

Springer Proceedings in Advanced Robotics 5
Series Editors: Bruno Siciliano · Oussama Khatib

Marco Hutter
Roland Siegwart *Editors*

Field and Service Robotics

Results of the 11th International
Conference



Springer Proceedings in Advanced Robotics

Volume 5

Series editors

Prof. Bruno Siciliano
Dipartimento di Ingegneria Elettrica
e Tecnologie dell'Informazione
Università degli Studi di Napoli
Federico II
Via Claudio 21 80125 Napoli
Italy
e-mail: siciliano@unina.it

Prof. Oussama Khatib
Robotics Laboratory
Department of Computer Science
Stanford University
Stanford, CA 94305-9010
USA
e-mail: khatib@cs.stanford.edu

Editorial Advisory Board

Gianluca Antonelli, University of Cassino, Italy
Dieter Fox, University of Washington, USA
Kensuke Harada, Osaka University, Japan
M. Ani Hsieh, University of Pennsylvania, USA
Torsten Kröger, Karlsruhe Institute of Technology, Germany
Dana Kulić, University of Waterloo, Canada
Jaehung Park, Seoul National University, South Korea

More information about this series at <http://www.springer.com/series/15556>

Marco Hutter · Roland Siegwart
Editors

Field and Service Robotics

Results of the 11th International Conference



Springer

Editors

Marco Hutter
ETH Zurich
Zürich
Switzerland

Roland Siegwart
ETH Zurich
Zürich
Switzerland

ISSN 2511-1256

ISSN 2511-1264 (electronic)

Springer Proceedings in Advanced Robotics

ISBN 978-3-319-67360-8

ISBN 978-3-319-67361-5 (eBook)

<https://doi.org/10.1007/978-3-319-67361-5>

Library of Congress Control Number: 2017954874

© Springer International Publishing AG 2018

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by Springer Nature

The registered company is Springer International Publishing AG

The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Foreword

Robots! Robots on Mars and in oceans, in hospitals and homes, in factories and schools; robots fighting fires, making goods and products, saving time and lives. Robots today are making a considerable impact from industrial manufacturing to healthcare, transportation, and exploration of the deep space and sea. Tomorrow, robots will become pervasive and touch upon many aspects of modern life.

The *Springer Tracts in Advanced Robotics (STAR)* was launched in 2002 with the goal of bringing to the research community the latest advances in the robotics field based on their significance and quality. During the latest 15 years, the STAR series has featured publication of both monographs and edited collections. Among the latter, the proceedings of thematic symposia devoted to excellence in robotics research, such as ISRR, ISER, FSR and WAFR, have been regularly included in STAR.

The expansion of our field as well as the emergence of new research areas has motivated us to enlarge the pool of proceedings in the STAR series in the past few years. This has ultimately led to launching a sister series in parallel to STAR. The *Springer Proceedings in Advanced Robotics (SPAR)* is dedicated to the timely dissemination of the latest research results presented in selected symposia and workshops.

The Eleventh edition of *Field and Service Robotics* edited by Marco Hutter and Roland Siegwart offers in its seven-part volume a collection of a broad range of topics clustered in control, computer vision, inspection, machine learning, mapping, navigation and planning, systems and tools. The contents of the forty-five contributions represent a cross section of the current state of robotics research from one particular aspect: field and service applications, and how they reflect on the theoretical basis of subsequent developments. Pursuing technologies aimed at field robots that operate in complex and dynamic environments, as well as at service robots that work closely with humans to help them with their lives, is the big challenge running throughout this focused collection.

Rich by topics and authoritative contributors, FSR culminates with this unique reference on the current developments and new directions in field and service robotics. A fine addition to the series!

Naples, Italy
Stanford, USA
July 2017

Bruno Siciliano
Oussama Khatib
SPAR Editors

Preface

Field and Service Robotics (FSR) is the leading single-track conference on applications of robotics in challenging environments. Its goal is to report and encourage the development and experimental evaluation of field and service robots, and to generate a vibrant exchange and discussion in the community. Field robots are non-factory robots, typically mobile, that operate in complex and dynamic environments: on the ground (Earth or other planets), under the ground, underwater, in the air or in space. Service robots are those that work closely with humans to help them with their lives. The first FSR was held in Canberra, Australia, in 1997. Since that first meeting, FSR has been held roughly every 2 years, cycling through Asia, the Americas and Europe. This book presents the results of the 11th edition of *Field and Service Robotics*, FSR 2017, held in Zurich, Switzerland, from 12 to 16 June 2017. This was the first time it has been held in Canada. This year we had 74 submitted papers from which we accepted 21 for oral presentations and 24 for interactive presentations. FSR 2017 was organized by Marco Hutter and Roland Siegwart, ETH Zurich, together with the regional chairs Tim Barfoot, Cedric Pradalier and Kazuya Yoshida. The organizers would like to thank the International Program Committee that generously provided their time to carry out detailed reviews of all the papers, namely Peter Corke, Carrick Dettweiler, Philippe Giguere, Genya Ishigami, Michael Jakub, Alonzo Kelly, Jonathan Kelly, Ross Knepper, Simon Lacroix, Christian Laugier, Josh Marshall, David P. Miller, Keiji Nagatani, Juan Nieto, Steve Nuske, François Pomerleau, Cédric Pradalier, Jonathan Roberts, Miguel Angel Salichs, Sanjiv Singh, Gaurav Sukhatme, Salah Sukkarieh, Takashi Tsubouchi, Arto Visala, Kazuya Yoshida and Uwe Zimmer. Additional thanks go to the keynote speakers Salah Sukkarieh, Terry Fong, Jean-Christophe Zufferey and Aparna Rao.

FSR 2015 would not have been possible without the generous support of our sponsors ETH Zurich, Maxon Motors AG, ABB, Wyss Zurich, NCCR robotics, NCCR digital fabrication and Wyss Zurich.

Zürich, Switzerland

Marco Hutter
Roland Siegwart

About the Book

This book contains the proceedings of the 11th FSR (Field and Service Robotics), which is the leading single-track conference on applications of robotics in challenging environments. This conference was held in Zurich, Switzerland, from 12 to 15 September 2017. The book contains 45 full-length, peer-reviewed papers organized into a variety of topics: control, computer vision, inspection, machine learning, mapping, navigation and planning, systems and tools.

The goal of the book and the conference is to report and encourage the development and experimental evaluation of field and service robots, and to generate a vibrant exchange and discussion in the community. Field robots are non-factory robots, typically mobile, that operate in complex and dynamic environments: on the ground (Earth or other planets), under the ground, underwater, in the air or in space. Service robots are those that work closely with humans to help them with their lives. The first FSR was held in Canberra, Australia, in 1997. Since that first meeting, FSR has been held roughly every two years, cycling through Asia, Americas and Europe.

Contents

Part I Control

Controlling Ocean One	3
Gerald Brantner and Oussama Khatib	
Safe Self-collision Avoidance for Versatile Robots Based on Bounded Potentials	19
David Gonon, Dominic Jud, Péter Fankhauser and Marco Hutter	
Towards Controlling Bucket Fill Factor in Robotic Excavation by Learning Admittance Control Setpoints	35
Heshan A. Fernando, Joshua A. Marshall, Håkan Almqvist and Johan Larsson	
Trajectory Optimization for Dynamic Grasping in Space Using Adhesive Grippers	49
Roshena MacPherson, Benjamin Hockman, Andrew Bylard, Matthew A. Estrada, Mark R. Cutkosky and Marco Pavone	
Generation of Turning Motion for Tracked Vehicles Using Reaction Force of Stairs' Handrail	65
Yuto Ohashi, Shotaro Kojima, Kazunori Ohno, Yoshito Okada, Ryunosuke Hamada, Takahiro Suzuki and Satoshi Tadokoro	

Part II Computer Vision

Finding Better Wide Baseline Stereo Solutions Using Feature Quality	83
Stephen Nuske and Jay Patravali	
High-Throughput Robotic Phenotyping of Energy Sorghum Crops	99
Srinivasan Vijayarangan, Paloma Sodhi, Prathamesh Kini, James Bourne, Simon Du, Hanqi Sun, Barnabas Poczos, Dimitrios Apostolopoulos and David Wettergreen	

Improved Tau-Guidance and Vision-Aided Navigation for Robust Autonomous Landing of UAVs	115
Amedeo Rodi Vetrella, Inkyu Sa, Marija Popović, Raghav Khanna, Juan Nieto, Giancarmine Fasano, Domenico Accardo and Roland Siegwart	
Fast and Power-Efficient Embedded Software Implementation of Digital Image Stabilization for Low-Cost Autonomous Boats	129
S. Aldegheri, D. D. Bloisi, J. J. Blum, N. Bombieri and A. Farinelli	
Evaluation of Combined Time-Offset Estimation and Hand-Eye Calibration on Robotic Datasets	145
Fadri Furrer, Marius Fehr, Tonci Novkovic, Hannes Sommer, Igor Gilitschenski and Roland Siegwart	

Part III Inspection

Field Report: UAV-Based Volcano Observation System for Debris Flow Evacuation Alarm	163
Keiji Nagatani, Ryosuke Yajima, Seiga Kiribayashi, Tomoaki Izu, Hiromichi Kanai, Hiroyuki Kanasaki, Jun Minagawa and Yuji Moriyama	
Cooperative UAVs as a Tool for Aerial Inspection of the Aging Infrastructure	177
Sina Sharif Mansouri, Christoforos Kanellakis, Emil Fresk, Dariusz Kominiak and George Nikolakopoulos	
Autonomous Aerial Inspection Using Visual-Inertial Robust Localization and Mapping	191
Lucas Teixeira, Ignacio Alzugaray and Margarita Chli	
Sensing Water Properties at Precise Depths from the Air	205
John-Paul Ore and Carrick Detweiler	
Autonomous and Safe Inspection of an Industrial Warehouse by a Multi-rotor MAV	221
Alexandre Eudes, Julien Marzat, Martial Sanfourche, Julien Moras and Sylvain Bertrand	

Part IV Machine Learning

Online Multi-modal Learning and Adaptive Informative Trajectory Planning for Autonomous Exploration	239
Akash Arora, P. Michael Furlong, Robert Fitch, Terry Fong, Salah Sukkarieh and Richard Elphic	

Season-Invariant Semantic Segmentation with a Deep Multimodal Network	255
Dong-Ki Kim, Daniel Maturana, Masashi Uenoyama and Sebastian Scherer	
StalkNet: A Deep Learning Pipeline for High-Throughput Measurement of Plant Stalk Count and Stalk Width	271
Harjatin Singh Baweja, Tanvir Parhar, Omeed Mirbod and Stephen Nuske	
Learning Models for Predictive Adaptation in State Lattices	285
Michael E. Napoli, Harel Biggie and Thomas M. Howard	
 Part V Mapping	
Field Deployment of the Tethered Robotic eXplorer to Map Extremely Steep Terrain	303
Patrick McGarey, David Yoon, Tim Tang, François Pomerleau and Timothy D. Barfoot	
Towards Automatic Robotic NDT Dense Mapping for Pipeline Integrity Inspection	319
Jaime Valls Miro, Dave Hunt, Nalika Ulapane and Michael Behrens	
Real-Time Semantic Mapping for Autonomous Off-Road Navigation	335
Daniel Maturana, Po-Wei Chou, Masashi Uenoyama and Sebastian Scherer	
Boundary Wire Mapping on Autonomous Lawn Mowers	351
Nils Einecke, Jörg Deigmöller, Keiji Muro and Mathias Franzius	
A Submap Joining Based RGB-D SLAM Algorithm Using Planes as Features	367
Jun Wang, Jingwei Song, Liang Zhao and Shoudong Huang	
Mapping on the Fly: Real-Time 3D Dense Reconstruction, Digital Surface Map and Incremental Orthomosaic Generation for Unmanned Aerial Vehicles	383
Timo Hinzmann, Johannes L. Schönberger, Marc Pollefeyts and Roland Siegwart	
Aerial and Ground-Based Collaborative Mapping: An Experimental Study	397
Ji Zhang and Sanjiv Singh	

Part VI Navigation and Planning

I Can See for Miles and Miles: An Extended Field Test of Visual Teach and Repeat 2.0	415
Michael Paton, Kirk MacTavish, Laszlo-Peter Berczi, Sebastian Kai van Es and Timothy D. Barfoot	
Dynamically Feasible Motion Planning for Micro Air Vehicles Using an Egocylinder	433
Anthony T. Fragoso, Cevahir Cigla, Roland Brockers and Larry H. Matthies	
Informed Asymptotically Near-Optimal Planning for Field Robots with Dynamics	449
Zakary Littlefield and Kostas E. Bekris	
Strategic Autonomy for Reducing Risk of Sun-Synchronous Lunar Polar Exploration	465
Nathan Otten, David Wettergreen and William Whittaker	
Towards Visual Teach and Repeat for GPS-Denied Flight of a Fixed-Wing UAV	481
M. Warren, M. Paton, K. MacTavish, A. P. Schoellig and T. D. Barfoot	
Local Path Optimizer for an Autonomous Truck in a Harbor Scenario	499
Jennifer David, Rafael Valencia, Roland Philippsen and Karl Iagnemma	

Part VII Systems and Tools

Field Experiments in Robotic Subsurface Science with Long Duration Autonomy	515
Srinivasan Vijayarangan, David Kohanbash, Greydon Foil, Kris Zacny, Nathalie Cabrol and David Wettergreen	
Design and Development of Explosion-Proof Tracked Vehicle for Inspection of Offshore Oil Plant	531
Keiji Nagatani, Daisuke Endo, Atsushi Watanabe and Eiji Koyanagi	
Life Extension: An Autonomous Docking Station for Recharging Quadrupedal Robots	545
Hendrik Kolvenbach and Marco Hutter	
Autonomous Mission with a Mobile Manipulator—A Solution to the MBZIRC	559
Jan Carius, Martin Wermelinger, Balasubramanian Rajasekaran, Kai Holtmann and Marco Hutter	

Towards a Generic Solution for Inspection of Industrial Sites	575
Marco Hutter, Remo Diethelm, Samuel Bachmann, Péter Fankhauser, Christian Gehring, Vassilios Tsounis, Andreas Lauber, Fabian Guenther, Marko Bjelonic, Linus Isler, Hendrik Kolvenbach, Konrad Meyer and Mark Hoepflinger	
Foresight: Remote Sensing for Autonomous Vehicles Using a Small Unmanned Aerial Vehicle	591
Alex Wallar, Brandon Araki, Raphael Chang, Javier Alonso-Mora and Daniela Rus	
Dynamic System Identification, and Control for a Cost-Effective and Open-Source Multi-rotor MAV	605
Inkyu Sa, Mina Kamel, Raghav Khanna, Marija Popović, Juan Nieto and Roland Siegwart	
AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles	621
Shital Shah, Debadeepta Dey, Chris Lovett and Ashish Kapoor	
Design and Development of Tether-Powered Multirotor Micro Unmanned Aerial Vehicle System for Remote-Controlled Construction Machine	637
Seiga Kiribayashi, Kaede Yakushigawa and Keiji Nagatani	
Human-Robot Teaming: Concepts and Components for Design	649
Lanssie Mingyue Ma, Terrence Fong, Mark J. Micire, Yun Kyung Kim and Karen Feigh	
An Analysis of Degraded Communication Channels in Human-Robot Teaming and Implications for Dynamic Autonomy Allocation	665
Michael Young, Mahdieh Nejati, Ahmetcan Erdogan and Brenna Argall	
LEAF: Using Semantic Based Experience to Prevent Task Failures	681
Nathan Ramoly, Hela Sfar, Amel Bouzeghoub and Beatrice Finance	
State Estimation and Localization for ROV-Based Reactor Pressure Vessel Inspection	699
Timothy E. Lee and Nathan Michael	

Part I

Control

Controlling Ocean One

Gerald Brantner and Oussama Khatib

Abstract Using robots to explore venues that are beyond human reach has been a longstanding aspiration of scientists and expeditionists alike. The deep sea exemplifies such an unchartered environment that is currently inaccessible to humans. Ocean One (O_2) is an anthropomorphic underwater robot, designed to operate in deep aquatic conditions and equipped with an array of sensor modalities. Central to the O_2 concept is a human interface that connects the robot and human operator through haptics and vision. In this paper, we focus on O_2 's control architecture and show how it enables an avatar-like synergy between the robot and human pilot. We establish functional autonomy by resolving kinematic and actuation redundancy, allowing the pilot to control O_2 in a lower-dimensional space. We illustrate O_2 's hierarchical whole-body control tasks including manipulation and posture tasks, feed-forward compensation as well as constraint handling. We also describe how to coordinate the dynamics of body and arms to achieve superior performance in contact and demonstrate O_2 's capabilities in simulation, experiments in the pool as well as deployment to its archeological maiden mission to the ‘Lune’, a French naval vessel that sunk to 91 m depth in 1664 in the mediterranean sea.

1 Introduction

Over the past decades, advances in automation have enabled robots to replace humans in performing manual labor [1, 2]. This was possible because the manufacturing floor is tightly structured and tasks are highly repetitive. The next evolution for robots is

G. Brantner (✉) · O. Khatib
Stanford Robotics Laboratory, 353 Serra Mall, Gates Computer Science,
Stanford, CA 94305, USA
e-mail: geraldb@cs.stanford.edu

O. Khatib
e-mail: ok@cs.stanford.edu

to proxy for humans in unstructured, inhospitable environments and advancing the boundaries of human exploration by entering places that are currently inaccessible. The deep sea exemplifies an environment that is inhospitable and largely inaccessible to humans. The field of ROVs has recently brought major advancements to underwater robots that can navigate, observe and map [3–5] and the need for underwater operations has led to the development of submersible manipulators [6, 7]. The O₂ concept offers the capability to perform operations typical for human divers by synthesizing a humanoid robot that is functionally autonomous with a human pilot, who provides higher-lever cognitive abilities, perception and decision making. In this paper, we focus on O₂'s control architecture. We illustrate how the robot acquires functional autonomy in coordinating manipulation tasks with posture and constraints in a hierarchical manner. Subsequently we establish an avatar-like synergy by interfacing the human pilot with the robot through vision and bimanual haptic devices. We demonstrate O₂'s capabilities in simulation, experiments at the pool and eventually its maiden deployment, where it explored a french naval vessel that sank to 91 m depth in the Mediterranean sea, and retrieved archeological artifacts (Fig. 1).

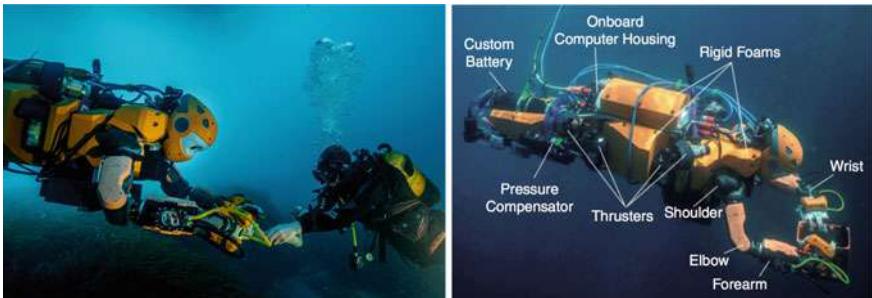


Fig. 1 Left: Ocean One interacting with a human diver. The human pilot—located on the research vessel *Andre Malraux*—interfaces with the robot through bimanual haptic devices and direct stereoscopic vision from the cameras mounted in its head. The robot mimics the pilot's hands while the pilot feels the forces perceived at the robot's hands. Right: Ocean One System Components. O₂ is a humanoid underwater robot with a body, two arms and a head. The series elastic arms are torque controlled and oil-filled. Each hand has three under-actuated tendon-driven fingers. O₂ has a pair of stereo cameras in the head and a wide-angle camera on the chest. The body is actuated by eight thrusters

2 Robot

Ocean One is a humanoid underwater robot of approximately adult-human dimensions and 200 kg overall weight. Its body is conceived in an anthropomorphic shape with shoulders, two arms and a head. Each arm has 7-DoF with electrically driven, torque-controlled joints adapted from the original Meka arm design. The arms are fitted with series elastic actuators that provide torque feedback to enhance compliant motion as well as force control, and safety. In order to withstand the pressure at oceanic depths, the arms are oil-filled and positively pressurized by spring-loaded compensators. Each hand has three fingers with three phalanges per finger that are driven with a tendon, which attaches to the distal finger phalanx, passes through the medial and proximal phalanges and loops around an axis driven by the single actuator [8]. The head contains a pair of high-definition cameras with global shutters. Pan and tilt actuation at the neck is currently being implemented. Another camera is attached on O₂'s chest, offering a wide-angle perspective on the surroundings in front and below. The body is actuated by eight thrusters. More details on hardware components can be reviewed in [9].

3 Pilot Interface

The O₂ concept is not only the underwater robot itself but a distributed system of hardware and software components. The surface station allows the human pilot to connect to a set of interfaces: Haptic devices, GUI, live vision, and world display. These interfaces play different roles in different modes of operation. In Avatar-mode, the haptic devices and live vision are central, while GUI and virtualization are more predominant in autonomous modes (Fig. 2).

Fig. 2 Pilot Interface. The pilot's interaction with O₂ is established through an immersive interface consisting of bimanual haptic devices, stereoscopic live vision, GUI and a world display giving a global perspective of the scene



4 Modeling

O_2 is modeled as a tree-like structure, with two arms branching out from the body. We model its kinematics using generalized coordinates, with 6 virtual DoF for the body and 7 DoF for each arm. Each link is a rigid body with associated mass properties. This leads us to a multi-body dynamic system represented by twenty-dimensional equations of motion

$$A(q)\ddot{q} + b(q, \dot{q}) + g(q) + h = \Gamma, \quad (1)$$

where q is the vector of generalized coordinates, $A(q)$ is the kinetic energy matrix, $b(q, \dot{q})$ is the vector of centrifugal and coriolis forces, $g(q)$ the gravitational forces, h the hydrodynamic contribution and Γ is the vector of generalized forces. We extract the mass properties of the body and arms from the CAD models and experimental inspection.

5 Control

O_2 is a force controlled robot, which allows us to take advantage of the robot's dynamics at a global level, coordinating the slow dynamics of the body with the fast dynamics of the arms in order to achieve superior performance in motion and force control.

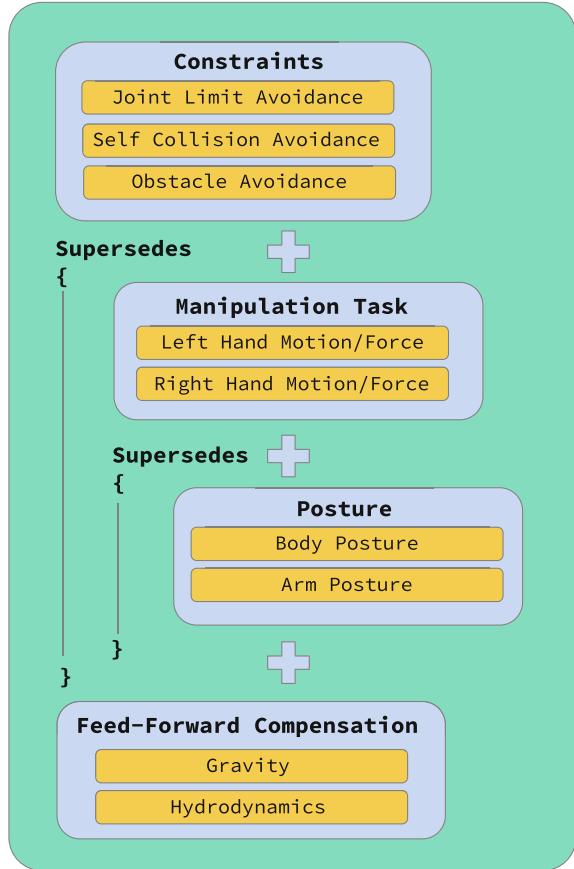
5.1 Whole Body Architecture

The objective behind O_2 's control architecture is to enable a connection to the human interface, where a small set of control inputs is sufficient to pilot the entire robot, while achieving a high degree of autonomy already at the controller level. Because the interaction between robot and environment happens primarily through physical contact, we directly control the two hands while the body autonomously follows the hands. Beyond this, the controller monitors the pilot, the robot and its environment, and overrides actions that would lead to constraint violations, such as collisions with obstacles or joint limits. The remaining null-space is used to optimize arm and body posture for a given task. This behavior is realized by the control law

$$\Gamma = \underbrace{(J_c^\top F_c)}_{\text{Constraints}} + \underbrace{(J_{t|c}^\top F_{t|c})}_{\text{Manipulation}} + \underbrace{(J_{p|t|c}^\top F_{p|t|c})}_{\text{Posture}} + \underbrace{(J_h^\top F_h)}_{\text{Compensation}}, \quad (2)$$

a hierarchical architecture comprised of tasks. The four additive terms in (2) are associated with the contributions of *Constraints*, *Manipulation Task*, *Posture* and *feed-forward compensation*, respectively. The controller coordinates these tasks in

Fig. 3 Whole-Body control architecture. A hierarchical structure allows for prioritization of manipulation task, posture and constraints. The only control inputs are motion and forces of the left and right hand. Joint-limit, self-collision and obstacle-avoidance have highest priority. Posture is defined as the remaining controllable sub-space after constraints and manipulation-task. For gravitational and hydrodynamic compensation, the controller has an additional feed-forward term



a prioritized manner. The notation $t|c$, for instance, reads *Task t consistent with Task c*. In case tasks are not simultaneously feasible, a lower-priority task will only be executed to the extent that it does not interfere with a higher-priority task. For instance, a new position goal at the hands might lead the body to collide with a rock. In such a case the constraint task will engage, and make the body evade the obstacle while still performing the task (Fig. 3).

For each task t , we specify an operational-space ϑ_t , and a control force F_t . The task Jacobian J_t , establishes a dual velocity-force mapping between generalized coordinate space q to operational space ϑ with

$$\vartheta_t = J_t \dot{q} \quad (3)$$

$$\Gamma = J_t^T F_t. \quad (4)$$

To prevent a lower-priority task from interfering with a higher-priority task, we filter Jacobians and control forces through null-space projections. Details on this implementation can be seen in [10].

5.2 Manipulation Task

The central element of the controller is the Manipulation Task, directly programming the hands. It is the only task that takes direct control inputs. These inputs are specified by six-dimensional operational space coordinates for each arm.

$$\vartheta_t^\top = [v_{\text{Left}}^\top, \omega_{\text{Left}}^\top, v_{\text{Right}}^\top, \omega_{\text{Right}}^\top] \quad (5)$$

$$\vartheta_t = J_t \dot{q} \quad (6)$$

v and ω represent the linear and angular velocities of O_2 's hands. In this space we implement unified motion and force control laws expressed in F_t [11]. This abstraction also allows us to specify the manipulation task in a way that is agnostic of the robot's morphology.

5.3 Posture Task

After specifying the twelve-dimensional manipulation task, there are (in general) 8 uncontrolled DoF left. This remaining null-space (Fig. 4) is occupied by the posture task, which consists of two sub-tasks *Body Posture* and *Arm Posture*. *Body Posture* positions the body relative to the hands. Its goal is to align the body's coronal plane with the horizontal plane while aligning its longitudinal axis with the horizontal perpendicular to the axis connecting the two operational points and keeping a constant linear offset to the mid-point of the hands. This sub-task occupies 6 DoF.

$$\vartheta_{p,\text{Body}}^\top = [v_{\text{Body}}^\top, \omega_{\text{Body}}^\top] \quad (7)$$

$$\vartheta_{p,\text{Body}} = J_{p,\text{Body}} \dot{q} \quad (8)$$

Finally, there is 1 DoF of null-space left in each of the arms, which is controlled by the *Arm Posture* sub-task.

$$\vartheta_{p,\text{Arms}}^\top = [\dot{q}_{\text{Left}}^\top, \dot{q}_{\text{Right}}^\top] \quad (9)$$

$$\vartheta_{p,\text{Arms}} = J_{p,\text{Arms}} \dot{q} \quad (10)$$



Fig. 4 Left: Task specification. Every task is associated with an operational-space ϑ_{Task} , illustrated by the cartesian coordinate frames. The manipulation task consists of ϑ_L and ϑ_R , the six-dimensional spaces of the left and right hand—three coordinates for the frame origin (purple sphere) and three for the frame’s orientation. These spaces are controlled directly by the manipulation task, which is the only task receiving control inputs. This abstraction allows for unified motion and force control that is agnostic of the robot’s morphology. The body’s operational space ϑ_{Body} is controlled by the Posture task such that it autonomously follows the hands. Right: Null-space of the manipulation task. The kinematics of O_2 are modeled with 20 DoF. The manipulation task occupies 6 DoF for each of the two arms. The remaining eight are controlled by the posture tasks. Here, O_2 performs a manipulation task on a red valve. The null-space is illustrated by various postures that leave the manipulation task variables (position and orientation of the white hands) undisturbed. This eight-dimensional posture can be used to execute the task in an optimized manner

In order to merge the two sub-tasks into a combined posture task, we stack the Jacobians $J_{p,\text{Body}}$ and $J_{p,\text{Arms}}$ and receive

$$\vartheta_p^T = [\vartheta_{p,\text{Body}}^T, \vartheta_{p,\text{Arms}}^T] \quad (11)$$

$$\vartheta_p = J_p \dot{q}. \quad (12)$$

5.4 Constraints Task

To prevent the robot from damaging itself or obstacles in the environment, we insert a constraint task to which we assign the highest priority. These constraints consist of joint limit avoidance, self collision avoidance and obstacle avoidance. Any action arising from the manipulation and posture tasks that would violate these constraints are filtered directly in the control loop. All three constraints rely on artificial potential fields [12] and use efficient distance computation using capsules. An example is given in Sect. 7.3.

5.5 Hydrodynamic Feed-Forward Compensation

O_2 experiences hydrodynamic effects when operating in an underwater environment, which is captured by the last term in (1). In order to compensate for these forces, we add a feed-forward term ($J_h^\top F_h$) to the controller in (2). Because this computation is part of the controller, we need to rely on models that are executable in real-time. For this purpose, we model O_2 's body, upper arms, lower arms, and hands as rigid links. For each link, we compute two forces: Viscous Damping and Buoyancy. Buoyancy is computed using each link's volume and center-of-buoyancy extracted from the CAD files. For viscous damping, we use the standard model of a cylinder

$${}^B F_D = -\frac{1}{2} \rho \begin{bmatrix} C_x \bar{r}^2 \pi | {}^B \dot{x} | {}^B \ddot{x} \\ C_y 2\bar{r} \bar{L} | {}^B \dot{y} | {}^B \ddot{y} \\ C_z 2\bar{r} \bar{L} | {}^B \dot{z} | {}^B \ddot{z} \end{bmatrix}. \quad (13)$$

We assume a local coordinate system in each link, originating at the center and \hat{x} along the cylinder axis. C_x , C_y and C_z are constant parameters, \bar{m} , \bar{L} and \bar{r} are cylinder parameters. The combined hydrodynamic force on each link is computed with

$${}^0 F_{h,i} = {}^0 R_B {}^B F_D + {}^0 F_B. \quad (14)$$

With the associated Jacobians we can now compute the total hydrodynamic compensation

$$\Gamma_h = (J_h^\top F_h) = \sum_i J_i^\top {}^0 F_{h,i}. \quad (15)$$

5.6 Thruster/Body Control

Ocean One has eight thrusters, four horizontal thrusters arranged in diamond-shape to control yaw and planar translates, four vertical thrusters arranged in square-shape to control roll, pitch and vertical translation. This redundancy allows for holonomic actuation and full maneuverability in case of a thruster failure. The mapping from the eight-dimensional thruster force vector T to six-dimensional generalized force vector Γ acting on the body is given by $\Gamma = J_{\text{Thruster}}^\top T$, more specifically

$$\Gamma = [J_1^\top n_1 \ J_2^\top n_2 \ \dots \ J_8^\top n_8] \begin{bmatrix} T_1 \\ T_2 \\ \vdots \\ T_8 \end{bmatrix}, \quad (16)$$

where J_i^\top is the Jacobian of the i th thruster and n_i its associated unit thrust vector. The robot is controlled in terms of generalized coordinates thus the inversion of (16)

is needed. Because J_{Thruster}^\top is a wide matrix, it is not immediately invertible but two more constraints are required. These constraints arise from the elimination of internal forces and moments given in

$$T_1 + T_2 + T_3 + T_4 = 0 \quad (17)$$

$$T_5 + T_7 - T_6 - T_8 = 0. \quad (18)$$

Inverting this system of equations is equivalent to solving the optimization problem

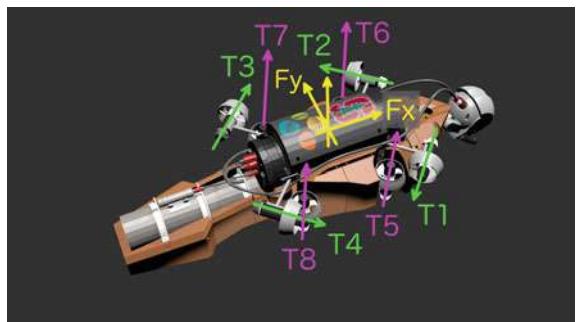
$$\begin{aligned} & \text{minimize} && T^\top T \\ & \text{s.t.} && J_{\text{Thruster}}^\top T. \end{aligned} \quad (19)$$

The thrusters are limited by their force capacity T_{\max} . This limit is imposed by proportionally scaling back all thruster forces until none exceeds this limit.

6 Teleoperation and Haptic Interaction

The pilot and robot are directly coupled at their hands via bimanual haptic devices. The robot mimics the pilot's movements and the pilot receives force feedback that is perceived through the robot's 6D force sensors at the wrists. We refer to this mode of collaboration as *Avatar-mode*. The pilot is stationary at the console while the robot is navigating through space in a holonomic manner. To achieve this mapping, we superimpose position and rate control to compute goal position and orientation of the two hands. For this purpose we introduce an intermediary coordinate frame referred to as *Manipulation-Frame* or *Mframe*. This frame is responsible for the rate contribution, it drifts in translation and yaw in proportion to the sum and difference to the haptic devices' linear positions. The position contributions are hand translations and orientations, directly mapped from the haptic devices' positions and orientations (Fig. 5).

Fig. 5 Four planar thrusters (green T1–T4) and four vertical thrusters (magenta T5–T8) enable holonomic actuation with 2° of redundancy. Three linear and three angular forces acting on a body-fixed coordinate system (yellow) are mapped to thruster forces (T1–T8)



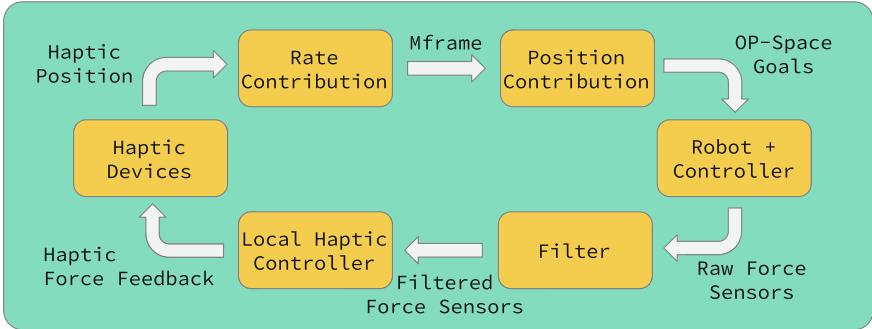


Fig. 6 Haptic teleoperation. The operational-space goals of the manipulation task are computed by a hybrid rate and position signal. The rate contribution is applied to a virtual coordinate frame (*Mframe*), and the position contribution is expressed in this frame. The controller enforces these goals. The raw force signals, measured at the wrists are filtered to remove very high frequency noise and forwarded and superimposed with a local haptic device controller, which allows the pilot to perform guided motions

These goals are then forwarded and enforced by the Manipulation task. O_2 's contact forces are measured by force sensors located at the wrists. The raw signals are passed through filters in order to eliminate high frequency noise while maintaining haptic transparency. The haptic devices do not only reflect the filtered contact forces but are actively controlled. This allows the pilot to perform guided motions, which simplifies the teleoperation task for the pilot by reducing its dimensionality. For instance, certain fetching tasks only require 1 active DoF in orientation, and a docking maneuver only requires 1 linear DoF (Fig. 6).

7 Simulation

Simulation played an integral part throughout the development of O_2 . Most importantly, it enabled us to develop and test O_2 's software stack prior to fabrication and deployment on the physical robot. It also informed mechanical design choices and allowed us to train the pilots and practice entire missions. SAI2 is a real-time interactive simulation environment comprised of a collection of libraries that include the simulation of multi-body dynamic systems, contact and collision resolution. In addition, we utilized the Chai3D [13] libraries to facilitate haptic and visual rendering.

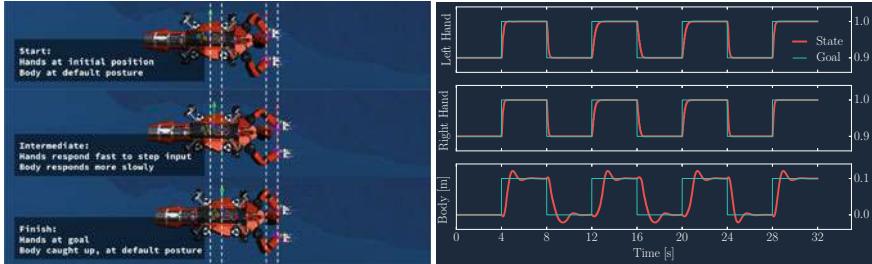


Fig. 7 A square wave goal position input is applied to the hands’ operational points. We observe that the hands’ dynamic responses are much faster than the body’s alone. The combined macro-mini dynamics of slow body and fast arms allows for fast overall dynamics. The body is limited by its large inertia and comparatively low actuation capabilities

7.1 Step Response in Operational Space

O_2 ’s kinematic structure can be decomposed into three parts. The body, referred to as *macro-manipulator* with 6 DoF and two arms referred to as *mini-manipulators* with 7 DoF each. This is a valid decomposition because the minis have full range in operational space and the macro has at least 1 DoF [14]. The serial combination of macro and mini offers two advantages. First, the effective inertias of the macro-mini combination is upper bounded by the effective inertias of the mini-manipulator alone. Second, the dynamic performance of the macro-mini can be made comparable to that of the mini.

This behavior is illustrated in Fig. 7. A square wave function is applied as position-goal for both hands in forward-direction, while lateral and vertical position-goals remain constant. The step-response of the body alone (bottom) shows slow dynamics with large overshoot, oscillation and long duration for convergence. This behavior is due to the body’s large inertia, and comparatively weak actuation capabilities due to thruster force limitations. The step-responses of the hands’ combined macro-mini dynamics display fast response with small rise-time and critical convergence. Hence, the fast, lightweight arms compensate for the slower body and overall response is fast and accurate.

7.2 Docking Maneuver with Force Control

To illustrate the advantages of whole-body coordination in force control mode, we perform a docking maneuver, where both hands apply and maintain a force normal to a given plane. The maneuver initiates at close proximity to the contact surface, where

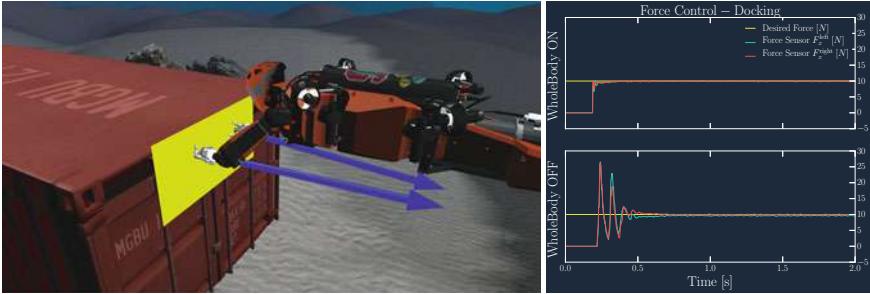


Fig. 8 We compare a docking maneuver with and without whole-body control. The hands initialize at close proximity to the red container. The collision plane is indicated by the yellow rectangle. We apply force control in forward-direction and position control in the orthogonal plane, on both hands, the desired force is 10N. The forces are rendered by the blue vectors. We observe that whole-body control leads to superior transitions (reduced spikes) and steady state behavior (faster convergence, smaller errors)

force control is activated in the forward-direction and position control is maintained in the orthogonal sub-space. Figure 8 shows the results comparing force control with and without whole-body coordination. We see that whole-body coordination leads to superior transitional and steady state behavior. The spikes during transition are greatly reduced, convergence is faster, and steady state errors are smaller.

7.3 Obstacle Avoidance

We simulate a scenario, where O_2 manipulates a container while avoiding local obstacles. To do this, we enclose O_2 in five (green) collider capsules and the obstacle in a (red) collision capsule (Fig. 9). In every servo loop, we monitor the distances between colliders and obstacles. In case a distance is smaller than the specified activation distance ρ_0 , the constraint task is activated and an artificial potential field is applied to avoid the collision. In the given scenario, we program O_2 to unscrew the container’s lid by specifying circle-segment trajectories at its hands. Without obstacle avoidance, O_2 ’s body would be sweeping through the obstacle during this motion. The smallest distance between the obstacle and O_2 ’s body is rendered by the red line segment between the blue and red spheres. Instead of colliding with the obstacle, the artificial potential field leads the body to glide over the barrel while the trajectory of the hands remains unaltered. The comparison between active and inactive obstacle avoidance is given in Fig. 9.

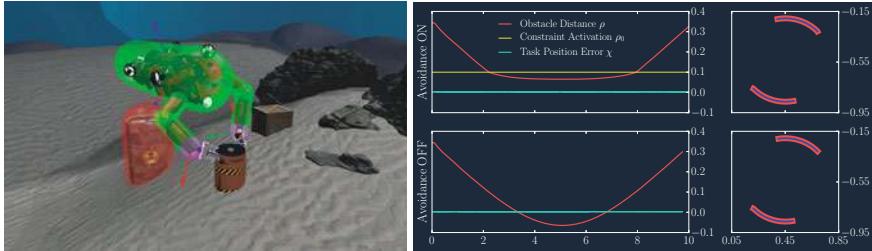


Fig. 9 Left: O_2 performs a manipulation task on the red container, while avoiding obstacles. In this scenario, O_2 unscrews the black lid of the red container. The manipulation task is programmed by applying a circular trajectory to the hands, which leads the body to rotate along. Without avoidance, this specification would lead to a collision between the tail and the rusty barrel. In order to complete the task without collision, we model the robot’s body and arms collider hulls with green capsules, and the obstacle with a red capsule. The red line between the smaller red and blue sphere indicates the smallest distance between the body-collider and the obstacle. We see that as the lid is rotated, the body sweeps above the barrel while maintaining a safety margin. Right: On the left column we plot the obstacle distance (red line) and task position error (sum of the two hands’ error, blue line). We see that with deactivated avoidance, the distance becomes negative, indicating collision and intersection. With activated avoidance, the constraint task triggers when $\rho < \rho_0$ (yellow line), avoiding the obstacle. The right column shows the hands’ goal (thick red line) and state (slim blue line) trajectories sweep. We observe that in both cases the task is performed with the same precision

8 Deployment and Experimental Results

After O_2 ’s hardware components were assembled, we deployed it in shallow depth at the Stanford Aquatic Center. We experimentally tuned parameters for buoyancy compensation and validated the kinematic and dynamic models as well as the sensors and communication protocols. The pool also offered the first opportunity to practice piloting the robot during navigation, grasping, and docking operations. In Fig. 10 (left) we compare the body’s dynamic responses between simulated and physical robot by applying sinusoidal trajectories at 0.05 Hz and 45° amplitude to yaw and 0.3 m to depth. The responses align well with the exception of some coupling that is likely due to unmodeled hydrodynamic contributions. In Fig. 10 (right) we compare the hands’ responses in operational-space position control by applying sinusoidal trajectories at 0.1 Hz to all three cartesian directions. Again, we observe good alignment between simulation and physical robot with additional coupling. The physical robot exhibits slightly decreased amplitudes, which is likely a result of under-estimated hydrodynamics and friction in the arm joints.

O_2 ’s maiden mission took place at an archeological site in the mediterranean sea near the coast of Toulon, France. The *Lune* is a two-decked, fifty-four-gun french naval vessel of *Lois XIV*’s that sunk in 1664 with nearly a thousand men on board to 91 m of depth, where it was discovered in 1993 by *Nautile*, a submarine of the French Oceanographic Institute. The mission was executed from the *Andre Malraux*, a research vessel operated by DRASSM [15]. After initial tests at 15 m depth, collaborating with a human diver, O_2 descended to a 4 h long mission to the *Lune*,

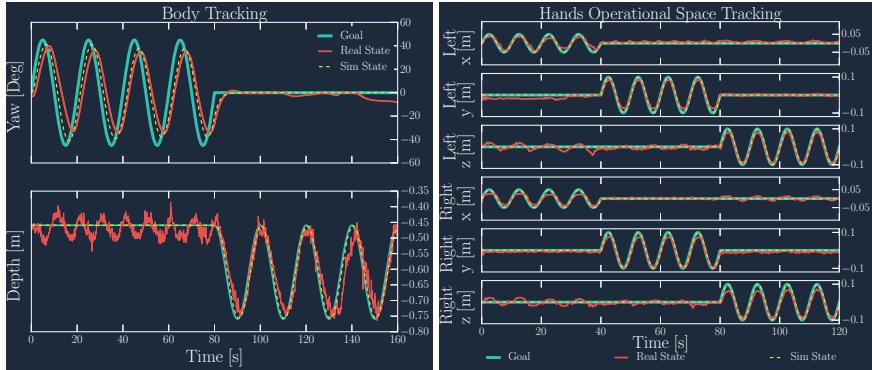


Fig. 10 Simulation Versus Real robot. Left: Dynamic Body Tracking. Sinusoidal trajectories at 0.05 Hz are consecutively tracked in yaw and depth coordinates. Simulation and real robot responses are well aligned with the exception of some coupling that is likely due to unmodeled hydrodynamic components. Right: Dynamic Operational-Space tracking. Sinusoidal trajectories at 0.1 Hz are tracked in forward (x), lateral (y), and vertical (z) directions, consecutively. Simulation and real robot are well aligned, there is more dynamic coupling and a decreased amplitude at the real robot, which is likely due to underestimated hydrodynamics and friction in the joints



Fig. 11 Ocean one deployed in Toulon, France. Left: collaboration with a diver at 15 m depth. top center/right: O₂ fetching a vase and dropping it into the collection basket. bottom center/right: researchers collecting the floated basket, Catalan vase on deck showing its acquired biofilm. (Photos courtesy of Frederic Osada and Teddy Seguin, DRASSM/stanford university)

where it explored the site, fetched a vase and deposited it in a collection box that was subsequently floated to the surface.

