configuration. The uncertainty distribution at a configuration is history dependent, i.e., it depends on the trajectory used to reach the configuration. This breaks the optimal substructure assumption required by prior asymptotically optimal motion planners such as RRT* [12] since the cost of any subpath is dependent on what succeeds and precedes it.

5.1 Multiple Independent RRTs (MIRRT)

Our motion planning method, Multiple Independent RRTs (MIRRT), builds upon the rapidly-exploring random tree (RRT) [6], a well-established sampling-based motion planner for finding feasible plans in configuration space. Our method begins by building an RRT and terminating as soon as a plan is found. For the computed plan, we compute the corresponding LQG controller and estimate the probability of collision of the plan as described in Sect. 4. We then launch a completely new RRT to compute another plan. We continue executing independent RRTs until a maximum time threshold is reached or the user stops the planner. (We note that this planning approach is trivially parallelizable.) As the plans are generated, we save the plan with minimum estimated probability of collision.

A single RRT for general cost functions will not converge to an optimal plan [12]. We show in Sect. 5.2 that, with some reasonable assumptions, MIRRT is asymptotically optimal for minimizing the probability of collision estimated using Sect. 4.

5.2 Analysis of Asymptotic Optimality

We analyze MIRRT for holonomic robotic manipulators for goal-oriented problems, i.e., we require the optimal plan π^* to have the property that state $\mathbf{q}_t^*$ along the plan π^* is closer (based on the Euclidean distance in configuration space) to the state $\mathbf{q}_{t-1}^*$ than to any state $\mathbf{q}_{t'}^*$ where $t' < t - 1$. For any goal-oriented problem, the manipulator will never return to a previous configuration to gain information. We show that for a goal-oriented problem in which a plan is represented by control inputs over a finite number of time steps, the plan returned by MIRRT asymptotically approaches the optimal plan π^* with probability 1.

For purposes of motion planning, we define cost function $c(\pi, \mathbf{q}_0)$ as the estimated probability of collision when plan π is executed starting at configuration $\mathbf{q}_0$. Given $\mathbf{q}_0$ and two feasible plans π_1 and π_2, we define the distance between π_1 and π_2 as $\|\pi_1 - \pi_2\| = \max_{\zeta \in [0,1]} \|\mathbf{q}(\zeta T_1 \Delta, \pi_1) - \mathbf{q}(\zeta T_2 \Delta, \pi_2)\|$, where T_i is the number of time steps of π_i and $\mathbf{q}(\tau, \pi)$ represents the configuration at time τ while executing the plan π. For our cost function, similar plans have similar cost. The cost function $c(\pi, \mathbf{q}_0)$ is Lipschitz continuous, i.e., there exists some constant K such that starting from $\mathbf{q}_0$, for any two feasible plans π_1 and π_2, $\|c(\pi_1, \mathbf{q}_0) - c(\pi_2, \mathbf{q}_0)\| \le K\|\pi_1 - \pi_2\|$.

To expand an RRT in the direction of a sampled configuration $\mathbf{q}_{\text{sample}}$, the algorithm uses the function *Steer* : $(\mathbf{q}, \mathbf{q}_{\text{sample}}) \mapsto \mathbf{q}_{\text{new}}$, which is defined as in [12]. *Steer* returns a configuration $\mathbf{q}_{\text{new}}$ that moves linearly from $\mathbf{q}$ toward $\mathbf{q}_{\text{sample}}$ up to a predefined distance $\eta \in \mathbb{R}^+$. For the optimal plan π^*, we define $d^* = \max_{0 \le i \le T-1} \|\mathbf{q}_i^* - \mathbf{q}_{i+1}^*\|$. We assume η is sufficiently large that $\exists \beta \in \mathbb{R}^+, d^* + \beta = \eta$.

Because we consider uncertainty, we leverage the assumption that an optimal plan π^* is α-collision free, i.e., the nominal plan avoids obstacles by a clearance distance of at least α for some $\alpha \in \mathbb{R}^+$. This assumption is reasonable since moving adjacent to an obstacle would almost surely cause collision.