9 Conclusion

In this paper, we focussed on O₂'s control architecture. We illustrated the hierarchical implementation of whole-body control and showed how to create an immersive interface with a human pilot that enabled an avatar-like collaboration. We demonstrated the system's capabilities in simulated whole-body control, force-controlled docking maneuvers as well as a manipulation task involving autonomous obstacle avoidance. We validated the dynamic models and controller with experiments in the pool and finally established O₂'s effectiveness with its deployment to its maiden mission in the mediterranean sea (Fig. 11).

References

1. Hägele, M., Nilsson, K., Pires, J.N., Bischoff, R.: Industrial Robotics in Springer Handbook of Robotics. Springer, pp. 1385–1422 (2016)
2. Groover, M.P.: Automation, Production Systems, and Computer-Integrated Manufacturing. Prentice Hall Press (2007)
3. Dudek, G., Giguere, P., Prahacs, C., Saunderson, S., Sattar, J., Torres-Mendez, L.-A., Jenkin, M., German, A., Hogue, A., Ripsman A., et al.: Aqua: an amphibious autonomous robot. *Computer*, **40**(1) (2007)
4. Vasilescu, I., Varshavskaya, P., Kotay, K., Rus, D.: Autonomous modular optical underwater robot (amour) design, prototype and feasibility study. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005. pp. 1603–1609. IEEE (2005)
5. Kimball, P.W., Rock, S.M.: Mapping of translating, rotating icebergs with an autonomous underwater vehicle. *IEEE J. Oceanic Eng* **40**(1), 196–208 (2015)
6. Schilling Robotics, LLC, 260 Cousteau Place, Davis, CA 95618, USA, Homepage. <http://www.schilling.com> (Accessed: December. 14, 2011). Internet (2011)
7. ECA Group, Homepage. <http://www.ecagroup.com/en/solutions/arm-5e-micro> (Accessed: September. 18, 2016), Internet (2016)
8. Stuart, H., Wang, S., Gardineer, B., Christensen, D.I., Aukes, D.M., Cutkosky, M.R.: A compliant underactuated hand with suction flow for underwater mobile manipulation. In: ICRA 2014, pp. 6691–6697 (2014)
9. Khatib, O., Yeh, X., Brantner, G., Soe, B., Kim, B., Ganguly, S., Stuart, H., Wang, S., Cutkosky, M., Edsinger, A., et al.: Ocean one: a robotic avatar for oceanic discovery. *IEEE Robot. Autom. Mag.* **23**(4), 20–29 (2016)
10. Khatib, O., Sentis, L., Park, J., Warren, J.: Whole-body dynamic behavior and control of human-like robots. *Int. J. Humanoid Robot.* **1**(01), 29–43 (2004)
11. 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)
12. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. In: Autonomous Robot Vehicles 1986, pp. 396–404. Springer (1986)
13. Conti, E., Barbagli, F., Balaniuk, R., Halg, M., Lu, C., Morris, D., Sentis, L., Warren, J., Khatib, O., Salisbury, K.: The chai libraries. In: Proceedings of Eurohaptics 2003, pp. 496–500. Dublin, Ireland (2003)
14. Khatib, O.: Inertial properties in robotic manipulation: an object-level framework. *Int. J. Robot. Res.* **14**(1), 19–36 (1995)
15. L'Hour, M.: The french department of underwater archaeology: a brief overview. *Eur. J. Archaeol.* **15**(2), 275–284 (2012)

Safe Self-collision Avoidance for Versatile Robots Based on Bounded Potentials

David Gonon, Dominic Jud, Péter Fankhauser and Marco Hutter

Abstract We present a novel and intrinsically safe collision avoidance method for torque- or force-controlled robots. We propose to insert a dedicated module after the nominal controller into the existing feedback loop to blend the nominal control signal with repulsive forces derived from an artificial potential. This blending is regulated by the system’s mechanical energy in a way that guarantees collision avoidance and at the same time allows navigating close to collisions. Although using well-known ingredients from previous reactive methods, our approach overcomes their limitations in respect of achieving reliability without significantly restricting the set of reachable configurations. We demonstrate the fitness of our approach by comparing it to a standard potential-based method in simulated experiments with a walking excavator.

1 Introduction

When deploying robots in the field instead of factories, new challenges arise in the design of safety systems that are able to cope with the varying modes of operation. This particularly applies to forestry, agriculture and construction, where research is focusing on (semi-)automation and teleoperation of the involved heavy machinery [4, 5, 13]. These machines are especially prone to colliding with *themselves* due to their high versatility, which is essential for tasks in unstructured environments. A prime example are walking excavators such as the M545 developed by Menzi Muck

D. Gonon (✉) · D. Jud · P. Fankhauser · M. Hutter
Robotic Systems Lab, Swiss Federal Institute of Technology Zurich,
8092 Zürich, Switzerland
e-mail: david.gonon@gmx.net

D. Jud
e-mail: dominic.jud@mavt.ethz.ch

P. Fankhauser
e-mail: pfankhauser@ethz.ch

M. Hutter
e-mail: mahutter@ethz.ch

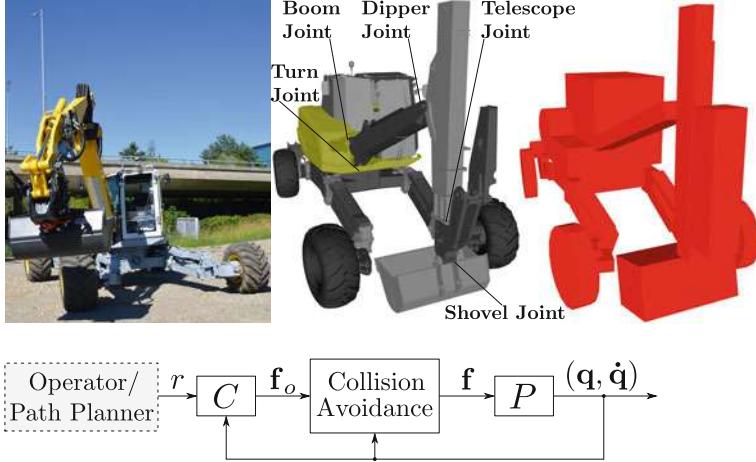


Fig. 1 Top row: The M545 (on the left), its CAD model with joint labels (in the middle) and the approximate geometric model we use for collision avoidance (on the right). Bottom: Schematic view of the control loop with our collision avoidance module inserted. It modifies the force/torque commands \mathbf{f}_o , which the nominal controller C generates from the reference r and the state feedback $(\mathbf{q}, \dot{\mathbf{q}})$, to steer the mechanical system P with the resulting commands \mathbf{f}

AG¹ (depicted in Fig. 1), which motivated our development of a suitable collision avoidance method.

In order to be generally applicable, a method should be (a) usable both for assisting human operators and for supervising autonomous maneuvers and (b) easily integrable into an existing control system. Probably the most straightforward way to meet these requirements is to base collision avoidance on insertion of a dedicated module into the existing control loop. In this work, we focus on such a design where the respective module is inserted after the nominal controller (as shown in Fig. 1). We present a corresponding solution, which is applicable to any fully actuated robotic system with a controller producing force or torque commands utilizing full state feedback.

Our approach builds up on *artificial potentials*, which have been used by previous works to plan a path and at the same time obtain a corresponding feedback law [6, 9, 10]. When using artificial potentials for robot control, one can elegantly guarantee collision avoidance via energy considerations. One possibility is to let the potential function approach infinity towards the boundary of the set of collision-free configurations [6]. However, this is impractical as it requires unbounded joint torques. Conversely, it has been suggested to use finite potential functions and to confine the robot to states whose (artificial) energy is lower than the value of the potential function for any collision configuration [9]. Our method employs the same reasoning and similarly guarantees collision avoidance based on the principle of energy conservation.

¹<http://www.menzimuck.com>.

On the other hand, our approach implements a form of *reactive control*, which denotes a generic class of methods that are designed to react spontaneously to events (such as the robot getting close to an obstacle) and typically integrate collision avoidance into a hierarchy or “stack” of tasks [2, 8]. In contrast, other reactive collision avoidance approaches [1, 10] simply add repulsive forces to the nominal commands and therefore exhibit the modularity we desire. They also derive the repulsive actions from artificial potentials, as it is common for reactive methods. Consequently, they inherit the well-known properties of artificial potentials to prohibit navigation through narrow passages and to trap the robot in local minima [7]. Our method overcomes these limitations to a large extent by applying the potential force only when it is found to be necessary to ensure collision avoidance (again employing the same energy consideration).

However, the most significant improvement is the guarantee for collision avoidance (provided that the required forces or torques are feasible), which additive methods [1, 10] cannot have by design. With them, reliability at best results from a proper tuning of the potential height and needs to be verified by extensive testing. Note that for reactive methods based on velocity commands [8, 11, 12] such a theoretical guarantee is not attainable either, as they do not take into account the robot’s dynamics, which are formulated on the force level. To the best of our knowledge, our way to incorporate artificial potentials into reactive control is novel, especially as it achieves safe collision avoidance while largely preserving workspace accessibility.

The remainder of this article is organized as follows: Sect. 2 introduces the preliminary notations and results as well as an additive reactive collision avoidance method, which will serve as a baseline for comparison with our approach. Then, in Sect. 3 we describe our method and in Sect. 4 we experimentally compare it to the baseline method. Finally, Sect. 5 gives a conclusion.

2 Preliminaries

Notation: We denote column vectors by bold face lower-case letters. Then, for a scalar a and two vectors \mathbf{b} , \mathbf{c} , let $\partial a / \partial \mathbf{b}$ and $\partial \mathbf{c} / \partial \mathbf{b}$ denote the row vector and the matrix having the entries $\partial a / \partial b_j$ and $\partial c_i / \partial b_j$, respectively. Further, we use dots to indicate differentiation with respect to time (e.g. $\dot{a} \equiv \frac{d}{dt} a$).

2.1 Artificial Potentials and Energy Conservation

We consider mechanical systems subject to holonomic constraints whose equations of motion take the form

$$\frac{d}{dt} \frac{\partial T}{\partial \dot{\mathbf{q}}} - \frac{\partial T}{\partial \mathbf{q}} + \frac{\partial V_g}{\partial \mathbf{q}} = \mathbf{f}^T, \quad (1)$$

with the kinetic energy T , the gravitational potential V_g , the generalized coordinates $\mathbf{q} \in \mathbb{R}^N$ and the generalized forces $\mathbf{f} \in \mathbb{R}^N$. We let \mathbf{q} represent the joint variables (angles) and \mathbf{f} the joint forces or torques, which constitute the control vector, as we focus on fixed-base robotic systems.² With this particular choice of \mathbf{q} , the kinetic energy is not explicitly dependent on time and takes the form $T = \frac{1}{2}\dot{\mathbf{q}}^T \mathbf{M}(\mathbf{q})\dot{\mathbf{q}}$, where \mathbf{M} is the mass matrix.

We then consider a control vector of the form $\mathbf{f} = \partial(V_g - V_A)/\partial\mathbf{q}^T + \tilde{\mathbf{f}}$, where $V_A(\mathbf{q})$ is an artificial potential field and $\tilde{\mathbf{f}}$ is a new (yet unspecified) control vector. Substituting this in (1) and rearranging terms yields

$$\frac{d}{dt} \frac{\partial T}{\partial \dot{\mathbf{q}}} - \frac{\partial T}{\partial \mathbf{q}} + \frac{\partial V_A}{\partial \mathbf{q}} = \tilde{\mathbf{f}}^T. \quad (2)$$

Then, given the functional forms of $T = T(\mathbf{q}, \dot{\mathbf{q}})$ and $V_A = V_A(\mathbf{q})$, one can derive from (2) the rate of change of the (artificial) energy $E_A := T + V_A$ as

$$\dot{E}_A = \dot{\mathbf{q}}^T \tilde{\mathbf{f}}, \quad (3)$$

implying $\dot{E}_A = 0$ for $\tilde{\mathbf{f}} = 0$, which is the principle of energy conservation [3].

2.2 Obstacles, Closest Points and Collisions

A scene consisting of a robotic system and its static environment can be seen as a collection of rigid bodies, and we define that a collision occurs if for any pair of bodies $(i, j) \in P_c$ the mutual distance d_{ij} equals zero. Here, P_c denotes a subset of all possible body pairs, and d_{ij} is defined as the shortest distance two points can have when one is chosen on body i and the other on body j . We denote a pair of such points that have this distance and lie on body i, j , respectively, by $(\mathbf{p}_{ij}, \mathbf{p}_{ji})$. We further define the corresponding unit vectors \mathbf{e}_{ij} and Jacobian matrices \mathbf{J}_{ij} as

$$\mathbf{e}_{ij} := (\mathbf{p}_{ij} - \mathbf{p}_{ji})/|\mathbf{p}_{ij} - \mathbf{p}_{ji}|, \quad \mathbf{J}_{ij} := (\partial \mathbf{p}_{ij} / \partial \mathbf{q})_{\text{body-point}}, \quad (4)$$

where \mathbf{p}_{ij} is regarded as a fixed body point during differentiation (according to the standard definition of the Jacobian).

Defining that only certain body pairs can produce a collision is useful in practice, since some bodies are normally in contact (e.g. adjacent links) and others need not to be taken into account because they can never collide. Observe that for static obstacles the d_{ij} are continuous functions in \mathbf{q} only.

²We argue that they are an accurate approximation of the M545 and similar machines with a heavy and sufficiently stiff base. In the remainder of this article, we therefore make this simplifying assumption. However, in general it is not necessary for our method, which only requires that all degrees of freedom are actuated.

Further, we define the globally shortest distance d^* as

$$d^*(\mathbf{q}) := \min_{(i,j) \in P_c} d_{ij}(\mathbf{q}) \quad (5)$$

and let $(\mathbf{p}_1^*, \mathbf{p}_2^*)$, $(\mathbf{e}_1^*, \mathbf{e}_2^*)$ and $(\mathbf{J}_1^*, \mathbf{J}_2^*)$ denote the corresponding pairs of globally closest points, unit vectors and Jacobian matrices, respectively (as defined above). Then, let $Q_c := \{\mathbf{q} \in \mathbb{R}^N | d^*(\mathbf{q}) > 0\}$ denote the set of collision-free configurations.

2.3 Classic Collision Avoidance with Artificial Potentials

As a baseline to compare our method to in Sect. 4, we briefly introduce a standard approach to collision avoidance presented in textbooks [10]. It is a straightforward implementation of additive potential-based reactive control (see Sect. 1).

The control vector takes the form $\mathbf{f} = \mathbf{f}_o + \mathbf{f}_A + \mathbf{f}_D$, where we add to the nominal control \mathbf{f}_o the artificial potential force \mathbf{f}_A and some damping \mathbf{f}_D . Let $\mathbf{f}_A := -\partial V_A / \partial \mathbf{q}^T$, and we define the artificial potential V_A using the terminology introduced in Sect. 2.2 as

$$V_A(\mathbf{q}) = \sum_{(i,j) \in P_c} U(d_{ij}(\mathbf{q})), \quad (6)$$

with the unilateral spring potential U being defined as

$$U(d) = \begin{cases} \frac{\alpha}{2}(d_0 - d)^2 & d \leq d_0 \\ 0 & d > d_0, \end{cases} \quad (7)$$

where $d_0 > 0$ is the range of repulsion and $\alpha > 0$ is the spring stiffness. Then, \mathbf{f}_A takes the explicit form

$$\mathbf{f}_A = \sum_{\{(i,j) \in P_c | d_{ij} \leq d_0\}} \alpha(d_0 - d_{ij})(\mathbf{J}_{ij}^T \mathbf{e}_{ij} + \mathbf{J}_{ji}^T \mathbf{e}_{ji}). \quad (8)$$

Further, we define \mathbf{f}_D such that it is equivalent to the forces of viscous dampers attached to the closest points. Let

$$\mathbf{F}_{D,ij} := -\kappa(1 - d_{ij}/d_0)(\mathbf{e}_{ij} \cdot \mathbf{J}_{ij} \dot{\mathbf{q}})\mathbf{e}_{ij}, \quad d_{ij} \leq d_0 \quad (9)$$

denote such a force with a damping factor increasing linearly up to $\kappa > 0$ with decreasing distance. Then, we define the corresponding generalized forces as

$$\mathbf{f}_D := \sum_{\{(i,j) \in P_c \mid d_{ij} \leq d_0\}} (\mathbf{J}_{ij}^T \mathbf{F}_{D,ij} + \mathbf{J}_{ji}^T \mathbf{F}_{D,ji}). \quad (10)$$

3 Method

Before describing our method, we outline the underlying generic idea (Sect. 3.1), which allows it to guarantee collision avoidance while preserving workspace accessibility. The subsequent sections then describe several extensions we make to derive our method from this idea.

3.1 Energy Bounding Control Strategy

We guarantee collision avoidance based on energy considerations [9]. Namely, we employ an artificial potential V_A , which reaches its supremum over \mathcal{Q}_c , denoted by

$$\hat{V}_c := \sup_{\mathbf{q} \in \mathcal{Q}_c} V_A(\mathbf{q}), \quad (11)$$

everywhere on the boundary of \mathcal{Q}_c , denoted by $\partial \mathcal{Q}_c$, i.e.

$$\forall \mathbf{q} \in \partial \mathcal{Q}_c, \quad V_A(\mathbf{q}) = \hat{V}_c. \quad (12)$$

Then, a way to ensure collision avoidance is to control the (artificial) energy E_A to stay below \hat{V}_c , since this implies $V_A(\mathbf{q}(t)) < \hat{V}_c$ (as $T \geq 0$), which implies that $\mathbf{q}(t)$ cannot cross $\partial \mathcal{Q}_c$. A suitable control law would be e.g.

$$\mathbf{f} = \begin{cases} \mathbf{f}_o & E_A \leq E_{A,\max} \\ (\partial V_g / \partial \mathbf{q} - \partial V_A / \partial \mathbf{q})^T + \mathbf{f}_D & E_A > E_{A,\max}, \end{cases} \quad (13)$$

where $E_{A,\max} < \hat{V}_c$ and the damping $\mathbf{f}_D(\mathbf{q}, \dot{\mathbf{q}})$ fulfills $\mathbf{f}_D^T \dot{\mathbf{q}} < 0$. Then, as soon as $E_A > E_{A,\max}$, this control law causes $\dot{E}_A < 0$ (according to Sect. 2.1) and therefore E_A can never reach \hat{V}_c .

3.2 Artificial Potential Force

Using the terminology introduced in Sect. 2.2, we define our particular choice of artificial potential as

$$V_A(\mathbf{q}) := \max\{0, \alpha(d_0 - d^*(\mathbf{q}))\}, \quad (14)$$

where $d_0 > 0$ is the range and $\alpha > 0$ the strength of the potential force. The negative gradient of V_A yields the generalized artificial potential force as

$$\mathbf{f}_A = \begin{cases} \alpha(J_1^{*T} \mathbf{e}_1^* + J_2^{*T} \mathbf{e}_2^*) & d^* \leq d_0 \\ 0 & d^* > d_0. \end{cases} \quad (15)$$

3.3 Dissipative Repulsion and Damping

The control law (13) disables any nominal control if we assume that the robot is in a state with $E_A > E_{A,\max}$. However, this situation is only temporary due to damping. Additionally, in order to assist the process of energy dissipation, we partially suppress the artificial potential force when its effect would be to increase the system's kinetic energy. To this end, we scale the contribution of \mathbf{f}_A to the control vector \mathbf{f} by the factor $\omega_A \in [0, 1]$ defined as

$$\omega_A := \max\{0, \min\{1, 1 - \varepsilon \dot{d}^*\}\}, \quad (16)$$

where $\varepsilon > 0$ and \dot{d}^* is the rate of change of the globally shortest distance, given as $\dot{d}^* = \mathbf{e}_1^{*T} (J_1^* - J_2^*) \dot{\mathbf{q}}$. Observe that ω_A vanishes as soon as the globally closest points are moving away from each other with a speed greater than $1/\varepsilon$. Conversely, the absorption of energy due to \mathbf{f}_A is undiminished, as $\omega_A = 1$ for $\dot{d}^* < 0$.

The particular damping term we use corresponds to viscous forces acting on the pair of globally closest points. With the damping constant $\kappa > 0$, we define \mathbf{f}_D as

$$\mathbf{f}_D := -\kappa (J_1^{*T} J_1^* + J_2^{*T} J_2^*) \dot{\mathbf{q}}. \quad (17)$$

3.4 Transitional Switching

We define a transitional zone between some energy level $E_{A,\text{trans}} < E_{A,\max}$ and $E_{A,\max}$, where both nominal control and collision avoidance forces are applied to some extent. This replaces the switching behavior from (13), where always exclusively one of the two can be active. How this relates to our energy considerations (Sect. 3.1) and the potential function (Sect. 3.2) is illustrated in Fig. 2.

Now, let \mathbf{f}_o^L denote the *limited nominal control*, which is the contribution of \mathbf{f}_o to the actual control signal. If the robot is in a state with $E_A > E_{A,\text{trans}}$, we restrict this contribution by demanding that

$$\|\mathbf{f}_o^L\| \leq \lambda_E f_{\max}, \quad \text{if } E_A \geq E_{A,\text{trans}}, \quad (18)$$

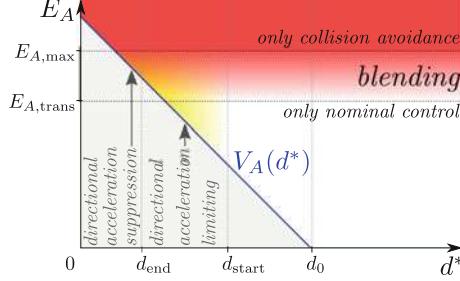


Fig. 2 Zones in the d^* - E_A -diagram with different associated modes of blending nominal control and collision avoidance. Note that the blending is regulated exclusively by the globally shortest distance d^* and the (artificial) energy E_A that the robot exhibits. The zones’ shapes depend on the artificial potential function $V_A(d^*)$ as well as the parameters $E_{A,\max}$, $E_{A,\text{trans}}$, d_0 , d_{start} and d_{end}

where $\|\cdot\|$ denotes the euclidean norm, $f_{\max} > 0$ and λ_E is defined as

$$\lambda_E(E_A) := \max \left\{ 0, \frac{E_{A,\max} - E_A}{E_{A,\max} - E_{A,\text{trans}}} \right\}. \quad (19)$$

How we actually compute \mathbf{f}_o^L from \mathbf{f}_o such that it respects this limit is the subject of the next section.

Also, λ_E regulates the extent to which damping and artificial potential forces are applied. Mainly for the sake of a cleaner notation, we define two corresponding scaling factors, which we derive from λ_E , as

$$\lambda_D(\lambda_E) := \max\{0, 1 - \lambda_E\}, \quad (20)$$

$$\lambda_A(\lambda_E) := \max\{0, 1 - \gamma \lambda_E\}, \quad (21)$$

where $\gamma > 1$ (we consistently use $\gamma = 8$). We then state the transitional control law

$$\mathbf{f} = \mathbf{f}_o^L + \mathbf{g} + \lambda_A \omega_A \mathbf{f}_A + \lambda_D \mathbf{f}_D, \quad (22)$$

where $\mathbf{g} := \partial V_g / \partial \mathbf{q}^T$ denotes gravity compensation.

For $E_A > E_{A,\max}$, we have again $\dot{E}_A < 0$ since then $\mathbf{f}_o^L = 0$ (according to (18) and (19)) while the damping force \mathbf{f}_D and the “dissipative potential force” $\omega_A \mathbf{f}_A$ are fully active (according to (20) and (21)). Therefore, (22) guarantees collision avoidance in the same way as (13).

3.5 Limiting of Nominal Control

Our method computes the limited nominal control \mathbf{f}_o^L in two stages. Beforehand, we subtract the gravity compensation term \mathbf{g} from \mathbf{f}_o as it is already present in (22).

3.5.1 Limiting the Collision Acceleration

The first limit only applies if d^* is smaller than some threshold $d_{\text{start}} > 0$. Then, this step computes a modification of its input $\tilde{\mathbf{f}}_o := \mathbf{f}_o - \mathbf{g}$, whose contribution to the relative acceleration of the globally closest points towards each other does not exceed a certain bound. This bound is proportional to

$$\lambda_d(d^*) := \max \left\{ 0, \frac{d^* - d_{\text{end}}}{d_{\text{start}} - d_{\text{end}}} \right\}, \quad (23)$$

with d_{start} and d_{end} specifying where limiting starts and where it reaches its full effect, respectively. They are to be chosen such that $0 < d_{\text{end}} < d_{\text{start}}$.

We first compute the joint accelerations that $\tilde{\mathbf{f}}_o$ would contribute according to

$$\ddot{\mathbf{q}}_o = \mathbf{M}^{-1} \tilde{\mathbf{f}}_o. \quad (24)$$

Then, we compute a new acceleration vector $\ddot{\mathbf{q}}'_o$, which differs from $\ddot{\mathbf{q}}_o$ only in its component along the steepest descent of shortest distance $-\partial d^*/\partial \mathbf{q}^T$, according to

$$\ddot{\mathbf{q}}'_o = \ddot{\mathbf{q}}_o - \mu (-\mathbf{J}_1^{*T} \mathbf{e}_1^* - \mathbf{J}_2^{*T} \mathbf{e}_2^*), \quad (25)$$

with $\mu \geq 0$ being chosen as small as possible such that

$$a'_c := (-\mathbf{J}_1^{*T} \mathbf{e}_1^* - \mathbf{J}_2^{*T} \mathbf{e}_2^*)^T \ddot{\mathbf{q}}'_o \leq \lambda_d a_{\text{max}}, \quad (26)$$

where a'_c is the acceleration of the closest points towards each other due to $\ddot{\mathbf{q}}'_o$ and $a_{\text{max}} > 0$ specifies the maximum value of the bound on a'_c . The μ satisfying this is given by

$$\mu = \max \left\{ 0, \frac{(-\mathbf{J}_1^{*T} \mathbf{e}_1^* - \mathbf{J}_2^{*T} \mathbf{e}_2^*)^T \ddot{\mathbf{q}}_o - \lambda_d a_{\text{max}}}{(\mathbf{J}_1^{*T} \mathbf{e}_1^* + \mathbf{J}_2^{*T} \mathbf{e}_2^*)^T (\mathbf{J}_1^{*T} \mathbf{e}_1^* + \mathbf{J}_2^{*T} \mathbf{e}_2^*)} \right\}.$$

With it, we compute $\ddot{\mathbf{q}}'_o$ from (25), and multiplication with the mass matrix then gives the corresponding generalized forces achieving this acceleration. The output of this step is therefore

$$\mathbf{f}'_o = \begin{cases} \mathbf{M} \ddot{\mathbf{q}}'_o & d^* \leq d_{\text{start}} \\ \tilde{\mathbf{f}}_o & d^* > d_{\text{start}}. \end{cases} \quad (27)$$

3.5.2 Limiting the Nominal Control's Norm

Next, only if $E_A \geq E_{A,\text{trans}}$, we enforce a limit on the norm of \mathbf{f}'_o , which is the output of the first limiting procedure described above. To be specific, we compute the limited nominal control \mathbf{f}_o^L according to

$$\mathbf{f}_o^L = \begin{cases} v \mathbf{f}'_o & E_A \geq E_{A,\text{trans}} \\ \mathbf{f}'_o & E_A < E_{A,\text{trans}}, \end{cases} \quad (28)$$

with $v \in [0, 1]$ in the first case being chosen as large as possible such that the limit (18) is respected. The value of v then follows as $v = \min\{1, \lambda_E f_{\max}/||\mathbf{f}'_o||\}$.

3.6 Summary

The final transitional control law (22) adds several new features to the basic variant (13). First of all, it increases energy dissipation via the factor ω_A , which partially hinders \mathbf{f}_A to feed back the energy stored in V_A . Next, we introduced a blending zone, where both collision avoidance and nominal control are active, to prevent high-frequency oscillations due to switching (“jittering”). The latter would occur in the control signal e.g. when E_A reaches $E_{A,\max}$ due to acceleration by the nominal control. The limiting procedure (Sect. 3.5) implements a measure to prevent oscillating motions when \mathbf{f}_o actively pushes the robot into an obstacle. Namely, it selectively and rigorously reduces the ability of \mathbf{f}_o to accelerate the robot towards an obstacle whenever it is near one (i.e. $d^* < d_{\text{start}}$, also marked in Fig. 2).

Regarding workspace accessibility, the transitional control law makes only a little restriction, as it only enforces slower motions near obstacles (as visible in Fig. 2, where the white space above $V_A(d^*)$ corresponds to the allowed kinetic energies). Particularly, the robot can take any path through the configuration space along which $d^* > d_{\text{end}}$ and $V_A < E_{A,\text{trans}}$, provided that one can arbitrarily slow down the motion on this path in order to respect the kinetic energy and acceleration limits. Excavators and manipulators typically comply with this condition as they are strong enough to continuously compensate gravity. Thus, they do not need to exploit their dynamics as they can perform every motion in a quasi-static fashion.

4 Experiments

We compare our method to a standard approach based on artificial potentials (Sect. 2.3) in experiments with a simulation of the walking excavator M545. In this section, we refer to the standard approach as AP and to our method as AP+.

When applying the geometric considerations described in Sect. 2.2, we represent the excavator’s links by simplistic triangular meshes (depicted in Fig. 1). Note that the shapes are not strictly convex [11] and consequently the closest points ($\mathbf{p}_{ij}, \mathbf{p}_{ji}$) are not uniquely defined for certain parallel configurations. However, these are not important in practice since they only occur in infinitely short time intervals.

4.1 Experimental Design

We compare both methods in three experiments testing different requirements:

1. *Driving with full force into a collision*: In this safety check, the excavator's shovel is pushed down into one of its wheels (as shown in Fig. 3) to test the methods' ability to prevent a collision even if the operator deliberately tries to cause one.
2. *Steering through a narrow passage*: Here, the excavator's shovel is steered into the space between its front legs (as illustrated by Fig. 4) in order to examine how workspace accessibility and versatility are affected.
3. “*Scratching*” collision: In this experiment, the excavator turns while the boom is not sufficiently extended such that a small avoiding motion is needed to prevent the shovel from colliding with a wheel (as depicted in Fig. 5). Here, we test the usefulness in a practical situation, when slight mistakes need to be corrected.

Throughout the experiments, the methods' parameters remain the same (as given in Table 1), and this also holds for the nominal controller. It tracks a shovel position reference trajectory, and it composes \mathbf{f}_o from a PD-feedback term and a feedforward term encompassing inverse dynamics.

4.2 Results and Discussion

The results of the three experiments are presented in Figs. 3, 4 and 5, respectively. Consistently, the top right plot shows for a selected joint the nominal and modified actuator forces/torques (which are related to the joint forces/torques by a

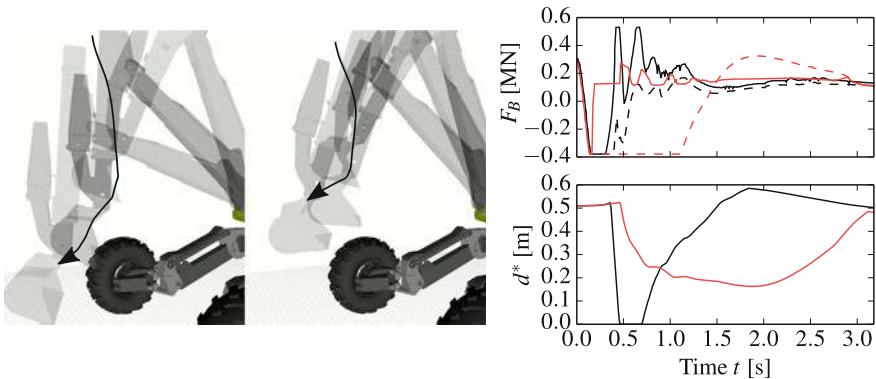


Fig. 3 Experiment 1: The left and the middle image show simulation snapshots for AP (classic method) and AP+ (our method), respectively, where the curved arrows mark the path of the shovel joint. The plots show the nominal (--) and the actual (—) boom joint force $F_B(t)$ (top) and the globally shortest distance $d^*(t)$ (bottom) for both AP (black) and AP+ (red)

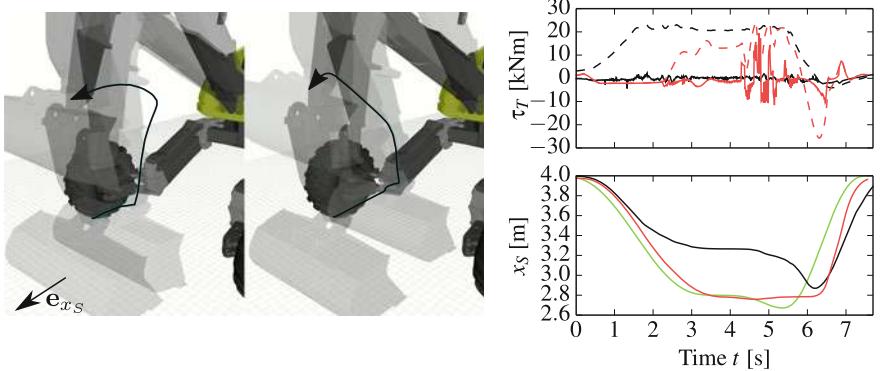


Fig. 4 Experiment 2: The left and the middle image show simulation snapshots for AP (classic method) and AP+ (our method), respectively, where the curved arrows mark the path of the shovel joint. The plots show the nominal (--) and the actual (—) turn joint torque $\tau_T(t)$ (top) and the shovel extension $x_s(t)$ (bottom) for both AP (black) and AP+ (red). The bottom plot also shows the reference for the nominal controller (green). The left image also shows \mathbf{e}_{x_S} as the direction of increasing x_S

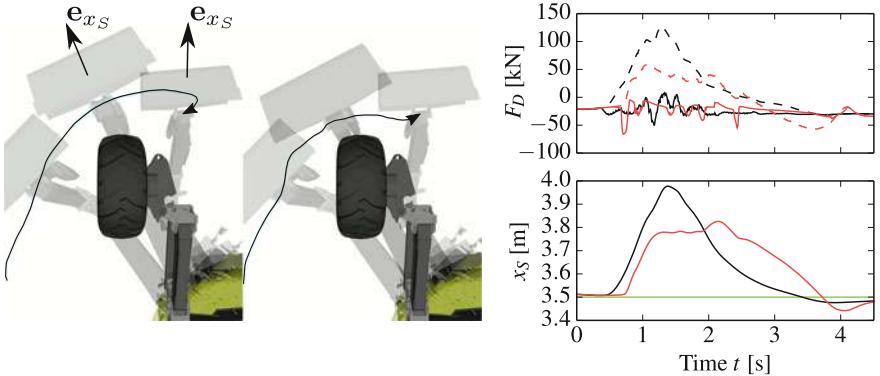


Fig. 5 Experiment 3: The left and the middle image show simulation snapshots for AP (classic method) and AP+ (our method), respectively, where the curved arrows mark the path of the shovel joint. The plots show the nominal (--) and the actual (—) dipper joint force $F_D(t)$ (top) and the shovel extension $x_s(t)$ (bottom) for both AP (black) and AP+ (red). The bottom plot also shows the reference for the nominal controller (green). In left image, the \mathbf{e}_{x_S} indicate the local directions of increasing x_S

configuration-dependent scaling factor). See Fig. 1 for the names given to the excavator's joints.

The upper plot in Fig. 3 shows that in the first experiment, AP+ starts pulling back the shovel earlier and thereby avoids the collision, whereas AP fails to do so, as shown by the shortest distance $d^*(t)$ reaching zero in the bottom plot. The collision for AP is also visible in the respective simulation snapshots, in contrast to the smooth deflection for AP+ (also shown in Fig. 3).

Table 1 Parameters for both methods AP and AP+

	$d_0, d_{\text{start}}, d_{\text{end}}$ (m)	α	κ (kNs/m)	ε (m/s)	$E_{A,\text{trans}}, E_{A,\text{max}}$ (J)	a_{\max} (m/s 2)	f_{\max} (-)
AP	0.8, –, –	25 kN/m	10	–	–	–	–
AP+	0.8, 0.2, 0.144	10 kN	2.5	10	4800, 6800	5	50000

In Figs. 4 and 5, the bottom right plots show the shovel extension $x_S(t)$, which is the coordinate of the shovel’s center point with respect to an axis that lies within the boom’s plane of motion and points to where the excavator is facing (as illustrated in the left images). The plots also show the reference for x_S that is tracked by the nominal controller and one clearly sees that it cannot be followed with AP in the second experiment, at least not beyond a certain proximity to the passage (also shown by simulation snapshots on the left of Fig. 4). With AP+ on the other hand, entering the narrow space is possible (as shown by the middle image). The top plot in Fig. 4 shows for AP+ oscillations of τ_T , which occur within the narrow passage and are caused by the switching direction of repulsion as the shovel is closer to either one or the other leg.

In the third experiment, the avoiding motion is larger with AP than with AP+, as the higher peak in the bottom plot in Fig. 5 shows. On the other hand, it shows a “sticky” behavior of AP+ near to the obstacle, as towards the end the reference is not reached for awhile. This is explained by the limiting, which is applied to \mathbf{f}_o if E_A is high (according to Sect. 3.5).

In summary, AP+ outperforms AP regarding both safety and versatility or workspace accessibility as the first and second experiment demonstrate. In the third experiment, both AP and AP+ perform well in the sense that they can successfully deflect the dangerous motion allowing to resume tracking of the original trajectory. However, we argue that from a safety point of view AP+ appears superior since a slower behavior close to collisions is preferable over large unexpected deflections.

5 Conclusion

We presented a novel collision avoidance method (Sect. 3) and compared it to a standard approach based on artificial potentials (Sect. 2.3) in simulated experiments with a walking excavator (Sect. 4). The experiments demonstrate that our method is superior due to its intrinsic safety and the significantly smaller restriction it imposes on the robot’s accessible workspace. Also, our method performs well in transforming slightly colliding (“scratching”) motions into harmless ones, allowing to quickly resume the original task.

Primarily for conceptual simplicity, we used a potential that only depends on the shortest distance, which however may cause oscillations in enclosures. As the

class of suitable potential functions is very broad, the best choice remains yet to be determined.

On the other hand, our method is very general as it is applicable to a very generic type of torque- or force-controlled robots. Further, due to its modular design, it is easy to integrate our method into existing control systems. In future work, we consider it worthwhile to explore e.g. the insertion of a second control module *before* the nominal controller. This would provide another means apart from the transitional blending we use to achieve a good interplay of collision avoidance with nominal control.

Finally, we have validated our approach as a way to overcome the limitations of standard reactive methods based on artificial potentials. Most importantly, provided that the required finite forces or torques are feasible, our method guarantees collision avoidance for any nominal control signal.

Acknowledgements This work was supported by the National Centre of Competence in Research Digital Fabrication.

References

1. De Santis, A., Albu-Schaffer, A., Ott, C., Siciliano, B., Hirzinger, G.: The skeleton algorithm for self-collision avoidance of a humanoid manipulator. In: 2007 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pp. 1–6. IEEE (2007)
2. Dietrich, A., Wimbock, T., Albu-Schaffer, A., Hirzinger, G.: Integration of reactive, torque-based self-collision avoidance into a task hierarchy. *IEEE Trans. Robot.* **28**(6), 1278–1293 (2012)
3. Goldstein, H., Poole, C., Safko, J.: Classical Mechanics, 3rd edn. Pearson Higher Ed (2002)
4. Hutter, M., Leemann, P., Stevscic, S., Michel, A., Jud, D., Hoepflinger, M., Siegwart, R., Figi, R., Caduff, C., Loher, M., et al.: Towards optimal force distribution for walking excavators. In: 2015 international conference on advanced robotics (ICAR), pp. 295–301. IEEE (2015)
5. Kalmari, J., Pihlajamäki, T., Hyyti, H., Luomaranta, M., Visala, A.: Iso 11783 compliant forest crane as a platform for automatic control. *IFAC Proc. Vol.* **46**(18), 164–169 (2013)
6. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. In: Autonomous Robot Vehicles, pp. 396–404. Springer (1986)
7. Koren, Y., Borenstein, J.: Potential field methods and their inherent limitations for mobile robot navigation. In: 1991 IEEE International Conference on Robotics and Automation, 1991. Proceedings, pp. 1398–1404. IEEE (1991)
8. Mansard, N., Chaumette, F.: Task sequencing for high-level sensor-based control. *IEEE Trans. Robot.* **23**(1), 60–72 (2007)
9. Rimon, E., Koditschek, D.E.: Exact robot navigation using artificial potential functions. *IEEE Trans. Robot. Autom.* **8**(5), 501–518 (1992)
10. Siciliano, B., Sciacicco, L., Villani, L., Oriolo, G.: Robotics: Modelling, Planning and Control. Springer Science & Business Media (2010)
11. Stasse, O., Escande, A., Mansard, N., Miossec, S., Evrard, P., Kheddar, A.: Real-time (self)-collision avoidance task on a hrp-2 humanoid robot. In: IEEE International Conference on Robotics and Automation, 2008. ICRA 2008, pp. 3200–3205. IEEE (2008)

12. Sugiura, H., Gienger, M., Janssen, H., Goerick, C.: Real-time collision avoidance with whole body motion control for humanoid robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007, pp. 2053–2058. IEEE (2007)
13. Zolynski, G., Schmidt, D., Berns, K.: Safety for an autonomous bucket excavator during typical landscaping tasks. In: New Trends in Medical and Service Robots, pp. 357–368. Springer (2014)

Towards Controlling Bucket Fill Factor in Robotic Excavation by Learning Admittance Control Setpoints

Heshan A. Fernando, Joshua A. Marshall, Håkan Almqvist
and Johan Larsson

Abstract This paper investigates the extension of an admittance control scheme toward learning and adaptation of its setpoints to achieve controllable bucket fill factor for robotic excavation of fragmented rock. A previously developed Dig Admittance Controller (DAC) is deployed on a 14-tonne capacity robotic load-haul-dump (LHD) machine, and full-scale excavation experiments are conducted with a rock pile at an underground mine to determine how varying DAC setpoints affect bucket fill factor. Results show that increasing the throttle setpoint increases the bucket fill factor and increasing the bucket's reference velocity setpoint decreases the bucket fill factor. Further, the bucket fill factor is consistent for different setpoint values. Based on these findings, a learning framework is postulated to learn DAC setpoint values for a desired bucket fill factor over successive excavation iterations. Practical implementation problems such as bucket stall and wheel-slip are also addressed, and improvements to the DAC design are suggested to mitigate these problems.

1 Introduction

In underground mining, articulated wheel loaders known as load-haul-dump (LHD) machines are utilized to excavate blasted rock from *muck piles* (i.e., piles of blasted rock, sometimes mixed with water and fine particles) at draw-points and haul this

H. A. Fernando (✉) · J. A. Marshall
Mining Systems Laboratory, Queen's University, 25 Union St,
Kingston, ON K7L 3N6, Canada
e-mail: heshan.fernando@queensu.ca

J. A. Marshall
e-mail: joshua_marshall@queensu.ca

H. Almqvist · J. Larsson
Rocktec Automation Division, Atlas Copco Rock Drills AB,
Klerkgatan 21, 701 91 Örebro, Sweden
e-mail: hakan.almqvist@se.atlascopco.com

J. Larsson
e-mail: johan.larsson@se.atlascopco.com

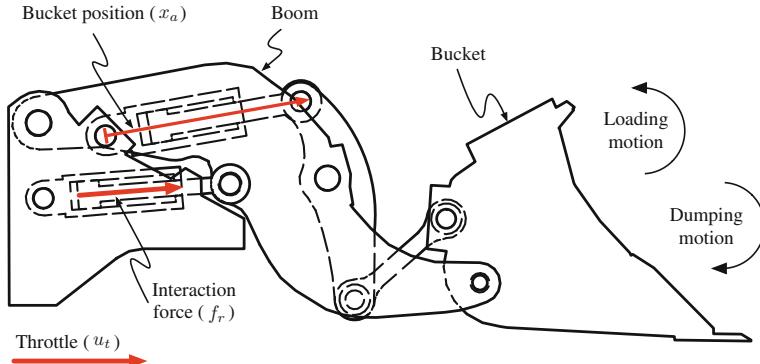


Fig. 1 A boom and bucket manipulator commonly found on mobile wheel loaders. The boom link and bucket end-effector are actuated by hydraulic cylinders. The Dig Admittance Controller (DAC) requires measurement of the bucket cylinder position (x_a) and measurement of the interaction force (f_r) at the boom cylinders

material to dump locations. Automating LHDs can improve safety and productivity of underground mining operations by removing operators from these hazardous and repetitive tasks. State-of-the-art technologies now offer autonomous tramping and dumping for LHDs, but commercially viable technologies for robotic excavation (i.e., loading) of fragmented rock are still in early stages of development.

The main challenge in developing a successful robotic system for excavating fragmented rock is designing a controller that can effectively regulate the motion of an excavator bucket (e.g., Fig. 1) to dig through a rock pile and *fill* the bucket with material to a desired level at each excavation iteration (i.e., does not overfill or under-fill). For excavating homogeneous media such as a sand and gravel piles, it is possible to use stereo-vision cameras or laser scanners to first scan the pile profile, plan a path trajectory to excavate a volume of material equivalent to the excavator's bucket, and apply robot motion control strategies to track the path with the bucket [8]. However, pure motion control strategies are ineffective when excavating fragmented rock because subsurface obstacles can cause large position errors that result in saturated actuation [2, 12]. Further, cameras and laser scanners are not practical to implement in dark and dusty underground mining environments.

We believe that the path to successful excavation of fragmented rock is through “feel”, rather than through “sight”. Specifically, we believe that feedback of measured bucket-rock interaction forces to an admittance control scheme is effective for regulating an excavator's bucket motion to dig through a rock pile and fill the bucket with material [13]. Further, we hypothesize that the bucket fill factor (or payload weight) can be controlled by varying the setpoint parameters of an admittance control scheme for robotic excavation of fragmented rock. In this paper, we present results of full-scale experiments with a 14-tonne capacity LHD machine and an underground muck pile that support our hypothesis.

1.1 Related Work

Admittance control for robotic excavation of fragmented rock was initially postulated by Marshall et al. [13] based on results of full-scale excavation experiments with expert operators and an instrumented wheel loader. Analysis of the force and motion data at the loader's bucket hydraulic cylinder showed that during excavation, operators controlled the bucket motion in an attempt to maintain a level of interaction force—though the operators were not conscious of this underlying fact. This led Marshall et al. [13] to postulate an admittance control scheme for robotic excavation of fragmented rock that regulates bucket motion based on feedback of interaction forces. Compared to intelligence-based approaches to robotic excavation [3] that require substantial training data and exhibit unpredictable behaviour in untrained situations, the admittance control scheme proposed by Marshall et al. [13] provides an effective framework for use in practical implementations.

Recent research work [5, 6] has developed a Dig Admittance Controller (DAC) for robotic excavation based on the admittance control scheme proposed by Marshall et al. [13]. The DAC was tested in full-scale robotic excavation experiments, which showed that a 14-tonne capacity LHD that was robotically controlled using the DAC excavated larger payloads of fragmented rock, more consistently, compared to an expert operator excavating the same rock pile using the same machine [5]. However, a challenge in implementing the DAC was selecting the proper control parameters for a desired bucket fill factor; parameter tuning was unintuitive due to lack of an explicit model for bucket-rock interaction. Manually tuning the DAC parameters worked well for a rock pile with particular characteristics (e.g., bulk density, fragmentation size and cohesion), but the control parameters required re-tuning to excavate a desired bucket fill factor from a pile with different characteristics.

In practice, pile characteristics are difficult to model explicitly, and these characteristics can evolve throughout the life of an excavation operation. Thus, for practical implementation in robotic excavation, the DAC requires extension by learning and adaptation so that optimal parameters for a desired bucket fill factor are automatically learned and adapted at each excavation iteration.

Research in learning-based interaction control has developed in recent years due to increased interest in emerging field and service robot applications such as human-robot collaboration [7] and robot-assisted surgery [9]. A fundamental requirement for robots in these applications is to maintain desired performance of a particular task while physically interacting with unstructured and evolving environments. Due to their robustness and stability in compliant manipulation tasks, interaction controllers based on the impedance and admittance control paradigms [14] are often chosen for extension by learning and adaptation in these emerging applications.

Much of the research for learning-based interaction control has focused on extension of impedance controllers toward adaptive controllers through learning of impedance parameters or desired trajectories [7]. Many of these “variable impedance” control approaches are based on iterative learning strategies where the objective is to learn the optimal impedance/admittance parameters through successive

interaction with the environment in repetitive tasks. Applications have mainly focused on human-robot collaboration [1, 4, 11, 15, 16]. Typically, a cost function is defined using interaction force and/or trajectory tracking error. Impedance parameters, such as a virtual damping coefficient, are then updated over a trajectory (i.e., gain scheduling) by minimizing the cost function. Stability of these methods have been analyzed by Kronander and Billard [10], and it was found that varying impedance control parameters can lead to instability. Impedance control with constant gains are typically stable when interacting with passive environments, but instability can occur when the gains are varying throughout the task.

1.2 About This Paper

In this paper, through full-scale field experiments, we investigate learning and adaptation of Dig Admittance Control (DAC) setpoints in order to achieve controllable bucket fill factor in robotic excavation of unstructured and evolving media, such as fragmented rock. First, the DAC scheme is presented in Sect. 2. Details of the LHD, muck pile and experiment procedure are provided in Sect. 3. Experiment results for varying DAC setpoints on bucket fill factor are presented and analyzed in Sect. 4. These results are further discussed in Sect. 5 in order to suggest improvements, and to postulate a learning framework for adapting the DAC setpoints. Finally, future directions and upcoming experimental work for continuation of this research are presented in Sect. 6.

2 Approach to Robotic Excavation: The Dig Admittance Controller

A block diagram of the Dig Admittance Controller (DAC), which was postulated by Marshall et al. [13] and tested by Dobson et al. [6] for robotic excavation of fragmented rock using wheel loaders, is shown in Fig. 2. In this scheme, the admittance controller $Y(s)$ attempts to comply the wheel loader's bucket motion with the rock pile interaction and react to measured interaction forces by modifying the reference motion trajectory x_c to a position-controlled bucket actuator (i.e., hydraulic cylinder). Thus, in a discrete-time implementation with a constant period T and a proportional admittance $Y(s) = K_A$ (as used by Dobson et al. [6]), the reference motion command to the bucket hydraulic cylinder at a time-step k becomes

$$x_{c,k} = x_{c,k-1} + T(v_{f,k} + v_d), \quad (1)$$

where v_d , is a constant nominal velocity setpoint for the bucket curl rate, and

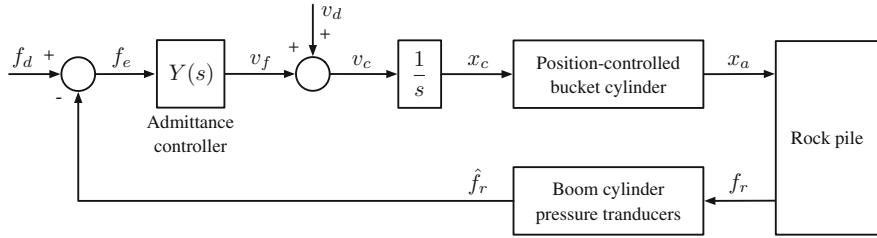


Fig. 2 A block diagram of the Dig Admittance Controller (DAC) for robotic excavation of fragmented rock postulated by Marshall et al. [13] and tested by Dobson et al. [6] in full-scale experiments

$$v_{f,k} = K_A(f_d - \hat{f}_{r,k}) \quad (2)$$

is a computed velocity change that is proportional to the error between a constant target force setpoint f_d and the measured interaction force $\hat{f}_{r,k}$. The interaction force $\hat{f}_{r,k}$ is calculated from pressure measurements at the boom cylinders as the boom is not actuated during excavation. Thus, the boom pressures measure only interaction forces and are not contaminated by other forces due to input signals. A constant forward throttle percent u_t is applied to the loader during excavation in order for the bucket to maintain contact with the pile.

The challenge in implementing the DAC in practice is determining how to best select control setpoints f_d and v_d , and throttle setpoint u_t to achieve desired bucket fill factor (typically measured as payload in tonnes) at each excavation iteration. The values of these parameters are dependent on the physical characteristics of the excavation media (e.g., bulk density, fragmentation and cohesion), which are difficult to model prior to excavation and typically evolve throughout the life of the excavation operation.

In their experiments, Dobson et al. [6] manually tuned the setpoints until a desired bucket fill factor was achieved. Once tuned, the DAC performed well in excavating consistent payloads from a particular rock pile, but the parameters required retuning when the pile characteristics changed. In our investigation, we are interested in determining the effects of varying the DAC setpoints on the bucket fill factor. If trends can be observed in the fill factor for varying setpoint values, then a learning framework can be developed to automatically learn and adapt the parameters for a desired fill factor over successive excavation iterations.

3 Experiment Apparatus and Methodology

Full-scale robotic excavation experiments described in this paper were conducted using an automation-ready Atlas Copco ST14 load-haul-dump (LHD) machine and a muck pile located at an underground mining test-facility at Kvarntorp, Sweden. Figure 3 shows images of the ST14 and muck pile.



Fig. 3 Images of the ST14 LHD machine and muck pile used for experiments described in this paper. A 1 m long rod is placed on a sample of rocks as a length reference

3.1 ST14 LHD Specifications

The Atlas Copco ST14 LHD machine is an articulated, low-profile, wheel loader that is designed for loading, hauling and dumping blasted rock in underground mining operations. This LHD has a 14-tonne payload capacity and a loading mechanism with a boom and bucket linkage as shown in Fig. 1. Two parallel hydraulic cylinders actuate the boom, and a single hydraulic cylinder, connected via a Z-bar linkage, actuates the bucket. As depicted in Fig. 1, bucket cylinder extension corresponds to bucket *curling* motion and bucket cylinder retraction corresponds to bucket *dumping* motion. The ST14 transmission is powered by a 335 HP diesel engine, and the hydraulic actuators (i.e., for boom, bucket and steering) are powered by two load-sensing variable displacement pumps. The hydraulic system is designed in a closed-centre configuration, so all hydraulic actuators receive maximum power from the variable displacement pumps. Electrohydraulic servo valves control flow to the hydraulic cylinders for actuation.

3.2 Sensors and Data Acquisition

Automation-ready ST14 LHD machines are equipped with programmable control systems and sensors that provide information about the machine's status. These sensors include pressure transducers that measure the boom cylinders' base and rod-side hydraulic pressures, a linear position transducer that measures the bucket cylinder extension length, an absolute encoder that measures the boom angle, and a cogwheel that measures the transmission speed. Thus, no additional hardware was required to implement the admittance controller on the ST14.

The ST14 control system operates in real-time at a frequency of 20 Hz for reading sensor measurements, sending actuator commands and logging data. This control frequency is adequate for implementing the admittance controller because bucket-rock interaction dynamics typically have very low bandwidth [12]. To conduct the

robotic excavation experiments, the admittance control scheme described in Fig. 2 was programmed onto the ST14's control system. The interaction force \hat{f}_r was calculated from hydraulic pressure measurements at the base and rod sides of the boom cylinders as $\hat{f}_r = P_1 A_1 - P_2 A_2$ where P_1 and A_1 are the boom cylinders' base pressure and cross-sectional area, respectively, and P_2 and A_2 are the boom cylinders' rod-side pressure and cross-sectional area, respectively.

Actuator commands to the bucket cylinder's servo valve are sent as digital signals that correspond to the valve's spool displacement. These command signals are normalized between -1 and $+1$, where negative values correspond to cylinder retraction and positive commands correspond to cylinder extension. Valve dead-bands are reduced in the control software so that the valve commands correspond to a near-linear response from the cylinder.

Payload weight W is calculated using a static load analysis with the boom raised to a specified angle. Boom cylinder pressure measurements are used to obtain the forces required for the analysis. We were unable to rigorously test the accuracy of the load weighing system due to lack of a calibration weight, but the system's precision was determined by raising and lowering the boom five times with a filled bucket. The largest absolute deviation (i.e., the largest difference between a measurement and the sample mean) was determined to be 0.2 tonnes.

3.3 Excavation Media

A muck pile, shown in Fig. 3, was used for all robotic excavation trials presented in this paper. The back of the pile was confined to a wall in a test area of an underground mine at Kvarntorp, Sweden. This muck pile consisted of a mixture of clay, fine gravel and large fragments of blasted rock (30–70 cm in nominal diameter), which is representative of a real muck pile found in underground mining. However, real muck piles are typically larger and more confined with a continuous flow of material from a blasted *stope* above the muck pile.

3.4 Robotic Excavation Trials Procedure

Robotic excavation trials followed a procedure of manually positioning the loader in front of the pile, executing the excavation algorithm, manually backing out of the pile, weighing the excavated payload and dumping the excavated material back on to the pile. At the start of each excavation trial, the admittance controller's throttle parameters u_t and reference velocity setpoint v_d were set in the ST14 control system, and the excavation algorithm was executed to automate the following steps:

1. Use position controllers to move boom and bucket to entry positions.
2. Apply throttle u_t to drive the LHD forward into the pile and penetrate the pile with the bucket.
3. When measured force \hat{f}_r exceeds an entry force threshold, f_{entry} , execute the admittance controller to dig through the pile and fill the bucket.
4. When bucket is fully curled, deactivate the admittance controller and return throttle to zero.

The lack of overhead confinement on our muck pile resulted in material build-up at the tip of the bucket at the end of excavation. Thus, at the end of each dig, an operator manually shook the bucket to allow this material to fall back into the bucket prior to load weighing.

4 Experiments and Analysis

Robotic excavation experiments were conducted to determine the effects of changing the Dig Admittance Controller's (DAC) target force setpoint f_d , throttle setpoint u_t and reference velocity setpoint v_d on excavated payloads W . Results and analysis from these experiments are presented in the following subsections.

4.1 Preliminary Experiments

In preliminary robotic excavation experiments that varied the constant target force setpoint f_d , it was determined that setting a low f_d can lead to bucket stall during excavation. Measured force and motion data from a stalled trial is shown in Fig. 4. It can be seen that as the bucket curls, the measured interaction force \hat{f}_r continues to increase. Eventually, \hat{f}_r increases beyond the target force f_d , causing the velocity change v_f to negate the reference velocity v_d to the bucket. This effectively stalls bucket motion and prevents the LHD from moving into the pile.

4.1.1 Target Force Adaptation

To prevent bucket stall situations from degrading the performance of a setpoint learning framework, we implemented an averaging filter to adapt the target force setpoint f_d to be a moving average of the measured interaction force

$$\bar{f}_{d,k} = \bar{f}_{d,k-1} \frac{n-1}{n} + \frac{\hat{f}_{r,k}}{n}, \quad (3)$$

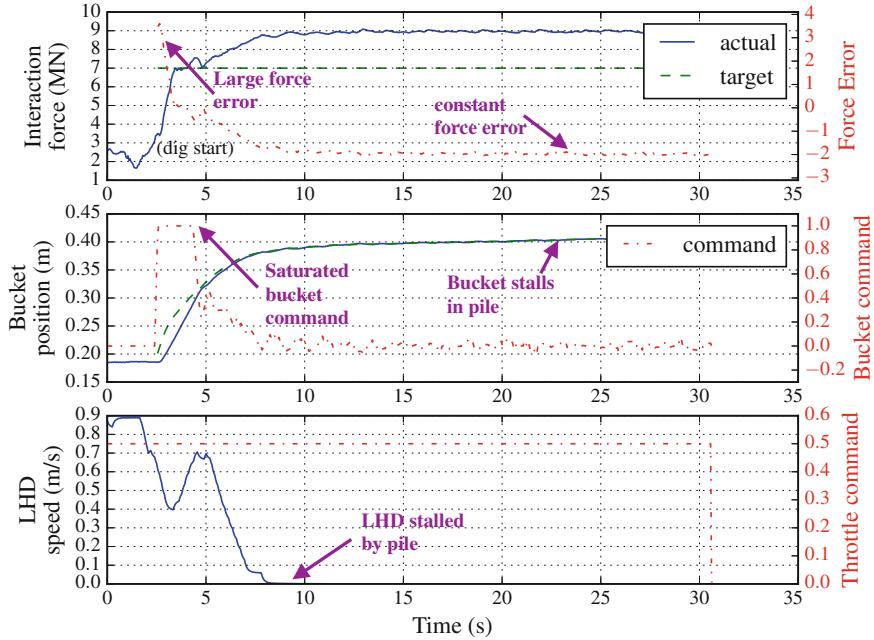


Fig. 4 Force and motion data from excavation trials with a constant target force setpoint $f_d = 7.0$ MN, reference velocity setpoint $v_d = 0.040$ m/s and throttle setpoint $u_t = 50\%$. Setting a low, constant, f_d can result in bucket stall during excavation

where $\bar{f}_{d,k}$ is the new average target force setpoint at the current timestep k , $\bar{f}_{d,k-1}$ is the average target setpoint from the previous timestep, n is the averaging window size, and $\hat{f}_{r,k}$ is the measured interaction force at the current time step. This averaging algorithm is computationally more efficient than calculating the true moving average.

The target force adaptation scheme proposed above is based on analysis of data from 25 robotic excavation trials with different, constant, f_d values. Analysis of the force signals from these consistently showed that the measured interaction forces always increase as the bucket curls, but the magnitude and profile of the force signal is dependent on many factors (e.g., material accumulated in the bucket, contact with the pile, throttle, etc.), so it is difficult to predetermine a suitable profile for the target force setpoint at each excavation iteration.

Ideally, the target force should follow the nominal level of interaction forces, so that changes in the interaction forces beyond this level correspond to appropriate velocity changes v_f by the admittance controller. We believe that the moving average technique achieves this, although a lag is present due to averaging. Preliminary testing found that $n = 20$ provides a suitable averaging window size.

4.2 Robotic Excavation Trials for Varying u_t and v_d

The excavation experiments were conducted by deploying the DAC, described in Sect. 2 (with the target force setpoint adaptation modification described in Sect. 4.1.1), on the ST14 LHD machine, and following the excavation procedure outlined in Sect. 3.4. The experiment variables are the DAC throttle setpoint u_t and reference velocity setpoint v_d . All other control parameters: admittance controller gain K_A , pre-entry boom and bucket positions, entry force threshold f_{entry} , and target force setpoint moving average window size n were manually tuned at the start and kept constant throughout the trials.

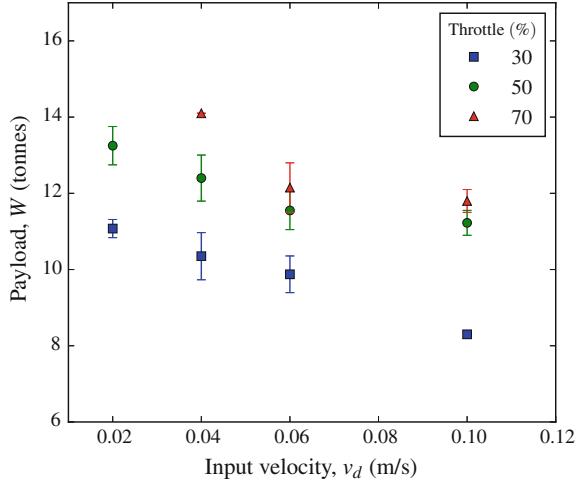
The uncontrollable variables in these trials are pile shape and pile consistency, which introduce some variation in the payload results. Because the muck pile used in these experiments did not have a continuous flow of material, the excavated payloads were dumped back onto the pile after each excavation trial. The pile shape was adjusted twice during the trials as a lack of confinement at the sides caused the pile to push out and decrease in depth. We are not concerned with pile changes between trials in this experiment as the DAC should be robust to pile variations.

The experiment matrix for the excavation trials is given in Table 1. Combinations of four different v_d values and three different u_t values were tested. A v_d value of 0.100 m/s was tested because this results in saturation of the bucket command during digging; these trials provide reference payloads for simply curling the bucket at maximum speed after pile penetration. The initial plan was to conduct four trials for each parameter combination; however, significant wheel slip at 70% throttle setpoint created deep trenches in roadway at the tip of the muck pile. Wheel slip can also cause the LHD's tires to heat and possibly burst, so we did not pursue many excavation trials at high throttle values. In the end, we were able to conduct five excavation trials at $u_t = 70\%$, which gave 37 successful trials for results and analysis. All trials were conducted in a random order to avoid pile variations from biasing payload results.

Table 1 Experiment matrix showing the number of excavation trials completed for different combinations of throttle setpoint u_t and reference velocity setpoint v_d in the DAC

v_d (m/s)	u_t (%)		
	30	50	70
0.020	4	4	0
0.040	4	4	1
0.060	4	4	2
0.100	4	4	2

Fig. 5 Payload results from robotic excavation trials for varying throttle parameter u_t and reference velocity v_d in the Dig Admittance Controller. Data points show the average values from all trials for each parameter combination (see Table 1 for trial information); error bars indicate the standard deviations



4.2.1 Results and Analysis

The DAC, modified with target force setpoint adaptation, worked well in our robotic excavation experiments—there were no bucket stall problems. Thus, the ST14 LHD was able to autonomously excavate the muck pile and load the bucket in all excavation trials. Trials with $u_t = 70\%$ were deactivated when significant wheel slip was observed; however, we expect that these trials would also have had successful excavation results if wheel slip had not occurred. The issue of wheel slip is addressed after presenting the payload results for the successful excavation trials below.

Figure 5 shows plots of average payloads from robotic excavation trials with different DAC setpoints, u_t and v_d . This plot shows that increasing u_t increases payload, and increasing v_d decreases payload. Despite pile variations, the excavated payloads for each parameter combination are consistent. These results affirm our hypothesis that controllable bucket fill factor can be achieved by controlling the admittance of the bucket rock interaction. For the trial sets with $u_t = 30\%$ and $u_t = 50\%$, the maximum standard deviation for payloads was 0.6 tonnes (recall that the load weighing system's precision is 0.2 tonnes). Thus, the minimum controllability of bucket payloads that we would be able to achieve with this robotic excavation system is approximately 0.6 tonnes.

Wheel slip was observed for trials with low v_d and high u_t . Based on observations during the excavation trials, we believe that this is due to slow bucket curl rate at the start of the dig. Slow bucket curl rate suggests low admittance in the control scheme, which means that the controller is not reacting quickly enough to increasing forces. Figure 6 shows measured force and motion data from a trial with $u_t = 50\%$ and $v_d = 0.020$ m/s. The force data shows that the moving average force target lags behind the measured force when forces are increasing. Because the force error is always negative, the velocity change v_f acts to reduce the reference velocity v_c ,

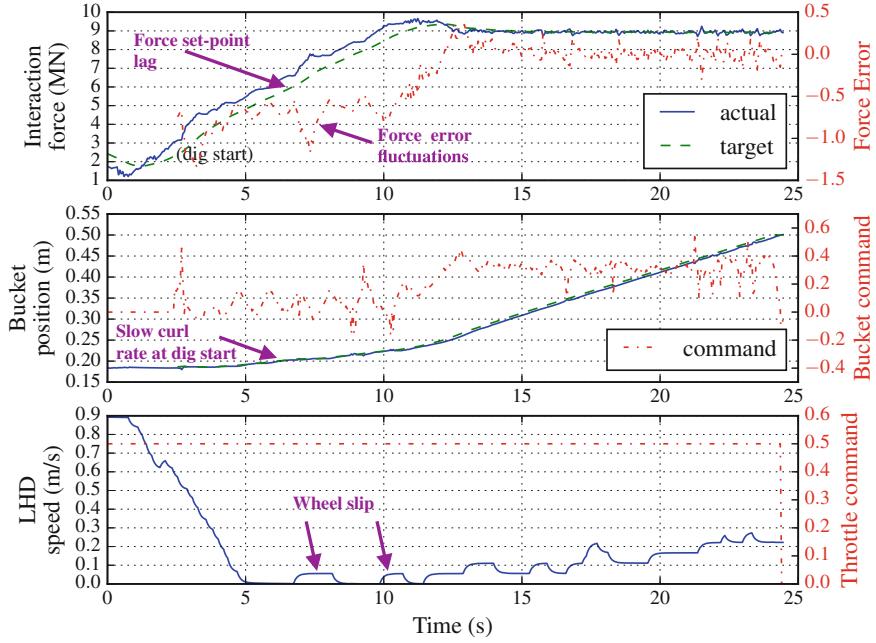


Fig. 6 Force and motion data from a robotic excavation trial using the Dig Admittance Controller modified with the moving average target force setpoint (\bar{f}_d). This data is from a trial with throttle setpoint $u_t = 50\%$ and reference velocity setpoint $v_d = 0.020 \text{ m/s}$

resulting in slow curl rate at the start of the dig cycle. Slow bucket curl rate causes the LHD to become stalled by the pile, which results in small amounts of wheel slip. Analysis of the force error signal indicates sharp increases in forces at the start of the dig, which should correspond to increases in bucket curl rate to admit the LHD into the pile. We believe that introducing a derivative control action to the admittance controller could address this issue.

5 Setpoint Learning for Controllable Bucket Fill: Discussion

Results from robotic excavation trials suggest that controlling the admittance of the excavation process by changing the throttle setpoint u_t and reference velocity setpoint v_d allow controllable bucket fill. Based on these results and subsequent analysis of data, the following discussion proposes improvements to the admittance controller design and postulates a learning framework for adapting these two setpoints over excavation iterations.

The moving average target force setpoint \bar{f}_d modification to the Dig Admittance Controller (DAC) worked well in eliminating bucket stall situations during excavation; however, lag in target-force adaptation results in slow bucket curl at the dig start, which leads to wheel slip. Increasing the admittance gain K_A would increase the response of the velocity change v_f resulting in oscillatory behaviour for the bucket, which is not ideal. Analysis of the force error signals from excavation trials suggest that adding a derivative control action to the admittance controller as $Y(s) = K_A f_e + B_A \dot{f}_e$ might improve the reaction of the velocity change for greater rates of change in the interaction forces. Consideration will need to be given to filter the significant amount of noise present in the force measurement signal prior to calculating its derivative.

There remains the question of how to best learn u_t and v_d to control bucket fill-factor. Both setpoints contribute to bucket payload, so it might be possible to learn both by using a simple gradient following algorithm at each excavation iteration, j :

$$u_{t,j+1} = u_{t,j} + \alpha(W_d - W_j) \quad (4)$$

$$v_{d,j+1} = v_{d,j} - \beta(W_d - W_j), \quad (5)$$

where W_d is the desired bucket fill factor (or payload) and $\alpha, \beta > 0$ are learning rates that must be tuned. Updating both setpoints at each excavation iteration may cause oscillations in the learning, so it might be better to give a fixed value to one setpoint and learn the remaining setpoint as per above learning laws. More excavation experiments are required to test these hypotheses.

6 Conclusions and Future Work

Results of full-scale robotic excavation experiments with a 14-tonne capacity load-haul-dump vehicle and an underground muck pile show that varying the throttle and reference velocity setpoints of a previously developed Dig Admittance Controller (DAC) for robotic excavation of fragmented rock allows controllable bucket fill factor (i.e., payload). Based on these results, we postulate a learning algorithm that learns the DAC throttle and velocity setpoints based on a performance metric of payload error at each excavation iteration. Further, we suggest improvements to the DAC for adapting the target force set-point and introducing derivative action in the control law for better reaction to large increases in measured interaction forces. Future work proposes to test the proposed learning scheme with full-scale excavation experiments on different types of excavation media such as a gravel pile and a fragmented rock pile.

Acknowledgements This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under project RGPIN-2015-04025, the Swedish Knowledge Foundation (KK-stiftelsen) under project 20150282, and by Atlas Copco Rock Drills AB (Sweden).

This work was completed while the second author was a Visiting Professor at the Centre for Applied Autonomous Sensor Systems (AASS) in the School of Science and Technology at Örebro University, Sweden.

References

1. Buchli, J., Stulp, F., Theodorou, E., Schaal, S.: Learning variable impedance control. *Int. J. Robot. Res.* **30**(7), 820–833 (2011)
2. Dadhich, S., Bodin, U., Andersson, U.: Key challenges in automation of earth-moving machines. *Autom. Constr.* **68**, 212–222 (2016)
3. Dadhich, S., Bodin, U., Sandin, F., Andersson, U.: Machine learning approach to automatic bucket fill. In: Proceedings of 24th Mediterranean Conference on Control and Automation, Athens, Greece, pp. 1260–1265 (2016)
4. Dimeas, F., Aspragathos, N.: Reinforcement learning of variable admittance control for human-robot co-manipulation. In: Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, pp. 1011–1016 (2015)
5. Dobson, A.A., Marshall, J.A., Larsson, J.: Admittance control for robotic loading: Underground field trials with an LHD. In: Proceedings of the 10th Conference on Field and Service Robotics (FSR), Toronto, Ontario, Canada (2015)
6. Dobson, A.A., Marshall, J.A., Larsson, J.: Admittance control for robotic loading: Design and experiments with a 1-tonne loader and a 14-tonne LHD. *Field Serv. Robot. J. Field Robot.* **34**(1), 123–150 (2017)
7. Haddadin, S., Croft, E.: Springer Handbook of Robotics, 2nd edn. Chap 69—Physical Human-Robot Interaction, pp. 1835–1874. Springer International Publishing, Switzerland (2016)
8. Hemami, A., Hassani, F.: An overview of autonomous loading of bulk material. In: Proceedings of the 26th International Symposium on Automation Robotics in Construction (ISARC), Austin, Texas, pp. 405–411 (2009)
9. Kim, Y.J., Seo, J., Kim, H., Kim, K.G.: Impedance and admittance control for respiratory motion compensation during robotic needle insertion—a preliminary test. *Int. J. Med. Robot. Comput. Assisted Surg.* 1–10 (2016)
10. Kronander, K., Billard, A.: Stability considerations for variable impedance control. *IEEE Trans. Robot.* **32**(5), 1298–1305 (2016)
11. Li, Y., Ge, S.: Impedance learning for robots interacting with unknown environments. *IEEE Trans. Control Syst. Technol.* **22**(4), 1422–1432 (2014)
12. Maeda, G.J., Manchester, I.R., Rye, D.C.: Combined ILC and disturbance observer for the rejection of near-repetitive disturbances, with application to excavation. *IEEE Trans. Control Syst. Technol.* **23**(5), 1754–1769 (2015)
13. Marshall, J.A., Murphy, P.F., Daneshmend, L.K.: Toward autonomous excavation of fragmented rock: full-scale experiments. *IEEE Trans. Autom. Sci. Eng.* **5**(3), 562–566 (2008)
14. Villani, L., De Schutter, L.: Springer Handbook of Robotics, 2nd edn. Chap 9—Force Control, pp. 195–220. Springer, Switzerland (2016)
15. Wang, C., Li, Y., Ge, S.S., Lee, T.H.: Reference adaptation for robots in physical interactions with unknown environments. *IEEE Trans. Cybern.* <https://doi.org/10.1109/TCYB.2016.2562698> (2016)
16. Yamawaki, T., Ishikawa, H., Yashima, M.: Iterative learning of variable impedance control for human-robot cooperation. In: Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, pp. 839–844 (2016)

Trajectory Optimization for Dynamic Grasping in Space Using Adhesive Grippers

Roshena MacPherson, Benjamin Hockman, Andrew Bylard,
Matthew A. Estrada, Mark R. Cutkosky and Marco Pavone

Abstract Spacecraft equipped with gecko-inspired dry adhesive grippers can dynamically grasp objects having a wide variety of featureless surfaces. In this paper we propose an optimization-based control strategy to exploit the dynamic robustness of such grippers for the task of grasping a free-floating, spinning object. First, we extend previous work characterizing the dynamic grasping capabilities of these grippers to the case where both object and spacecraft are free-floating and comparably sized. We then formulate the acquisition problem as a two-phase optimization problem, which is amenable to real time implementation and can handle constraints on velocity, control, as well as integer timing constraints for grasping a specific target location on the surface of a spinning object. Conservative analytical bounds for the set of initial states that guarantee feasible grasping solutions are derived. Finally, we validate this control architecture on the Stanford free-flyer test bed—a 2D microgravity test bed for emulating drift dynamics of spacecraft.

R. MacPherson (✉) · B. Hockman · M. A. Estrada · M. R. Cutkosky

Department of Mechanical Engineering, Stanford University,
Stanford, CA 94305, USA

e-mail: roshenam@stanford.edu

B. Hockman

e-mail: bhockman@stanford.edu

M. A. Estrada

e-mail: estrada1@stanford.edu

M. R. Cutkosky

e-mail: cutkosky@stanford.edu

A. Bylard · M. Pavone

Department of Aeronautics and Astronautics, Stanford University,
Stanford, CA 94305, USA

e-mail: bylard@stanford.edu

M. Pavone

e-mail: pavone@stanford.edu

1 Introduction

Recently, in an effort to alleviate some of the tasks performed by astronauts, there has been increased interest in the use of small assistive free-flying robots (AFF) for grasping and manipulating objects inside and outside spacecraft. One such example is the Smart SPHERES teleoperated test bed, which was developed to perform various intra-vehicular activities aboard the International Space Station (e.g., camera work and environmental monitoring), as well as to serve as a robotics research platform in microgravity [1]. Enabling AFFs to autonomously grasp and manipulate objects has the potential to make many human operations safer and more efficient by reducing time spent performing repetitive tasks and on EVAs (see Fig. 1). Autonomous object manipulation may also enable a wide range of new applications that are too dangerous, complex, or expensive for astronauts, such as the assembly of large-scale space structures or the removal of space debris [2].

Traditionally, most grasping devices, especially in space, have relied on robotic hands that either pinch opposing faces of the target (“force closure”) or grapple around its features to secure it (“caging grasp”). The precision of this operation typically requires that the target be stationary relative to the gripper for successful acquisition. For example, in [3, 4], the authors assume that target objects have a grappling fixture for caging [3] or pinching [4] and plan the spacecraft’s trajectory such that its end effector velocity matches that of the grappling feature. However, velocity matching often imposes a heavy burden on control precision and fuel expenditure.

Grippers that utilize *dry surface adhesion* represent a promising alternative. Inspired by the adhesive properties of geckos’ feet, several grippers have been developed using gecko-like materials that can adhere to any smooth, flat or curved surface

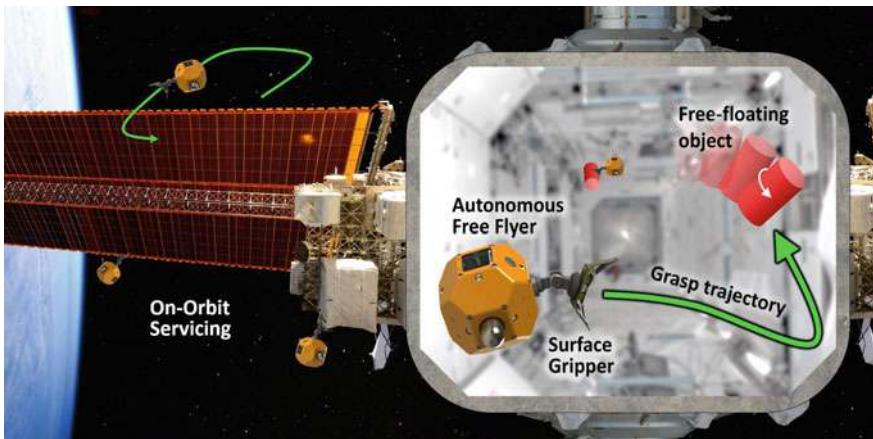


Fig. 1 Autonomous free flying spacecraft equipped with dry adhesion surface grippers may assist astronauts inside and outside the space station. This paper investigates optimal control strategies for autonomous perching and acquisition of free-floating, tumbling objects

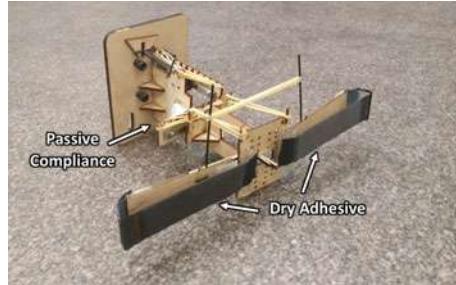


Fig. 2 A curved-surface gripper utilizes gecko-inspired adhesive materials to robustly grasp a variety of objects. Two opposing fingers passively collapse onto any curved smooth surface upon contact, by triggering a bistable mechanism. The gripper is mounted on a passive compliant wrist that allows it to absorb impact energy. See [8] for details

simply by touching them [5, 6]—thus, broadening the class of possible grasp locations from a small set of features to a larger (continuous) space of feature-*less* surfaces. Furthermore, when paired with a compliant wrist mechanism, these grippers can *dynamically* engage objects with high relative velocity—a key advantage for capturing drifting objects in space [7]. Previous work by the authors investigated the performance of one such gripper designed to grasp a translating and rotating object [6] (see Fig. 2). A passive cylindrical object, free-floating on frictionless air bearings, was thrown at a stationary gripper on Stanford’s planar microgravity test bed (see Fig. 4). The gripper, fixed to the inertial frame, was able to catch the object over a wide range of contact velocities. By systematically probing the dynamic limitations of the gripper in simulations and experiments, an envelope of contact states amenable for reliable grasping was empirically constructed—henceforth referred to as the “grasping envelope”. In this paper, we investigate how such dynamic surface grasping can be leveraged to develop robust control laws for grasping objects in space.

From a control standpoint, adhesive grippers eliminate the need to deliberately coordinate finger contact forces, allowing the precision grasping task to be simplified to a *rendezvous and docking problem*—a well-studied problem having a rich body of literature. Specifically, a variety of optimization-based approaches have been devised for the problem of spacecraft rendezvous and docking, including [9], which treats some of the constraints as soft penalties in the cost function. This allows the problem to be formulated as a Quadratic Program (QP), thus enabling real-time implementation. Similarly, [10] restricts each phase of the problem (long-range rendezvous, short-range docking, etc.) to be formulated as either a Linear Program (LP) or a QP for fast, online execution. In [11], the authors applied MPC to the rendezvous and docking of a spacecraft with a non-rotating platform in circular orbit around the Earth. They extended this work in [12] to the case of a rotating/tumbling object, imposing state constraints to avoid debris. In a similar vein, we propose an optimization-based approach to the related problem of dynamic grasping, consisting of a two-phase

optimal control architecture that is amenable to the complex dynamics and terminal constraints characterizing adhesive grippers, and integer timing constraints for grasping a specific location on a spinning body.

Specifically, the contribution of this paper is threefold. First, in Sect. 2, we extend our previous work in [6] on characterizing the grasping envelope of a curved surface gripper to the case where both spacecraft and object are free floating and in relative motion. Second, in Sect. 3 we formulate the problem of grasping spinning, featureless objects as a two-phase optimal control problem and derive conservative analytical bounds for the set of initial states that guarantee feasible grasping solutions. Finally, we validate the controller in simulation and through a variety of experiments on a custom free-floating spacecraft test bed (Sect. 4).

2 Grasping Envelope

In order to leverage the dynamic grasping capabilities of adhesive grippers for robust object acquisition, some model of the set of “graspable” contact states is required. This *grasping envelope* is a complex function of the gripper design, object shape and surface, and the highly nonlinear behavior of the dry adhesives. First order insights for defining this envelope can be derived from analytical models and simulations (as was done in [6]), but a more complete characterization relies on systematically probing the boundaries experimentally. In previous work, Estrada et al. [6] characterized the envelope of a gripper fixed to the inertial frame through a passive compliant wrist, which is akin to the case in which the target object is significantly less massive than the spacecraft. However, for small AFFs that often perch or grasp larger objects, this is often not the case. Accordingly, our first step is to extend those results to the more general case in which both object *and* spacecraft are floating and of comparable mass.

For planar motion, the contact state can be uniquely described by four parameters, namely, the offset of the contact from the center of the gripper (d), and the relative velocity, decomposed as the linear (v) and angular (Ω) speeds and angle of attack (ϕ) (see Fig. 3). Thus, the grasping envelop can be viewed as a closed set $(v, \Omega, \phi, d) \in \mathbb{R}^4$ centered at $(\mu_v, 0, 0, 0)$ and symmetric about $d = 0$.

For imposing terminal velocity constraints in the grasping problem, it is most important to characterize the relationship between speed and angle of attack, which can then be translated into normal and lateral velocity constraints. In other words, by varying v and ϕ and holding d and Ω constant, one can experimentally construct a 2D slice within the 4D grasping envelope by observing successful and unsuccessful grasps.

All experiments were conducted on the Stanford free-flyer test bed—a 3×4 m granite table calibrated to be extremely flat and level—on which robotic platforms can float using frictionless air bearings, simulating a 2D microgravity environment (See Fig. 4). In nearly the exact same setup as in [6], a smooth cylindrical object (1.6 kg, 11 cm radius) was fixed to a floating platform such that it could be spun

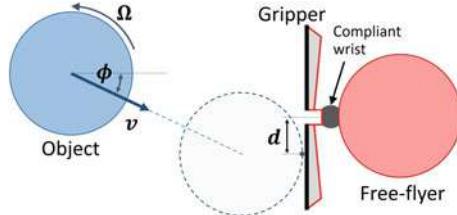


Fig. 3 The contact state between the spinning cylindrical object and free-flyer is parameterized with four variables as show in this top-down view: relative speed (v), angular velocity (Ω), angle of attack (ϕ), and offset (d). Note that these parameters are defined with respect to the free-flyer, which may also be moving

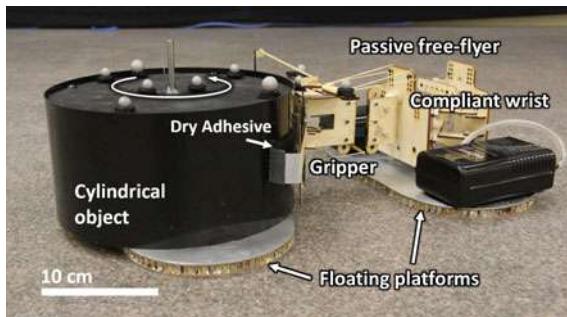


Fig. 4 Grasping experiment on the Stanford free-flyer test bed. A cylindrical object mounted on a frictionless air-bearing platform collides and attaches to another free floating platform equipped with a curved surface gripper. The dry adhesive fingers and compliant wrist are able to reliably secure the object over a wide range of dynamic contacts

and launched towards a gripper, which was also mounted on a floating platform. An OptiTrack motion capture system was used to measure the trajectories of the object, free-flyer robot, and its attached gripper to sub-millimeter precision at 120 Hz.

About fifty trials were run, varying the object's speed and angle of attack for each of two scenarios: (1) a high-mass free-flyer (4.2 kg or roughly 2.5 times the mass of the object), and (2) a low-mass free-flyer (1.7 kg or roughly the same mass as the object). The results in Fig. 5 show the data for both of these scenarios compared with data for a fixed gripper from [6]. Two analytical bounds were proposed in [6] to segment the successful and unsuccessful grasps and correlate them to the two dominant failure modes, which were: (A) a minimum normal impulse that was required to depress the gripper's passive trigger mechanism,¹ and (B) a maximum angular impulse that the gripper's compliant wrist could absorb after attaching. These bounds are still good predictors of failure for a floating free-flyer, however the normal and angular impulse must now also account for the movement of the free-flyer after collision.

¹Future gripper designs will incorporate an automatic trigger, eliminating the minimum normal impulse requirement.

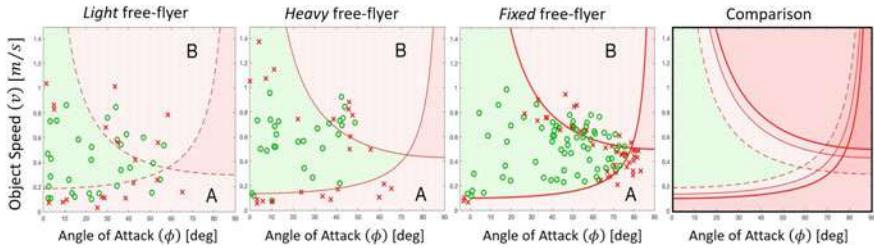


Fig. 5 Grasping envelopes relating speed (v) and angle of attack (ϕ) for a non-spinning object contacting the gripper with zero offset (d). The left three plots show data collected for a light (1.7 kg), heavy (4.2 kg), and fixed free-flyer, respectively. The green o's and red x's depict successful and unsuccessful trials. The right plot overlays the approximate envelope bounds for each of the three cases, indicating generally tighter bounds for lighter free-flyers

Thus, as the mass of the free-flyer is reduced, the minimum speed needed for the object to passively engage the gripper increases, the tolerable angular momentum of the object decreases, and overall, the grasping envelope shrinks.

For high-speed collisions, an additional failure mode was observed, whereby the floating free-flyer rebounds before the gripper can fully close around the object. This phenomenon involves the mechanical response of gripper's compliant mount and the response time of the bistable closing mechanism. In [13], Yoshida discusses the contact dynamics between a robotic arm and a floating satellite and shows that appropriate *impedance matching* can mitigate this effect. Future work will consider similar methods of impedance matching using tunable wrist compliance [8] to reduce this rebound effect. For the grasping controller discussed in Sect. 3, we will simply enforce a constraint on the maximum speed.

3 Autonomous Grasping

In this section we formally state the control problem we wish to address, devise a two phase formulation for its solution using optimal control techniques, and discuss feasibility guarantees and implementation details. We highlight that our problem formulation and tests are limited to planar motion; the generalization to 3D is possible and will be addressed in future work. Furthermore, we make two key assumptions: (1) the environment is obstacle-free, and (2) orbital dynamics can be ignored. In practice, the full motion planning problem for grasping would be decoupled into an initial rendezvous phase using a kinodynamic motion planner (e.g., [14]) to negotiate obstacles over an arbitrary distance, which transitions to this final controller within a close, obstacle-free vicinity. Similarly, the short timescales for this grasping problem make higher-order effects due to orbital dynamics negligible. Section 3.1 states the dynamics of the problem, Sect. 3.2 discusses the decoupled control architecture and

desired contact geometry, Sect. 3.3 derives the control law for phase 1, Sect. 3.4 derives the control law for phase 2, and Sect. 3.5 derives conservative analytical bounds for the region of attraction.