Motivated by [13], we build "balls" along the optimal plan π^*. Given any $\varepsilon \in (0, \min(\frac{\beta}{2}, \alpha))$, for any $\mathbf{q}_t^*$, we define (ε, t)-ball $B_{\mathbf{q}_t^*}$ as all $\mathbf{q}$ such that $\|\mathbf{q}_t^* - \mathbf{q}\| \le \varepsilon$. Because of the choice of ε, at time step t, if the robot is at a state $\mathbf{q}$ within $B_{\mathbf{q}_t^*}$, and a sample $\mathbf{q}_{\text{sample}}$ ends up within $B_{\mathbf{q}_{t+1}^*}$, the function *Steer*$(\mathbf{q}, \mathbf{q}_{\text{sample}})$ can connect $\mathbf{q}$ and $\mathbf{q}_{\text{sample}}$ by a straight line in $\mathcal{C}$. The straight line is also collision free. For any feasible plan π that has the same number of time steps as π^*, we call π an ε-close path if and only if for any time step t along π, $\mathbf{q}_t \in B_{\mathbf{q}_t^*}$.

Theorem 1 (MIRRT is asymptotically optimal) *Let π_i denote the best plan found after i RRTs have returned solutions. Given the assumptions above and assuming the problem is goal-oriented and admits a feasible solution, as the number of independent RRT plans generated in MIRRT increases, the best plan almost surely approaches the optimal plan π^*, i.e., $P(\lim_{i\to\infty} \|c(\pi_i, \mathbf{q}_0) - c(\pi^*, \mathbf{q}_0)\| = 0) = 1$.*

Proof For any ε (without loss of generality, we assume $\varepsilon \in (0, \min(\alpha, \frac{\beta}{2}))$), we build (ε, t)-balls along π^*. Consider a sequence of events that can generate an ε-close path. In one RRT, we start from the initial state $\mathbf{q}_0$ (we assume $\mathbf{q}_0 = \mathbf{q}_0^*$). The first sample $\mathbf{q}_1$ ends in $B_{\mathbf{q}_1^*}$ with nonzero probability. The steering function connects $\mathbf{q}_0^*$ and $\mathbf{q}_1$. The second sample $\mathbf{q}_t$ ends in $B_{\mathbf{q}_2^*}$ with nonzero probability. Based on the goal-oriented assumption, $\mathbf{q}_t$ is closer to $\mathbf{q}_1$ and hence the steering function connects $\mathbf{q}_1$ and $\mathbf{q}_2$. We repeat until the last sample $\mathbf{q}_T$ ends in $B_{\mathbf{q}_T^*}$ with nonzero probability and the steering function connects $\mathbf{q}_{T-1}$ and $\mathbf{q}_T$. Thus, the probability of generating an ε-close path by one execution of RRT is nonzero, which we express as $P_\varepsilon \in \mathbb{R}^+$. Hence we have $P(\|\pi_i - \pi^*\| > \varepsilon) = (1 - P_\varepsilon)^i = \bar{P}_\varepsilon^i$. Thus, $\sum_i P(\|\pi_i - \pi^*\| > \varepsilon) = \sum_i \bar{P}_\varepsilon^i \le \frac{1}{1-\bar{P}_\varepsilon}$ is finite. Based on a Borel-Cantelli argument [10], we have $P(\lim_{i\to\infty} \|\pi_i - \pi^*\| = 0) = 1$. Since the cost function of estimating probability of collisions is Lipschitz continuous, $P(\lim_{i\to\infty} \|c(\pi_i, \mathbf{q}_0) - c(\pi^*, \mathbf{q}_0)\| = 0) = 1$. □

We have shown that when the above assumptions hold and the number of feasible plans approaches infinity, the plan returned by MIRRT will almost surely approach a solution that minimizes the estimated probability of collision c.

6 Experiments

We evaluated our methods on two simulated scenarios: (1) a 6-DOF manipulator in a 3-D environment with narrow passages, and (2) a concentric tube robot that has applications to surgical procedures. In both scenarios, the robots have substantial actuation uncertainty and limited sensing feedback. We evaluated our C++ implementation on a 3.33 GHz Intel i7 PC.