3.1 System Dynamics

We consider an autonomous docking between a target object (T) and a spacecraft (S) equipped with a dry adhesive gripper. The target object has a circular shape of radius r_T , mass m_T , and rotates with constant angular velocity ω_T . The spacecraft has a gripper located distance l_S from its center of mass S_{cm} , and rotates with angular velocity $\omega_S(t)$. We define a point T_g on the surface of T that represents the target point for contact (e.g., a part of the target surface that is particularly suitable for grasping). Right-handed orthogonal bases, n , t , and s are fixed in the inertial frame, target object, and spacecraft, respectively, rotated by angles θ_s and θ_T . The position vector from T_{cm} to S_{cm} can be written as $\mathbf{r}_S = x\hat{n}_x + y\hat{n}_y$ and its derivative, $\dot{\mathbf{v}}_S = \dot{x}\hat{n}_x + \dot{y}\hat{n}_y$. This notation is summarized in Fig. 6. The double-integrator dynamics of the spacecraft are simply,

$$\ddot{x} = u_x, \quad \ddot{y} = u_y, \quad \ddot{\theta}_S = u_{\theta_S}, \quad (1)$$

where u_x and u_y represent the translational control inputs (actuated, e.g., via thrusters), and u_{θ_S} represents the independent angular control input (actuated, e.g., via a reaction wheel). For convenience, we can rewrite the dynamics with respect to a new basis, e , as

$$\dot{v}_r - \frac{v_\theta^2}{r} = u_r, \quad \dot{v}_\theta + \frac{v_r v_\theta}{r} = u_\theta, \quad (2)$$

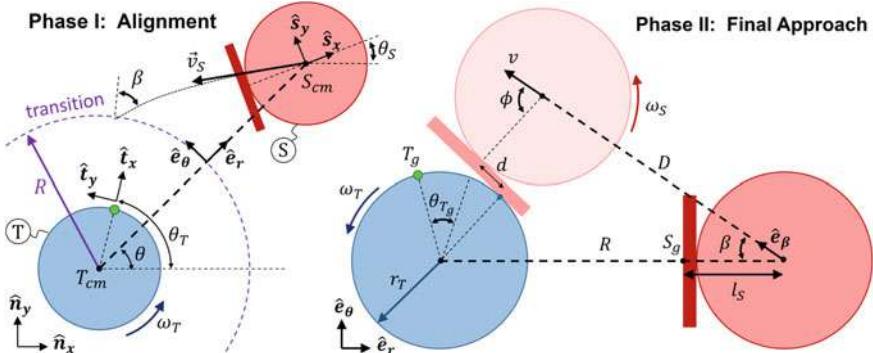


Fig. 6 Geometry of the grasping problem. The initial alignment phase (left) steers the spacecraft towards some desired approach trajectory, defined by β , at which point the final approach phase (right) tracks a straight-line to ensure proper timing and contact geometry

where $\mathbf{r}_S = r\hat{e}_r$, and $\mathbf{v}_S = v_\theta\hat{e}_\theta + v_r\hat{e}_r$. This form will be useful for deriving the alignment controller in Sect. 3.3. Note that thruster arrangements on spacecraft typically yield nonuniform maximum thrust capabilities *in the body frame*. Thus, most generally, $u_{r,\max}$ and $u_{\theta,\max}$ are functions of θ_S . However, for simplicity, we will impose a conservative inner approximation on the control constraints:

$$u_r^2 + u_\theta^2 \leq u_{\max}^2, \quad u_{\max} = \min[u_{\max}(\theta_S)], \quad (3)$$

which allows the exact mapping to thruster firings to be abstracted as a lower level controller.

3.2 Control Architecture

The grasping problem is constrained in three fundamental ways: (1) constraints on the control input, (2) a constraint on the contact location on the surface of the rotating target, and (3) dynamic contact constraints imposed by the gripper, as characterized in [6] and Sect. 2. For a spinning object, the constraint on the contact location imposes a coupled relationship on the pose and timing of contact, according to $t_f = \frac{\theta_{T_g} + 2\pi n}{\omega_T}$, where n is an integer number of rotations before collision, and θ_{T_g} encodes the contact pose. This integer constraint on the final time, combined with the complex 4D grasping envelope, makes this problem challenging to solve end-to-end as a single optimal control problem. We note that for $\omega_T \rightarrow 0$, we have $t_f \rightarrow \infty$, which leads to prohibitively slow solutions. Indeed this control approach is tailored for the case in which the target's angular velocity is faster than a simpler linear controller can handle (e.g., by “chasing” the target point). In other words, our control approach should be considered complimentary to a controller that can handle static or quasi-static cases.

Accordingly, we decompose the grasping problem into two phases. Phase 1 aligns the spacecraft's velocity vector with the desired approach vector (Fig. 6, left) and phase 2 simply tracks this straight line trajectory and ensures proper contact timing (Fig. 6, right). Importantly, the phase switch is assumed to occur sufficiently far from the target as to guarantee a feasible, time-optimal solution—thus imposing constraints on initial conditions, as discussed in Sect. 3.5.

To define this switching condition, we must work backwards from the desired contact state within the grasping envelope. In principle, an appropriate selection of approach trajectory can map to *any* desired point in the grasping envelope. Practically, however, the spacecraft cannot spin arbitrarily fast to match the object. In fact, it is often desired for the spacecraft to have *zero* angular velocity for robust trajectory tracking (i.e., so that thrusters are not spinning). Therefore, by forcing $\omega_S = 0$, the relative angular velocity at contact is simply that of the target object, ω_T .

There are many ways in which the remaining contact states (v^*, ϕ^*, d^*) may be chosen. Given some estimate of the grasping envelope, one strategy would be to inscribe a maximum radius sphere within the (3D) slice defined by $\Omega = \omega_T$.

The center of this sphere is one measure of the most robust target point. Therefore, given some appropriate selection of contact state $(v^*, \omega_T, \phi^*, d^*)$, the geometry of the approach trajectory in phase 2 (see Fig. 6, right) can be uniquely defined as:

$$v = v^*, \quad \phi = \phi^*, \quad \omega_S = 0, \quad \beta = \sin^{-1} \left(\frac{d^* \cos \phi^* + (l_S + r_T) \sin \phi^*}{R} \right), \quad (4)$$

where R is the distance of the spacecraft at the beginning of phase 2. Interestingly, as discussed in [6], this optimal target point often corresponds to a non-zero offset and angle of attack for spinning objects—a key difference from traditional grippers.

Note that this paper does not address attitude control, which is a function of the specific arrangement of actuators for a given spacecraft. For the planar motion with a reaction wheel considered here, the (1D) solution is trivial. We simply assume that the spacecraft is able to rotate to the desired heading for grasping within t_f .

3.3 Phase 1: Alignment

The goal of the initial alignment phase is to drive the spacecraft to the desired approach vector computed by (4) in minimum time. Specifically, the final switching condition is met at t_s when,

$$\frac{v_\theta(t_s)}{-v_r(t_s)} = \tan \beta. \quad (5)$$

Intuitively, this can be thought of as applying some control input to effectively “rotate” the velocity vector until it points at the desired contact location. Note from Eq. (4) that β is a function of R , which is time-varying. Thus, while β cannot be computed exactly a priori, Eq. (5) can easily be evaluated at each time step to check for the switching condition.

The control input to achieve this in *minimum time* is simply a maximum thrust *normal* to the approach vector, specifically:

$$u_\theta^*(t) = u_{\max} \cos \beta \frac{-v_\theta(t)}{|v_\theta(t)|}, \quad u_r^*(t) = u_{\max} \sin \beta \frac{v_r(t)}{|v_r(t)|}. \quad (6)$$

Since β is often small (within 10° for typical parameters and zero in the nominal case), the dominant component of thrust is normal to the spacecraft’s position vector (\mathbf{r}_S), effectively arresting the spacecraft’s angular momentum about the target. Furthermore, the geometry of this thrust is such that the spacecraft’s speed will always decelerate. Thus, a total speed constraint (that is not initially violated) will remain obeyed. An apparent drawback of this approach is the inability to directly consider position constraints, which may arise due to, e.g., narrow corridors within the ISS.

However, as a last stage in a higher level planning framework, this controller can make some assumptions about the allowable set of initial states (e.g., from a kinodynamic planner) that guarantee a collision-free “approach corridor”.

3.4 Phase 2: Final Approach

After aligning the velocity vector (along \hat{e}_β) in phase 1, the goal for phase 2 (starting at t_0) is to track this straight-line trajectory to intercept the target object at state (v^*, ϕ^*) and location T_g using *minimum fuel*. Indeed, a periodic constraint is imposed on the final time:

$$t_f = \begin{cases} \frac{\theta_{T_g}(t_0) + 2\pi n}{\omega_T}, & \omega_T < 0 \\ \frac{2\pi(1+n) - \theta_{T_g}(t_0)}{\omega_T}, & \omega_T > 0 \end{cases}, \quad \theta_{T_g}(t_0) = \theta_T(t_0) - \theta(t_0) - \phi^* + \beta + \frac{d^*}{r_T}, \quad (7)$$

where n is the integer number of full revolutions of the target object before contact. The minimum feasible n also corresponds to the *minimum time* solution. For some choice of n , we can formally state the 1D input-constrained minimum fuel optimal control problem:

$$\begin{aligned} \min \quad & \int_0^{t_f} |u(t)| dt \\ \text{s.t.} \quad & \dot{X} = AX + Bu \\ & X(0) = [r, v_0]^T \\ & X(t_f) = [D, v_f]^T \\ & u_{\min} \leq u(t) \leq u_{\max} \end{aligned} \quad (8)$$

where $X = [r, v]$, $\dot{r} = v$, $D = \frac{1}{\cos \beta} [R - r_T \cos(\phi - \beta) - l_S \cos(\beta - \phi) - d \sin(\beta - \phi)]$, and A and B represent the dynamics of a 1D double integrator. It is known that the solution to an input constrained minimum-fuel optimal control problem (where the system is controllable) will have a bang-off-bang form [15]. Additionally, for our specific problem, there are a family of fuel-optimal solutions corresponding to the choice of n . For an initial radius (D) sufficiently large, an optimal solution is to fire the thrusters one time in an off-bang-off regime, whereby the timing and duration of the firing determines the impact speed (v_f) and time (t_f). The total time is given by the sum of the initial coast phase (τ_1), acceleration phase (τ_2), and final coast phase (τ_3). Similarly, the total distance traveled (D) can be decomposed into three phases. With appropriate manipulation, this allows the timing to be computed as:

$$\tau_1 = \frac{D + v_f(\tau_2 - t_f) - \frac{|v_f^2 - v_0^2|}{2u_{\max}}}{|v_f - v_0|}, \quad \tau_2 = \frac{|v_f - v_0|}{u_{\max}}. \quad (9)$$

Note that this solution is only valid for $0 \leq \tau_1 \leq t_f - \tau_2$ and $\tau_2 < t_f$. In other words, a single-fire solution may not exist for spacecraft that starts too far away, approaches too fast, too slow, or when $v_0 \approx v_f$. For this case when $\tau_1 \geq t_f - \tau_2$, a two-fire, bang-off-bang control is optimal, whereby the spacecraft immediately thrusts for duration τ_1^* , coasts for τ_2^* at speed v_2 , and thrusts for the remaining τ_3^* . Similar to (9), the timing of the firing can be computed as:

$$\tau_1^* = \frac{|v_2 - v_0|}{u_{\max}}, \quad \tau_3^* = \frac{|v_f - v_2|}{u_{\max}}, \quad v_2 = \frac{Du_{\max} - \frac{1}{2}|v_2^2 - v_0^2| - \frac{1}{2}|v_f^2 - v_2^2|}{u_{\max}t_f - |v_2 - v_0| - |v_f - v_2|}. \quad (10)$$

This solution is also only valid for $\tau_1^* + \tau_3^* \leq t_f$. Otherwise, the timing mismatch at t_0 is too large for a bang-off-bang regime to compensate. However, an appropriate constraint on the initial state (discussed in Sect. 3.5), can guarantee that *either* a single-fire solution (Eq. (9)) or two-fire solution (Eq. (10)) exists.

3.5 Approximate Region of Attraction

In summary, given some initial state, the two-phase control proceeds as follows:

1. Select a desired location on the surface of the target to grasp, T_g .
2. Using some model for the grasping envelope, select a robust target point $(v^*, \omega_T, \phi^*, d^*)$ as the desired contact state.
3. Execute the control for phase 1 according to (6).
4. Watch for terminal condition given by (5) and switch to phase 2 when triggered.
5. Compute optimal single-fire control inputs according to (9).
6. If infeasible, compute the two-fire optimal control solution according to (10).
7. Execute phase 2 controller, optionally with a closed-loop tracking controller (e.g., LQR), to drive the spacecraft to the desired contact state.

In order to stitch this controller to a preceding planner, we would like to formally characterize the set of initial states from which a feasible solution is guaranteed—corresponding to, for example, the goal region of a kinodynamic planner. First, Eqs. (9) and (10) will be used to derive a minimum distance, D_{\min} at which phase 2 must begin to guarantee a feasible solution *for any possible target point*. Then, a conservative linearization of the dynamics given by Eqs. (2) and (6) will provide an inner approximation of the backwards reachable set to achieve this transition.

3.5.1 Region of Attraction for Phase 2

To derive the minimum distance D_{\min} for phase 2, we start by realizing that in order to guarantee feasibility for *any* choice of T_g (i.e., at least one feasible choice of n), then it is sufficient to guarantee that a solution exists for all $t_{f,\min} \leq t_f \leq t_{f,\min} + 2\pi/\omega_T$ (i.e., the time the target takes to complete one full rotation).

For the single-fire solution given by Eq. (9), the minimum distance D_{\min} can be derived by setting the difference in timing between the slowest solution and fastest solution exactly equal to one rotation period: $t_{f,\max} - t_{f,\min} = \frac{2\pi}{\omega_T}$. For example, in the case when the spacecraft needs to slow down (i.e., $v_f < v_0$), the slowest solution is to apply u_{\max} immediately and coast at v_f until impact, and the fastest solution is to wait until just before impact to apply u_{\max} . Substituting this into (9) and (10) and solving for D_{\min} , we have

$$D_{\min} = \frac{2\pi v_0 v_f}{\omega_T |v_f - v_0|} + \frac{|v_f^2 - v_0^2|}{2u_{\max}}. \quad (11)$$

The second term corresponds to the distance traveled during thrusting, and the first term represents the distance required to adjust phasing of contact by up to 2π .

For a two-fire solution, we can take the same approach by computing $t_{f,\max} - t_{f,\min} = \frac{2\pi}{\omega_T}$. In this regime, $t_{f,\min}$ is achieved by accelerating as long as possible before immediately decelerating to hit T at v_f (i.e., $\tau_2^* = 0$), and $t_{f,\max}$ is the exact opposite. However, in some cases $t_{f,\max} = \infty$, corresponding to the case when the spacecraft can fully stop before accelerating. Thus Eq. (10) can be manipulated in a similar way to solve for D_{\min}^* , the minimum distance required for a guaranteed solution in a two-fire regime:

$$D_{\min}^* = \min \left\{ \frac{v_0^2 + v_f^2}{2u_{\max}}, \quad \left| \frac{\pi}{2\omega_T} - \frac{v_0 + v_f}{2u_{\max}} \right| \sqrt{(v_0 - v_f)^2 + \frac{2\pi u_{\max}(v_0 + v_f)}{\omega_T} - \frac{\pi^2 u_{\max}^2}{\omega_T^2}} \right\}. \quad (12)$$

3.5.2 Region of Attraction for Phase 1

Now that we have characterized the minimum distance required at the phase 2 transition, we would like to compute the backwards reachable set through the control input during phase 1 to find a set of initial states for which a solution is guaranteed. However, the coupled, second order nonlinear dynamics from Eqs. (2) and (6) cannot be solved in closed form. Instead, we can solve for a conservative approximation of the minimum time, $\hat{t}_s \geq t_s$ by linearizing the dynamics:

$$\dot{v}_r = u_r + \frac{v_\theta^2}{r} \approx 0, \quad (13)$$

$$\dot{v}_\theta = u_\theta - \frac{v_r v_\theta}{r} \approx u_{\theta,\text{eff}} = \frac{u_\theta - v_r(0)v_\theta(0)}{r(0)}, \quad (14)$$

where $u_{\theta,\text{eff}}$ represents the reduced effective control in the \hat{e}_θ direction. For $v_r(0) < 0$ (i.e., moving towards the target), u_θ and $C(t) = v_r(t)v_\theta(t)/r(t)$ always have the same sign. Furthermore, it can be shown that $\text{sgn}(\frac{dC}{dt}) = -\text{sgn}(C(t)) \forall t \in (0, t_s)$, if $(v_\theta^2 + 2|v_r v_\theta|)/r \leq u_{\max}$. In other words, the magnitude of C is monotonically

decreasing, and $|\dot{v}_\theta| \geq |u_{\theta,\text{eff}}| \forall t \in (0, t_s)$. Thus, $\dot{v}_\theta \approx u_{\theta,\text{eff}}$ will serve as a conservative approximation for computing \hat{t}_s .

Similarly, since it is assumed that $v_r < 0$, the approximation that $\dot{v}_r \approx 0$ yields a conservative approximation of the inward radial distance traveled (Δr_{\min}) if $\dot{v}_r \geq 0$, which is true for $\beta \leq \sin^{-1}(v_\theta^2/(ru_{\max}))$. Finally, we can use the linearized dynamics given by (13) and (14) to compute \hat{t}_s and Δr_{\min} :

$$\hat{t}_s = \frac{v_\theta(0) + v_r(0) \tan \beta}{u_{\theta,\text{eff}}}, \quad \Delta r_{\min} = v_r(0)\hat{t}_s. \quad (15)$$

Combining Eqs. (11), (12), and (15), we can express the total initial distance the spacecraft must be from the target as:

$$r \geq \min(D_{\min}, D_{\min}^*) + \Delta r_{\min}. \quad (16)$$

Note that the initial velocity in phase 2 (v_0 in Eqs. (11) and (12)) is now approximated by $|v_r(0)|$. In summary, the set of initial states for persistent feasibility is defined by (16) and the following assumptions:

$$v_r < 0, \quad \frac{v_\theta^2 + 2|v_r v_\theta|}{r} \leq u_{\max}, \quad \beta \leq \sin^{-1} \left(\frac{v_\theta^2}{ru_{\max}} \right). \quad (17)$$

While these are complicated, interdependent expressions, in the context of a sampling-based motion planner, they are cheap to evaluate (i.e., query the goal state).

4 Experimental Results

The two-phase grasping controller developed in Sect. 3 was implemented on the Stanford free-flyer test bed (see Fig. 7). A simple PD controller was used to control free-flyer attitude in Phase 2. A passive target object was manually spun and pushed at some initial coasting velocity. The free-flying robot equipped with eight thrusters and a reaction wheel was pushed at varying initial velocities (within the region of attraction), immediately executing the grasping controller. Figure 8 displays five (of 16) example trajectories overlaid on the (ideal) simulated trajectory. A video of one example trajectory can be found at: <https://www.youtube.com/playlist?list=PL8-2mtIIIFIJrHMGNeKmHnkB1XBD0iKvto>.

Overall, there is good agreement between the measured and simulated trajectories. Most of the deviation can be attributed to modeling errors—in particular, the changing mass of the free-flyers as the CO₂ tanks drain (to be addressed with online system identification in future work). We also observed some timing errors during the second approach phase which caused the free-flyer to occasionally miss the target point. This is because the reference trajectory for the second phase was computed immediately after phase one, which can result in mis-timed grasps when unanticipated modeling

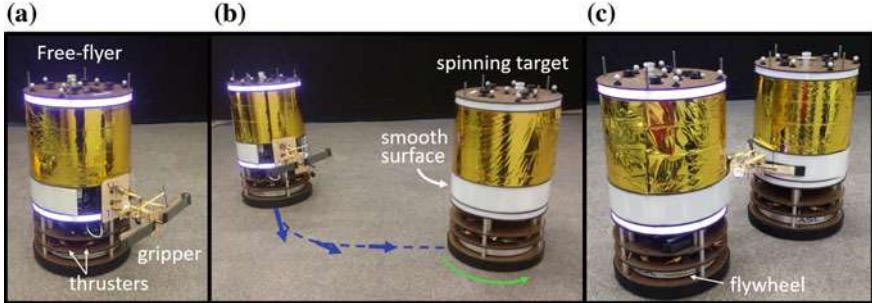


Fig. 7 Autonomous grasping experiments on the Stanford free-flyer test bed. **a** A free-flying robot floats on frictionless air bearings and is equipped with eight compressed gas thrusters and a flywheel. **b** The trajectory controller developed in Sect. 3 is executed on-board to dynamically grasp **c** a translating and spinning target

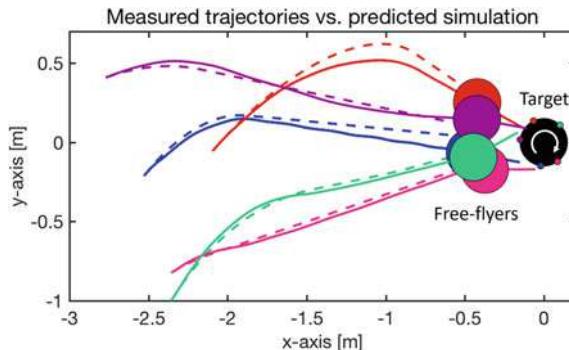


Fig. 8 Measured motion capture data for three example trajectories (solid lines) of a free-flying robot grasping a spinning target (black) overlaid with simulated predictions (dashed lines). The points on the surface of the target represent the locations of the target point (T_g) upon impact for the corresponding color. Gripper orientation is indicated by a straight line

errors such as table friction are present. Future experiments will incorporate an MPC-style implementation of the phase two controller that constantly recomputes the reference trajectory for more robust grasping.

5 Conclusions

In this paper we presented an optimal control approach for the problem of dynamic grasping of tumbling objects in space using gecko-inspired adhesive grippers. We extended the characterized grasping envelope for a curved surface gripper to the case when both the spacecraft and target object are free floating and of comparable mass. We developed a two-phase control architecture that decomposes the grasping

problem into an initial alignment phase and final approach phase, each of which with time optimal solutions. A conservative inner approximation of the region of attraction for initial states was derived analytically to serve as a terminal goal region for a preceding motion planner. Experimentation is ongoing, but the preliminary results constitute one of the first successful demonstrations of autonomous surface grasping in a high-fidelity spacecraft analog test bed.

This paper leaves numerous important extensions open for future research. First, it is important to extend the controller to handle non-cylindrical objects, whereby surface target selection should be addressed in a more principled way. Second, we plan to introduce an actuated arm that allows for more robust acquisition through active damping and impedance matching, and also for manipulation tasks. Third, we plan to extend this controller and gripper design to allow for out-of-plane motion. Finally, future experiments will be integrated with a preceding kinodynamic motion planner to negotiate obstacle-rich environments.

Acknowledgements This work was supported by NASA under the Space Technology Research Grants Program, Grant NNX12AQ43G, and Early State Innovations, Grant NNX16AD19G.

References

1. Fong, T., Micire, M., Morse, T., Park, E., Provencher, C., To, V., Wheeler, D.W., Mittman, D., Torres, T., Smith, E.: Smart SPHERES: a telerobotic free-flyer for intravehicular activities in space. In: AIAA SPACE Conferences and Exposition (2013)
2. Ambrose, R., Nesnas, I.A.D., Chandler, F., Allen, B.D., Fong, T., Matthies, L., Mueller, R.: 2015 NASA Technology Roadmaps: TA 4: Robotics and Autonomous Systems. Technical Report, NASA (2015)
3. Aghili, F.: A prediction and motion-planning scheme for visually guided robotic capturing of free-floating tumbling objects with uncertain dynamics. *IEEE Trans. Robot.* **28**(3), 634–649 (2012)
4. Motaghedi, P.: On-orbit performance of the orbital express capture system. *Proc. SPIE* 6958(0E) (2008)
5. Jiang, H., Hawkes, E.W., Arutyunov, V., Tims, J., Fuller, C., King, J.P., Seubert, C., Chang, H.L., Parness, A., Cutkosky, M.R.: Scaling controllable adhesives to grapple floating objects in space. In: Proceedings of IEEE Conference on Robotics and Automation (2015)
6. Estrada, M.A., Hockman, B., Bylard, A., Hawkes, E.W., Cutkosky, M.R., Pavone, M.: Free-flyer acquisition of spinning objects with gecko-inspired adhesives. In: Proceedings of IEEE Conference on Robotics and Automation (2016)
7. Hawkes, E.W., Christensen, D.L., Eason, E.V., Estrada, M.A., Heverly, M., Hilgemann, E., Jiang, H., Pope, M.T., Parness, A., Cutkosky, M.R.: Dynamic surface grasping with directional adhesion. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2013)
8. Estrada, M.A., Jiang, H., Noll, B., Hawkes, E.W., Pavone, M., Cutkosky, M.R.: Force and moment constraints of a curved surface gripper and wrist for assistive free flyers. In: Proceedings IEEE Conference on Robotics and Automation (2017)
9. Weiss, A., Baldwin, M., Erwin, R.S., Kolmanovsky, I.: Model predictive control for spacecraft rendezvous and docking: Strategies for handling constraints and case studies. *IEEE Trans. Control Syst. Technol.* **23**(4), 1638–1647 (2015)

10. Hartley, E., Trodden, P.A., Richards, A.G., Maciejowski, J.M.: Model predictive control system design and implementation for spacecraft rendezvous. *Control Eng. Pract.* **20**(7), 695–713 (2012)
11. Park, H., Di Cairano, S., Kolmanovsky, I.: Model predictive control of spacecraft docking with a non-rotating platform. *IFAC World Congr.* **44**(1), 8485–8490 (2011)
12. Di Cairano, S., Park, H., Kolmanovsky, I.: Model predictive control approach for guidance of spacecraft rendezvous and proximity maneuvering. *Int. J. Robust Nonlinear Control* **22**(12), 1398–1427 (2012)
13. Yoshida, K., Nakanishi, H., Ueno, H., Inaba, N., Oda, M.: Dynamics, control and impedance matching for robotic capture of a non-cooperative satellite. *Adv. Robot.* **18**(2), 175–198 (2004)
14. Schmerling, E., Janson, L., Pavone, M.: Optimal sampling-based motion planning under differential constraints: the drift case with linear affine dynamics. In: Proceedings of IEEE Conference on Decision and Control (2015)
15. Kirk, D.E.: *Optimal Control Theory: An Introduction*. Courier Corporation (2012)

Generation of Turning Motion for Tracked Vehicles Using Reaction Force of Stairs' Handrail

**Yuto Ohashi, Shotaro Kojima, Kazunori Ohno, Yoshito Okada,
Ryunosuke Hamada, Takahiro Suzuki and Satoshi Tadokoro**

Abstract Inspections by mobile robots are required in chemical and steel plants. The robots are required to ascend and descend stairs because equipment components are installed on different-level floors. This paper proposes turning motion for tracked vehicles on stairs. A characteristic of the proposed turning motion is that it is generated using the reaction force from the safety wall of the stairs' handrail. The safety wall is commonly used in plants because it prevents objects from dropping down and damaging equipments. Proper turning motion is generated based on the motion model of the tracked vehicle. Experimental results show that the proposed turning motion can change the heading direction on the stairs. In addition, the proposed turning motion enables the vehicle to run with less slippage, as compared to other turning motions. The proposed method can reduce slippage by 88% while climbing up the stairs and by 44% while climbing down the stairs. The proposed method is more effective on the upward stairs than on the downward stairs. An autonomous turning motion control is implemented on the tracked vehicle, and it is evaluated on the upward stairs.

1 Introduction

Inspection using robot technologies is required to prevent accidents caused by equipment issues or deterioration in chemical and steel plants. It is considerably risky for human workers to inspect dangerous equipment components, such as blast furnace, during operation. The use of robot technologies can reduce the risk involved in the inspection of equipment during operation.

Tracked vehicles, which can climb up and down on stairs, are suitable for inspection because equipment components are installed on different-level floors. Therefore, we propose an inspection method using a tracked vehicle with sub-tracks, which is shown in Fig. 1 [1]. This vehicle can climb up and down on stairs. Instead of human workers, the vehicle can inspect equipments on different-level floors.

Y. Ohashi (✉) · S. Kojima · K. Ohno · Y. Okada · R. Hamada · T. Suzuki · S. Tadokoro
Tohoku University, Sendai, Miyagi 9800845, Japan
e-mail: ohashi.yuto@rm.is.tohoku.ac.jp

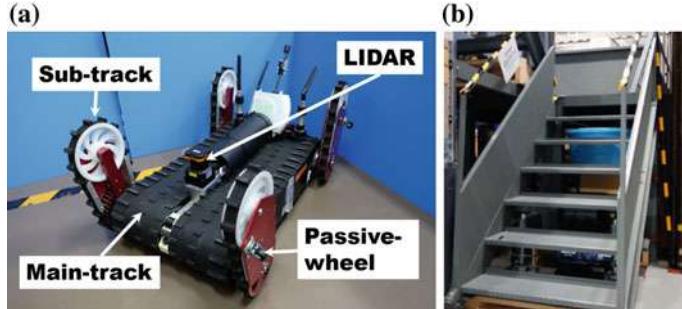


Fig. 1 Tracked vehicle “Quince” (a) and target stairs with safety walls (b)

An important function of tracked vehicles is to change the heading direction on the stairs. A track is a mechanism in which slippage occurs in principle on the ground and stairs. During the climbing up/down motion on the stairs, the heading direction changes because of slippage and gravity. Therefore, it is necessary to adjust the heading direction to the upward/downward direction on the stairs.

This paper proposes turning motion using the reaction force from the safety wall of the stairs’ handrail, which prevents objects from dropping down. This is a new approach for changing the heading direction on the stairs. A characteristic of the proposed method is to generate the turning motion using the turning moment caused by the reaction force from the safety wall of the stairs’ handrail. In general, tracked vehicles change heading directions using the difference between the velocities of the left and right tracks. The proposed method generates turning motion using the turning moment caused by the reaction force, in addition to the velocity difference. Experimental results show that the proposed method can reduce slippage on stairs, as compared to the turning motion based on the velocity difference.

The remaining part of the paper is organized as follows: Related works are explained in Sect. 2. The turning motion using the reaction force is proposed in Sect. 3. Proper velocity is derived from kinematics constraints. The proposed method is evaluated on the upward and downward stairs, and the results are provided in Sect. 4. The results show that the proposed method is effective on the upward stairs. Autonomous turning motion control is implemented on the tracked vehicle, and its evaluation on the upward stairs is described in Sect. 5. The paper is concluded in Sect. 6.

2 Related Works

Collisions with obstacles and the environment prevents proper motion of mobile robots because the reaction force caused the collision can damage the robot. Therefore, collision avoidance is an important research topic in the case of mobile robots. Several studies have been conducted on collision avoidance [2–6]. We consider that

the reaction force can be used to control motion. In this paper, the tracked vehicle does not prevent collision and positively uses the reaction force to change the heading direction on the stairs.

Motion control based on the reaction force is an important research topic in the case of mobile robots. Compliance control is a widely used technique. Compliance control enabled mobile robots or robotics arms can reduce the reaction force caused by a collision and move along the surface of objects or an environment [7–9]. Rude proposed a compliance control method using mechanical dampers and springs [10]. The proposed method could change the heading direction of mobile robots using the reaction force caused by collisions in a clutter environment. However, compliance control cannot move a vehicle in the appropriate direction. The purpose of compliance control is to reduce the reaction force. The motion generated using compliance control does not ensure that a mobile robot will face the target direction. This paper proposes turning motion using the reaction force. A kinematic model is used to determine the proper motion.

The reaction force from walls prevents mobile robots from generating proper turning motion. Kojima et al. proposed a control method to prevent this problem [11]. The study suggests that the proper control rule enables to generates turning motion by using the reaction force. Additionally, the method did not require direct measurement of the reaction force, and it generated proper turning motion. The approach proposed in this paper is based on the same perspective as that of the previously mentioned research, and it can generate proper turning motion on the stairs.

3 Turning Motion Using Reaction Force

This paper proposes a new method for generating turning motion on the stairs. The proposed method uses the difference between the speeds of left and right tracks and the reaction force to generate turning motion. In general, turning motion is generated using the difference between the speeds of the left and right tracks. Turning moment is generated using only the difference between these speeds. The turning moment generated using the reaction force is larger than that generated using the difference between the speeds of the left and right tracks. We consider that a combination of the speed difference and the reaction force can generate an even larger turning moment.

The safety wall of the stairs' handrail is used to generate the reaction force. The safety wall is commonly used in chemical and steel plants because it prevents objects from dropping down and damaging equipment.

Passive wheels are attached to the side of the sub-tracks, which is shown in Figs. 1a and 2. These wheels can reduce the friction force between the safety wall and wheels. Without passive wheels, the friction force prevents the tracked vehicle from turning on the stairs. In addition, the use of passive wheels reduces the damage to the safety wall and tracked vehicle.

Fig. 2 Passive wheel attached to sub-track

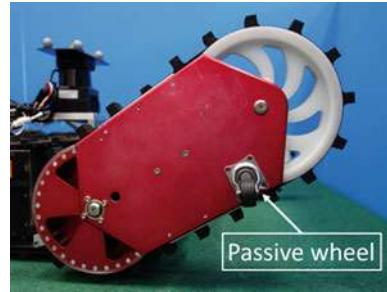


Fig. 3 Proposed turning motion and its trajectory for the tracked vehicle on stairs

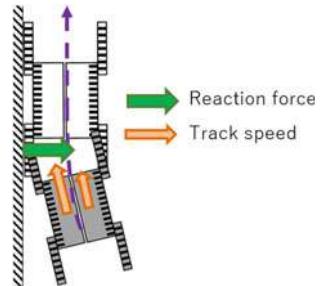
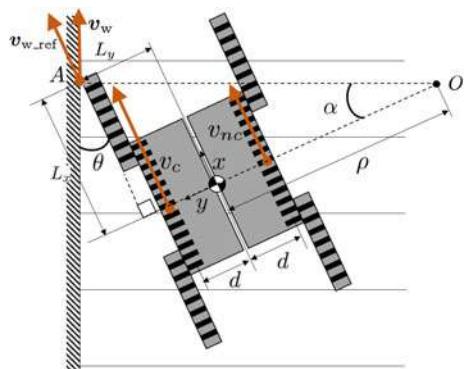


Fig. 4 Kinematic model of the tracked vehicle that contacts the stairs' handrail



The tracked vehicle moves as shown in Fig. 3; it moves and turns along the safety wall using the reaction force. To generate this turning motion, we need to consider how to derive the proper torque and speed of the left and right tracks. This paper proposes a speed control method based on the kinematic model of the turning motion of the tracked vehicle shown in Fig. 4. A coordinate frame is attached to the center of the vehicle body, in which the x -axis faces the vehicle's front and the y -axis faces its left. The turning center is O . Table 1 defines the parameters of this kinematic model. It is possible to derive the velocity condition using the kinematic model even if contact state varies.

Table 1 Parameters of the kinematic model

v_c	Peripheral velocity of contact side track
v_{nc}	Peripheral velocity of non-contact side track
v_w	Actual velocity vector of contact point with the wall
v_{w_ref}	Commanded velocity vector of contact point with the wall
$2d$	Tread
L_x	Distance between center of gravity and contact point on x -axis
L_y	Distance between center of gravity and contact point on y -axis
θ	Angle between wall and robot
α	Angle between line segment from contact point to axis of rotation and the line segment from axis of rotation to center of gravity
ρ	Turning radius

To generate the reaction force, it is necessary for the tracked vehicle to move toward the wall direction that is represented by velocity v_{w_ref} at contact point A in Fig. 4. This is the commanded velocity at contact point A. When the tracked vehicle moves along the wall because of the reaction force, the actual velocity, v_w , is parallel to the safety wall at point A. In this case, the center of the turning motion exists on the line that is perpendicular to the wall at contact point A. When the tracked vehicle does not contact the wall, the center of the turning motion does not exist on the line.

The condition under which the robot maintains contact with the wall is derived using the kinematic model as

$$\theta + (90^\circ - \alpha) \geq 90^\circ \quad (1)$$

Angle α is

$$\alpha = \arctan \left(\frac{L_x}{L_y + \rho} \right) \quad (2)$$

Based on Eqs. (1) and (2), the condition of contact with the wall is

$$\tan \theta \geq \frac{L_x}{L_y + \rho} \quad (3)$$

Velocity v and angular velocity ω are given by

$$v = \frac{v_c + v_{nc}}{2} \quad (4)$$

$$\omega = \frac{v_c - v_{nc}}{2d} \quad (5)$$

Therefore, turning radius ρ is

$$\rho = \frac{v}{\omega} = \frac{d(v_c + v_{nc})}{v_c - v_{nc}} \quad (6)$$

As the robot wishes to turn around turning center O clockwise, the following relationship is obtained:

$$v_c > v_{nc} > 0 \quad (7)$$

The condition under which the robot maintains contact with the wall can be obtained using Eqs. (3), (6), and (7) as follows:

$$v_{nc} \geq \left(\frac{L_x - (d + L_y) \tan \theta}{L_x + (d - L_y) \tan \theta} \right) v_c \quad (8)$$

By moving such that this condition is satisfied, it is possible to perform a turn along the wall while continuously obtaining the moment due to the wall reaction force, as shown in Fig. 3.

4 Evaluation of Turning Motion Generated Using Reaction Force from Safety Wall of Stairs' Handrail

4.1 Evaluation Method

The following three turning methods are implemented on the tracked vehicle and compared on the stairs:

- A: **Reaction force based turning motion (reaction force)** The tracked vehicle moves forward, without turning motion. The heading direction is changed by the reaction force caused by the forward motion.
- B: **Turning motion based on differential between left and right track speeds (differential)** The tracked vehicle turns when it contacts the wall. The turning motion is generated using only the difference between the speeds of the left and right tracks.
- C: **Proposed turning motion (reaction force + differential)** The tracked vehicle turns when it contacts the wall. The turning motion is generated using the reaction

force from the wall and the difference between the speeds of the left and right tracks.

The same initial conditions and pose were used for these evaluations. The vehicle started the motion upon contact with the wall, and its angle, θ , was 15° . The vehicle climbed the stairs, which were at an inclination of 40° , as shown in Fig. 1b. Typically, the tracked vehicle stretches the front and rear sub-tracks when it climbs the stairs for stabilizing itself. In motions B and C, the robot started turning from the beginning, and after completing the turn, it proceeded straight at the rotational velocity of the main tracks' motor, which was 2000 rpm. Each running test was performed three times.

The rotational speed of the motor at the contact side was set at 2000 rpm. The velocity of the track at the contact side was obtained using the gear ratio from the motor to the output shaft, n , and the pulley diameter, D , using the following equation:

$$v_c = 2000 n D \text{ [m/s].} \quad (9)$$

The velocity of the track at the non-contact side, v_{nc} , can be obtained from the Eqs. (8) and (9). Therefore, the track velocity on the non-contact side was determined such that it satisfied. Here, $2d = 0.37$ m, $L_x = 0.41$ m, and $L_y = 0.28$ m. Therefore, v_{nc} is derived as follows:

$$v_{nc} \geq 1485 n D \text{ [m/s].} \quad (10)$$

The velocity which is used in Eqs. (4)–(8) can be used with the actual track velocity or the approximated value of the commanded track velocity. Here, we use the command velocity. Table 2 shows the track speeds at the contact and non-contact sides for each type of motion. The rotational speed of the motor at the non-contact side was derived using Eq. (10) for motions B and C. v_{nc} of motion C is $1500 n D$, which satisfies the condition of the robot maintaining contact with the wall as shown in Eq. (10). Turning time, which is the time required by the robot to directly face the stairs, was determined empirically based on the result of a preliminary experiment.

The above three motions are evaluated using movie and motion capture data. Turning angular velocity is calculated using the motion capture data during the turning motion. For motion A, turning time is the time required to climb the first step. For motions B and C, turning times are provided in Table 2. In addition, moving speed and slippage of robot are evaluated using the motion capture data because slippage is an important factor for turning motion on the stairs.

Slippage is evaluated using a vertical velocity ratio as shown in Eq. (11).

$$\text{Vertical velocity ratio} = \left| \frac{v_{z_ref} - v_z}{v_{z_ref}} \right|. \quad (11)$$

where v_z is the current vertical velocity and v_{z_ref} is the commanded vertical velocity. If the vertical velocity ratio is larger than 1, it can be judged that slippage occurs in the z direction. This equation is determined in reference to slip ratio. Slip distance is

Table 2 Rotational speed and turning time

Motion	Rotational speed of contact side [rpm]	Rotational speed of non-contact side [rpm]	Turning time [s]
A: Reaction force	2000	2000	—
B: Differential	2000	1000	1.8
C: Reaction force + differential	2000	1500	2.0

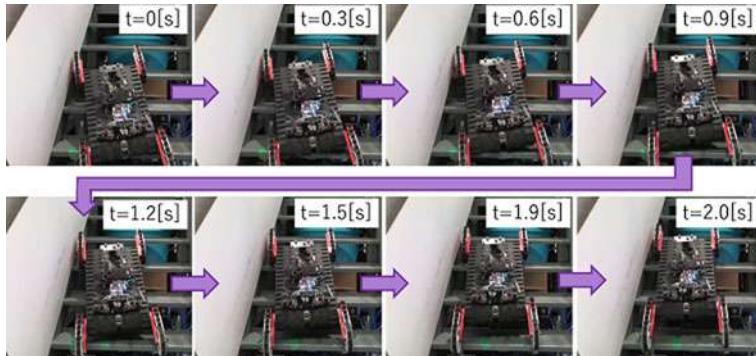
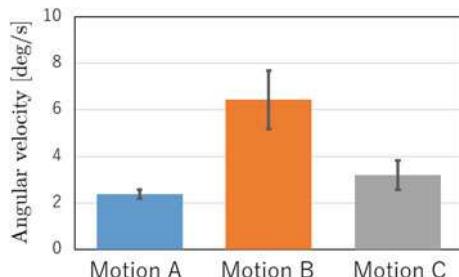
**Fig. 5** Proposed turning motion, which uses reaction force from the wall while climbing

Fig. 6 Turning angular velocities while climbing; A:
Reaction force, B:
Differential, C: Reaction
force + differential



derived by integrating the velocity in the vertical direction while slipping. The slip and the time required to run the stairs was evaluated as the time over which the vertical displacement changed by one step, which is at a height of 0.21 m from the start of trial. These two parameters were evaluated using tracking software (Tracker3.3, Vicon), based on the images obtained using a motion capture camera (VANTAGE V5, 5 million pixels, Vicon). The sampling frequency of the motion capture camera was 100 [Hz]. Motion capture is used only to evaluate the motions.

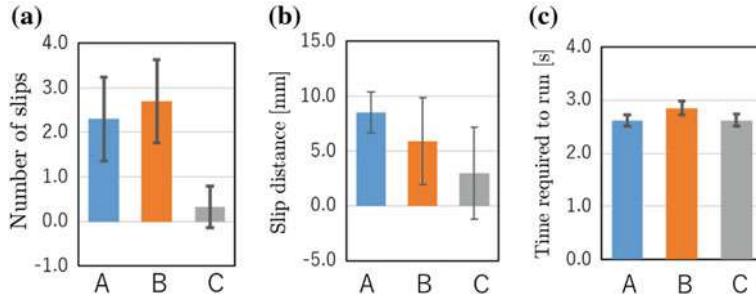


Fig. 7 Number of slips, and slip distance, and time required to run while climbing. A: Reaction force, B: Differential, C: Reaction force + differential

4.2 Evaluation of the Turning Motion on the Upward Stairs

Movie and motion capture data were recorded during the evaluations. The recorded movies showed that motions A, B, and C generated the turning motion on the stairs. Even though these behaviors were different, it was difficult to observe the difference in the movies. Figure 5 shows the turning motion generated by motion C (reaction force + differential).

Motion data were used to analyze the difference between these turning motions on the upward stairs. The angular velocity of each motion while commanding the turn is shown in Fig. 6. The figure shows that turning is the fastest in motion B at an angular velocity of $6.4^{\circ}/\text{s}$ and the second fastest in proposed motion C at $3.2^{\circ}/\text{s}$. In motion A, turning is at $2.4^{\circ}/\text{s}$.

The number of slips, slip distance, and time required to run are shown in Fig. 7. It can be seen from Fig. 7a that the number of slips is 0.33 times for proposed motion C, which is the smallest value, 2.3 times for motion A, and 2.7 times for motion B. Figure 7b shows that the slip distance is 2.97 mm for proposed motion C, which is the smallest value, 5.90 mm for motion B, and 8.51 mm for motion A. It can be observed from Fig. 7c that the time required to run 2.74 s for motion A and proposed motion C, which is shorter than that for motion B, i.e., 2.96 s. Based on these results, proposed motion C (reaction force + difference) was determined to be the turning motion with the least slip and the fastest speed of climbing the stairs.

4.3 Evaluation of the Turning Motion on the Downward Stairs

The recorded movie and motion capture data were used to analyze the difference between the turning motions generated by motions A, B, and C on the downward stairs. Figure 8 shows the turning motion generated by motion C (reaction force +

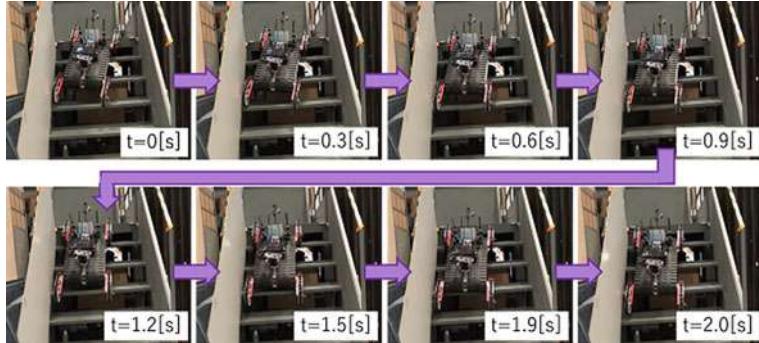
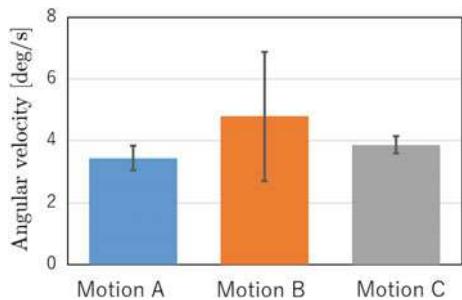


Fig. 8 Proposed turning motion, which uses reaction force from the wall while descending

Fig. 9 Turning angular velocities while descending;
A: Reaction force, B:
Differential, C: Reaction
force + differential



differential). The angular velocity of each motion while commanding the turn is shown in Fig. 9. The figure shows that the angular velocity is 4.8°/s for motion B, which is highest value, 3.9°/s for proposed motion C, which is the second highest value, and 3.4°/s for motion A.

The, number of slips, slip distance, and time required to run are shown in Fig. 10. It can be seen from Fig. 10a that the number of slips is 5 times for proposed motion C, which is the smallest value, 6 times for motion A, and 9 times for motion B. Figure 10b shows that slip distance is 38.9 mm for proposed motion C, which is the smallest value, 71.1 mm for motion A, and 82.7 mm for motion B. Figure 10c shows that the time required to run is 1.99 s for motion A, which is shorter than motion B and proposed motion C, i.e., 2.39 s. Based on these results, proposed motion C (reaction force + differential) was determined to be the turning motion with the least slip while descending.

4.4 Discussion

The results show that the proposed method (motion C: reaction force + differential) generated proper turning motion and reduced the number of slips and slip distance

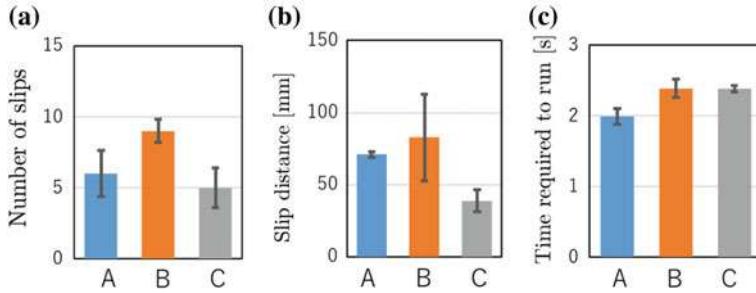


Fig. 10 Number of slips, slip distance, and time required to run while descending. A: Reaction force, B: Differential, C: Reaction force + differential

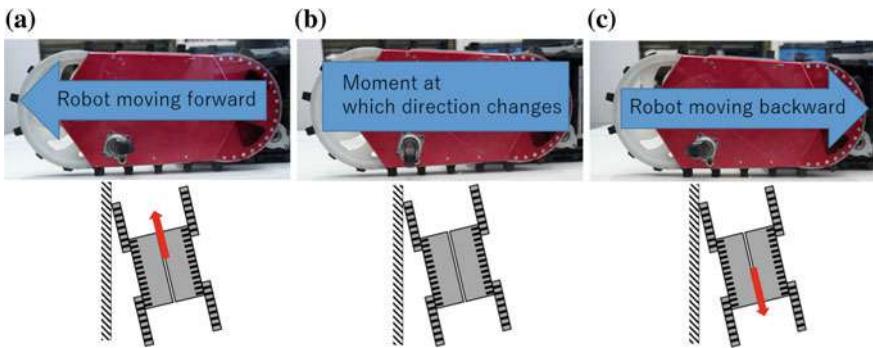


Fig. 11 Rotational characteristic of the caster: robot moving forward (a), moment at which direction of progression changes (b), robot moving backward (c)

while climbing up/down the stairs. A combination of reaction force and differential speed is a suitable solution for generating turning motion on the stairs.

There was considerable difference between the number of slips and total slip distance while climbing up and down the stairs. We analyzed the reason for this difference. We consider that the passive wheel attached to the sub-tracks causes this difference. The passive wheel consists of one wheel and one rotational axis, which is offset between the wheel and rotational axis. The rotational direction of the passive wheel changes depending on the motion of the tracked vehicle, as shown in Fig. 11. When the tracked vehicle slips while ascending, the direction of the wheel changes as Figs. 11a and b. At that instant, friction force is generated between the passive wheel and wall. This friction prevents the vehicle from slipping down on the stairs.

On contrary, when the tracked vehicle climbs down the stairs, the direction of the vehicle and slip is the same, and the direction of the passive wheel does not change. Therefore, friction force is not generated.

Based on these observations, it can be said that the passive wheel reduces the friction force between the sub-track and wall, and increases the friction during slippage when climbing up the stairs. This characteristic is considerably important for improving the turning motion when climbing up the stairs.

5 Autonomous Turning Motion on the Stairs

5.1 Implementation Method

To use the proposed motion for plant inspection or pilot assistance, turning motion is implemented on the tracked vehicle. To automate turning control, the control method requires the detection of contact and the measurement of the angle, θ , between the wall and the vehicle. The control flow, which decides velocity from detection of the contact and contact angle, θ , is shown in Algorithm 1.

Contact with the wall is detected using the roll angle of the main body and the electrical current value difference between the left and right main tracks' motors. It was experimentally determined that the current value of the main motor on the non-contact side increases and that on the contact side decreases upon contact with the wall during forward motion. Therefore, contact with the wall is detected when the current value difference is more than 4.5 A. This value is decided empirically. When running in a diagonal direction on the stairs, the roll angle of the robot is used to judge whether the robot is facing left or right with respect to the stairs (positive or negative roll angle). This can prevent the misjudgment of contact.

Algorithm 1 Decides velocity from detection of the contact and contact angle, θ .

```

1: loop
2:    $\{I_{L\_ave}, I_{R\_ave}\} \leftarrow$  Average of current value of {left, right} track motor for the past 5 times
3:    $\phi \leftarrow$  Roll angle of the robot from IMU
4:    $\theta_{ave} \leftarrow$  Average of contact angle for the past 5 times which obtained from point cloud data
5:   if ( $I_{L\_ave} - I_{R\_ave} < -I_{th}$ ) and  $\phi$  is negative then // Detect left side contact with the wall
6:     while  $|\theta| > \theta_{th}$  do //  $\theta_{th}$  is threshold of contact angle
7:       Decide the right side motor speed using Eqs.(7) and (8)
8:     end while
9:   else if ( $I_{L\_ave} - I_{R\_ave} > I_{th}$ ) and  $\phi$  is positive then // Detect right side contact with the wall
10:    while  $|\theta| > \theta_{th}$  do
11:      Decide the right side motor speed using Eqs.(7) and (8)
12:    end while
13:   else
14:     Set the both left and right side motor speed to default
15:   end if
16:   Command rotational speed to motor
17: end loop

```

The contact angle is detected using a 2D LIDAR(HOKUYO: UTM-30LX) attached to top of the vehicle (Fig. 1a). The point cloud data on the wall are extracted

from scan data, and the wall is detected using liner approximation. To reduce false detection, the point cloud data that are located between 0.3 and 1.0 m from the vehicle are used for wall detection. The contact angle, θ , is obtained from the wall detected using the point cloud data and the heading direction of the vehicle. Then, proper track velocity is obtained from the Eqs. (7) and (8). The rotational speed of the motor at the contact side is set as 2000 rpm. The track velocity at the non-contact side can be obtained. In addition, when the contact angle is less than 5° or the wall is not detected, the velocity at the non-contact side is not updated using Eqs. (7) and (8). This can reduce the improper turning motion.

5.2 Evaluation Method

Autonomous turning control is evaluated based on the comparison between the following two methods:

Reaction force based turning motion The tracked vehicle moves forward, without turning motion. The heading direction is changed by the reaction force caused by the forward motion.

Proposed turning motion The tracked vehicle turns when it detects contact with the wall. Turning motion is generated using the reaction force from the wall and the difference between the speeds of the left and right tracks.

During the evaluation, the vehicle starts running without contact with the wall. The running distance is three steps, which is equal to a height of 0.63 m from the start. Other conditions are the same as those described in Sect. 4. A slip is counted when the slip distance exceeds approximately 2.4 mm, which is 10% of the interval of the grossers in the vertical direction of the stairs.

5.3 Evaluation Result

Figure 12 shows the images of the motion generated by the proposed control method. The vehicle turns left using the reaction force from the wall. Figure 13 shows the rotational velocity of the motor at the third trial of the proposed control method. Figure 13 shows that the rotational velocity of the motor at the non-contact side automatically changes at 2.2 s. Turning motion is automatically generated by the proposed method based on the detection of contact and the contact angle.

Figure 14 shows the number of slips, slip distance, and time required to climb the stairs for both turning control methods. The proposed turning control method reduces slippage during autonomous control. The number of slips for the proposed method is 1.76 times smaller than that for the other control method (Fig. 14a). The slip distance for the proposed method is 2.31 times smaller than that for the other method

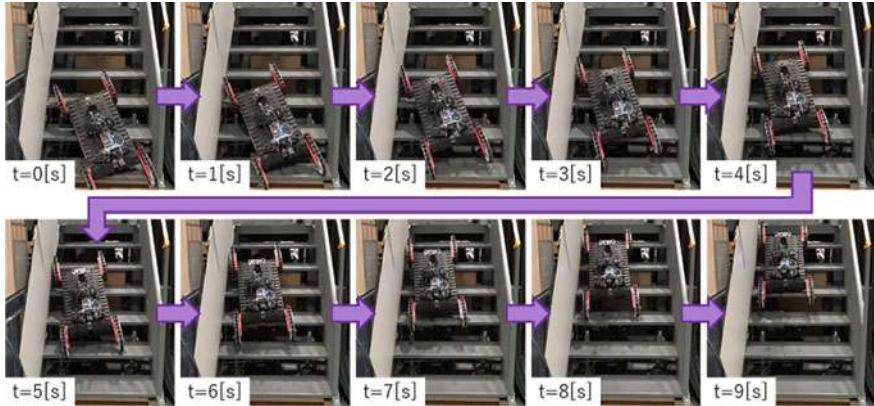


Fig. 12 Proposed automatic turning motion, which uses reaction force from the wall while climbing

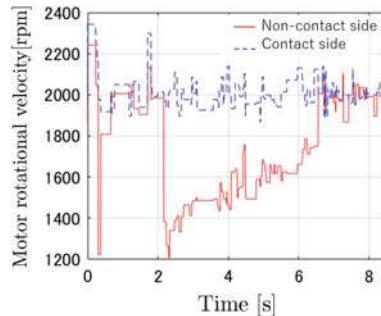


Fig. 13 Motor rotational velocity during the proposed turning motion which uses reaction force from the wall while climbing

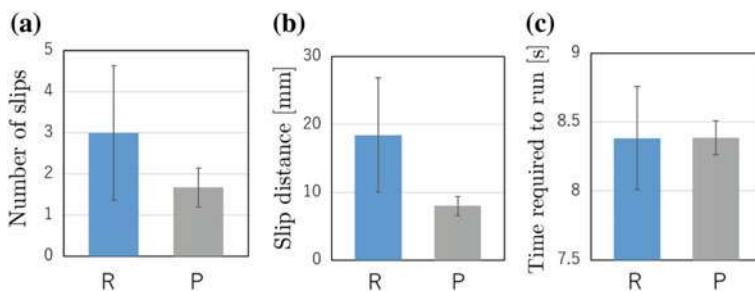


Fig. 14 Number of slips, slip distance, and time required to run while climbing. R: Reaction force based turning motion, P: Proposed turning motion

(Fig. 14b). The time required to climb the stairs is almost the same for both methods (Fig. 14c). These results show that proposed motion C (reaction force + differential) can generate proper turning motion and reduce slippage during the upward motion.

6 Conclusion

This paper proposed turning motion for a tracked vehicle on stairs. This turning motion was generated using the difference between the speeds of the left and right tracks and the reaction force from the safety wall. Movie and motion capture data were used to confirm that the proposed method generated proper turning motion on the stairs. In addition, the occurrence of slippage during the turning motion was evaluated. The results showed that the proposed method enables the vehicle to turn with less slippage compared to other methods (differential speed based, reaction force based). The proposed turning motion was more effective on the upward stairs than on the downward stairs. The passive-wheel caused less slippage during the upward motion because of the friction force between the passive-wheel and safety wall. Autonomous turning motion control was implemented on the tracked vehicle, and it was tested during the upward motion. It was observed that the proposed method enabled the vehicle to run with less slippage, as compared to the case in which turning motion was generated using only the reaction force from the wall.

Acknowledgements This research was supported by JST CREST Recognition, Summarization and Retrieval of Large-Scale Multimedia Data, Grant Number JPMJCR1403, Japan.

References

1. Rohmer, E., Yoshida, T., Ohno, K., et al.: Quince: a collaborative mobile robotic platform for rescue robots research and development. In: Proceedings of the 5th International Conference on Advanced Mechatronics: Toward Evolutionary Fusion of IT and Mechatronics, pp. 225–230 (2010)
2. Fox, D., Wolfram Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. *IEEE Trans. Robot. Autom.* **4**(1), 23–33 (1997)
3. Borenstein, J., Koren, Y.: Real-time obstacle avoidance for fast mobile robots. *IEEE Trans. Syst. Man Cybern.* **19**(5), 1179–1187 (1989)
4. Fox, D., Burgard, W., Thrun, S., et al.: A hybrid collision avoidance method for mobile robots. In: Proceedings of 1998 IEEE International Conference on Robotics and Automation vol. 2, pp. 1238–1243 (1998)
5. Fernández, J.L., Sanz, R., Benayas, J.A., et al.: Improving collision avoidance for mobile robots in partially known environments: the beam curvature method. *Robot. Auton. Syst.* **46**(4), 205–219 (2004)
6. Minguez, J., Montano, L.: Nearness diagram (ND) navigation: collision avoidance in troublesome scenarios. *IEEE Trans. Robot. Autom.* **20**(1), 45–59 (2004)
7. Mason, M.T.: Compliance and force control for computer controlled manipulators. *IEEE Trans. Syst. Man Cybern.* **11**(6), 418–432 (1981)

8. Salisbury, J.K.: Active stiffness control of a manipulator in Cartesian coordinates. In: Proceedings of 1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes, vol. 19, pp. 95–100 (1980)
9. Kim, K.S., Kwok, A.S., Thomas, G.C., et al.: Fully omnidirectional compliance in mobile robots via drive-torque sensor feedback. In: Proceedings of 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4757–4763 (2014)
10. Rude, M.: A flexible, shock-absorbing bumper system with touch-sensing capability for autonomous vehicles. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2, pp. 410–417 (1996)
11. Kojima, S., Ohno, K., Suzuki, T., et al.: Motion control of tracked vehicle based on contact force model. In: Proceedings of 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1177–1183 (2016)

Part II

Computer Vision

Finding Better Wide Baseline Stereo Solutions Using Feature Quality

Stephen Nuske and Jay Patravali

Abstract Many robotic applications that involve relocalization or 3D scene reconstruction, have a need of finding geometry between camera images captured from widely different viewpoints. Computing epipolar geometry between wide baseline image pairs is difficult because often there are many more outliers than inliers computed at the feature correspondence stage. Abundant outliers require the naive approach to compute a huge number of random solutions to give a suitable probability that the correct solution is found. Furthermore, large numbers of outliers can also cause false solutions to appear like true solutions. We present a new method called UNIQSAC for giving weights for features to guide the random solutions towards high quality features, helping find good solutions. We also present a new method to evaluate geometry solutions that is more likely to find correct solutions. We demonstrate in a variety of different outdoor environments using both monocular and stereo image-pairs that our method produces better estimates than existing robust estimation approaches.

1 Introduction

Computing the relative geometry between a wide-baseline camera pair is an important task for many robotics problems such as loop-closure in SLAM systems or 3D reconstructions. The task can be difficult in challenging real-world scenes, mainly because it is hard to determine which point correspondences are correct (inliers) among a set containing many incorrect correspondences (outliers).

The standard approach, RANSAC [7], is to robustly estimate an epipolar geometry between the pair by testing multiple geometry hypotheses. RANSAC and its improved variants [2, 3, 6, 13, 16], are impressive in their ability to correctly estimate

S. Nuske · J. Patravali (✉)

Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh,

PA 15213, USA

e-mail: jaypatravali@cmu.edu

S. Nuske

e-mail: nuske@cmu.edu

geometry given small percentages of true correspondences and large percentages of false point correspondences. However, when the percentages of inliers and outliers are pushed further out of favor then even these robust approaches struggle to find the correct geometry. To that end, we propose a new metric for the quality of the features, which is their uniqueness amongst all other features in the image. We use this metric to weight the sampling of correspondences to help find good solutions, and we also present a new method to evaluate solutions that can better differentiate between true and false solutions. Figure 1 shows an example of our algorithm’s functioning.

Our sampling approach is different to existing weighted sampling strategies in that we first robustly quantize each feature and then calculate its uniqueness among all other features in the image, as opposed to just a feature correlation score to the nearest neighbor or a ratio of correlation scores to the second nearest neighbor.

We also propose to use feature quality to evaluate geometry solutions. The traditional approach to evaluate solutions is to just use the count of supporting correspondences that pass an epiline distance threshold. With many outliers present, many false correspondences can pass the inlier test causing bad solutions to appear as good solutions. Our approach is more likely to find correct solutions because we have down weighted low-quality features whereas the traditional approach treat all features equal.

We first test our approach on a number of challenging outdoor monocular image-pairs, each with a very small fraction of inliers in the set of correspondences. The results are evaluated against manual ground-truth and compared against other robust estimation algorithms. The result is that, in comparison to the commonly used approaches, our approach finds more accurate solutions.

Secondly, we apply our uniqueness weighting approach for faster and accurate estimations in wide-baseline stereo visual odometry. Finally, the paper is organized by presenting the methodology, following which the results for monocular and stereo image-data are described in consecutive sections.

2 Related Work

Here we present an overview of state of the art in estimation of wide baseline geometry. The task ultimately requires correlating some type of visual information between the image-pair. In the literature there are a number of different categories of information that are used for wide baseline; region based [12], edge/line based methods [9], but by far the most prevalent methods used are those using point features such as SIFT [10]. Appearance based methods, such as [8] learn to predict matchable descriptors, attaining faster geometry computation with high success rates of matching. The point-based methods are popular because firstly they have algorithms to repeatedly find the same image keypoints in a scene even when the viewing pose changes, and secondly they can form descriptions of the area surrounding the keypoints that have a reasonable level of robustness to viewing changes.

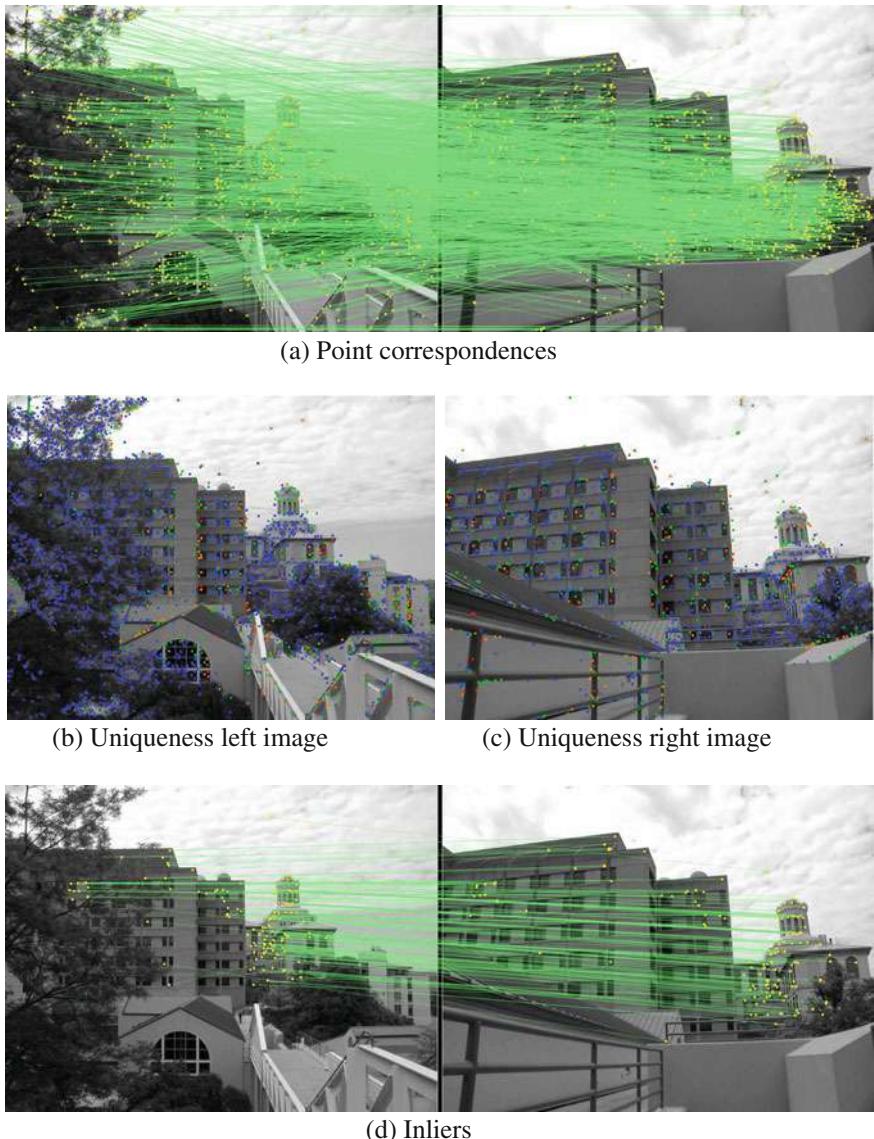


Fig. 1 Example of weighted epipolar geometry estimation: Campus Scene. The feature uniqueness coloring ranges from *unique* to *repetitive* as follows; red, orange, green, light-blue, blue. The low-quality repetitive foliage and many repetitive building features are downweighted

However, even though the process is comparatively robust, there are often more outliers correspondences between a wide baseline image pair than inliers. The most prevalent and successful approaches to dealing with cases where the outliers are above 50% of all correspondences is RANSAC [7] and its variants. The RANSAC process is to form a hypothesized solution to the geometry from a randomly selected minimal sample of point correspondences and evaluate the hypothesis by counting correspondences in agreement. The process is repeated with different samples until a good solution is found. To achieve higher accuracy, Locally Optimized RANSAC (LoRANSAC) [3] locally improves the estimated model hypothesis when the best model so far is found. For faster convergence, [15] precludes computation of incorrect feature matches using spatial order information in images.

There are approaches to improve RANSAC at the sampling step, where each correspondence is weighted and the likelihood of it being chosen in the sampling is adjusted [2, 6, 13, 16]. SCRAMSAC [13] achieves this by ranking correspondences based on the spatial consistency of neighboring correspondences. USAC [2] combines [2, 3, 11] into a single comprehensive pipeline.

The work in [16] replaces random sampling by guided sampling, computing a weight for every correspondence based on an image correlation score. PROSAC [2] uses correspondence quality measures to speed up RANSAC based on the correlation score of keypoints. In EVSAC [4], hypothesis generation is accelerated by computing confidence values by modelling the matching scores. BEEM [6] presents another method of weighting correspondences by a correlation quality score determined by the ratio of distance of nearest to second nearest feature in the descriptor-space.

A majority of these existing approaches calculate weights from the correlation score between two features or the ratio of correlation scores to the next best correspondence. These existing weighting metrics are formed considering just one or two correspondences. This will result in a low weighting for semi-distinct features, and this weighting will be as low as the weighting for very non-distinct features. Whereas our approach is different in that it calculates the uniqueness score of a feature against all other features in the image, and will provide an appropriate weight for semi-distinct features that are similar to only a few other features. Thus our uniqueness weighting is less susceptible to the case of semi-distinct features in the scene, which are features that are similar to a small number of other features.

Furthermore the convention in these approaches is to use the number of correspondences in the consensus set that have passed an epiline distance threshold. However, when there are many outliers, many false correspondences can mistakenly pass the inlier test causing bad solutions to appear as good solutions. In our work we evaluate solutions based on the feature quality in the consensus set, which is more likely to find correct solutions because we have down weighted low-quality features whereas the traditional approach treat all features equal.

3 Sampling Using Feature Uniqueness: UNIQSAC

We first detail our approach to weight the sampling of correspondence subsets for the RANSAC procedure. We reiterate the necessity for weighting the sampling of correspondences to achieve timely convergence in difficult image pairs as previously demonstrated in [2, 6, 16] and propose an alternative strategy for determining the weights. The first step is to quantize the feature descriptors into distinct labels, similar to the bag-of-words approach [14]. We count how many times each type of feature appears in an image, giving us an estimate of that feature’s uniqueness. The bag-of-words approach requires a large prior database relevant to the current scene at hand and a large amount of preprocessing to cluster the database of features. We propose an alternative feature-quantization approach that requires no learning and is efficient.

Our approach is to first take a N -dimensional feature descriptor and randomly sample M -dimensional sub-features from it K times. Then, we quantize these K sub-features into K integers. The reason for a set of identifiers for each feature is that quantizing large N -dimensional features is difficult and error prone, because if any one dimension is disrupted (from any number of potential viewing disruptions; lighting, sensor noise, viewing pose changes, non-planarity, partial-occlusions etc.) then the resulting integer identifier will be wrong. So by choosing an appropriate value for M and K , the resulting set of K integers will still each on their own be a reasonable means for identifying a feature, and even though it is still likely that a number of the K integer identifiers will be erroneous because of viewing disruptions, it is likely that there will be enough identifiers for each feature that are still valid.

The procedure is as follows. Let \mathbf{f}_i be the value of the i th dimension of feature, \mathbf{f} , which is a normalized value between 0 and 1. Then let $r(k, m, N)$ be a function that returns a random hasher between 1 and N , which will return the same random hasher for specific k and m values (i.e. it will pick the same dimensions to sample for each feature we are quantizing). Given a feature, \mathbf{f} , we randomly generate a sub-feature, \mathbf{s} , several times to form the set of sub-features, \mathbf{S} , where \mathbf{s}_j is the j th sub-feature. We then quantize the sub-features into an integer identifier, d , with the quantization function $q(v, p, z)$, which takes a feature value v between 0 and 1, an exponent value, p , and the number of quantizations for each dimension, z ;

$$q(v, p, z) = \lfloor v^p z \rfloor \quad (1)$$

Combining the sub-features we form a set of integer identifiers, \mathbf{D} , for each feature. Here \mathbf{D}_l is the l th integer identifier. This process is defined as the QUANTIZE() function, given in Algorithm 1. Therefore by creating K integer identifiers for each feature descriptor, \mathbf{f} , we can count each time one of these identifiers appears in an image, $C(d)$, compute the most commonly appearing feature identifier, C_{MAX} , and then compute an average over all K identifiers to compute a score of how unique that type of feature is amongst the other features in that image, U_f . This details our method for calculating the uniqueness scores of a set of feature descriptors.

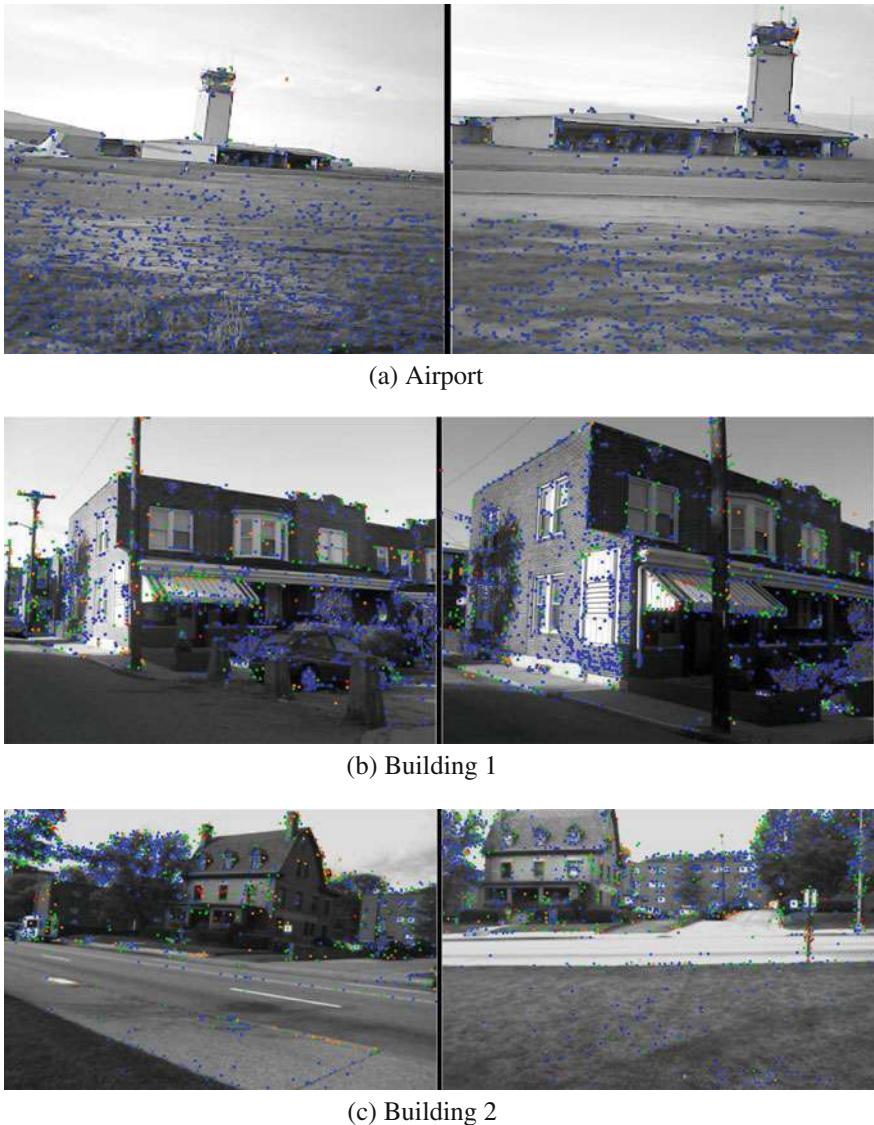


Fig. 2 Visualization of Uniqueness Computation. The calculated uniqueness are colored for the features detected in a set of challenging outdoor image pairs. Coloring ranges from *unique* to *repetitive* as follows; red, orange, green, light-blue, blue. The airport scene noticeably has many low-quality ground features downweighted by our technique. Building scenes have many repetitive low-quality features from brickwork and foliage. Whereas most of the high-quality features have been correctly identified and given higher weights

Algorithm 1 Uniqueness Sampling and Consensus (UNIQSAC) Algorithm.

Input: Set of features \mathbf{F}	function QUANTIZE(\mathbf{f})
Output: Set of uniqueness values \mathbf{U}	for $k = 0$ to K do
forall \mathbf{f}_k in \mathbf{F} do	$d = 0$
$\mathbf{D} = \text{QUANTIZE}(\mathbf{f})$	for $m = 0$ to M do
$\mathbf{U}_\mathbf{f} = \sum_{k=0}^K \frac{1-C(\mathbf{D}_k)}{C_{\text{MAX}}}$	$i = r(k, m, N)$
	$\mathbf{s}_m = \mathbf{f}_i$
	$d = d + q(\mathbf{s}_m, e, z)z^m$
	$\mathbf{D}_k = d$
	end function

Figure 2 provides a visualization of uniqueness computation in three monocular wide-baseline settings. The coloring follows a *heat-map* color scheme, where from red is a unique feature and blue is a frequently occurring feature. Here it is clearly identified the features are that are unique and which are low-quality. In the airport scene it is noticeable that most of the ground features have been correctly down-weighted as they are low-quality features, and many of the features on the hangers and tower have correctly been identified as the useful *unique* features. In the building scenes, most of the foliage features and many of the features on the brickwork, repeated windows have been suitably down-weighted.

4 Evaluating Hypotheses Using Feature Uniqueness

The previous section described the sampling stage of geometry estimation, the next stage of the RANSAC framework is to evaluate the resulting model hypothesis. Traditionally, almost all methods use the cardinality of the consensus set [7] (inlier set), \mathbf{c} , as the metric to decide the quality of the model, which has limitations in being able to determine good hypotheses from bad.

The common approach is to test if a feature correspondence, i , is in consensus and therefore an *inlier*, is if its distance, ϵ , away from the epipolar line computed from the n th model is less than the threshold, τ :

$$\mathbf{c}_n = \forall \mathbf{c}(i) \in \mathbf{C}, \epsilon_i < \tau \quad (2)$$

Then given N RANSAC iterations, the final estimated model, \hat{m} , is simply the model with the largest consensus set as follows:

$$\hat{m} = \arg \max_n (|\mathbf{c}_n|) \quad (3)$$

where \mathbf{c}_n is the consensus set from the n th RANSAC iteration. However, there can be many correspondences that pass the inlier test that are in fact outliers. Therefore, there can be poor geometry hypotheses that appear good because of many false correspondences in the consensus set.

As a result of the limitations of using just the cardinality of the consensus set as the evaluation metric, we have developed a new metric to evaluate RANSAC hypotheses, that is a measure of the quality of the features in the consensus set. Our approach is less likely to find false solutions because we down-weight low-quality features in the evaluation process, whereas the traditional approach treats all features as equal. We propose to use feature weighting approach described in the previous section as a measure of quality, and use the sum of the inliers' feature weights as the metric for hypothesis evaluation. So our formulation for the best model becomes:

$$\hat{m} = \arg \max_n \left(\sum_{\forall \mathbf{f}_i \in \mathbf{c}_n} w(\mathbf{f}_i) \right) \quad (4)$$

Now we plot two graphs comparing our new metric against conventional inlier count as shown in the Fig. 3. The first graph Fig. 3a, shows the correlation between size of the consensus set (inliers) and the actual true inliers (validated inliers), recorded for 1,000,000 iterations of RANSAC (computing 1,000,000 iterations would most likely take too long in practice, but used here to give a clear picture of the relationship to the true number of inliers) on the wide baseline image pair seen in Fig. 2a. Similarly, the second graph in Fig. 2b shows the correlation between summation of inlier weights and validated inliers.

It is obvious from Fig. 3b, that the new metric produces a far better correlation to true inliers when compared to the conventional inlier count. Although we see an upwards trend in Fig. 3a, relating number of inliers to number of validated inliers,

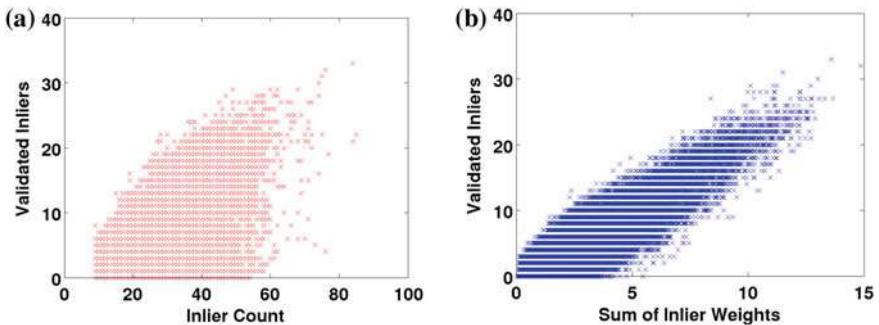


Fig. 3 Hypothesis Evaluation and Accuracy score for a million iterations. **Left:** At higher inlier counts, RANSAC is inaccurate with few validated inliers. **Right:** Feature Uniqueness provides a linear correlation between summation of inlier weights and validated inliers

there is an alarming variance in this distribution, where some hypotheses with many overall inliers have very few validated inliers. This illustrates, that our new metric is far less likely at wrongly selecting a poor hypothesis.

5 Monocular Wide-Baseline Geometry Estimation

In this section, we provide a detailed description of implementing feature uniqueness weighting for sampling correspondences in a RANSAC-type geometry estimation for wide-baseline monocular images. The correspondence set, \mathbf{C} , is the set of matched features between \mathbf{F}_1 and \mathbf{F}_2 , the two features sets from the image pair. From this correspondence set, we take a minimal subset, \mathbf{s} , of the entire correspondence set using a weighting function that computes the likelihood of \mathbf{f} being sampled, $w(\mathbf{f})$:

$$w(\mathbf{f}) = \mathbf{U}_{\mathbf{f}}^2 \quad (5)$$

The minimal sample set is used to estimate our hypothesized model which for us estimating epipolar geometry is the fundamental matrix. The cardinality of \mathbf{s} being 8, because we are using the 8-point algorithm [7]. When selecting the 8 point-correspondences of \mathbf{s} from \mathbf{C} we use a Monte-Carlo sampling strategy based on the weights of the two features $\mathbf{f}_1(i)$ and $\mathbf{f}_2(i)$ of a specific correspondence, i :

$$p(\mathbf{C}(i)) = w(\mathbf{f}_1(i))w(\mathbf{f}_2(i)) \quad (6)$$

6 Experimental Results: Monocular Image-Data

We collected a variety of particularly challenging wide baseline image pairs in a number of different outdoor settings to evaluate our approach. Here we present results collected from four different scenes, shown previously in Figs. 1 and 2.

We evaluate our UNIQSAC approach against three standard epipolar geometry estimation methods; RANSAC [7], LoRANSAC [2] and BEEM [6]. All these methods sample a small number of sparse feature point correspondences, fit a epipolar geometry model, evaluate the model, store if it is the best found and iterate. As an input, we use the SIFT features [10] for keypoint detection and SIFT feature descriptor for each of these keypoints.

The parameters for feature quantization, given in Eq. 1, are set as per the following: Dimension of sub-feature M as 6, Number of sub-features K as 6, Dimension of feature N as 128, Exponent p as 1, Number of quantization z as 2. These parameters are set for a coarse quantization making it suitable to distinguish frequent features from the more unique features. Taking the quantizations and computing a uniqueness for each feature according to Algorithm 1, yields a visualization of uniqueness shown in Fig. 2.

Table 1 Epipolar geometry estimation results

Image-pair	Statistic	RANSAC	LoRANSAC	BEEM	UNIQSAC
Airport	Correspondences	1237	1237	1237	1237
	Inliers	82.5	98.0	102.4	96.66
	Validated inliers	22.3	29.9	36.4	43.2
	Epiline error (pix)	20.0	11.4	4.7	4.1
Building 1	Correspondences	1204	1204	1204	1204
	Inliers	187.7	214.5	216.9	217.0
	Validated inliers	88.9	113.1	124.5	125.8
	Epiline error (pix)	7.9	5.4	5.2	3.6
Building 2	Correspondences	1192	1192	1192	1192
	Inliers	146.6	181.3	177.7	182.5
	Validated inliers	87.8	116.0	115.3	125.0
	Epiline error (pix)	4.1	2.8	2.3	1.7
Campus	Correspondences	1451	1451	1451	1451
	Inliers	143.7	157.2	162.8	175.7
	Validated inliers	103.3	121.6	130.7	128.7
	Epiline error (pix)	14.9	5.9	8.0	4.5

Once uniqueness has been computed, the next step is to find point correspondences. We compute the correspondences between the pair using a KD-tree [1] (using 250 as the maximum number of leaves to visit). In each of the image pairs the number of inliers is low, somewhere between 10 and 15%. We evaluate the performance of the algorithm by manually designating a set of groundtruth correspondences (usually around 30) between image pairs. We compute a *groundtruth* fundamental matrix using the standard 8-point algorithm and use this fundamental matrix to validate how many of the inliers found also agree with the *groundtruth* fundamental matrix (we call these the *validated inliers*). We also compute an epipolar line from the estimated fundamental matrix and compute the mean distance of the *groundtruth* correspondences to this line (we call this the *epiline error*).

To maintain a fair comparison we fix the number of iterations for all algorithms to the same number: 50,000. For LoRANSAC we use an inner loop of 5,000 iterations, where each inner iteration contributes to 50,000 iterations in total. We run 50,000 iterations for each algorithm, 10 times on each image-pair, compute the mean and present these in Table 1.

We highlight in bold the best performing algorithm for the number of validated inliers and the *groundtruth* epiline error. Overall our *uniqueness* approach to weighted sampling performs better than all other algorithms, as it provides more accuracy in terms of epiline distance and has more validated inliers.

We can see that in some of the tests that some of the algorithms produce more inliers, however as we have discussed throughout this paper, the number of inliers

does not necessarily mean a better solution (as false correspondences can be counted as inliers). The important statistics are the number of validated inliers and the epiline distance.

7 Wide-Baseline Stereo Camera Pose Estimation

In this section we extend the monocular method to solve geometry between sets of stereo pairs. The results presented in previous section provides significant motivation to apply our Uniqueness Sampling and Consensus (UNIQSAC) technique to compute a 6 Degree-of-Freedom pose for a wide-baseline stereo camera.

Our Stereo image dataset comprises of plant images captured in a wide-baseline stereo setup from Grape Vineyards, Sorghum Fields and Apple Orchards. Due to repetitive features present in plant foliage and background clutter, sampling unique features becomes essential in computing motion estimates accurately within a feasible number of iterations. To compute a complete 6 DoF motion, we implement a stereo visual odometry pipeline consisting of three stages: Feature Extraction and Matching, Uniqueness score computation and Geometry estimation.

7.1 *Feature Extraction and Matching*

With a moving stereo camera setup, we obtain four images at any given time: left and right images of two consecutive frames. To compute camera transformation, we require feature matches between all four images. Starting from Previous Left image, we look for correspondences in Current Left Image. After establishing correspondences between Previous and Current Left image pair, we utilize these matched features as templates to search for corresponding features in the Previous and Current right images along the epipolar scanlines. The template image matching uses a Normalized Cross Correlation Score. For feature matching process, keypoints are detected using SIFT features. Subsequently, SIFT descriptors are extracted and matched using K-nearest neighbor technique.

7.2 *Uniqueness Score Computation*

Before computing the uniqueness score, features are quantized keeping the same parameters as described in the earlier section. In addition to the Uniqueness score U_{score} , we also consider depth consistency and distance-ratio of matched features in form of a new hybrid score T_{score} . This hybrid score is used to assign weights to all the matched features as given in the Eq. 7. Depth consistency D_{score} is computed by calculating the differences in depth values from Previous Left- Current Left and

Current Left- Previous Left, and then normalizing the difference values. The distance-ratio R_{score} of matched features is determined by a process described in [6].

$$\mathbf{T}_{score} = U_{score} + D_{score} + R_{score} \quad (7)$$

7.3 Geometry Estimation

Once we have pixel coordinates of all matched features, we can apply triangulation to acquire two sets of 3D coordinates. The camera motion (r, t) can be computed by minimizing the sum of reprojection errors using a Gaussian-Newtonian optimization. Our motion estimation scheme from sparse feature matches is similar to the method presented by Geiger et. al [5]. Instead of randomly drawing 3 correspondences, we replace RANSAC with our Uniqueness sampling technique to increase the probability of selecting high quality features. For testing model hypothesis, we use our metric given in Eq. 4 to obtain the final inlier set.

8 Experimental Results: Stereo Image-Data

To test the performance in grape and sorghum image dataset, we consider both time-efficiency and accuracy for evaluation. Since the run-time for execution of one cycle of an algorithm can be relative, a good way to measure the time-efficiency is to check the number of iterations the algorithm takes to converge.

In our setup, the robot is equipped with a stereo camera moving at an average velocity of 0.45 m/s, capturing images at 5 frames/sec facing a plantation wall at a distance of 3 feet. Figure 4 illustrates our complete Stereo Visual Odometry pipeline for the grape dataset Fig. 4a and sorghum image dataset Fig. 4b. We can see that the raw images have considerable portions that will yield low quality features—ground, sky and plant foliage. As a result, many feature correspondences are incorrect. Computing a hybrid Uniqueness score downweights a majority of features extracted from ground and plant foliage. Only a tiny fraction of these correspondences are correctly identified as inliers (inlier ratio <9%). Similar results from Apple orchard dataset are illustrated in Fig. 5.

Our algorithm UNIQSAC is tested against the three standard methods: RANSAC, LoRANSAC and BEEM. The metric for evaluation is the validated inlier score (total validated inliers), and computed for 1000 and 10,000 iterations. To ensure robustness and repeatability, the scores are averaged over 1000 times. The results presented in Fig. 6 show that our proposed technique provides 25% (Grape) and 80% (Sorghum) improvement in Accuracy-Time Efficiency as compared to RANSAC.