For both scenarios, we first evaluated our method for estimating probability of collision by comparing the estimated collision probability with the ground truth probability. We computed ground truth by running at least 5,000 Monte Carlo simulations of the given motion plan and reporting the percentage of collision free simulations. Each simulation was executed in a closed-loop fashion using the given linear feedback controller and a Kalman filter, and with artificially generated motion and sensing noise. We also demonstrated the ability of MIRRT to compute plans that approach optimality as computation time is allowed to increase.

6.1 6-DOF Manipulator Scenario

We first applied our method to a holonomic 6-DOF articulated robotic manipulator in a 3-D environment as shown in Fig. 2a. The robotic manipulator must move from its initial configuration to a configuration in which its end-effector is inside the predefined goal region (red ball) while avoiding the cyan obstacles. To reach the goal, the manipulator must pass through one of two narrow passages; the left narrow passage is wider than the upper narrow passage. Although our geometric representation resembles an industrial manipulator, we model the robot as a low-cost, compliant manipulator with uncertainty in actuation and with an encoder only at the base joint, resulting in limited state feedback.

We define the configuration of the robot as $\mathbf{q} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$, the manipulator's joint angles. The control inputs are the angular velocities of the joints, $\mathbf{w} = (w_1, w_2, w_3, w_4, w_5, w_6)$, each corrupted by a process noise $\mathbf{m} = (\tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \tilde{w}_4, \tilde{w}_5, \tilde{w}_6) \sim \mathcal{N}(\mathbf{0}, M)$ where $M = \sigma_1^2 I$ with $\sigma_1 = 0.03$ rad/s. We define the robot's discrete dynamics model as

$$\mathbf{q}_{t+1} = \mathbf{q}_t + \Delta(\mathbf{w} + \mathbf{m}) \tag{9}$$

where Δ is the time step size. We assume the robot has an encoder only at its base joint and hence define the sensing model as

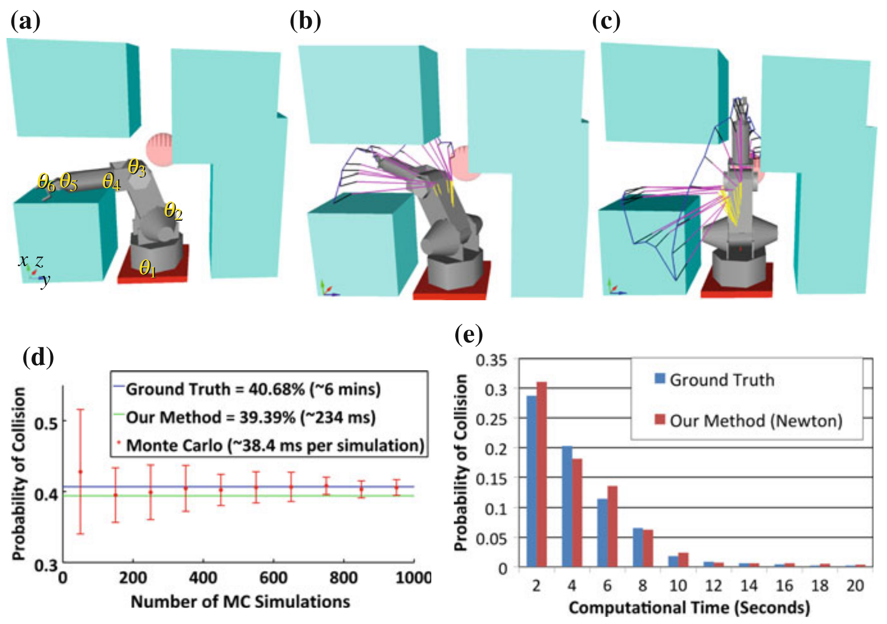


Fig. 2 The 6-DOF manipulator with an encoder only at its base joint must move from its initial configuration shown in **a** to a goal configuration in which its end-effector reaches the *red* ball while avoiding the *cyan* obstacles. An example feasible plan (**b**), where the trajectory of the robot over time is shown by its *yellow*, *magenta*, and *blue* links. Our method estimates the probability of collision of this plan at 39.39 % and ground truth is 40.68 %. We compare our probability of collision estimation method and Monte Carlo simulation for the example plan (**d**). For Monte Carlo simulation to achieve similar accuracy to our method, 700 simulations are needed, which requires over 26 s rather than 234 ms for our method. The optimal plan computed by MIRRT run for 20 s passes through the upper passage (**c**). The estimated probability of collision is 4.19 % while the ground truth is 3.12 %. The performance of the MIRRT converges as the computation time is allowed to increase (**e**)

$$h(\mathbf{q}, \mathbf{n}) = \theta_1 + \mathbf{n} \tag{10}$$

where the observation is corrupted by noise $\mathbf{n} \sim \mathcal{N}(0, \sigma_2^2)$ with $\sigma_2 = 0.03$ rad/s. This is indeed a formally unobservable sensing model.

We first evaluate the ability of our method to accurately estimate the probability of collision for a particular plan. For the example feasible plan shown in Fig. 2b that was computed by an RRT, we compare our method to the alternative of using Monte Carlo simulations for estimating probability of collision. Our method required 234 ms of computation time. Figure 2d shows the deviation in the probability estimates computed using Monte Carlo simulations with varying number of samples (averaged over 100 trials). As expected, the variance decreases as the number of Monte Carlo simulations increases. It takes over 700 Monte Carlo simulations, each simulation requiring an average of 38 ms, to arrive within the accuracy bounds of our method, which corresponds to over 26 s of computation time just to estimate the collision probability. Hence, our method is over 100 times faster than Monte Carlo simulation.

To assess the accuracy of our method across a broader range of plans, we also randomly generated 100 feasible plans using independent RRTs. The average absolute error between the ground truth probability of collision (computed using 10,000 Monte Carlo simulations) and the estimation by our method is 9.78 %. The average computational time of our method for each plan is 158 ms.

We also evaluate our MIRRT approach for computing motion plans with low probability of collision. We executed MIRRT for varying computation times up to 20 s, which allows for computation of $\sim$100 plans. In Fig. 2e, we show the estimated probability of collision of the best plan computed by our method as well as the ground truth probability of collision for that plan. Each bar is the average of 20 executions. As computation time is allowed to increase, the MIRRT approach returns a plan that is almost guaranteed to avoid collisions, and its performance is verified by the ground truth value. In the optimal plan (shown in Fig. 2c) the robot pulls back and then passes through the upper narrow passage rather than through the left passage, even though the left passage would allow a path that is both shorter and has greater clearance from obstacles. The reason for selecting a path through the upper narrow passage is because the robot has low uncertainty for θ_1 due to its ability to sense that joint's orientation, and that decreases the chances of collision when moving through the upper passageway because the robot's kinematics for the remaining joints will restrict uncertainty primarily along the z axis.

6.2 Concentric Tube Robot Scenario

We also apply our method to a concentric tube robot, a tentacle-like robot composed of nested, pre-curved elastic tubes. These devices have the potential to enable physicians to perform new minimally-invasive surgical procedures that require maneuvering through narrow passages or around anatomical obstacles. Potential clinical applications include surgeries of the pituitary gland or nearby structures in the skull base [5] as well surgeries that require maneuvering inside the heart [29].

Each tube of a concentric tube robot is pre-curved and can be inserted and axially rotated independently of the other tubes. A device having n tubes thus has $2n$ degrees of freedom. Due to the elastic interaction of the tubes, the kinematics of concentric tube robots is complex and must be computed numerically [22, 31]. Computing the kinematic model of the concentric tube robot requires over 50 times more computation time than for the 6-DOF manipulator.

We consider a 3-tube robot for which the state $\mathbf{q} = (\theta_1, \theta_2, \theta_3, \beta_1, \beta_2, \beta_3)$ consists of an axial angle θ_i and insertion distance β_i for each tube. We define the control input as $\mathbf{u} = (w_1, w_2, w_3, v_1, v_2, v_3)$, where w_i and v_i represent the axial rotation angular velocity and insertion speed, respectively, for the i'th tube. We assume the control input is corrupted by a process noise $\mathbf{m} = (\tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \tilde{v}_1, \tilde{v}_2, \tilde{v}_3,) \sim \mathcal{N}(\mathbf{0}, M)$. This results in the dynamics model