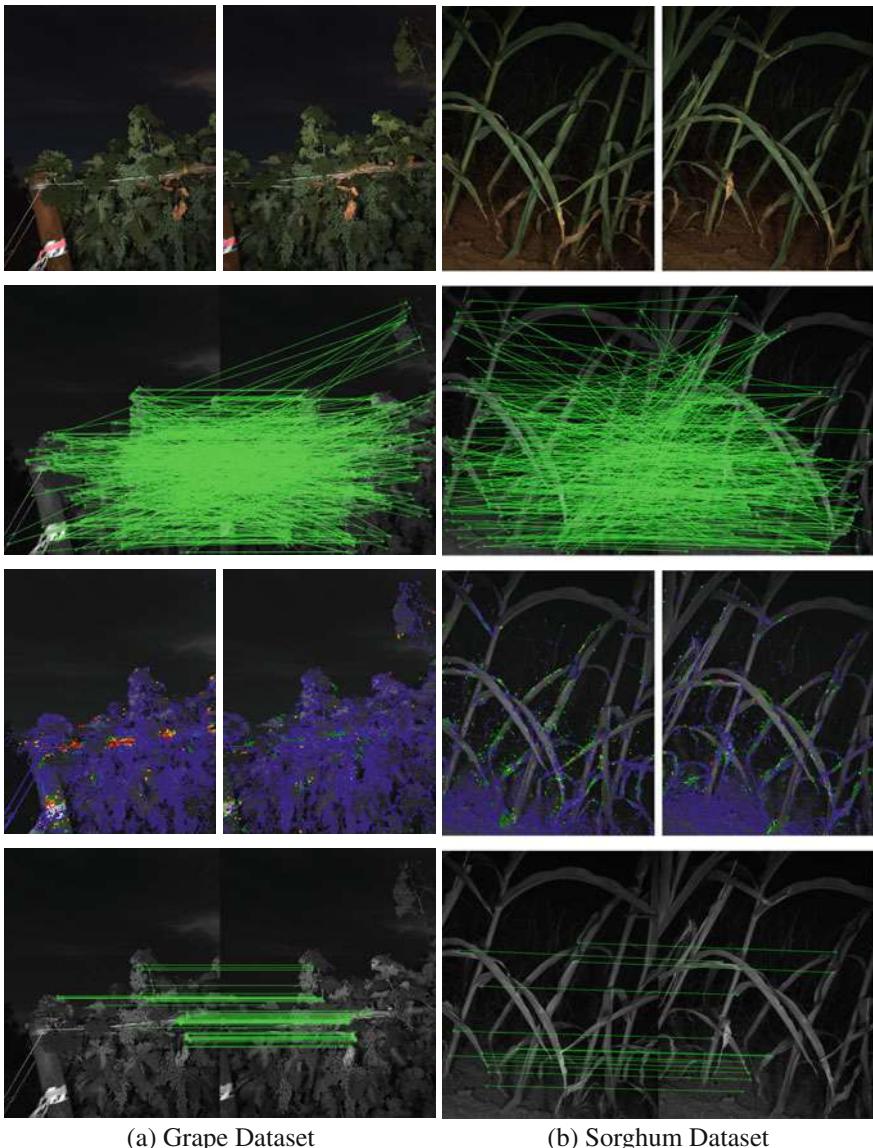


Fig. 4 Wide-baseline stereo geometry estimation. An example of wide-baseline geometry estimation on stereo image-pairs. Illustrations on the left column are obtained from the Grape dataset and right column from the Sorghum dataset. **First Row:** Raw image-pairs as captured from the Stereo Camera. **Second Row:** Correspondence Matching using K-nearest neighbors, between Previous Left and Current Left Image. **Third Row:** Feature uniqueness computation and colored heat map for Previous Left and Current Left Image. **Fourth Row:** Plotted Inliers are obtained using our Uniqueness sampling technique

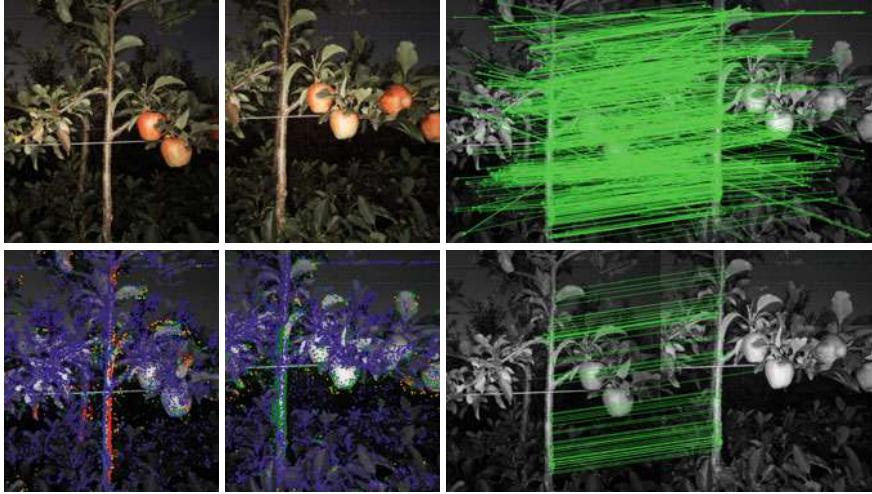


Fig. 5 Apple Orchard Dataset. **First Row-Left to Right:** Raw image-pairs, Correspondence Matching using K-nearest neighbors. **Second Row-Left to Right:** Uniqueness computation as colored heat maps, plotted inliers using UNIQSAC, between Previous Left and Current Left Image

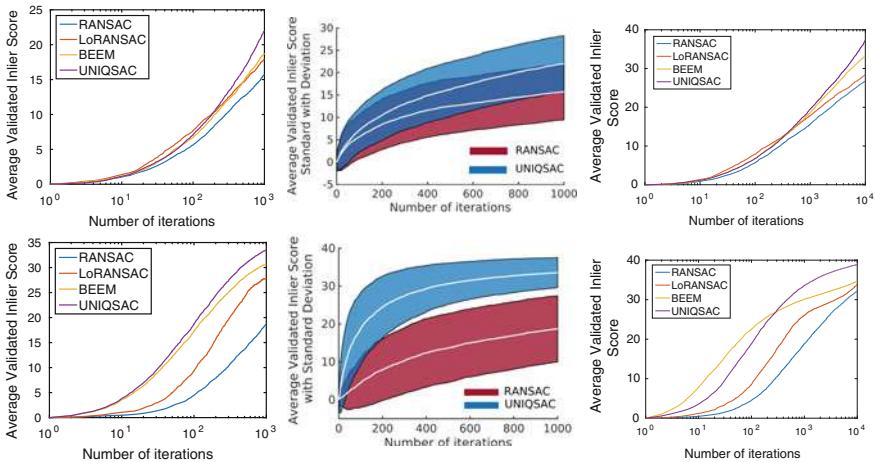


Fig. 6 Time Efficiency-Accuracy Score. Each figure in top row left-to-right are results acquired from the Grape Dataset. **Left:** Comparison for 1000 iterations. **Centre:** Comparison for RANSAC versus UNIQSAC, with standard deviation from the mean validated inlier score. **Right:** Comparison for 10,000 iterations. Similarly, each figure in the bottom row left-to-right, are results obtained from the Sorghum Dataset

9 Conclusion

This paper presented a new weighted sampling method called UNIQSAC, based on feature quality for computing epipolar geometry between wide baseline image pairs. Also presented is a new metric for evaluating model hypothesis based on the feature quality of the consensus set that performs better than simply counting the cardinality of the consensus set. Together, these two methods were demonstrated in a variety of different outdoor environments, where the low-quality repetitive features are correctly down-weighted and the more unique structural features are correctly up-weighted, resulting in more accurate solutions.

To evaluate the performance of our methods, a set of monocular and stereo image-pairs was ground-truthed and the results were compared against standard robust estimation approaches like RANSAC, LoRANSAC, and BEEM. It was demonstrated in monocular settings, that our methods produces in all but one case the most accurate estimates. These results are further validated in the stereo settings, where we acquire the most accurate and time-efficient estimates while operating at very low inlier ratios (<9%). Utilizing feature quality to guide random samples, opens up several possibilities for future work. In particular, we are interested to extend our methods to obtain faster and accurate registration of dense 3D point clouds.

Acknowledgements This work is supported under award numbers; USDA 2015-51181-24393, USDA 2016-67021-24535 and ARPA-e 1830-219-2020937.

References

1. Beis, J.S., Lowe, D.G.: Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In: 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997. Proceedings, pp. 1000–1006. IEEE (1997)
2. Chum, O., Matas, J.: Matching with prosac-progressive sample consensus. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, pp. 220–226. IEEE (2005)
3. Chum, O., Matas, J., Kittler, J.: Locally optimized ransac. In: Joint Pattern Recognition Symposium, pp. 236–243. Springer (2003)
4. Fragoso, V., Sen, P., Rodriguez, S., Turk, M.: Evsac: accelerating hypotheses generation by modeling matching scores with extreme value theory. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2472–2479 (2013)
5. Geiger, A., Ziegler, J., Stiller, C.: Stereoscan: Dense 3d reconstruction in real-time. In: Intelligent Vehicles Symposium (IV) (2011)
6. Goshen, L., Shimshoni, I.: Balanced exploration and exploitation model search for efficient epipolar geometry estimation. IEEE Trans. Pattern Anal. Mach. Intell. **30**(7), 1230–1242 (2008)
7. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer vision. Cambridge University Press (2003)
8. Hartmann, W., Havlena, M., Schindler, K.: Predicting matchability. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9–16 (2014)
9. Lee, J.A., Yow, K.C., Chia, A.Y.S.: Robust matching of building facades under large viewpoint changes. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 1258–1264. IEEE (2009)

10. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **60**(2), 91–110 (2004)
11. Matas, J., Chum, O.: Randomized ransac with sequential probability ratio test. In: Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005, vol. 2, pp. 1727–1732. IEEE (2005)
12. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. *Image Vis. Comput.* **22**(10), 761–767 (2004)
13. Sattler, T., Leibe, B., Kobbelt, L.: Scramsac: improving ransac’s efficiency with a spatial consistency filter. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 2090–2097. IEEE (2009)
14. Sivic, J., Zisserman, A.: Video google: efficient visual search of videos. In: Toward Category-level Object Recognition, pp. 127–144. Springer (2006)
15. Taler, L., Moses, Y., Shimshoni, I.: Using spatial order to boost the elimination of incorrect feature matches. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1809–1817 (2016)
16. Tordoff, B., Murray, D.W.: Guided sampling and consensus for motion estimation. In: European Conference on Computer Vision, pp. 82–96. Springer (2002)

High-Throughput Robotic Phenotyping of Energy Sorghum Crops

Srinivasan Vijayarangan, Paloma Sodhi, Prathamesh Kini,
James Bourne, Simon Du, Hanqi Sun, Barnabas Poczos, Dimitrios
Apostolopoulos and David Wettergreen

Abstract Plant phenotyping is a time consuming, labour intensive, error prone process of measuring the physical properties of plants. We present a scalable robotic system which employs computer vision and machine learning to phenotype plants rapidly. It maintains high throughput making multiple phenotyping measurements during the plant lifecycle in plots containing thousands of plants. Our novel approach allows scanning of plants inside the plant canopy in addition to the top and bottom section of the plants. Here we present our design decisions, implementation challenges and field observations.

S. Vijayarangan (✉) · P. Sodhi · P. Kini · J. Bourne · S. Du · H. Sun · B. Poczos ·
D. Apostolopoulos · D. Wettergreen
Carnegie Mellon University, Pittsburgh, USA
e-mail: srinivasan@cmu.edu

P. Sodhi
e-mail: psodhi@andrew.cmu.edu

P. Kini
e-mail: pkini@andrew.cmu.edu

J. Bourne
e-mail: jbourne@nrec.ri.cmu.edu

S. Du
e-mail: ssdu@andrew.cmu.edu

H. Sun
e-mail: hanqis@andrew.cmu.edu

B. Poczos
e-mail: bapoczos@cs.cmu.edu

D. Apostolopoulos
e-mail: da1v@cs.cmu.edu

D. Wettergreen
e-mail: dsw@cmu.edu

1 Introduction

Biofuels are a significant source of renewable energy. However, only limited amounts of biofuels are produced from non-grain feedstocks because production costs are not competitive. Increased yield of bioenergy crops would mitigate this barrier to utilization. While crop breeding and improved management practices have increased productivity, the rate of yield gain in most crops is 1–2% per year with very significant investment. The ability to correlate phenotypic traits with their genotypes plays a crucial role in improving plant breeding techniques. Phenotyping is the bottleneck in the plant breeding pipeline and high throughput automated methods are crucial to improved production [1, 2]. We have created our high-throughput robotic phenotyping system which is capable of measuring a comprehensive set of phenotypes associated with traits that impact biomass yield with high accuracy and repeatability (Fig. 1).

Plant phenotyping is the quantitative assessment of plant traits like physiology, yield, etc. Currently, plant phenotyping is a manually intensive, slow, error prone



Fig. 1 High-throughput phenotyping robot in a sorghum field in Weslaco, Texas. Sorghum grows 6 m tall in 4 months

process which involves humans making measurements in the field or greenhouse. In most cases, the phenotyped data is analyzed post-season. However with our high-throughput system, we can collect plant phenotype data much faster and accurately. We can also collect more measurements compared to the current manual methods, which helps to increase the accuracy and completeness of the estimates. Unlike some manual methods, our system is non intrusive which means we can scan the plant multiple times during its life cycle and not affect its development. Energy sorghum is used in this program because it is a highly productive, annual, drought tolerant C4 grass with an excellent genetic (diploid, inbreeding) and genomics platform [3, 4].

Our main contribution is the design and development of a high throughput plant phenotyping robot which was deployed in the field to measure phenotype traits associated with energy crop traits that impact biomass yield, with high precision and reproducibility. Its novel properties are:

1. Capable of plunging its sensor boom into the plant canopy to measure traits not visible from above or below without damaging the plants which enables multiple measurement during the life-cycle of the plant.
2. A system capable of handling the large-scale data processing, feature extraction, plant modeling and phenotype estimation.

2 Related Work

2.1 High-Throughput Phenotyping Platforms

Field phenotyping platforms include ground-based and aerial-based methods [5]. Aerial-based platforms enable greater coverage and rapid characterization of field plots, but are limited in spatial resolution and payload capacity. They are more suitable for estimating macro-phenotypic traits like plant location and densities [6].

Ground-based platforms can yield more detailed phenotypic information at the cost of relatively lower coverage rates. Lemnatec is one of the leaders in automated phenotyping with in-field platforms like Bonirob [7] and Scanalyzer Field [8]. Bonirob is an autonomous field robot designed to be a reusable platform for multiple agricultural applications like phenotyping, precision spraying and penetrometer measurement [7]. The Scanalyzer Field is an overhead gantry system with a sensor payload constituting imaging systems like fluorescence imaging, multi-spectral cameras and LiDAR [8]. While the Bonirob is a mobile platform capable of navigating between crop rows, the Scanalyzer Field is a gantry system on rails limited to small sites in which it can operate. Both platforms, however, are overhead phenotyping systems limited to top-down views of the plant and unable to see into closed canopy. This paper describes a mobile tractor-based phenotyping platform capable of collecting plant images at multiple vertical and horizontal viewpoints inside the closed plant canopy.

2.2 Plant Imaging and Phenotype Estimation

High-throughput phenotyping platforms deploy a variety of imaging modalities like 2D visible imaging, 3D imaging, multispectral imaging, thermal infrared imaging and fluorescence imaging [5]. Given its low cost and ease of operation, 2D/3D visible imaging has been commonly used for applications like plant mapping and detection [9], weed control [10], fruit counting and yield estimation [11]. For the purpose of phenotyping, the use of 3D visible imaging is important in order to be able to make ground truth metric measurements purely from imaging. However, most of the current state-of-the-art in 3D plant reconstruction, segmentation and phenotyping is in controlled greenhouse environments [12–14]. This paper demonstrates results for in-field 3D reconstruction and segmentation of plant structures from imaging data collected by the phenotyping platform. It compares these against reconstructions obtained from a greenhouse environment, and also proposes preliminary phenotype estimation methods on the greenhouse data.

3 High-Throughput Phenotyping Robot

The robot resides on a trailer system that provides an inexpensive support and transport system for the sensor boom and mast. The trailer will move in the alley ways between sub plots and once in position, drop leveling legs to stabilize the mast so that the sensor boom can be deployed reliably.

The trailer is a six-wheeled vehicle with *bogie* suspension. This type of suspension provides for a certain amount of terrain averaging without requiring a spring/damper-type suspension system. Hubs, axles, tires, wheels, trailer coupler and stabilizing jack (auto leveling) system are all commercial off- the-shelf items. The total weight of the system is ~ 1000 kg including the mast and sensor boom. The overall footprint of the trailer is 1.5 m wide by 3 m long. The solid front axle pivots for steering, providing an approximate 2.5 m turning radius. This vehicle requires an alley way of 1.8 m.

The system consists of a vertical column carrying opposing sensor booms as shown in Fig. 2a. Sensor pods are arranged along each of the sensor booms. The booms are mounted to a moveable carriage which is carried up the column by a linear actuator. The booms are extended and retracted by a linear actuator at the base of the sensor boom. Motion of this actuator, combined with motion of the carriage linear actuator, permits the booms to deploy while maintaining constant boom tip-to-ground distance. This substantially horizontal motion of the boom tip permits the sensor booms to enter the plant rows without damage to the plants. Once fully deployed the carriage linear actuator moves the booms upwards along the column to scan plants. After reaching the top of the scan, booms are retracted in reverse of the deployment motion and the system is moved to the next plant row to be scanned. The deployment sequence is depicted in Fig. 2b. The system operates at the rate of 72 plants/hour which is much higher than the manual methods.

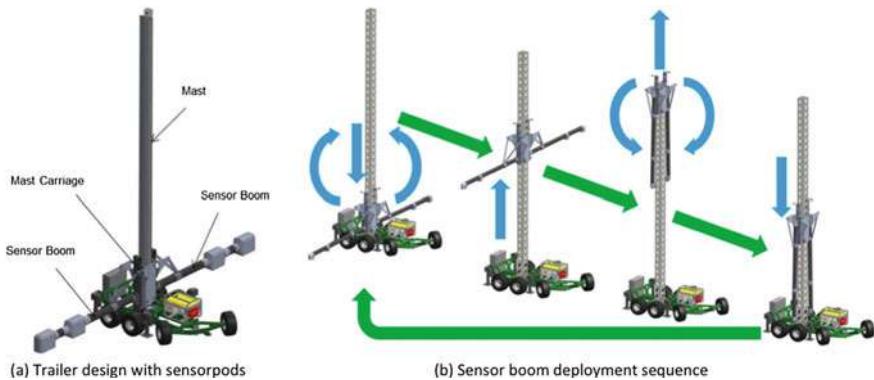


Fig. 2 High-throughput Phenotyping Robot—**a** trailer with the sensorpod **b** deployment sequence showing arms carrying sensorpods stretched out at the bottom; followed by sweeping up motion inside plant canopy; and finally folding back into the mast

3.1 Sensorpod

The sensorpod has 10 cameras with 8 (low resolution) cameras placed in two rows and remaining 2 (high resolution) cameras to the sides at a verged angle of 30°. The cameras are hardware synchronized and triggered every 10 cms when the boom moves up the canopy (Fig. 3). The top center pair cameras are designated for stereo reconstruction. The other low-resolution cameras carry narrow bandpass filters to capture images in multiple wavelengths. 450 and 550 nm filters correspond to the chlorophyll absorption bands 2 and 3, the 740 and 940 nm filters correspond to the normalized difference vegetation index and 950 nm filter correspond to nitrogen band. The high-resolution verged cameras are used for Structure-from-Motion reconstructions. A bright LED strip with diffusers is located below the camera array so as to provide uniform illumination inside densely covered plant canopies. An ambient light sensor is attached to the bottom of the sensorpod, looking at the plant, to help set the exposure values of the cameras. A PAR¹ sensor is mounted on top of the sensorpod to make photosynthetic light measurements when the system moves along the canopy. There is also an environmental sensor suite which measures the temperature, humidity and CO₂ mounted inside the sensorpod. All the cameras and sensors are connected to a embedded computer (Intel i7 3.4 GHz) running Ubuntu 14.04. In addition to the internal storage, a 1TB SSD is connected to the computer exclusively for data logging. An identical setup, with the cameras and computer is mounted on the other side of the sensorpod so that the system can scan two rows simultaneously. Since most LiDAR units do not provide high spatial resolution or close range (<10 cm), we do not use LiDAR for the purpose of dense 3D reconstruction of plants.

¹PAR—Photosynthetically Active Radiation is the spectral range of the solar radiation that plants use during photosynthesis.

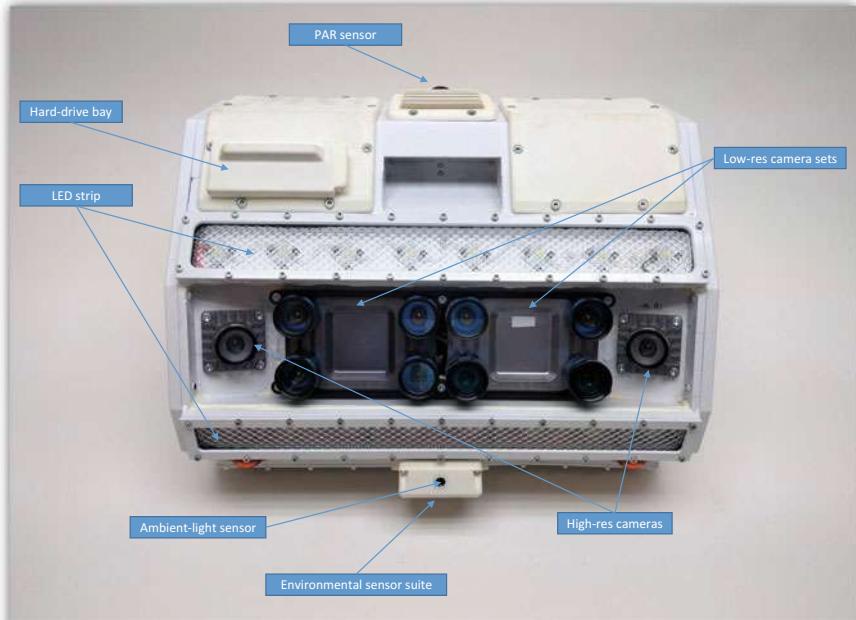


Fig. 3 Sensorpod—sensing module integrated with 10 cameras, environmental sensors, computing and storage

3.2 Data Collection and Interface

The robotic system has 2 sensorpods on either side and can scan 4 plants in one sweep. The design supports 8 or 12 sensorpods in total. This results in data flowing through 40 cameras and other environmental sensors simultaneously. In order to achieve this throughput, the data is logged locally as noted in the previous section. This gives us the freedom to add sensorpods to the boom without worrying about bandwidth issues. We still need to monitor the data from all the sensors including cameras and make sure it is logged properly. We accomplish this using the software architecture proposed in Fig. 4.

The *health and status* process monitors the health of all the processes running on the computer and also pulls images from the camera driver and sends image thumbnails and histograms to the *user interface (UI)* built using the Robot Operating System (ROS) framework. So if any camera fails, or if the image is not good (over/under exposed, foreground not in focus, etc.) the operator can work on it. The *logger* stores full resolution images from the camera drivers locally. After a scan is completed, the *logger* reports the number of frames logged to the *UI* so that dropped frames are noted immediately. The *computing* process monitors the state of the computer and sends vital information to the *UI* for tracking and monitoring. The *stereo* process generates disparities from the stereo pair and pushes them to the *UI* for inspection.

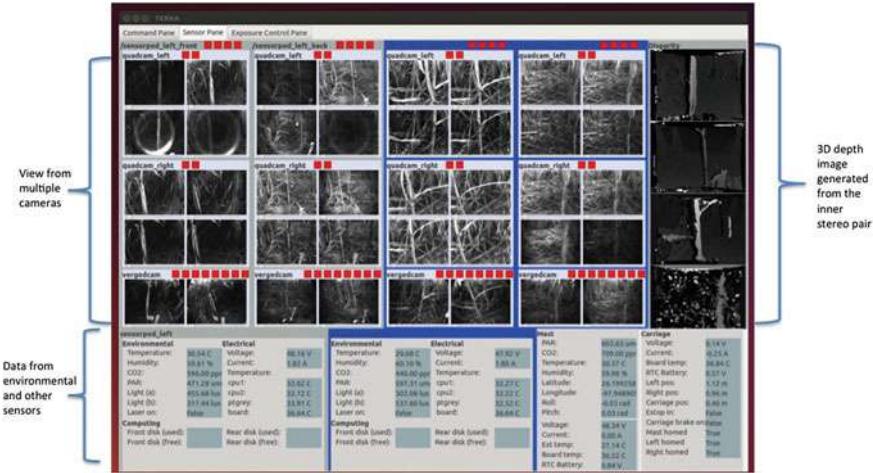


Fig. 4 User interface showing the camera view. The top section streams camera data from the different cameras along with a 3d depth image generated using the top inner stereo pair cameras. The bottom section shows the data streaming from the environmental and other sensors

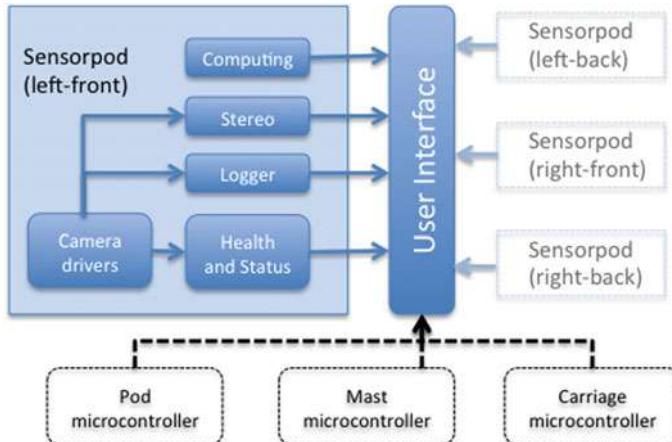


Fig. 5 Software architecture—the camera drivers feed image data to the different processes which in-turn feed their status to the user interface. An identical version of the software suite runs on each of the sensorpods. The encoder positions and other system level information flows from the micro-controllers to the user interface

A clone of similar processes run on all the sensorpod computers which talk to the central UI component. Figure 5 shows a snapshot of the UI with data streaming from the four sensorpod computers. It is crucial that we properly localize the robot so that the logs are tagged appropriately. To help on that front, the user interface displays a table of the plot layout and the operator would click on which plot the robot is scanning.

4 Modeling Plants

4.1 3D Reconstruction

Structure from Motion (SfM): Structure from Motion uses 2D grayscale images from different view-points to construct a point cloud of the scene. It takes in rectified images, finds features in the images and matches them. It then triangulates these feature matches to obtain 3D point positions using an initial estimate of the camera pose. The camera pose constituting intrinsics and extrinsics along with the 3D point positions are then collectively solved for in an optimization problem that seeks to minimize overall reprojection error [15, 16]. This gives a sparse point cloud which is then fed to a patch-growing algorithm like [17] to create a dense point cloud of the scene. An example of the point cloud constructed using SfM technique [15, 16] is shown in Fig. 6a. Using SfM techniques we can reconstruct the full plant model. The reconstructed plant model is geometrically consistent as it uses images from multiple known viewpoints.



Fig. 6 3D point cloud of Sorghum plant generated using **a** SfM technique **b** using stereo (SGBM) method

Stereo: For stereo point clouds we can only use the images from the inner pair of cameras on the top row. Since the plants are real close to the cameras (35–55 cms), using any other camera will make the disparity too big to be usable. Stereo reconstruction generally produces denser point clouds, but compared to the point clouds from SfM reconstruction, they are not as geometrically consistent as it uses fewer images and limited view-points. Also, this technique produces point clouds for each pair of frames and they have to be stitched (using techniques like Iterative Closest Point [18]) to generate the full 3D model. Figure 6b shows a 3d point cloud generated using stereo reconstruction technique.

4.2 Segmentation and Phytomer Extraction

A phytomer unit is a functional building block for the plant which consists of a leaf, its sheath and the stem segment on which the leaf resides. The pipeline [19] mapping input plant images to 3D phytomers is illustrated in Fig. 7. The first stage illustrates the generation of 3D point cloud from images. The next step computes point-level 3D features [20] using local geometries and a global distance metric to density modes. Each point is then classified as a stem or a leaf by learning a Support Vector Machine decision boundary, followed by spatial smoothing using a Conditional Random Field [21]. Next, the semantic segmentation step classifies each 3D point into one of two semantic classes, stem and leaf. The phytomer extraction step proceeds by performing RANSAC fit of a 3D cylinder model [22] to the points labeled as stem. The fitted cylinder is then expanded by ~25% to intersect leaves branching out from the stem. Each intersection point, referred to as a node, is then passed as a seed point to a region growing algorithm [23] so as to extract a single leaf

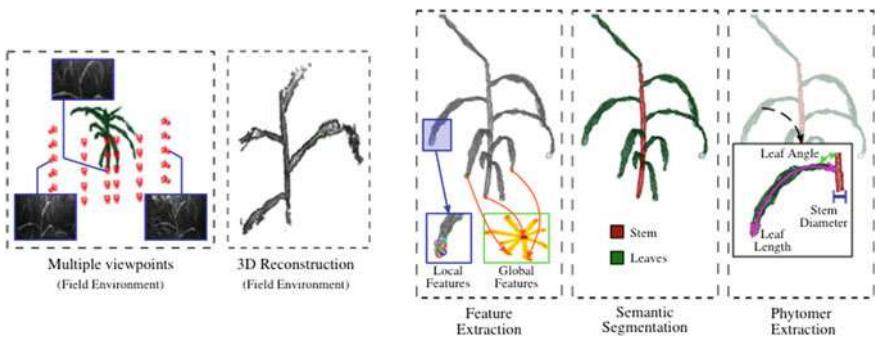


Fig. 7 Automated pipeline mapping input plant images to segmented 3D phytomer units. The first two modules take plant images captured from multiple viewpoints and generate a 3D point cloud reconstruction of the plant. Once the 3D reconstruction is obtained, local and global 3D point features are extracted. The next module uses a machine learning classifier to assign a semantic class label to each 3D point. Finally, the phytomer is extracted from the segmented point cloud

growing from that node. Once individual leaves at each of the nodes are extracted, these are then merged with a section of stem around the node to obtain a phytomer unit corresponding to that particular node.

5 Estimating Phenotypes

Since the field data is noisy due to occlusion, wind, etc., pure geometric approaches like surface fitting perform badly for phenotype estimation. Instead, we build a neural network model for this task. To be specific, the segmented phytomers are first binned into voxels. The voxels are then fed to the neural network for prediction. However, because the number of segmented phytomers are limited, the neural network cannot make good predictions if it is trained on real data alone. To alleviate this problem, we simulated plants based on the model from botanists and use this synthesized data set to train our model. We use the Multilayer Perceptron architecture with 4 hidden layers. The sizes of the layers are 5000, 1000, 200, 100. The plant point clouds are binned into voxels with dimension $30 \times 20 \times 50$. We serialize the $30 \times 20 \times 50$ voxels into vectors of size 30,000 before feeding them to the neural network. The neural network is trained on 10,000 synthesized plants with dropout probability 0.8 and learning rate 0.0001.

6 Field Experiments and Observations

We deployed a prototype in Puerto Rico in February 2016 to gain insights at the needs and challenges in designing the sensorpod. In October 2016 we deployed the robotic system in College Station, Texas and again in December 2016 in Weslaco, Texas.

6.1 *Exposure Control*

Camera exposure control is critical to obtaining high-quality data in this environment of dappled constantly changing lighting. Our system utilized light sensing to set camera exposure, as auto-exposure control algorithms were not feasible with our low frequency trigger-based capture process. The dynamic range of our scenes gradually increased along with required exposure time from the top of the plant moving downwards to the bottom of the plant creating dissimilarities in ideal exposure values. The varying relative intensities between ambient light and LED arrays as we moved from the top to the bottom, introduced a non-linearity to light-exposure relation. If images were under or over exposed, features would become indistinguishable in the image, underlining the importance of this process. In order to tackle this issue, we tried the

following techniques, however, the problem is not solved completely. (a) We used the high dynamic range (HDR) setting procedure to aid in compressing the dynamic range of the scene into a capturable image. (b) We incorporated brighter LED arrays into the sensorpod to increase the illumination at the bottom section of the plants. At the top of the canopy though, it couldn't be contrasted to the much brighter sunlight. (c) We used the ambient light sensor and PAR sensor on the sensorpod and mast, to set exposure of the cameras. Exposure controllers for the stereo camera were first calibrated to the light sensors. This was done in the field, using images of the scene at different heights so as to correctly expose the plants which covered most of the scene, while still keeping the brightest and darkest spots of the dynamic range unsaturated. An 18% gray card was used as a middle-gray reference to seed calibration. The resulting light-exposure relationship is given by:

$$\text{exposure} = K / (\text{light})$$

where K is the calibration constant. Therefore, doubling the light (increasing by one stop), would halve the exposure. Finally, offsets were calculated experimentally for the filtered imagers based off the relative intensities of solar radiation in the narrow wavelength bands. During field tests due to non-linearities mentioned earlier, we had to periodically adjust calibration thresholds based on the time of the day and cloud cover.

Though the current illumination helped at bottom of canopy, the imaging system needs much more illuminated light to fully subtract the ambient sunlight above about $500 \mu\text{mol}/\text{m}^2/\text{s}$.² Night scans with illumination yielded datasets with good amount of background subtraction which helped the plant modeling algorithms.

6.2 Field of View (FOV)

It was quickly established that the wider FOV lens was not ideal from many aspects. The wide FOV lens accepted light that the multispectral filters could not consistently block. Filters only function with a certain angle of incidence (AOI) of light, but the lens captured light outside of that region, which corresponded with a rainbow effect in the images.

Other problems with the wide field of view lens were pixel density (resolution) and depth of field related. A wider field of view lens would require a higher resolution imager to maintain the pixel density, which would increase the computing system demand from acquisition, to storage, to processing.

The depth of field of a lens in this relatively close imaging arrangement is significantly reduced when compared to the same lens focused at its hyperfocal distance. This problem is exacerbated with a wide field of view lens, where the distance a

² $\mu\text{mol}/\text{m}^2/\text{s}$ is based on the number of photons in a certain waveband incident per unit time on a unit area divided by the Avogadro constant.

subject is from the lens varies greatly from the center of the image to the ends. These concerns lead us to compromise on the amount of a row in view per camera, to obtain better image quality.

In our prototypes we used a lens with larger aperture f/2.6. But the depth of field was too shallow and we switched to a smaller aperture (f/4.0) lens. This yielded a larger depth of field at the expense of less light reaching the image sensor.

6.3 *Filters*

The commercially available off the shelf filters did not perform as expected. Some of the filters were reflective and applied to thick substrates while not evenly blocking light though the required FOV. This produced artifacts in the image. There also seemed to be a signal to noise component to this artifact. It was most notable in the filters for the 940 and 950 nm wavelengths where there isn't much sunlight due to water vapor absorption in the atmosphere, which reduced the filter efficiency.

6.4 *Ground Truth*

Taking ground truth measurement and associating the measurements to the robot collected imagery is a manually intensive and error prone process. For instance consider the leaf angle measurement process. The leaf angle phenotype generally refers to the first leaf below the flag leaf (top leaf). This leaf could have tillers coming out of the node. Tiller is a side shoot emanating from the main stem. Even for a keen human observer, the process of distinguishing a leaf from a tiller can be confusing. Assuming the distinction is made and the leaf angle measured, they need to be identified precisely from robot obtained imagery. This should be straight forward if we have the full 3D plant reconstruction. To tackle this problem, we tagged leaves for which we took ground truth leaf angle measurements. The tagging was done by tying a bright colored ribbon below the leaves. During post processing, the tag detection was automated and the manual measurements were then associated to the robot-collected data.

7 Results and Analysis

We scanned 991 fully grown sorghum plants using this system of which 170 plants were ground truthed. The ground truth phenotype measurements included leaf angle, stem diameter, leaf area and plant height.

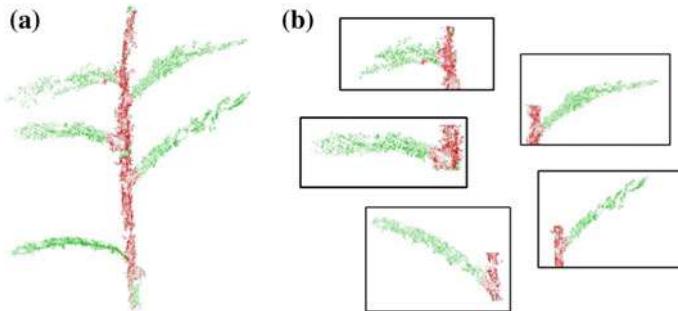


Fig. 8 **a** Shows qualitative segmentation outputs from SVM followed by CRF smoothing for field environment. **b** Shows segmented 3D phytomers extracted from the plant point cloud in (a)

Table 1 Relative root mean squared error on plant phenotypes measured using RGB-D sensor

Phenotype	Relative root mean squared error (%)
Leaf area	26.15
Leaf length	26.67
Leaf width	25.15

Figure 8 shows qualitative results for phytomer extraction on the field data collected in Dec 2016. It shows the semantically segmented point cloud followed by CRF smoothing along with the extracted 3D phytomers.

While the field data was processed and the segmentation algorithms were developed, we scanned the plants using a RGB-D sensor (indoors) and tested our neural network implementation. The neural network was trained on 10,000 synthesized plants with dropout probability 0.8 and learning rate 0.0001. The trained model was evaluated on 54 real plants to either predict phenotype values or determine whether the value is greater or smaller than the median value of all those real plants. Table 1 shows the predicted phenotypes along with their root mean squared errors relative to the range of the phenotype. Our current result is better than the naively using the mean all of data for prediction. The result can be improved in the future by passing the voxels into a CNN without serialization, increasing the resolution (number of voxels used) and adopting transfer learning techniques.

8 Conclusion

We presented a high-throughput non-intrusive phenotyping robot that takes multiple phenotypic measurements throughout the life of the crop with minimal human intervention. It is designed to handle the high volume data and make phenotypic

measurements. Through field validation of the system and the observations focused on bioenergy sorghum, the technology is deployable to a range of biomass crops as well as agronomic row crops.

References

1. Araus, J.L., Cairns, J.E.: Field high-throughput phenotyping: the new crop breeding frontier. *Trends Plant Sci.* **19**(1), 52–61 (2014)
2. Fahlgren, N., Gehan, M.A., Baxter, I.: Lights, camera, action: high-throughput plant phenotyping is ready for a close-up. *Curr. Opin. Plant Biol.* **24**, 93–99 (2015)
3. Mullet, J., Morishige, D., McCormick, R., Truong, S., Hilley, J., McKinley, B., Anderson, R., Olson, S.N., Rooney, W.: Energy sorghum—a genetic model for the design of C4 grass bioenergy crops. *J. Exp. Bot.* **65**, 3479–3489 (2014)
4. Vermerris, W.: Survey of genomics approaches to improve bioenergy traits in maize, Sorghum and sugarcane. *J. Integr. Plant Biol.* **53**, 105–119 (2011)
5. Li, L., Zhang, Q., Huang, D.: A review of imaging techniques for plant phenotyping. *Sensors* **14**(11), 20078–20111 (2014)
6. Ribera, J., et al.: Estimating Phenotypic Traits From UAV Based RGB Imagery
7. Bangert, W., et al.: Field-robot-based agriculture: remotefarming and bonirob-apps. *VDI-Berichte* **2193**, 439–446 (2013)
8. Virlet, N., et al.: Field scanalyzer: an automated robotic field phenotyping platform for detailed crop monitoring. *Functional Plant Biol.* **44**(1), 143–153 (2017)
9. Weiss, U., Biber, P.: Plant detection and mapping for agricultural robots using a 3D LIDAR sensor. *Robot. Auton. Syst.* **59**(5), 265–273 (2011)
10. Slaughter, D.C., Giles, D.K., Downey, D.: Autonomous robotic weed control systems: a review. *Comput. Electron. Agric.* **61**(1), 63–78 (2008)
11. Dey, D., Mumtaz, L., Sukthankar, R.: Classification of plant structures from uncalibrated image sequences. In: 2012 IEEE Workshop on Applications of Computer Vision (WACV). IEEE, APA (2012)
12. McCormick, R.F., Truong, S.K., Mullet, J.E.: 3d sorghum recon- structions from depth images identify qtl regulating shoot architecture. *Plant Physiol.* 00948 (2016)
13. Lehnert, C., et al.: Autonomous sweet pepper harvesting for protected cropping systems. *IEEE Robot. Autom. Lett.* **2**(2), 872–879 (2017)
14. Chaivivatrakul, S., et al.: Automatic morphological trait characterization for corn plants via 3D holographic reconstruction. *Comput. Electronics in Agric.* **109**, 109–123 (2014)
15. Wu, C.: Towards linear-time incremental structure from motion. In: Proceedings of—2013 International Conference 3D Vision, 3DV 2013, pp. 127–134 (2013)
16. Fuhrmann, S., Langguth, F., Goesele, M.: MVE—a multi-view reconstruction environment. In: Eurographics Workshops on Graphics and Cultural Heritage (2014)
17. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(8), 1362–1376 (2010)
18. Besl, P., McKay, N.: A method for registration of 3D shapes. *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)* **14**(2), 239–256 (1992)
19. Sodhi, P., Vijayarangan, S., Wettergreen, D.: In-field segmentation and identification of plant structures using 3D imaging. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2017)
20. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (FPFH) for 3D registration. In: IEEE International Conference on Robotics and Automation, 2009. ICRA'09. IEEE (2009)
21. Koltun, Vladlen: Efficient inference in fully connected crfs with gaussian edge potentials. *Adv. Neural Inf. Process. Syst.* **2**(3), 4 (2011)

22. Schnabel, R., Wahl, R., Klein, R.: Efficient RANSAC for pointcloud shape detection. In: Computer Graphics Forum, vol. 26, no. 2. Blackwell Publishing Ltd (2007)
23. Rusu, R.B., Cousins, S.: 3d is here: point cloud library (pcl). In: 2011 IEEE International Conference on Robotics and Automation (ICRA). IEEE (2011)

Improved Tau-Guidance and Vision-Aided Navigation for Robust Autonomous Landing of UAVs

Amedeo Rodi Vetrella, Inkyu Sa, Marija Popović, Raghav Khanna, Juan Nieto, Giancarmine Fasano, Domenico Accardo and Roland Siegwart

Abstract In many unmanned aerial vehicle (UAV) applications, flexible trajectory generation algorithms are required to enable high levels of autonomy for critical mission phases, such as take-off, area coverage, and landing. In this paper, we present a guidance approach which uses the improved intrinsic tau guidance theory to create spatio-temporal 4-D trajectories for a desired time-to-contact with a landing platform tracked by a visual sensor. This allows us to perform maneuvers with tunable trajectory profiles, while catering for static or non-static starting and terminating motion states. We validate our method in both simulations and real platform experiments by using rotary-wing UAVs to land on static platforms. Results show that our method achieves smooth landings within 10 cm accuracy, with easily adjustable trajectory parameters.

1 Introduction

Unmanned aerial vehicles (UAVs) play a significant role in providing services and enhancing safety thanks to their flexible and low-cost surveillance, monitoring, and risk assessment capabilities. In many emerging applications, including environmental monitoring [3], industrial inspection [4], and emergency response [13], high levels of system autonomy are vital to guarantee safe and reliable operation. This is especially true when UAVs act as sentinels monitoring large areas, which take-off from a *nest* and return to it for recharging before performing further tasks.

A typical mission can be divided into the following phases: (1) take-off, (2) mid-course, (3) area scanning [14, 15], (4) visual tracking, and (5) landing. Often, the final phase is the most critical as it involves performing delicate maneuvers; e.g., landing on a station for re-charging [2] or on a ground carrier for transportation [8]. These procedures are subject to constraints on time and space, and must be robust

A. R. Vetrella (✉) · G. Fasano · D. Accardo
University of Naples “Frederico II”, Naples, Italy
e-mail: amedeorodi.vetrella@unina.it

I. Sa · M. Popović · R. Khanna · J. Nieto · R. Siegwart
Autonomous Systems Lab, ETH Zürich, Zürich, Switzerland

to changes in environmental conditions, such as visibility and wind disturbances [6]. To achieve smooth landings, precise sensing and accurate control techniques are therefore required.

In this paper, we address these problems by integrating, within the end-to-end software system developed at the ETH Zürich, a trajectory generation algorithm based on a visual tracking system and a bio-inspired guidance method for autonomously landing on a specified target. Our motivation is to increase the reliability and versatility of landings in the example scenarios mentioned above. We use the improved intrinsic tau guidance theory [10, 19] to generate spatio-temporal (4-D) trajectories for a desired time-to-contact (TTC) based on the estimate of the relative pose of the UAV with respect to the target. This approach enables us to perform a maneuver with arbitrary initial and final motion states, and tailor its trajectory profile for various types of rotary- or fixed-wing tasks, such as landing, in-flight obstacle avoidance, and object-picking. The advantage of this approach is an “*user-oriented*” trajectory generation method, where fundamental parameters can be tuned based on the mission requirements. As such, the user may be interested in assigning predefined mission requirements for the trajectory, such as to maintain the course within the boundary of two intersecting planes (e.g., flying within a natural or man-made canyon), controlling the total mission time or energy consumption, or specifying the initial and final landing angles.

We simulate and experimentally validate our approach in outdoor and indoor experiments using rotary-wing UAV equipped with a downward-looking camera for detecting a static target. The main contributions of this work are:

1. A bio-inspired landing method which:
 - generates easily tunable 4-D trajectories to the target,
 - provides guidance starting or terminating with static or non-static motion states,
 - is applicable with different sensor configurations.
2. The validation of our method in both simulation and real platform experiments.

The paper is structured as follows: Sect. 2 begins by outlining the state-of-the-art in autonomous landing methods. The general landing problem is formulated in Sects. 3, and 4 describes our bio-inspired approach. We present our experimental set-ups and results in Sects. 5 and 6 before concluding in Sect. 7.

2 Related Work

Significant work has been done recently on autonomous landing methods for UAVs in various environments. As discussed by [6], landing navigation frameworks mainly use a Global Positioning System (GPS) and an Inertial Measurement Unit (IMU), with small, light-weight visual sensors often integrated for improved accuracy, especially in outdoor applications.

Landing strategies can be categorized based on the guidance techniques used to create trajectories to the landing position. We consider (i) position-based [18, 20] and (ii) biologically-inspired [9, 11, 17] techniques. Traditional position-based guidance approaches, such as the pursuit [20] and proportional [18] laws, leverage Line-of-Sight (LoS) to navigate the UAV towards the target. While these methods provide precise tracking, they are limited to position sensors only as positions and/or velocities must be accurately computed [8]. Moreover, they may require complex control algorithms and lack controllability of the UAV trajectory pattern and profile, thus restricting their applicability.

Bio-inspired guidance paradigms overcome these limitations by using visual information such as the TTC [11] or optical flow [17] to generate 4-D trajectories [9, 17], enabling the control of both their spatial and temporal components. In this category, the general tau theory [10, 11] has been popularly postulated to describe goal-directed movements, e.g., collision avoidance [19], docking, and landing [9]. [9] recently introduced a tau-based UAV autopilot and implemented it to perform high-accuracy maneuvers. Similarly, we apply the tau principles to generate a 4-D trajectory for landing. However, by using elements of improved intrinsic tau guidance [19], our method is not restricted to static initial and final states and is thus also suitable for landing on moving platforms.

3 Problem Definition

As outlined in Sect. 1, there are various UAV missions requiring large area coverage. For safe and robust autonomous operation, it is necessary to generate successive and consecutive trajectories by maintaining position and/or velocity continuities on the boundary waypoint between the n th and the $n+1$ th trajectories. Adjourning the trajectories is also required during the successive steps of following or landing on a moving object, taking into account the error deriving by the tracking algorithm, including noise. Consequently, the main problem is to develop and apply a fast and flexible algorithm enabling the UAV to perform the above-mentioned phases autonomously, while maintaining a small number of intuitive turning parameters.

The general goal-directed trajectory generation problem is defined as follows. We describe a spatio-temporal 4-D UAV trajectory by the state:

$$\mathbf{S}(t) = \{x(t), y(t), z(t), \dot{x}(t), \dot{y}(t), \dot{z}(t)\}$$

The aim is to guide the UAV from arbitrary initial states $\mathbf{S}(t_0)$ to goal states $\mathbf{S}(t_0 + T)$ in the execution time T . The UAV dynamics are:

$$\begin{cases} \dot{x} = u_x, \\ \dot{y} = u_y, \\ \dot{z} = u_z, \end{cases} \quad (1)$$

where $\mathbf{u} = \{u_x, u_y, u_z\}$ are the velocity components along each co-ordinate axes used as commands for the trajectory tracker, which is discussed for our application in Sect. 5.

Although the algorithm is applicable to a generic trajectory, this paper focuses on the most complex mission scenario involving landing on a stationary or moving target. The trajectory is adjourned based on visual information provided by a camera during the approach phase. The specific requirements are to:

1. reach the target point in a specified time T ,
2. arrive and stop at the target point with zero velocity at contact, such that $\mathbf{S}(t) = \{x(t), y(t), z(t), 0, 0, 0\}$ at $t = t_0 + T$,
3. reach the target point from a specific approach direction, as discussed in Sect. 4.1.

4 Approach

This section overviews the proposed bio-inspired guidance strategy for autonomously landing on a platform, where the perching trajectory is generated assuming that the target can be tracked with a camera. In brief, we use the intrinsic tau guidance strategy to generate tunable 4-D trajectories for smooth landings with static or non-static starting and terminating states. First, we present our method of trajectory parametrization as the basis of our approach. We then summarize the basic principles of general tau theory before detailing our specific guidance technique.

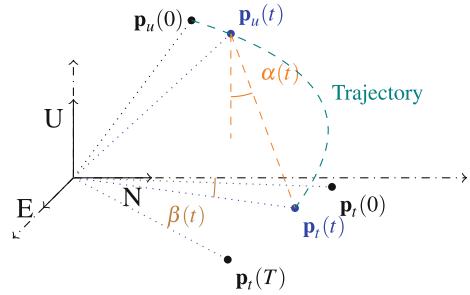
4.1 Trajectory Parametrization

With reference to Fig. 1, in the East-North-Up (ENU) frame, the landing maneuver usually requires arriving at a destination with a final state of the trajectory depending on the target morphology and dynamics. The main gaps are:

- the distance gap $d(t)$,
- the relative speed,
- the approaching angle of the trajectory $\alpha(t)$, between $d(t)$ and the normal to the EN plane,
- the approaching angle of the trajectory $\beta(t)$, which permits changing the UAV heading during perching.

$d(t)$ is the instantaneous distance along the LoS to the target, given by the difference between the UAV position $\mathbf{p}_u(t)$, and the target position $\mathbf{p}_t(t)$. The relative speed is given by $\dot{\mathbf{p}}_u(t) - \dot{\mathbf{p}}_t(t)$ which, in order to avoid collisions, must be zero at the touch-down. The approaching angle $\alpha(t)$ is based on the landing approach, and consequently depends on the UAV platform and application, e.g., along the tangent to the surface for a fixed-wing UAV, near-vertical for a rotary-wing UAV, or inclined at

Fig. 1 Reference frame and main gaps for autonomous tau-landing



a certain angle to enter an opening. The approaching angle $\beta(t)$ ensures that the body frames of the UAV and the target respect a required orientation at the touch-down, e.g., when recharging requires docking between the two.

To introduce these parameters in the mission design, we use the intrinsic tau guidance approach presented in Sect. 4.3.

4.2 General Tau Theory

Our guidance method for landing is based on the general tau theory [10], which uses the tau function to represent the TTC of a goal-directed movement. We define the tau variable of a motion gap χ as:

$$\tau_\chi = \begin{cases} \frac{\dot{\chi}(t)}{\ddot{\chi}(t)} & |\dot{\chi}(t)| \geq \dot{\chi}_{min}, \\ \text{sgn}\left(\frac{\dot{\chi}(t)}{\ddot{\chi}(t)}\right) \tau_{max} & |\dot{\chi}(t)| < \dot{\chi}_{min}, \end{cases} \quad (2)$$

where $\chi(t)$ can be the gap of any state in $\mathbf{S}(t)$.

4.3 Improved Intrinsic Tau Guidance Strategy

Specifically, our landing method uses the improved intrinsic tau guidance strategy [19] to allow for goal-directed movements and, in particular, to start the trajectory with an initial flight condition. We formulate the intrinsic guidance gap of a movement $G_v(t)$ as:

$$\begin{cases} G_v(t) = -0.5at^2 + V_G t + G_0, \\ \dot{G}_v(t) = -at + V_G, \\ \ddot{G}_v(t) = -a, \end{cases} \quad (3)$$

where a is the acceleration, V_G is an initial velocity, and G_0 specifies an initial intrinsic gap. In particular, the intrinsic tau guidance strategy (3) represents the vertical component of a projectile motion. Once the acceleration a is assigned, the initial velocity V_G and the initial gap G_0 must be computed according to the initial ($t = 0$) and the final ($t = T$) conditions of the actual movement, as described in the following.

Considering the movement along a generic x-axis from time 0 to T as an example, the position and velocity gaps can be expressed as $\Delta x = x_T - x$ and $\Delta \dot{x} = \dot{x}_T - \dot{x}$, respectively, where x_T and \dot{x}_T denote the goal states at time T . We apply the tau coupling strategy [9, 19] for synchronous gap-closing to obtain the movement states:

$$\begin{cases} x(t) = x_T + \dot{x}_T(t - T) - \frac{\chi_{x0}}{G_0^{1/k_x}} G_v^{1/k_x}, \\ \dot{x}(t) = \dot{x}_T - \frac{\chi_{x0}}{k_x G_0^{1/k_x}} \dot{G}_v G_v^{1/k_x - 1}, \\ \ddot{x}(t) = -\frac{\chi_{x0}}{k_x G_0^{1/k_x}} G_v^{1/k_x - 2} \left(\frac{1-k_x}{k_x} \dot{G}_v^2 + G_v \ddot{G}_v \right). \end{cases} \quad (4)$$

where k_x is a gain parameter controlling gap convergence along the x-axis, as discussed below.

From the definition of G_v , Eq. 4, and Eq. 3, it can be shown that:

$$\begin{cases} G_0 = \frac{\chi_{x0} g T^2}{2(\chi_{x0} + k_x \Delta \dot{x}_0 T)}, \\ V_G = \frac{k_x \Delta \dot{x}_0 g T^2}{2(\chi_{x0} + k_x \Delta \dot{x}_0 T)}. \end{cases} \quad (5)$$

G_0 and V_G can be viewed as bonding actual and intrinsic movements due to gravitational effects. If $k_x \in (0, 0.5)$, $(x, \dot{x}, \ddot{x}) \rightarrow (x_T, \dot{x}_T, 0)$ and the position and velocity can be steadily guided to the target values, as required. Hence, varying the components of $\mathbf{k} = \{k_x, k_y, k_z\}$ within this range allows for modifying the trajectory profile along each axis.

In addition, using the tau coupling strategy described by Eqs. (2) and (3), we can apply a more “user-oriented” approach by coupling to the intrinsic movements the gaps $d(t)$, $\alpha(t)$ and $\beta(t)$ as follows:

$$d(t) = \frac{d(0)}{G_0^{\frac{1}{k_d}}} G_v^{\frac{1}{k_d}} \quad (6)$$

$$\alpha(t) = \frac{\alpha(0)}{d(0)^{\frac{1}{k_\alpha}}} d(t)^{\frac{1}{k_\alpha}} \quad (7)$$

$$\beta(t) = \frac{\beta(0)}{d(0)^{\frac{1}{k_\beta}}} d(t)^{\frac{1}{k_\beta}} \quad (8)$$

where each gap has its own gain parameter (k_d, k_α, k_β). Combining these equations the position vector at time t of the UAV is simply given by:

$$\mathbf{p}(t) = \mathbf{p}(T) + \begin{bmatrix} -d(t)\sin\alpha(t)\cos\beta(t) \\ -d(t)\sin\alpha(t)\sin\beta(t) \\ d(t)\cos\alpha(t) \end{bmatrix} \quad (9)$$

5 Experimental Set-Up

This section details physical system architectures used to test the proposed method (Sect. 6.2). To demonstrate applicability, we conducted experiments in both outdoor and indoor environments.

The outdoor experiments are conducted on an empty 20×20 m farmland plot in clear weather conditions (Fig. 2a), where we demonstrate our landing algorithms running in real-time on an AscTec Firefly (Fig. 2b). Our UAV is equipped with an autopilot providing a low- and high-level control, and on-board computer (AscTec Mastermind), and the following sensors:

- 100 Hz IMU,
- 10 Hz Pixsi V2 (RTK) differential GPS,
- 20 Hz (VI-)sensor,
- 50 Hz downward-looking Point Grey Chameleon 2.0 camera.

The landing platform target (Fig. 2c) is an A3 (297×420 mm) arrangement of variable-dimension tags, obtained from the `ar_track_alvar` library. The nested tag layout on the platform allows for detecting and estimating the camera-to-target relative pose from different altitudes. Note that, in this paper, we use a static platform for proof of concept and leave the study of moving targets to future work.

The logical architecture of the algorithms running on the UAV is described in Fig. 3, which depicts the different interacting modules. In particular, the input data (Fig. 3, left) include: GPS measurements, images from the downward-looking camera and the VI-sensor, and accelerometer and gyroscope measurements from the VI-sensor and the on-board IMU.

For navigation, the UAV pose is estimated by integrating VI-sensor and GPS data within the ROBust Visual Inertial Odometry (ROVIO) framework [1]. This is then integrated with IMU data in the Multi-Sensor Fusion (MSF) framework [12] to obtain a refined state estimate Fig. 3. Guidance is based on the improved tau guidance strategy (red block in Fig. 3), which uses the actual UAV state and the camera-to-target relative pose to generate reference trajectories that are then tracked by a non-linear Model Predictive Controller (MPC) [7].



Fig. 2 **a** Shows our experimental set-up on the field, with the GPS RTK base station visible on the right. **b** Depicts the AscTec Firefly positioned on the landing platform. **c** Exemplifies the tag arrangement on the platform as seen by the on-board camera

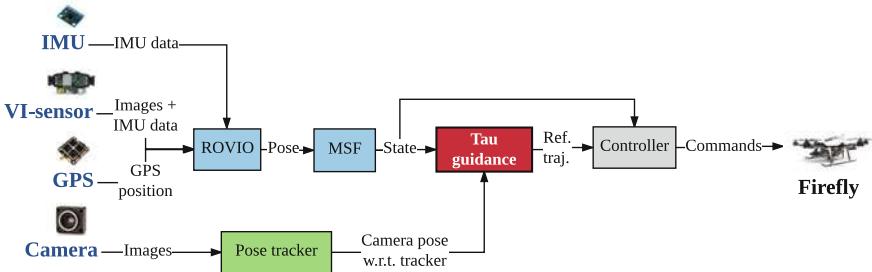


Fig. 3 Systems diagram for our outdoor experiments. A camera is used to track the landing platform pose, and localization is provided by the MSF framework output. The guidance unit passes reference trajectories to the MPC. Note that our RTK GPS requires a ground base station (Fig. 2a) to receive satellite signals and transmit position corrections to the UAV

We also performed indoor experiments in an empty $20 \times 20\text{ m}$ environment (Fig. 4) using the same landing platform. Here, our algorithms run in real-time on a DJI Matrice 100 with a 100 Hz IMU and Intel RealSense ZR300 camera providing 30 Hz images [16]. Our system comprises the general pipeline in Fig. 3; however, in the absence of GPS data, only images and IMU data are input to ROVIO and MSF.

6 Results

This section demonstrates the usage of the improved tau guidance strategy to land smoothly on a target platform. We first validate our method in simulation before presenting results from outdoor and indoor experiments.

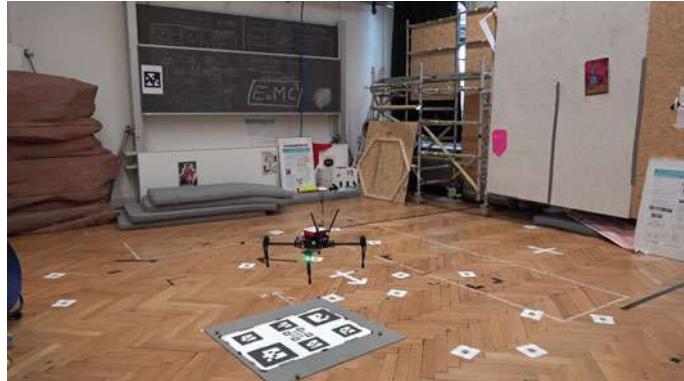


Fig. 4 A side view of our indoor experimental set-up showing the UAV and the landing platform

Table 1 Assumptions on the simulated bias and random errors for our simulation trials

Sensors	Bias instability	Random walk
Gyroscopes	$14.5^\circ/\text{hr}$	$0.66^\circ/\sqrt{\text{hr}}$
Accelerometers	0.25 mg	$0.11 \text{ m/sec}/\sqrt{\text{hr}}$

6.1 Simulations

The performance of the proposed landing method is validated preliminarily in simulation. The simulation environment is developed in the RotorS framework [5], which realistically replicates the flight dynamics of an Asctec Firefly and its on-board sensors. For the IMU, the orders of magnitude of biases and random errors are set to be consistent with Micro Electro-Mechanical Systems (MEMS) used on the AscTec Firefly (Table 1). Furthermore, the GPS position uncertainty in the ENU frame is modeled as a constant bias plus Gaussian white noise to account for the time correlation in GPS errors. For vision sensing, only the downward-looking camera was simulated with an Instantaneous Field of View (IFoV) uncertainty modeled as Gaussian white noise with a standard deviation of 0.05° . This camera is used to detect the landing target on the ground and to obtain its relative pose with respect to the UAV. Consequently, navigation is achieved here without simulating ROVIO, but only integrating simulated position and attitude measurements within the MSF framework.

The simulations serve to validate and integrate each component of the software framework before the experimental tests (Sect. 6.2). Since the focus is set on rotary-wing UAVs, the interest is also to land on the platform with a desired orientation $\alpha(t)$ and zero velocity. Hence, simulations have been performed to evaluate the behavior of trajectory shapes for varying coefficients k with an emphasis on vertical landing. Figure 5a shows a comparison of different landing trajectories where the maneuver starts from an height of 1.5 m and EN components of 0.8 m each. This figure evidences

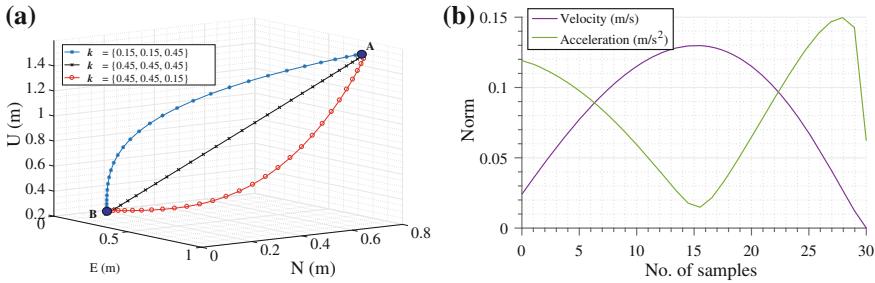


Fig. 5 **a** Depicts landing trajectories with different elements in \mathbf{k} . These parameters enable modifying the trajectory profile for LoS (black), near-vertical (blue), and near-horizontal (red) shapes. **b** Shows the velocity and acceleration norm profiles for the near-vertical (blue) curve in (a)

how the 3-D shape of each trajectory is determined by coupling the coefficients \mathbf{k} . In addition, Fig. 5b shows the velocity and acceleration profiles during the near-vertical landing trajectory (blue curve in Fig. 5a), demonstrating that zero velocity is reached at the end of the landing maneuver.

6.2 Experimental Tests

To test our approach in practical scenarios, we conducted multiple experiments using the set-ups described in Sect. 5. In each trial, the UAV was commanded to:

1. take-off and perform its mission,
2. fly back to the landing area,
3. fly a search path to detect the target, and
4. safely land on the target point in a specified time.

Figures 6 and 7 illustrate the two autonomous landing stages for an outdoor test using an AscTec Firefly. The first stage is target detection (at an height of about 2 m, as shown in Fig. 6c) followed by a refinement phase for noise reduction. Once the target is detected and tracked, the UAV starts the approaching stage during which the camera continues tracking the target to update its relative pose. If the difference between successive target positions is larger than a certain tolerance, the trajectory is re-planned accordingly and passed to the non-linear MPC.

These aspects are evident in Fig. 6, which shows the UAV position components in the ENU frame. A comparison of the reference trajectory generated by our guidance strategy with the UAV position output from the MSF framework confirms that the controller follows the reference throughout the maneuver until the touch-down.

Figure 7 compares the UAV position and velocity along the three co-ordinate axes. The plots illustrate the optimum velocity profile of the maneuver to reach the softest touch-down while achieving the accuracy required to land near the target center.

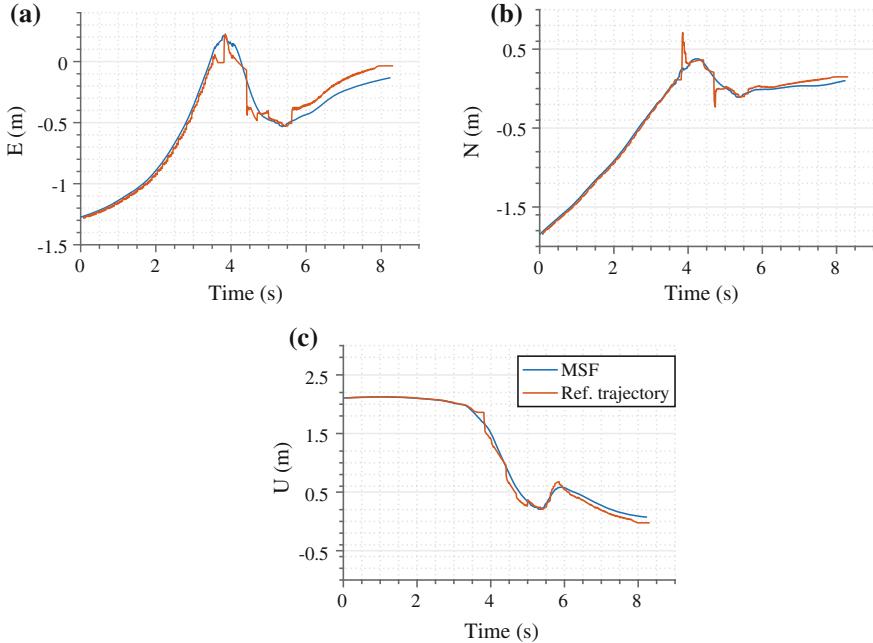


Fig. 6 Differences between the reference and MSF trajectories during the final phases of landing

As shown in Fig. 8, the final landing position remains within a maximum dispersion of several centimeters with respect to the target center. It is worth noting that, if the target is lost for a certain time period, the UAV is commanded to climb to increase the camera footprint, and returns on the landing path only after the target is detected and tracked again. Figure 8 demonstrates this effect as the target is lost at ~ 20 cm height mainly due to the UAV shadow causing occlusions impeding robust tracking. As a result, the UAV ascends, landing smoothly upon target re-detection.

In addition, we executed 63 successful indoor landings with the DJI Matrice 100. State estimation was obtained by integrating ROVIO within the MSF framework, the relative pose between the UAV and the landing platform was obtained by images acquired by the Intel RealSense ZR300 and the VICON system was used for ground truth reference. These tests show the repeatability and robustness of the proposed landing approach as well as its achievable accuracy. To this end, six case studies were considered varying in the \mathbf{k} coefficients and the height of the target detection stage. The approaching heights tested were 1.5, 2 and 2.5 m, with two different sets of \mathbf{k} coefficients: $k_x = k_y = 0.2$, $k_z = 0.4$, and $k_x = k_y = k_z = 0.2$. Statistical results are summarized in Table 2. Here, errors are computed based on UAV and target ground truth positions acquired from the VICON system.

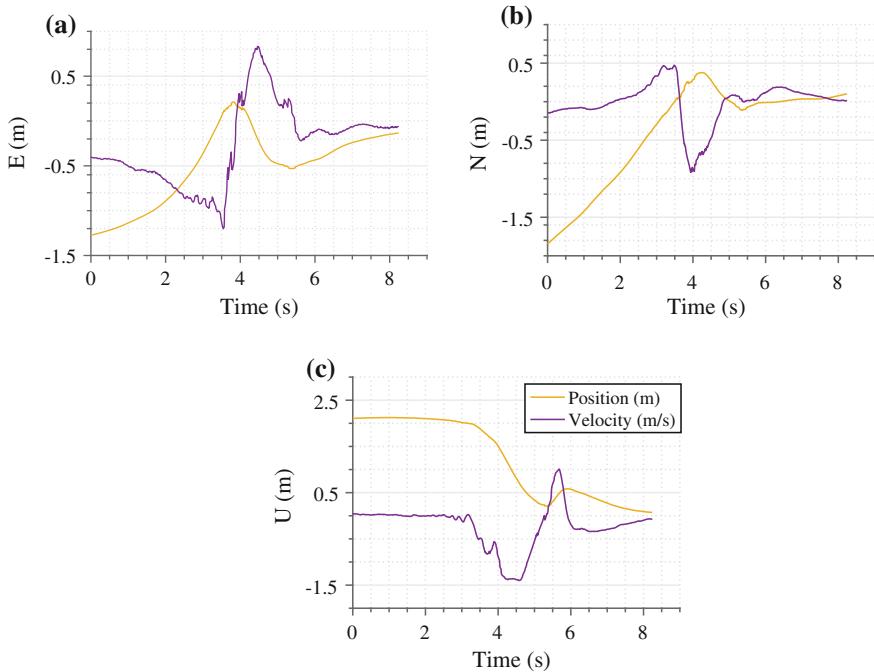


Fig. 7 Velocity profile during the landing maneuver with the position profile as a reference

Fig. 8 UAV landing position with respect to the target position in the EN plane

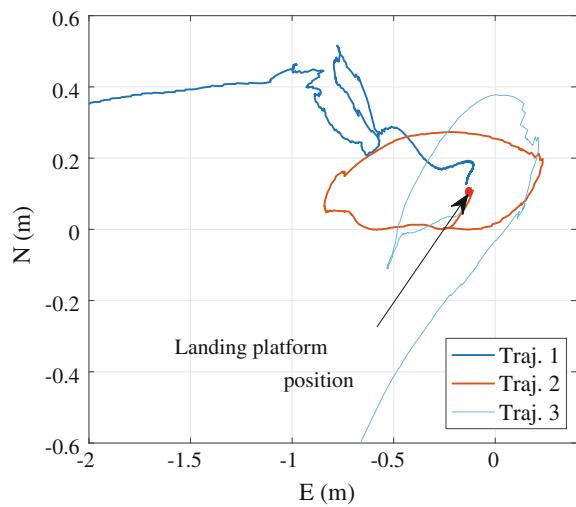


Table 2 Landing mean errors and standard deviations for the considered case studies

Height (m)	1.5		2.0		2.5	
	$k_z = 0.2$	$k_z = 0.4$	$k_z = 0.2$	$k_z = 0.4$	$k_z = 0.2$	$k_z = 0.4$
Mean (m)	0.15	0.16	0.14	0.14	0.2	0.19
Std (m)	0.02	0.04	0.04	0.05	0.04	0.04

7 Conclusions and Future Work

This paper presented a guidance approach based on the improved intrinsic tau guidance law for autonomous UAV landing on a static platform. The guidance theory generates smooth and computationally efficient 4-D trajectories that are both suitable for fixed- and rotary-wing UAV platforms. The framework was validated in simulations and multiple outdoor and indoor experiments with different platforms, showing that trajectories can be easily designed by varying the guidance coefficients. Results from over 60 indoor tests, using a VICON system only to provide ground truth reference, demonstrate landing with centimeter-level accuracy.

Future work will examine methods of developing a reliable target tracker based on the current visual target detector as well as GPS and IMU measurements to estimate a target pose with respect to the UAV. This would allow more robust target-tracking in challenging conditions, such as dynamic motion, and even in cases where the target is outside the camera field-of-view. An accurate target-tracker would also enable landing on a moving platform. In addition, we will consider developing a target detector more suitable for poorly lit environments.

Finally, we are interested in applying the proposed guidance law to fixed-wing landing missions.

Acknowledgements This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644227 and from the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number 15.0029. We would like to thank Marco Tranzatto and Michael Pantic for their valuable insights and platform support, and the ETH Crop Science Group for providing the testing facilities.

References

1. Bloesch, M., Omari, S., Hutter, M., Siegwart, R.: Robust visual inertial odometry using a direct EKF-based approach. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 298–304. IEEE, Hamburg (2015)
2. Coccioni, F., Mancini, A., Longhi, S.: Autonomous navigation, landing and recharge of a quadrotor using artificial vision. In: International Conference on Unmanned Aircraft Systems, pp. 418–429 (2014)
3. Dunbabin, M., Marques, L.: Robots for environmental monitoring: significant advancements and applications. IEEE Robot. Autom. Mag. **19**(1), 24–39 (2012)

4. Ezequiel, C.A.F., Cua, M., Libatique, N.C., Tangonan, G.L., Alampay, R., Labuguen, R.T., Favila, C.M., Honrado, J.L.E., Canos, V., Devaney, C., Loreto, A.B., Bacusmo, J., Palma, B.: UAV aerial imaging applications for post-disaster assessment, environmental management and infrastructure development. In: International Conference on Unmanned Aircraft Systems, pp. 274–283 (2014)
5. Furrer, F., Burri, M., Achtelik, M., Siegwart, R.: Rotors-a modular gazebo MAV simulator framework, vol. 1, pp. 595–625. Springer International Publishing (2016)
6. Gautam, A., Sujit, P.B., Saripalli, S.: A survey of autonomous landing techniques for UAVs. In: International Conference on Unmanned Aircraft Systems, pp. 1210–1218 (2014)
7. Kamel, M., Stastny, T., Alexis, K., Siegwart, R.: Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system, pp. 3–39. Springer International Publishing, Cham (2017)
8. Kendoul, F.: Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *J. Field Robot.* **29**(2), 315–378 (2012)
9. Kendoul, F.: Four-dimensional guidance and control of movement using time-to-contact: application to automated docking and landing of unmanned rotorcraft systems. *Int. J. Robot. Res.* **33**(2), 237–267 (2013)
10. Lee, D.N.: A theory of visual control of braking based on information about time-to-collision. *Perception* **5**(4), 437–459 (1976)
11. Lee, D.N., Davies, M.N.O., Green, P.R., (Ruud) Van Der Weel, F.R.: Visual control of velocity of approach by pigeons when landing. *J. Exp. Biol.* **180**, 85–104 (1993)
12. Lynam, S., Achtelik, M.W., Weiss, S., Chli, M., Siegwart, R.: A robust and modular multi-sensor fusion approach applied to MAV navigation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3923–3929. IEEE, Tokyo (2013)
13. Nex, F., Remondino, F.: UAV for 3D mapping applications: a review. *Appl. Geomat.* **6**(1), 1–15 (2014)
14. Popovic, M., Hitz, G., Nieto, J., Sa, I., Siegwart, R., Galceran, E.: Online informative path planning for active classification using UAVs. In: IEEE International Conference on Robotics and Automation. IEEE, Singapore (2017a)
15. Popovic, M., Vidal-Calleja, T., Hitz, G., Sa, I., Siegwart, R., Nieto, J.: Multiresolution mapping and informative path planning for UAV-based terrain monitoring. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Vancouver (2017b)
16. Sa, I., Kamel, M., Khanna, R., Popovic, M., Nieto, J., Siegwart, R.: Dynamic System Identification, and Control for a cost effective open-source VTOL MAV. [arXiv:1701.08623](https://arxiv.org/abs/1701.08623) (2017)
17. Strydom, R., Denuelle, A., Srinivasan, M.V.: Bio-inspired principles applied to the guidance, navigation and control of UAS. *Aerospace* **3**(21) (2016)
18. Yamasaki, T., Sakaida, H., Enomoto, K., Takano, H., Academy, N.D.: Robust trajectory-tracking method for UAV guidance. In: International Conference on Control, Automation and Systems. IEEE, Seoul (2007)
19. Yang, Z., Fang, Z., Li, P.: Decentralized 4D trajectory generation for UAVs based on improved intrinsic tau guidance strategy. *Int. J. Adv. Robot. Syst.* **13**(3), 1–13 (2016)
20. Yoon, S., Kim, Y., Kim, S.: Pursuit guidance law and adaptive backstepping controller design for vision-based net-recovery UAV. In: AIAA Guidance, Navigation and Control Conference and Exhibit, AIAA, Honolulu, HI, August, pp. 1–33 (2008)

Fast and Power-Efficient Embedded Software Implementation of Digital Image Stabilization for Low-Cost Autonomous Boats

S. Aldegheri, D. D. Bloisi, J. J. Blum, N. Bombieri and A. Farinelli

Abstract The use of autonomous surface vehicles (ASVs) is an efficient alternative to the traditional manual or static sensor network sampling for large-scale monitoring of marine and aquatic environments. However, navigating natural and narrow waterways is challenging for low-cost ASVs due to possible obstacles and limited precision global positioning system (GPS) data. Visual information coming from a camera can be used for collision avoidance, and *digital image stabilization* is a fundamental step for achieving this capability. This work presents an implementation of an image stabilization algorithm for a heterogeneous low-power board (i.e., NVIDIA Jetson TX1). In particular, the paper shows how such an embedded vision application has been configured to best exploit the CPU and the GPU processing elements of the board in order to obtain both computation performance and energy efficiency. We present qualitative and quantitative experiments carried out on two different environments for embedded vision software development (i.e., OpenCV and OpenVX), using real data to find a suitable solution and to demonstrate its effectiveness. The data used in this study is publicly available.

1 Introduction

Autonomous surface vehicles (ASVs), also known as Autonomous Surface Crafts (ASCs), are marine drones that can operate without the direct guidance of humans [6, 10]. The use of ASVs for performing large-scale monitoring of marine and aquatic

S. Aldegheri (✉) · D. D. Bloisi · J. J. Blum · N. Bombieri · A. Farinelli
Department of Computer Science, University of Verona, Verona, Italy
e-mail: stefano.aldegheri@univr.it

D. D. Bloisi
e-mail: domenico.bloisi@univr.it

J. J. Blum
e-mail: jason.blum@univr.it

N. Bombieri
e-mail: nicola.bombieri@univr.it

A. Farinelli
e-mail: alessandro.farinelli@univr.it

Fig. 1 IntCatch 2020 project uses Platypus Lutra boats, about 1 m long and 0.5 m wide



environments is receiving increasing attention, as it represents an efficient alternative to manual or static sensor network sampling for persistent environmental monitoring [5]. ASVs are capable of undertaking long-endurance missions and carrying multiple sensors to collect data about water quality indicators (e.g., temperature and dissolved oxygen) [3].

Figure 1 shows an example ASV specifically developed for water quality monitoring. It is a Lutra mono-hull boat produced by Platypus,¹ which can mount submerged propellers or an air fan for propulsion. Lutra boats are used in the EU-funded project IntCatch2020,² which will develop efficient and user-friendly monitoring strategies for facilitating sustainable water quality management by community groups and non-governmental organizations (NGOs). The innovative approaches developed within the IntCatch project will be tested and validated at sites including different scenarios, ranging from large lakes (i.e., Lake Garda in Italy) to small rivers (i.e., a rural river catchment in the east of England).

The possible presence of obstacles (above-, below-, and on-surface) and limited precision global positioning system (GPS) data are two of the main challenges for safe navigation of low-cost ASVs in narrow waterways. Due to cost limitations, visual information coming from an on-board camera can be used as a cheaper alternative to radar [13] and sonars, to efficiently deal with both the challenges [6], and *digital image stabilization* is a preliminary and fundamental step for achieving this capability. Although hardware solutions for damping the motion of the camera (e.g., gyroscopic stabilizers) exist and work well in practice, they are too expensive to be mounted on a low-cost boat.

In this work, we develop a software solution to the digital image stabilization problem designed for low-cost ASVs. In particular, we present a software implementation of the stabilization algorithm that is:

- Efficient in terms of performance as well as power consumption. This is enabled by heterogeneous (i.e., CPU- and GPU-equipped) low-power embedded systems.

¹senseplatypus.com.

²www.intcatch.eu.

- Power-scalable, allowing for obtaining a good performance to energy consumption ratio. We can exploit dual-mode operation (i.e., performance-oriented or energy-saving mode), switching from high frame rate to a lower one to trade accuracy for energy efficiency. For example, in open water reducing accuracy is acceptable, so throttling the algorithm to save energy is desirable.

The main contribution of this work consists of the qualitative and quantitative analysis of different configurations of the stabilization software. This paper presents the results obtained by implementations using two development environments (OpenCV³ and OpenVX⁴) and the corresponding libraries for embedded vision software. As demonstrated by the experiments we can obtain a sublinear growth in power consumption as the processed frame rate approaches 60 frames per second.

The remainder of this paper is structured as follows. Background information is provided in Sect. 2, together with an analysis of related work. The proposed method is presented in Sect. 3, while qualitative and quantitative experimental results are shown in Sect. 4. Finally, conclusions are drawn in Sect. 5.

2 Background and Related Work

Surface robotic platforms can be an invaluable cost-effective tool for several important applications ranging from flood mitigation to environmental monitoring [16]. While Autonomous Underwater Vehicles (AUVs) [12] and Unmanned Surface Vehicles (USV) [9] are often employed for data collection, such vehicles are typically expensive and require a significant logistical effort in terms of personnel and equipment for transportation and operation. The interest towards simple, low-cost surface robotic platforms for water quality monitoring is now significantly increasing with the advent of relatively cheap solutions, including NUSwan⁵ from Singapore, the ARC boats⁶ and the Platypus system. In particular, the Platypus autonomous boats are used in the EU-funded IntCatch project, which will demonstrate the use of low-cost ASVs integrated with sensors for water quality monitoring.

Figure 2 shows the functional architecture of the IntCatch system.

The boat can be controlled with a Wi-Fi connected tablet. The user can define a way-point path on the tablet that the boat follows, navigating autonomously. The tablet app generates a spiral path between the way-points to collect sensor data in the area. The data collected by the boat, including dissolved oxygen, electrical conductivity and temperature, can be processed to infer water quality.

A possible extension for the IntCatch system consists of mounting a camera on the bow of the boat to capture visual data (see Fig. 1). An important feature required by the

³opencv.org.

⁴www.khronos.org/openvx.

⁵<http://www.arl.nus.edu.sg/twiki6/bin/view/ARL/Swan>.

⁶www.hrwallingford.com/expertise/arc-boat.

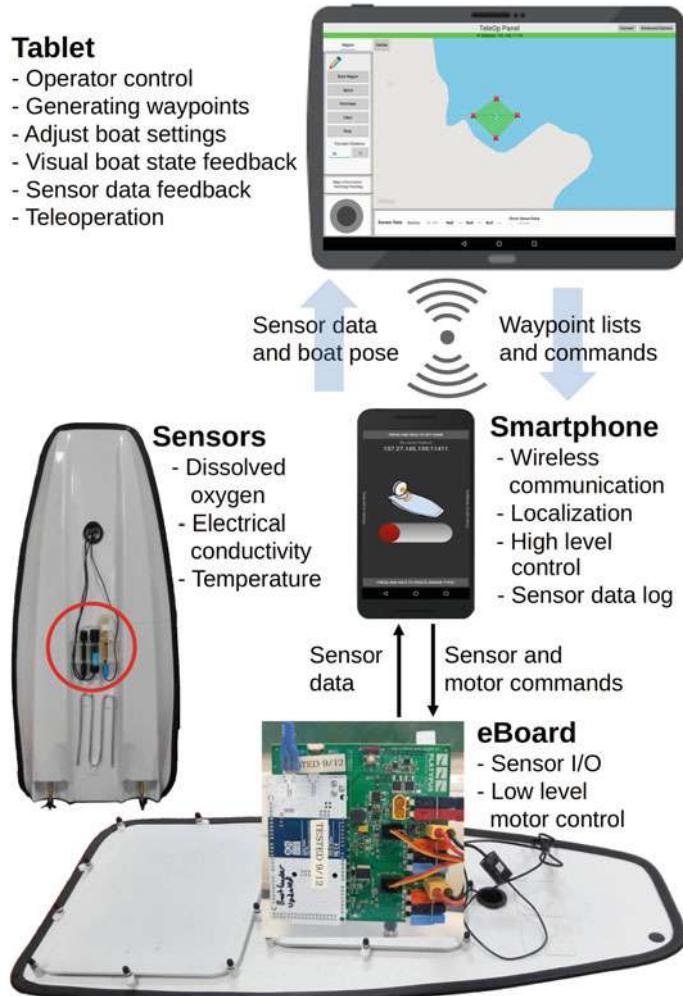


Fig. 2 Overall scheme of the IntCatch 2020 autonomous boat for water quality monitoring. We mounted different sensors to measure electrical conductivity, temperature and dissolved oxygen to generate maps of geolocalized data

IntCatch system is the capability of avoiding obstacles. Obstacles in the operational environment can be the boundary of the water or floating debris, which presents a significant challenge to continuous detection from images taken on-board [2]. Stereo rigs are used by Huntsberger et al. [11] for obstacle and moving objects detection on an ASV. However, since a large baseline is required for granting a large field of view, this can create instability for small vessels. A method for detecting water regions in videos by clustering color and texture features is proposed by Santana et al. [15]. However, color-based methods can suffer from significant variation of

the appearance of water, depending upon the color of the sky, the level of turbidity, the time of day, and the presence of wind, terrain reflections, underwater objects visible from the surface, surface vegetation, and shadows [14]. Fefilatyev et al. [8] use the horizon line position to eliminate all edges not belonging to floating objects by assuming that within an image all objects of interest lie above the horizon line. A similar idea is exploited by Wang et al. [19]. After detecting the horizon line, potential obstacles are searched in the region below the horizon. The main drawback of approaches based on the horizon line detection is that situations close to the shore cannot be easily handled, since the edge of the water does not always correspond to a simple horizon line.

In any of the above algorithms it is helpful to stabilize the images coming from the camera, mitigating the effect of physical motions of the sensor (e.g., caused by waves). Solutions involving hardware for damping the motion of the camera, e.g., gyroscopic stabilizers, cannot be used in the case of low-cost ASVs due to cost restrictions. Even if this equipment works well in practice, it is expensive. In this paper, we propose the use of an *embedded vision* system for digital image stabilization. With the term embedded vision, we refer to the deployment of practical computer vision methods on new-generation high-performance, low-cost, and energy-efficient embedded systems.

2.1 Low-Power Embedded Vision

In this work we adopted the *NVIDIA Jetson TX1* system-on-module, which is based on the Tegra X1 chip. Tegra X1 is one of the latest NVIDIA system-on-chip processor for embedded applications. It includes four ARM Cortex-A57 cores and a 256-core GPU based on the NVIDIA Maxwell architecture. For compute-intensive applications, the GPU is the most interesting feature of this chip, which can be programmed (through CUDA) to perform a wide range of parallel computation tasks in addition to handling 3D graphics.

OpenCV is a popular open source library of primitives for computer vision. It consists of a large set of primitives that can be executed on a CPU, and a subset of primitives implemented in CUDA and OpenCL to be accelerated on a GPU. *OpenCV4Tegra* is a closed-source porting of OpenCV provided by NVIDIA optimized specifically for the Tegra architecture. OpenCV4Tegra provides excellent compatibility with OpenCV, thus promising seamless porting of code developed with the open source OpenCV library.

OpenVX is an open standard from the Khronos Group aimed at enabling low power computer vision applications in mobile and embedded devices. OpenVX consists of a software framework and a library of common computer vision building blocks. The framework allows developers to describe their computer vision algorithms in the form of a dataflow graph. The framework can then execute the algorithm with dataflow

optimized for the device architecture. For example, the OpenVX framework can automatically apply *node-to-computing element* mapping or *image tiling* techniques to greatly reduce bandwidth to off-chip memory, improving speed and reducing power consumption.

VisionWorks is a software development package for computer vision and image processing based on OpenVX and provided by NVIDIA for Tegra architectures.

Different works have been presented to analyse the use of OpenVX for embedded vision [4, 18, 20]. In [20], the authors present a new implementation of OpenVX directed at platforms comprised of CPUs and GPUs that leverages various analytical techniques. In [4], the authors examine how OpenVX responds to different data access patterns, by testing three different OpenVX optimizations: kernels merge, data tiling and parallelization via OpenMP. In [18], the authors introduce ADRENALINE, a novel framework for fast prototyping and optimization of OpenVX applications for heterogeneous SoCs with many-core accelerators.

We present an optimized implementation of a real computer vision application embedded on a Jetson TX1 board and the optimization analysis targeting performance and energy consumption. In particular, we present an analysis of the study we conducted by comparing the implementation performance of OpenCV (with its optimization for the Tegra architecture, NVIDIA OpenCV4Tegra) and OpenVX (with its optimization for such a board NVIDIA VisionWorks).

3 Method

In this work, we aim at implementing the digital image stabilization of a visual stream captured by a camera mounted on a small ASV. An unstabilized video is an image sequence that exhibits unwanted perturbations in the apparent image motion. The goal of digital video stabilization is to improve the video quality by removing unwanted camera motion while preserving the dominant motions in the image sequence. For obtaining an obstacle detection solution, stabilization is a crucial pre-processing step before performing higher-level processing like object tracking. For example, the accuracy of predicted object trajectories can decrease in the case of unstabilized images [7].

Figure 3 shows an overview of the adopted video stabilization algorithm, which is represented through a dependency graph. The input sequence of frames is taken from a high-definition camera, and each frame is converted to the grayscale format to improve the algorithm efficiency without compromising the quality of the result. A *remapping* operation is then applied to the resulting frames to remove fish-eye distortions. A sparse optical flow is applied to the points detected in previous frame by using a feature detector (e.g., Harris or FAST detector). The resulting points are then compared to the original point to find the homography matrix. The last N matrices are then combined by using a Gaussian filtering, where N is defined by the user (higher N means more smoothed trajectory a the cost of more latency). Finally, each frame is inversely warped to get the final result. Dashed lines in Fig. 3 denote

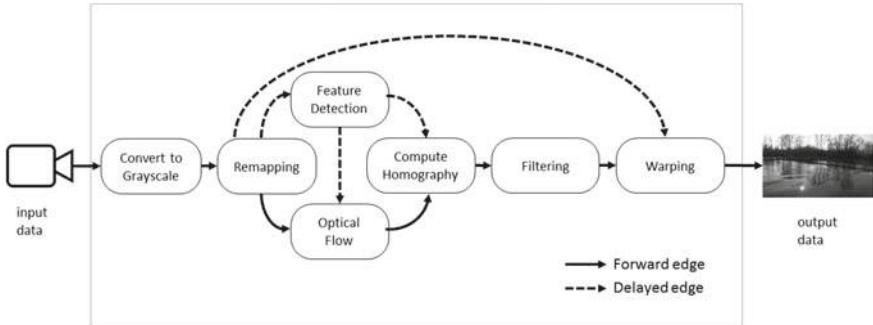


Fig. 3 Dependency graph of the video stabilization algorithm

inter-frame dependencies, i.e., parts of the algorithm where a temporal window of several frames is used to calculate the camera translation.

Although this algorithm does not represent particular challenges for the sequential implementation targeting CPU-based embedded systems, it presents a large design space to be explored when implemented for hybrid CPU-GPU systems. On the one hand, several primitives of the algorithm (graph nodes) can benefit from GPU acceleration while, on the other hand, their *offloading* on GPU involves additional memory-transfer overhead. The mapping exploration between nodes and computational elements (i.e., CPU or GPU) is thus crucial both for the performance and for the energy consumption.