$$\mathbf{q}_{t+1} = \mathbf{q}_t + \Delta(\mathbf{u} + \mathbf{m}), \tag{11}$$

where Δ is the time step size. We set $M = \begin{bmatrix} \sigma_1^2 I & 0 \\ 0 & \sigma_2^2 I \end{bmatrix}$ with $\sigma_1 = 0.02$ rad/s and $\sigma_2 = 0.001$ m/s. We assume the concentric tube robot's tip position can be tracked in 3-D using an electromagnetic tracker (e.g., the NDI Aurora Electromagnetic Tracking System as in [5]). This gives the stochastic measurement model

$$h[\mathbf{q}_t, \mathbf{n}] = \begin{bmatrix} x_t & y_t & z_t \end{bmatrix}^T + \mathbf{n}, \tag{12}$$

where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, N)$. We set $N = \sigma_3^2 I$ with $\sigma_3 = 0.001$ m.

For the concentric tube robot, we evaluate the ability of our method to accurately estimate the probability of collision. As shown in Fig. 3, we consider a tubular envi-

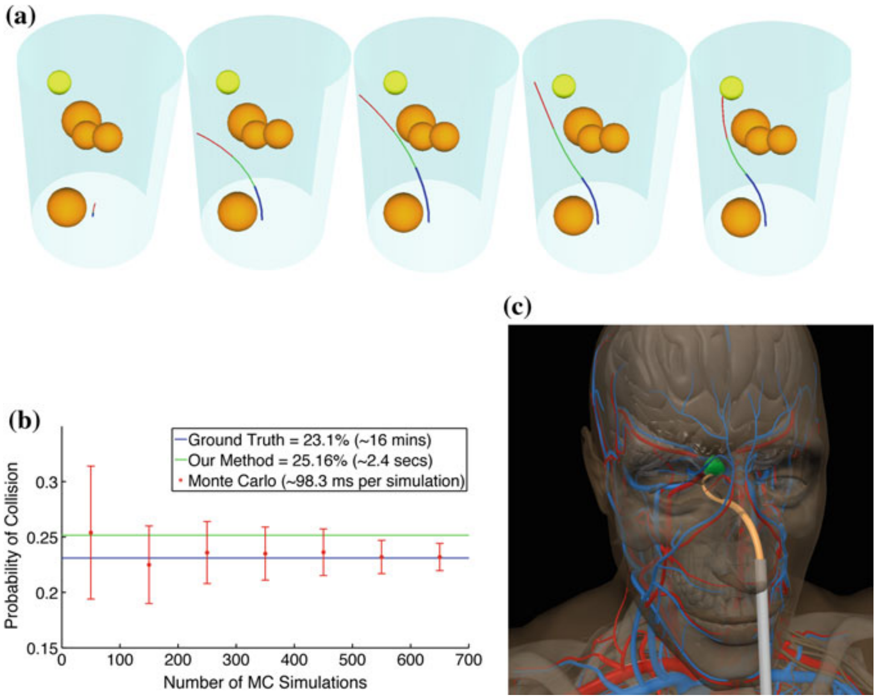


Fig. 3 The 3-tube concentric tube robot must reach the *yellow* goal sphere while avoiding the *orange* obstacles and remaining inside the *cyan* cylinder (**a**). We compare our method with Monte Carlo simulation for the example plan (**b**). The average computation time per Monte Carlo simulation is 98 ms (which is higher than for the 6-DOF manipulator due to the complex kinematics). For Monte Carlo simulation to achieve a similar accuracy to our method, over 450 simulations are needed, which requires over 44 s rather than 2.4 s total for our method. We also illustrate a clinically-motivated scenario in which a concentric tube robot enters the body via the nasal cavity and reaches a rightward facing pose at the pituitary gland (*green*) for skull base surgery (**c**)

ronment with spherical obstacles similar to the environments used in prior work [24]. For the example plan shown in Fig. 3a, our method required 2.4 s of computation time. Figure 3b compares the probability of collision estimates computed using Monte Carlo simulation with varying number of samples (averaged over 100 trials). As expected, the variance decreases as the number of Monte Carlo simulations increases. It takes over 450 Monte Carlo simulations to achieve the accuracy of our method, which corresponds to over 44 s of computation time just to estimate the collision probability. Hence, our method is over 10 times faster than Monte Carlo simulation for the concentric tube robot scenario.

To assess the accuracy of our method across a broader range of concentric tube robot plans, we also randomly generated 100 feasible plans using independent RRTs. The average absolute error between the ground truth probability of collision (computed using 10,000 Monte Carlo simulations) and the estimation by our method is 4.36 %. The optimal plan picked by MIRRT using our method has an estimated probability of collision of 9.11×10^{-7} % while ground truth is 0 %.

To treat cancers of the pituitary gland, surgeons often must resect the tumor, which requires inserting surgical instruments to reach the skull base. Enabling surgeons to access the pituitary gland (shown in green in Fig. 3c) via the nasal cavity and by drilling through thin sinus bones would be far less invasive than current surgical approaches, but requires controllable, curvilinear instruments for surgical access [5]. For the concentric tube robot, we generated 100 plans for the anatomy in Fig. 3c, avoiding obstacles such as bone and blood vessels and requiring a rightward facing tip pose for the surgical task. MIRRT executed for 100 plans returns a plan with an estimated probability of collision of 0.94 %, while the ground truth is 0 %. In Fig. 3c, we illustrate the optimal plan found by our method.

7 Conclusion

We introduced a new integrated motion planning and control algorithm for robotic manipulators that makes safety a priority by explicitly minimizing the probability of unwanted collisions. Our approach quickly computes distances to obstacles in the workspace and appropriately transforms this information into the configuration space using a Newton method to estimate the most relevant collision points in configuration space. We then presented a sampling-based motion planner that executes multiple independent RRTs and returns a plan that, under reasonable assumptions, asymptotically converges to a plan that minimizes the estimated collision probability. We applied our approach in simulation to a 6-DOF manipulator and a tentacle-like surgical robot. Our results show orders of magnitude speedup over Monte Carlo simulation for equivalent accuracy in estimating collision probability, and the ability of our motion planner to return high quality plans with low probability of collision.

Our method assumes that the manipulator operates under the assumptions of Gaussian motion and sensing uncertainty. Although the class of problems where Gaussian distributions are appropriate is large (as shown by the widespread use of the extended Kalman filter for state estimation), the approximation is not acceptable for some applications. In future work we plan to extend our method to non-Gaussian uncertainty and integrate with local optimization methods in belief space to quickly refine plans. We also will apply the method to meso-scale, flexible surgical robots to improve device safety, effectiveness, and clinical potential.

Acknowledgments This research was supported in part by the National Science Foundation (NSF) through awards IIS-1117127 and IIS-1149965 and by the National Institutes of Health (NIH) under awards 1R01EB017467 and 1R21EB017952.

References

1. Agha-mohammadi, A.-A., Chakravorty, S., Amato, N.M.: Sampling-based nonholonomic motion planning in belief space via dynamic feedback linearization-based FIRM. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4433–4440, October 2012
2. Alterovitz, R., Siméon, T., Goldberg, K.:The stochastic motion roadmap: a sampling framework for planning with Markov motion uncertainty. In: Proceedings of the Robotics: Science and Systems, pp. 1–8, June 2007
3. Brock, O., Khatib, O.: Elastic strips: a framework for motion generation in human environments. Int. J. Robot. Res. **21**(2), 1031–1052 (2002)
4. Bry, A., Roy, N.: Rapidly-exploring random belief trees for motion planning under uncertainty. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 723–730, May 2011
5. Burgner, J., Swaney, P.J., Rucker, D.C., Gilbert, H.B., Nill, S.T., Russell III, P.T., Weaver, K.D., Webster III, R.J.: A bimanual teleoperated system for endonasal skull base surgery. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2517–2523, September 2011
6. Choset, H., Lynch, K.M., Hutchinson, S.A., Kantor, G.A., Burgard, W., Kavraki, L.E., Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press, Cambridge (2005)
7. Degani, A., Choset, H., Wolf, A., Zenati, M.A.: Highly articulated robotic probe for minimally invasive surgery. In: Proceedings of the IEEE Internationl Conference on Robotics and Automation (ICRA), pp. 4167–4172, May 2006
8. Du Toit, N.E., Burdick, J.W.: Robot motion planning in dynamic, uncertain environments. IEEE Trans. Robot. **28**(1), 101–115 (2012)
9. Dupont, P.E., Lock, J., Itkowitz, B., Butler, E.: Design and control of concentric-tube robots. IEEE Trans. Robot. **26**(2), 209–225 (2010)
10. Grimmett, G., Stirzaker, D.: Probability and Random Processes, 3rd edn. Oxford University Press, New York (2001)
11. Guibas, L.J., Hsu, D., Kurniawati, H., Rehman, E.: Bounded uncertainty roadmaps for path planning. In: Proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR) (2008)
12. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. Int. J. Robot. Res. **30**(7), 846–894 (2011)
13. Kavraki, L.E., Kolountzakis, M.N., Latombe, J.-C.: Analysis of probabilistic roadmaps for path planning. IEEE Trans. Robot. Autom. **14**(1), 166–171 (1998)
14. Kurniawati, H., Hsu, D., Lee, W.: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: Proceedings of the Robotics: Science and Systems (2008)

15. Lee, A., Duan, Y., Patil, S., Schulman, J., McCarthy, Z., van den Berg, J., Goldberg, K., Abbeel, P.: Sigma hulls for gaussian belief space planning for imprecise articulated robots amid obstacles. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)
16. Nash, S.G., Sofer, A.: Linear and Nonlinear Programming. McGraw-Hill, New York (1996)
17. Patil, S., van den Berg, J., Alterovitz, R.: Motion planning under uncertainty in highly deformable environments. In: Proceedings of the Robotics: Science and Systems, June 2011
18. Patil, S., van den Berg, J., Alterovitz, R.: Estimating probability of collision for safe motion planning under Gaussian motion and sensing uncertainty. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 3238–3244, May 2012
19. Platt, R., Kaelbling, L.: Efficient planning in non-Gaussian belief spaces and its application to robot grasping. In: International Symposium on Robotics Research (ISRR) (2011)
20. Platt, R., Tedrake, R., Kaelbling, L., Lozano-Perez, T.: Belief space planning assuming maximum likelihood observations. In: Proceedings of the Robotics: Science and Systems (2010)
21. Prentice, S., Roy, N.: The belief roadmap: efficient planning in belief space by factoring the covariance. Int. J. Robot. Res. **31**, 1263–1278 (2009)
22. Sears, P., Dupont, P.E.: A steerable needle technology using curved concentric tubes, In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2850–2856, October 2006
23. Simaan, N.: Snake-like units using flexible backbones and actuation redundancy for enhanced miniaturization. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 3023–3028, April 200
24. Torres, L.G., Alterovitz, R.:Motion planning for concentric tube robots using mechanics-based models. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5153–5159, September 2011
25. van den Bergen, G.: SOLID. http://www.win.tue.nl/~gino/solid/ (2004)
26. van den Berg, J., Abbeel, P., Goldberg, K.: LQG-MP: optimized path planning for robots with motion uncertainty and imperfect state information. Int. J. Robot. Res. **30**(7), 895–913 (2011)
27. van den Berg, J., Patil, S., Alterovitz, R.: Motion planning under uncertainty using iterative local optimization in belief space. Int. J. Robot. Res. **31**(11), 1263–1278 (2012)
28. van den Berg, J., Patil, S., Alterovitz, R.: Efficient approximate value iteration for continuous Gaussian POMDPs. In: Proceedings of the Twenty-Sixth AAAI Conference (AAAI-12), pp. 1832–1838, July 2012
29. Vasilyev, N.V., Dupont, P.E.: Robotics and imaging in congenital heart surgery. Future Cardiol. **8**(2), 285–296 (2012)
30. Vitus, M.P., Tomlin, C.J.: Closed-loop belief space planning for linear, Gaussian systems. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 2152–2159, May 2011
31. Webster III, R.J., Okamura, A.M., Cowan, N.J.: Toward active cannulas: miniature snake-like surgical robots. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2857–2863, October 2006
32. Webster III, R.J., Romano, J.M., Cowan, N.J.: Mechanics of precurved-tube continuum robots. IEEE Trans. Robot. **25**(1), 67–78 (2009)