To best explore correctness, performance, and energy consumption of the algorithm, we implemented the software in all the possible configurations (nodes vs. CPU/GPU) and by adopting both OpenCV and OpenVX design environments.

3.1 OpenCV Implementation

OpenCV provides the implementation for CPU of all the primitives of the video stabilization algorithm. In addition, it provides the GPU implementation of the following five nodes:

- Convert to Grayscale.
- Remapping.
- Feature detection (either based on Harris or FAST algorithm).
- Optical Flow.
- Warping.

The complete design space exploration of OpenCV consists of 32 configurations with the Harris-based feature detection plus 32 configurations with the FAST-based one. We exhaustively implemented and compared all the possible configurations. We also conducted the system-level optimization for each configuration, which, by

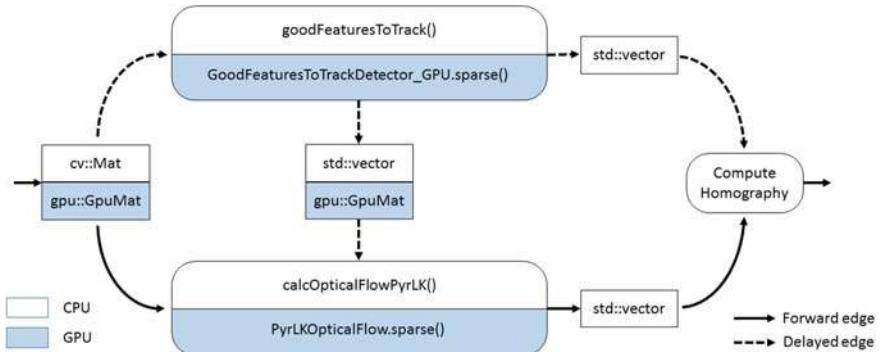


Fig. 4 OpenCV implementation of the most computational demanding nodes of the video stabilization algorithm

adopting OpenCV, is a manual and time consuming task. Indeed, although any single function downloading on GPU requires a quite straightforward code intervention (i.e., a function signature replacement), the system-level optimization involves a more accurate and time consuming analysis of the CPU-GPU data dependency in the overall data flow. As an example, consider the three nodes *feature detection*, *compute homography*, and *optical flow*. Any configuration requiring the first mapped on the CPU and the others on the GPU involves one data transfer from the CPU main memory (the output of the feature detection) to the GPU main memory (as input for either the optical flow or homography). A second (useless) CPU-GPU data transfer leads, in this algorithm implementations, to a 15% performance loss. Finding such data dependency and optimizing all the CPU-GPU data transfer, in the current OpenCV release, is let to the programmer.

The profiling analysis of all these code versions underlines that the *feature detection* and *optical flow* nodes are the two most computational demanding functions of the algorithm. Figure 4 depicts their OpenCV structure, by underlining how they are implemented (in terms of data exchange structures and the primitive signature) if run on the CPU or offloaded on the GPU. Their mapping on CPU or GPU involves the main important differences from the performance and power consumption point of view, as shown in Sect. 4.

Even though OpenCV primitives for GPUs are implemented both in OpenCL and CUDA, only CUDA implementations can be adopted for the Jetson board.

3.2 OpenVX Implementation

Listing 1 shows the most important parts of the OpenVX code. Some primitives have been tested both with the version released in the standard OpenVX library and in the VisionWorks library.

```

1 vx_context context = vxCreateContext();
2 /* create data structure */
3 vx_image gray = vxCreateVirtualImage(graph, 0, 0,
4 VX_DF_IMAGE_U8);
4 vx_image rect_image = vxCreateVirtualImage(graph, 0, 0,
5 VX_DF_IMAGE_U8);
6 vx_array curr_list = vxCreateVirtualArray(graph,
7 VX_TYPE_KEYPOINT, 1000);
6 vx_matrix homography = vxCreateMatrix(context,
8 VX_TYPE_FLOAT32, 3, 3);
7 /* create graph and relative structure */
8 vx_graph graph = vxCreateGraph(context);
9 vxColorConvertNode(graph, frame, gray);
10 vx_node remap_node = vxRemapNode(graph, gray, rect_image,
11 VX_INTERPOLATION_BILINEAR, remapped);
11 nvxCopyImageNode(graph, rect_image, out_frame, 0));
12 vxGaussianPyramidNode(graph, remapped, new_pyramid);
13 vx_node opt_flow_node = vxOpticalFlowPyrLKNode(graph,
14 old_pyramid, new_pyramid, points, points, curr_list,
15 VX_TERM_CRITERIA_BOTH, s_lk_epsilon, s_lk_num_iters,
16 s_lk_use_init_est, s_lk_win_size);
14 nvxFindHomographyNode(graph, old_points, curr_list,
15 homography, NVX_FIND_HOMOGRAPHY_METHOD_RANSAC, 3.0f,
16 2000, 10, 0.995f, 0.45f, mask);
15 homographyFilterNode(graph, homography, current_mtx,
16 curr_list, frame, mask);
16 matrixSmoothenNode(graph, matrices, smoothed);
17 truncateStabTransformNode(graph, smoothed, truncated, frame,
18 s_crop);
18 vxWarpPerspectiveNode(graph, frame_out_sync, truncated,
19 VX_INTERPOLATION_TYPE_BILINEAR, frame_stabilized);
19 nvxHarrisTrackNode(graph, rect_image, new_points, NULL,
20 curr_list, 0.04, 3, 18, NULL);
20 /* forced nodes to GPU */
21 vxSetNodeTarget(remap_node, NVX_TARGET_GPU, NULL);
22 /* forced node to CPU */
23 vxSetNodeTarget(opt_flow_node, NVX_TARGET_CPU, NULL);

```

Listing 1 OpenVX Code Example

The programming flow starts by creating a context (line 1). Based on this context, the program builds the graph (line 8) and the corresponding data objects (lines 3–6). The whole algorithm is then finalized as a dataflow graph by linking data objects through nodes (lines 9–19). Lines 21 and 23 show how processing nodes can be manually forced to be mapped on specific processing elements.

The OpenVX environment allows automatically changing the nodes-to-processing elements mapping and the corresponding data exchange system-level optimization. It also provides both the Harris-based and FAST-based feature detector, both available for CPU and GPU. In particular, it provides two different versions for each primitive: the first one is the standard “general purpose” OpenVX version, while the second one is provided in the VisionWorks library and is optimized for tracking algorithms.

In order to verify the best configuration targeting performance and to figure out the best one targeting power efficiency, we developed and tested all the possible configurations by forcing the nodes-to-processing elements mapping.

In the OpenVX mapping exploration, we considered the following differences from OpenCV:

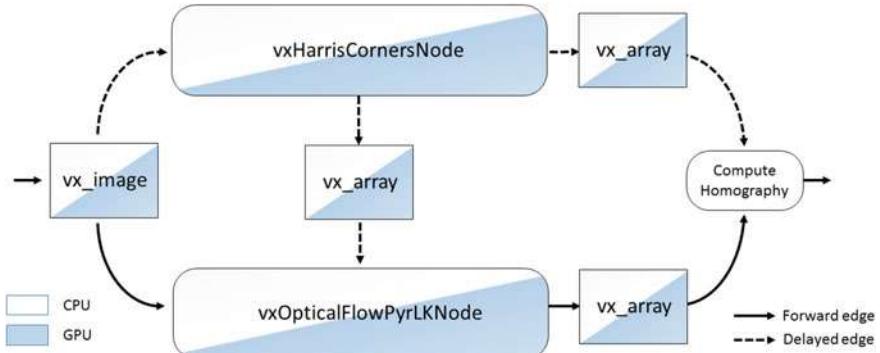


Fig. 5 OpenVX implementation of the most computational demanding nodes of the video stabilization algorithm

1. Harris/FAST tracker: OpenVX/VisionWorks provides the Harris/FAST tracker, which allows optimizing the data eflow by giving priority to the points tracked in the previous frame instead of the new detected ones, if they are in the same area.
2. OpenVX relies on column-major matrices, while OpenCV relies on major-row matrices. This is important especially in the remap cases, where the OpenCV backend is used to build the coordinates of the remapped points.
3. VisionWorks relies on a delay object to store an array of temporal objects (e.g., N frames back). This is not possible in OpenCV.

4 Experimental Results

Experiments have been carried out on real data collected in a small lake near Verona, Italy with a GoPro Hero 3 Black camera mounted on the bow of the Platypus Lutra boat (see Fig. 1). In particular, we analyzed three different image sequences (see Fig. 6), registered at 60 FPS with 1920×1080 wide angle resolution. Sequence S1 is particularly challenging due to large rolling movements, while Sequence S2 presents strong sun reflections on the water surface, and the boat is very close to the coast. The last Sequence S3 presents a similar view-point with respect to S1, but with lower rolling. The three sequences can be downloaded from the IntCach AI website,⁷ together with the source code of the different implementations that has been considered. Additional sequences, not analyzed in this work, can be downloaded from the *IntCatch Vision Data Set*.⁸

⁷<http://profsci.univr.it/bloisi/intcatchai/intcatchai.html>.

⁸<http://profsci.univr.it/bloisi/intcatchvisiondb/intcatchvisiondb.html>.

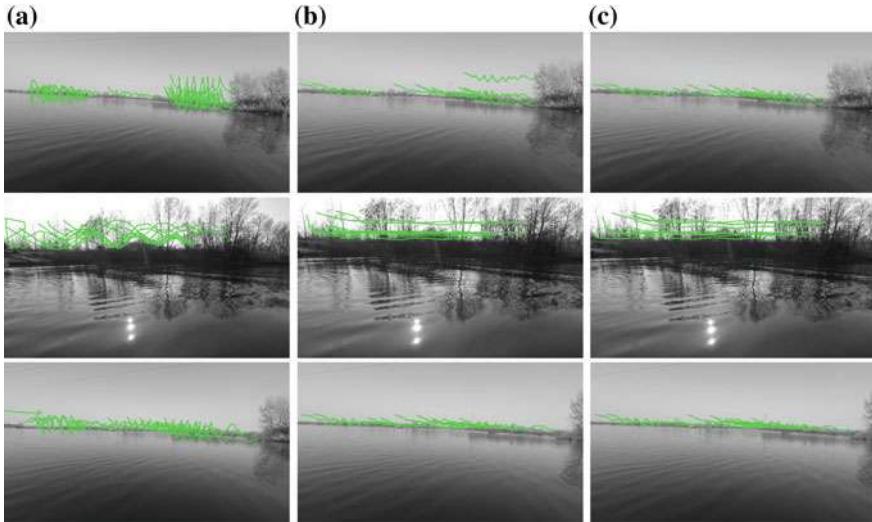


Fig. 6 Video stabilization results on sequences S1 (first row), S2 (second row), and S3 (third row). **a** A frame in the unstabilized video overlayed with lines representing point trajectories traced over time. **b** The corresponding frame in the OpenCV stabilized video. **c** The OpenVX stabilization results. Point trajectories are significantly smoother when the stabilization is activated

Stabilization results. In order to show the importance of video stabilization as a necessary preprocessing step for horizon line detection, we have considered the three sequences in Fig. 6, using them as input for a feature tracking algorithm. Supplemental videos can be downloaded at goo.gl/mg4fH1 for a clearer demonstration of our results.

We used the well-known Kanade-Lucas-Tomasi feature tracker (KLT) for tracking points around the horizon line in the camera field of view. The obtained feature points are visualized by tracing them through time [17]. The first column in Fig. 6 contains stabilization results when the input consists of images that have not been stabilized. The second and the third column show the results generated by the OpenCV and OpenVX based implementation, respectively.

We use the median absolute deviation with median as central point (*MAD*) to estimate the stabilization quality by measuring the statistical dispersion of the points generated by the KLT algorithm:

$$MAD = \text{median}(|x_i - \text{median}(X)|) \quad (1)$$

where $X = \{x_1, x_2, \dots, x_n\}$ is the n original observations.

Table 1 shows the *MAD* values obtained with the unstabilized images and with the preprocessed data generated by OpenCV and OpenVX. The stabilization allows to obtain lower *MAD* values. It is worth noticing that the alternatives generated by the same environment (i.e., OpenCV or OpenVX) give comparable results in terms

Table 1 Stabilization quantitative results

Sequence	Number of tracked points	MAD		
		Unstabilized	OpenCV	OpenVX
S1	10	31.5	11.3	10.4
S2	10	25.0	8.2	6.8
S3	10	23.4	10.4	10.7

Table 2 OpenCV implementation results

Remap	Cvt color	Warping	Optical flow	Features	P_{avg} (W)	P_{peak} (W)	FPS	$E(T_{end})$ (J)
GPU	GPU	GPU	GPU	CPU/HARRIS	5.1	11	7.5	772
GPU	GPU	GPU	GPU	GPU/FAST	5.7	15	16.3	855
GPU	GPU	GPU	GPU	GPU/HARRIS	6.0	17	15.3	906
GPU	GPU	GPU	GPU	CPU/FAST	5.3	15	13.7	810

of MAD , while differ in terms of performance and power consumption as discussed in the next paragraph.

Computational load results. Multiple mappings between routines and processing elements have been evaluated using different metrics. Live capture has been simulated from recorded video stored in the Jetson internal memory by skipping the frames in accordance with the actual processing speed of the considered implementation.

Tables 2 and 3 show the obtained results in terms of frames per second (FPS) together with the following metrics:

$$P_{peak} = \max_t P(t) \quad (2)$$

$$E(t) = \int_0^t P(t) \quad (3)$$

$$P_{avg} = \frac{E(T_{end})}{T_{end}} \quad (4)$$

where $P(t) = V(t)I(t)$ is the instant power usage and T_{end} is the total duration of the analyzed sequence in seconds. Table 3 has an asterisk at the beginning of rows which correspond to configurations that have been chosen by Visionworks automatically. The value $E(T_{end})$ denotes the absolute battery consumption measured in Joule (J), P_{avg} gives a measure of the average instant power usage during computation measured in Watt (W), and P_{peak} is the maximum instant power usage in Watt.

A Powermon board [1] has been used to measure the energy consumption. The absolute value of energy is an important measure that determines the battery capacity required for a mobile platform to perform the work. The number of processed frames

Table 3 OpenVX implementation results

	Remap	Cvt color	Warping	Optical flow	Features	P_{avg} (W)	P_{peak} (W)	FPS	$E(T_{end})$ (J)
	GPU	GPU	GPU	CPU	GPU/HARRIS OPENVX	5.4	12	36.5	810
	GPU	GPU	GPU	CPU	GPU/FAST VISION-WORKS	5.1	11	15.3	760
	GPU	GPU	GPU	GPU	CPU/FAST VISION-WORKS	4.3	11	16.2	639
*	GPU	GPU	GPU	GPU	GPU/FAST OPENVX	5.3	12	60	804
*	GPU	GPU	GPU	GPU	GPU/HARRIS OPENVX	5.2	11	60	776
*	GPU	GPU	GPU	GPU	GPU/FAST VISION-WORKS	5.2	12	60	780
*	GPU	GPU	GPU	GPU	GPU/HARRIS VISION-WORKS	5.1	11	60	764

is equally important to understand how well the algorithm performs. It is worth noticing that the OpenCV implementation is not able to achieve real-time performance with high-resolution high-frame rate data.

We investigated the maximum value of FPS that different configurations can reach and the corresponding power consumption. We also investigated how the power consumption scales with the performance. Since no OpenCV implementation allows achieving real-time performance (60 FPS), we used a high resolution timer to simulate the actual capture rate of the camera.

The results underline that the best OpenCV implementation allows reaching 16.3 FPS at the cost of 855 J energy consumption. They also underline that OpenCV does not allow implementing a dual performance-oriented or energy-saving mode, since the most appropriate energy saving configuration (-10% J) provides very insufficient performance. In contrast, OpenVX allowed us to implement four different configurations that guarantee appropriate performance (60 FPS) and energy scaling over FPS, as underlined in Fig. 7. Here, a linear increase in frames per second can be considered as a linear increase in computational load.

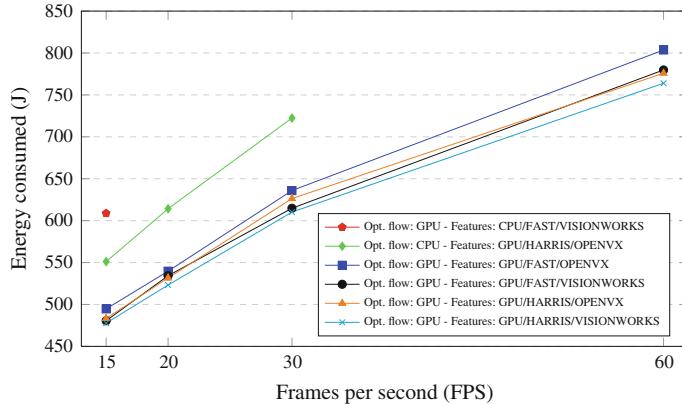


Fig. 7 OpenVX energy scaling per FPS

5 Conclusions

Low-cost autonomous surface vehicles (ASVs) can provide effective and efficient tools to help human operators performing water monitoring operations in lakes and rivers. However, their use in natural narrow waterways is challenging due to the possible presence of obstacles and limited precision GPS data.

In this paper, we investigate different solutions for digital image processing that can operate online in low-cost ASVs. In particular, we consider image stabilization, which is a key preprocessing step for building a horizon line detection module. Our focus is to provide solutions that can manage high frame rate images in real time so to allow the system to react when obstacles appears in the field of view of the on-board camera. To this end, we implement state of the art approaches in two development environments (OpenCV and OpenVX), investigating different mappings among the routines (e.g., warping, optical flow, etc.) and processing elements (i.e., CPU, GPU). Overall our results show that by using appropriate mappings we can run such state of the art approaches on an embedded system (NVIDIA Jetson TX1) achieving a high processing rate (i.e., 60 frames per second) while reducing the battery consumption (with respect to a traditional CPU based implementation).

These results pave the way for several interesting research directions. In particular, our future work in this space includes the implementation and evaluation of situation assessment and obstacle avoidance methods that exploit the embedded system and development environments investigated in this work.

Acknowledgements This work is partially funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 689341.

References

1. Bedard, D., Lim, M.Y., Fowler, R., Porterfield, A.: Powermon: fine-grained and integrated power monitoring for commodity computer systems. In: IEEE SoutheastCon, pp. 479–484 (2010)
2. Bloisi, D., Pennisi, A., Iocchi, L.: Background modeling in the maritime domain. *Mach. Vis. Appl.* **25**(5), 1257–1269 (2014)
3. Codiga, D.L.: A marine autonomous surface craft for long-duration, spatially explicit, multidisciplinary water column sampling in coastal and estuarine systems. *J. Atmos. Ocean. Technol.* **32**(3), 627–641 (2015)
4. Dekkiche, D., Vincze, B., Merigot, A.: Investigation and performance analysis of OpenVX optimizations on computer vision applications. In: International Conference on Control, Automation, Robotics and Vision, pp. 1–6 (2016)
5. Dunbabin, M., Grinham, A.: Quantifying spatiotemporal greenhouse gas emissions using autonomous surface vehicles. *J. Field Robot.* **34**(1), 151–169 (2017)
6. El-Gaaly, T., Tomaszewski, C., Valada, A., Velagapudi, P., Kannan, B., Scerri, P.: Visual obstacle avoidance for autonomous watercraft using smartphones. In: Autonomous Robots and Multirobot Systems workshop (2013)
7. Elliott, G.A., Yang, K., Anderson, J.H.: Supporting real-time computer vision workloads using OpenVX on Multicore+GPU platforms. In: Real-Time Systems Symposium, pp. 273–284 (2015)
8. Fefilatyev, S., Goldgof, D., Lembke, C.: Tracking ships from fast moving camera through image registration. In: International Conference on Pattern Recognition, pp. 3500–3503 (2010)
9. Ferri, G., Manzi, A., Fornai, F., Ciuchi, F., Laschi, C.: The HydroNet ASV, a small-sized autonomous catamaran for real-time monitoring of water quality: from design to missions at sea. *IEEE J. Ocean. Eng.* **40**(3), 710–726 (2015)
10. Giordano, F., Mattei, G., Parente, C., Peluso, F., Santamaria, R.: Integrating sensors into a marine drone for bathymetric 3D surveys in shallow waters. *Sensors* **16**(1) (2016)
11. Huntsberger, T., Aghazarian, H., Howard, A., Trotz, D.C.: Stereo vision-based navigation for autonomous surface vessels. *J. Field Robot.* **28**(1), 3–18 (2011)
12. Kalwa, J., Carreiro-Silva, M., Tempera, F., Fontes, J., Santos, R.S., Fabri, M.C., Brignone, L., Rida, P., Birk, A., Glotzbach, T., Caccia, M., Alves, J., Pascoal, A.: The morph concept and its application in marine research. In: MTS/IEEE OCEANS, pp. 1–8 (2013)
13. Pisa, S., Bernardi, P., Cavagnaro, M., Pittella, E., Piuzzi, E.: Monitoring of cardio-pulmonary activity with UWB radar: A circuital model. In: Asia-Pacific Symposium on Electromagnetic Compatibility and 19th Int. Zurich Symposium on Electromagnetic Compatibility, pp. 224–227 (2008)
14. Rankin, A., Matthies, L.: Daytime water detection based on color variation. In: International Conference on Intelligent Robots and Systems, pp. 215–221 (2010)
15. Santana, P., Mendona, R., Barata, J.: Water detection with segmentation guided dynamic texture recognition. In: International Conference on Robotics and Biomimetics, pp. 1836–1841 (2012)
16. Scerri, P., Kannan, B., Velagapudi, P., Macarthur, K., Stone, P., Taylor, M., Dolan, J., Farinelli, A., Chapman, A., Dias, B., Kantor, G.: Flood Disaster Mitigation: A Real-World Challenge Problem for Multi-agent Unmanned Surface Vehicles, pp. 252–269. Springer (2012)
17. Smith, B.M., Zhang, L., Jin, H., Agarwala, A.: Light field video stabilization. In: International Conference on Computer Vision, pp. 341–348 (2009)

18. Tagliavini, G., Haugou, G., Marongiu, A., Benini, L.: Adrenaline: an OpenVX environment to optimize embedded vision applications on many-core accelerators. In: International Symposium on Embedded Multicore/Many-core Systems-on-Chip, pp. 289–296 (2015)
19. Wang, H., Wei, Z., Wang, S., Ow, C.S., Ho, K.T., Feng, B.: A vision-based obstacle detection system for unmanned surface vehicle. In: International Conference on Robotics, Automation and Mechatronics, pp. 364–369 (2011)
20. Yang, K., Elliott, G.A., Anderson, J.H.: Analysis for supporting real-time computer vision workloads using OpenVX on Multicore+GPU platforms. In: International Conference on Real Time and Networks Systems, RTNS '15, pp. 77–86 (2015)

Evaluation of Combined Time-Offset Estimation and Hand-Eye Calibration on Robotic Datasets

Fadri Furrer, Marius Fehr, Tonci Novkovic, Hannes Sommer, Igor Gilitschenski and Roland Siegwart

Abstract Using multiple sensors often requires the knowledge of static transformations between those sensors. If these transformations are unknown, hand-eye calibration is used to obtain them. Additionally, sensors are often unsynchronized, thus requiring time-alignment of measurements. This alignment can further be hindered by having sensors that fail at providing useful data over a certain time period. We present an end-to-end calibration framework to solve the hand-eye calibration. After an initial time-alignment step, we use the time-aligned pose estimates to perform the static transformation estimation based on different prefiltering methods, which are robust to outliers. In a final step, we employ a non-linear optimization to locally refine the calibration and time-alignment. Successful application of this estimation framework is demonstrated on multiple robotic systems with different sensor configurations. This framework is released as open source software together with the datasets.

1 Introduction

The hand-eye calibration problem is among the most important calibration scenarios in robotics. Its name refers to the problem of calibrating the pose of a camera coordinate system relative to the reference frame of the robot arm’s end effector on which it is rigidly mounted. Another important instance of the problem is infer-

F. Furrer (✉) · M. Fehr · T. Novkovic · H. Sommer · I. Gilitschenski · R. Siegwart
Autonomous Systems Lab, ETH Zurich, Leonhardstrasse 21, 8092 Zurich, Switzerland
e-mail: fadri.furrer@mavt.ethz.ch

M. Fehr
e-mail: marius.fehr@mavt.ethz.ch

T. Novkovic
e-mail: tonci.novkovic@mavt.ethz.ch

H. Sommer
e-mail: igilitschenski@mavt.ethz.ch

R. Siegwart
e-mail: rsiegwart@mavt.ethz.ch

ring the relative pose of two sensors, such as cameras, even if their views do not overlap. More generally, hand-eye calibration systems aim at finding the transformation between two reference frames that are rigidly mounted with respect to each other.

Formally, solving the hand-eye calibration problem comes down to solving the $AX = XB$ equation in which A , B , and X represent rigid body motions. This formulation, originally proposed in [21], has been subject of an extensive body of research which focused on finding a solution to this equation. However, practical implementations of a hand-eye calibration system may present significant additional challenges. For instance, time-alignment needs to be taken into account when the two reference frames stem from sensors/actuators running on different systems. This is particularly true when these systems need to be (re-)calibrated online during regular operation and, therefore, cannot be specifically controlled for calibration.

For practical applications it is of particular interest to be able to solve the aforementioned problems within a single system making it widely applicable to different practical instances of the hand-eye calibration problem. Unfortunately, the broad body of research on the hand-eye calibration problem is not adequately matched by thorough evaluations in different scenarios and freely available software packages.

The goal of this work is to fill this gap by providing an open source toolbox¹ for hand-eye calibration that can be easily used within a broad range of applications and is at the same time easily adaptable to incorporating further algorithms and calibration procedures. Contributions presented in this paper not only involve the presentation of the software package but also its applicability to robotic systems. This is achieved through thorough evaluations on different types of datasets involving a robotic arm and multiple hand-held devices. Furthermore, we make all our datasets publicly available in order to simplify future evaluation of hand-eye calibration algorithms. The contributions of this work can be summarized as follows:

- A collection of datasets using different sensors and sensor configurations.
- Thorough validation of the hand-eye calibration system with different filtering methods on these datasets.
- A software toolbox for hand-eye calibration including time-alignment and handling noisy data.

The remainder of the paper is structured as follows. In the next section, we study related work. An overview of the methodology implemented in the proposed calibration toolbox is presented in Sect. 3. The datasets that are used for evaluation are presented in Sect. 4 followed by the validation of the proposed method. A conclusion with an outlook is provided in Sect. 6.

¹https://github.com/ethz-asl/hand_eye_calibration.

2 Related Work

The problem of hand-eye calibration has been well studied in late 80 and 90s. Classical approaches to solving $AX = XB$ problem decoupled rotational and translational parts of the calibration, resulting in simpler but more error-prone solutions [23]. Shiu and Ahmad [21] demonstrated how a hand-eye calibration problem can be expressed using an angle-axis representation and solved for rotation, then translation using a least squares fitting. Similar, but a more efficient approach was developed by Tsai and Lenz [25] using a closed-form solution. Wang [26] proposed another formulation using angle-axis representation and conducted an early comparison of the three methods, reporting that the one by Tsai and Lenz [25] performed the best on average. Formulation of the same problem using quaternions for rotations was introduced by Chou and Kamel [3]. Park and Martin [18] have formed an alternative closed-form solution using Lie group theory to simplify the problem, and Fassi and Legnani [6] demonstrated how to solve the calibration problem, in a least squares manner, first for rotation and then translation in presence of noisy data.

The same $AX = XB$ problem can be solved simultaneously for hand-eye rotation and translation. Horaud and Dornaika [13], in addition to proposing another closed-form solution using quaternions for the decoupled problem, also proposed an iterative method for solving the orientation (represented by quaternions) and translation components simultaneously. They applied a Levenberg-Marquardt technique, a robust non-linear optimization method, to obtain the solution. Furthermore, they performed a stability analysis for both of their approaches and the method proposed by Tsai and Lenz [25], concluding that the non-linear optimization method is the most robust with respect to measurement errors and noise, and much more accurate than the classical formulation by Tsai and Lenz [25]. Daniilidis [4] proposed another formulation, based on screw-theory, for the simultaneous hand-eye calibration. He obtained a singular value decomposition (SVD)-based solution by using a dual-quaternion representation for both rotations and translations. His work was extended by Schmidt et al. [20] who also implemented the screw-axis based selection of movement pairs for increasing numerical stability and random sample consensus (RANSAC)-based elimination of outliers. Another iterative method based on a parameterization of a stochastic model was introduced by Strobl and Hirzinger [23]. In order to evaluate optimality of different algorithms, they introduced a metric on the group of the rigid transformations $SE(3)$ and the corresponding error model for non-linear optimization.

Andreff and Espiau [2] demonstrated robot hand-eye calibration using structure-from-motion for computing camera motions, up to an unknown scale factor which is introduced in a linear formulation of the calibration problem. They also showed that their method is very accurate in rotation, however, for translations, in case of noisy data, other methods by Daniilidis [4] and Horaud and Dornaika [13] perform better. A modification to the structure-from-motion approach was presented by Heller et al. [12] which addresses the scale ambiguity by formulating the estimation of the hand-eye displacement as an L_∞ -norm optimization problem.

In most practical applications, in addition to estimating the hand-eye calibration, and due to asynchronous clocks from different devices, it is necessary to perform temporal alignment of the data. Ackerman et al. [1] used invariant quantities, coming from screw theory, between two pairs of measurements to align uniformly asynchronous data and account for data with gaps. Alignment was based on correlation of the measurement invariants using the Discrete Fourier Transform (DFT), however, the approach was evaluated only on simulated data. In their motion-based calibration method, Taylor and Nieto [24] compute the likelihood of a timing offset based on an angle through which each sensor rotates, taking the associated uncertainty into account. Additionally, using a probabilistic framework and based on the estimated motion of each individual sensor, estimated accuracy of each sensor's readings and appearance information, they compute the final calibration. Rehder et al. [19] have demonstrated a general framework, using a continuous-time state representation, for joint calibration of temporal offsets and spatial transformations between multiple sensors. In this approach, the time offset is estimated using basis functions which allows them to treat the problem within the rigorous theoretical framework of maximum likelihood estimation. An alternative approach, formulating the temporal calibration as a registration task, using an iterative closest point (ICP) algorithm, was introduced by Kelly and Sukhatme [14]. TICSync, an open source library implementing software for time-alignment was developed by Harrison and Newman [11]. They used a two-way timing mechanism to estimate the offset and realize unified and precise timing across distributed networked systems. Since sensors rarely have support for this two-way mechanism, another open-source framework, TriggerSync by English et al. [5] was developed based on TICSync library. This framework is used for synchronizing multiple triggered sensors with respect to the local clock.

Our approach is based on the method by Daniilidis [4] with a similar outlier rejection and motion selection as in Schmidt et al. [20]. However, our method incorporates several additional outlier rejection techniques, of which we prove that they can significantly improve the performance of the original algorithm. Furthermore, we perform initial time-alignment based on correlation between the angular velocities. As a final refinement step we perform non-linear maximum likelihood batch estimation with a continuous-time state representation as described in [9]. The overall approach for the this step is very similar to the one proposed in [19].

3 Method

In the presented work, we allow inputs to the hand-eye calibration to be pose estimations or camera images from which we estimate poses relative to a visual target. Therefore, we split this section into the subsections for target extraction, time-alignment, hand-eye calibration, and the refinement step. The pose estimations from camera images are described in Sect. 3.1. We use interpolated angular velocity norms for the time-alignment, which we describe in Sect. 3.2. With two timely aligned sets of poses, we perform the hand-eye calibration with outlier rejection, as described in

Sect. 3.3. Using the results from time-alignment and (global) hand-eye calibration as initial guesses, we additionally perform a final local refinement step using non-linear optimization to find a local, joint spatiotemporal maximum likelihood solution. This last step we describe in Sect. 3.3.2.

3.1 Target Extractor

In order to use the time-alignment and hand-eye calibration methods presented in the following sections, two sets of poses are required. There are several well known methods to estimate poses from camera images, based on feature matching, optical flow, etc. To avoid drift in the measurements it is beneficial to find features of an object that is known to be stationary in the environment. Camera pose estimates from matched features suffer from an unknown scale, in order to solve this issue, one can look for pairs of features with known metric distances, or use additional sensors with metric information, such as inertial measurement units, range sensors, radar, motor, or wheel encoders.

In our approach, we use visual *AprilTag* targets [17] of known size. We assume that the intrinsic camera calibrations are known.² When the target is visible in the camera frame, corner features are extracted. Additionally, by detecting *AprilTags* on the calibration target, the detected corners can produce one to one matches among different images. The successful observations of the target are appended to a vector. For all these observations we check, using a RANSAC based Perspective-n-Point method, if they agree with the camera model from the intrinsic calibration and extract a pose estimation of the camera (or eye) E , \mathbf{T}_{WE_i} , in the target (or world) frame W . If the corresponding inlier ratio λ_i is greater than an inlier threshold λ_{th} we keep the pose estimate \mathbf{T}_{WE_i} along with its timestamp t_{E_i} for the next steps.

3.2 Time Alignment

In order to compare poses originating from different sensors one can rarely rely on hardware-synchronized device clocks as the sensors might not be communicating at all, e.g. when calibrating a camera tracked by an external motion capture system. That is why the synchronization of the two sensor clocks, or to be more precise the two sets of timestamped sensor data is the first crucial step for hand-eye calibration. A popular method of computing the time-alignment for signals with constant time-offsets is to correlate the angular velocity norms of both pose signals. To that end, we first resample the poses at the lower frequency of the two pose signals and then compute the angular velocity norm based on both sets of quaternions. In order to

²For intrinsic camera calibration, the Kalibr framework (<https://github.com/ethz-asl/kalibr>) was used and we refer the reader to [7, 8, 16] for more details.

make the time-alignment more robust to outliers or missing data, e.g. caused by a signal drop, we first cap the signal at the 99th percentile of the magnitude and then apply a low-pass kernel. The time offset can then be computed from the maximum value of the convoluted signal. In order to provide feedback about the quality of the time-alignment, the user is presented a comprehensible graphical representation of the alignment results (see Fig. 1).

3.3 Hand-Eye Calibration

To perform a hand-eye calibration, at least two pose pairs are required. As depicted in Fig. 2, we can then solve the hand-eye calibration equation:

$$\mathbf{T}_{BH_1} \mathbf{T}_{HE} \mathbf{T}_{WE_1}^{-1} = \mathbf{T}_{BH_2} \mathbf{T}_{HE} \mathbf{T}_{WE_2}^{-1}, \quad (1)$$

where B is the body frame, H the hand frame, W and E , the world and eye frame introduced earlier. This can be reformulated using the transformation between consecutive poses of the two pose sources, using $\mathbf{T}_{H_1 H_2} = \mathbf{T}_{BH_2}^{-1} \mathbf{T}_{BH_1}$, and $\mathbf{T}_{E_1 E_2} = \mathbf{T}_{WE_2}^{-1} \mathbf{T}_{WE_1}$, respectively, into $\mathbf{T}_{H_1 H_2} \mathbf{T}_{HE} = \mathbf{T}_{HE} \mathbf{T}_{E_1 E_2}$.

In the context of this paper, the method presented in [4] was used for the hand-eye calibration. Therefore, we are solving the hand-eye calibration using dual quaternions $\dot{\mathbf{q}}_{H_1 H_2} = \dot{\mathbf{q}}_{HE} \dot{\mathbf{q}}_{E_1 E_2} \dot{\mathbf{q}}_{HE}^{-1}$. However, this method is sensitive to outlier and noise as it employs an SVD to solve the hand-eye calibration problem.

3.3.1 Filtering

To improve the robustness and the accuracy of the calibration results, we implemented and evaluated several outlier rejection and filtering methods. First, in order to reduce the amount of data points we need to process, we employ the following filtering technique:

Pose Filtering (PF) Since usual datasets can contain very large number of pose pairs for calibration, in the first step of our approach, we apply a filtering method based on [20]. This method first computes the screw motion axis of each dual quaternion

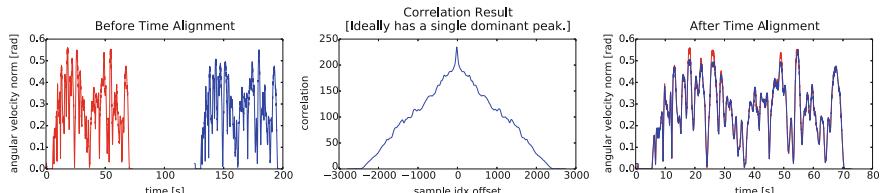
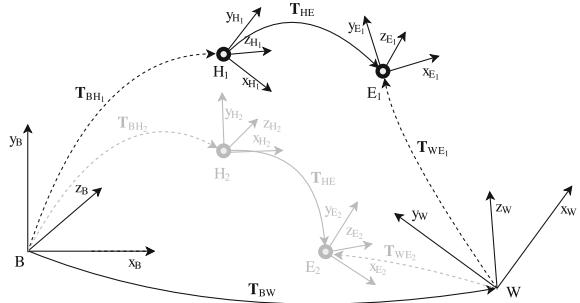


Fig. 1 Time-alignment result: The plot provides an intuitive understanding of the direction and magnitude as well as the quality of the alignment

Fig. 2 The transformations relevant for the hand-eye calibration. Black are the transformations of the first pose pair and in grey the second pose pairs. Solid lines indicate static transformations, where dashed lines indicate transformations that change over time



representing one transformation. The dot products for each combination of the screw-axis from the hand data, as well as the dot products of each combination of the eye data are computed. If one of these dot products is higher than a threshold then the respective hand-eye pair gets filtered out. The main idea for this filtering is that if the dot product is high, it means that the screw-axis for these transforms are almost parallel, meaning that they contain similar information and that we can filter one of them out since it is not so informative. Using this filtering method we can greatly reduce the number of pose pairs that will be passed to the calibration algorithm, thus, improving the efficiency, however, slightly reducing the accuracy.

RANSAC “Classic” (RC) In a first, step we reduce the number of dual quaternion pairs using the filtering method described above. RANSAC (see Algorithm 1) draws n ($n \geq 2$) time-aligned quaternion pairs at random, the *sample*. As described in [4], the scalar parts of two dual quaternions representing the same screw need to be equal in order for this method to succeed. We made use of this condition to first reject any *samples* that violate it early on. RANSAC then proceeds by identifying inliers that agree with the hand-eye calibration. Therefore, the standard way is to first estimate the calibration based on the drawn *samples*. This resulting calibration is then used to transform the quaternion pairs into the same base frame. We then compare their position and orientation errors which allows us to apply thresholds $\lambda_{t,min}$ (position) and $\lambda_{r,min}$ (orientation) to identify inliers and outliers. The calibration is refined by repeating the hand-eye calibration method on the inliers found in the previous step. We repeat the evaluation step we used to identify the inliers and compute the root mean square error (RMSE) of position and orientation across all the quaternion pairs. Finally, we keep the calibration that exhibits the lowest RMSE.

RANSAC Scalar (RS) based inlier check Furthermore, we propose and compare a second variant of this algorithm that employs a different, more efficient way of identifying inliers. We reduce the *sample* size to 1 and omit the *sample*-based hand-eye calibration computation and its expensive evaluation, and directly select the inliers based on the compatibility of the quaternion pairs’ scalar values. The algorithm then continues like the previous RANSAC variant by computing the calibration on the inliers and evaluating it based on the RMSE of position and rotation error of the aligned quaternion pairs.

We compare our proposed improvements to the following two algorithms.

- **Baseline (B):** Finds the first subset of quaternion pairs that fulfills the scalar value equality condition and compute the hand-eye-calibration.
- **Exhaustive search (EC and ES):** Is algorithmically identical to the proposed RANSAC algorithms, except that all possible sample combinations are explored. In order to keep the runtime within reasonable limits we employ this method only on the filtered set of quaternion pairs.

Algorithm 1: RANSAC based input pose pair selection for Eq. 1.

Data: A pair of vectors with time-aligned dual quaternions: $\mathbf{P}_{a,b} = [a, b]$
 $a = [\check{\mathbf{q}}_{a,1} \cdots \check{\mathbf{q}}_{a,k}]^T, \quad b = [\check{\mathbf{q}}_{b,1} \cdots \check{\mathbf{q}}_{b,k}]^T, \quad RMSE_{best} = \infty$

Result: Static transform dual quaternion $\check{\mathbf{q}}_{a,b}$ and corresponding RMSE

Function HandEyeCalibrationRANSAC($\mathbf{P}_{a,b}$)

```

 $\mathbf{F}_{a,b} \leftarrow \text{FilterPairs}(\mathbf{P}_{a,b}) \text{ // PF}$ 
While not reached probability of at least one inlier sample do
   $\mathbf{S}_{a,b} \leftarrow \text{SamplePairs}(\mathbf{F}_{a,b})$ 
  if not AllScalarPartsEqual( $\mathbf{S}_{a,b}$ ) then next ;
  if RC or EC then
     $\check{\mathbf{q}}'_{a,b} \leftarrow \text{ComputeHandEyeCalibration}(\mathbf{S}_{a,b})$ 
     $\mathbf{I}_{a,b} \leftarrow \text{GetInliersBasedOnPoseError}(\mathbf{F}_{a,b}, \check{\mathbf{q}}'_{a,b}, \lambda_{t,min}, \lambda_{r,min})$ 
  else
    // RS or ES
     $\mathbf{I}_{a,b} \leftarrow \text{GetInliersBasedOnScalarPartsEquality}(\mathbf{S}_{a,b}, \mathbf{F}_{a,b})$ 
  end
  if  $|\mathbf{I}_{a,b}| < \text{required number of inliers}$  then next ;
   $\check{\mathbf{q}}'_{a,b} \leftarrow \text{ComputeHandEyeCalibration}(\mathbf{I}_{a,b})$ 
   $(RMSE, \mathbf{I}_{a,b}) \leftarrow \text{EvaluatePairs}(\mathbf{P}_{a,b}, \check{\mathbf{q}}'_{a,b})$ 
  if  $RMSE < RMSE_{best}$  then
     $RMSE_{best} \leftarrow RMSE$ 
     $\check{\mathbf{q}}_{a,b} \leftarrow \check{\mathbf{q}}'_{a,b}$ 
  end
return  $(RMSE_{best}, \check{\mathbf{q}}_{a,b})$ 

```

3.3.2 Refinement Step

In Sects. 3.2 and 3.3 we address the global extrinsic spatiotemporal hand-eye calibration problem. However, the expected accuracy is limited mostly due to the fact that the assumed pose-trajectories are estimated individually and kept fix when aligning them to find the hand-eye calibration. A joint maximum likelihood optimization of calibration and trajectory given the measurements allows higher accuracy. This optimization is hard to solve as global problem but using the results from our global approach as an initial guess a local likelihood maximization can improve the accuracy

of the calibration. We perform this joint batch estimation step with a continuous-time representation for the trajectory, as in [9, 19], and overall very similar to what is described in [19]. Except for that we use Lie group valued B-splines, [22], to represent SO(3)-trajectories instead of vector space valued B-splines in an unconstrained parameter space of SO(3) [9, 19].³ To solve the non-linear optimization we use an extension of Levenberg-Marquardt to Lie groups (as documented e.g. in [22]).

More specifically, we model the joint problem with one trajectory for the moving hand frame, H , $\mathbf{T}_{BH}(t) =: \mathbf{X}(t)$. The eye-frame, E , is assumed to be rigidly connected to the hand frame by the spatial calibration, \mathbf{T}_{HE} , as depicted in Fig. 2. The pose measurement timestamps for H and E are assumed to be connected through a fixed time-offset Δt . Their errors we assume to be generated from isotropic multivariate Cauchy distributions⁴ with three degrees of freedom independently for both translation (with zero mean) and rotation (with identity mean)⁵ with respect to B and W -frame respectively. Or, if the eye is only emitting relative pose estimates (as, e.g., in visual inertial odometry), the same type of error source is assumed but with respect to the pose of the last measurement event. This yields the following negative log likelihood function, l , which we minimize:

$$\begin{aligned} l &(\mathbf{T}_{WE}, \mathbf{T}_{HE}, \Delta t, \mathbf{X} | (\mathbf{T}_{WEi}, t_{Ei})_{i=1}^k, (\mathbf{T}_{BH_i}, t_{Hi})_{i=1}^l) \\ &= \sum_{i=2}^{k_E} \rho(\|d_E(\mathbf{T}_{WEi-1}, \mathbf{T}_{WEi}, \mathbf{T}_{WE}(t_{Ei-1}), \mathbf{T}_{WE}(t_{Ei}))\|_{\Sigma_E}^2) \\ &\quad + \sum_{i=2}^{k_H} \rho(\|d_H(\mathbf{T}_{BH_i-1}, \mathbf{T}_{BH_i}, \mathbf{T}_{BH}(t_{Hi-1}), \mathbf{T}_{BH}(t_{Hi}))\|_{\Sigma_H}^2), \end{aligned} \quad (2)$$

where $\mathbf{T}_{WE}(t) := \mathbf{T}_{WB}\mathbf{T}_{BH}(t - \Delta t)\mathbf{T}_{HE}$, $\mathbf{T}_{BH}(t) = \mathbf{X}(t)$, $\rho(s) = \log(1 + s)$ the Cauchy-loss, and d_E and d_H are either relative, $(\mathbf{A}', \mathbf{A}, \mathbf{B}', \mathbf{B}) \mapsto d(\mathbf{A}'^{-1}\mathbf{A}, \mathbf{B}'^{-1}\mathbf{B})$, or absolute $(\mathbf{A}', \mathbf{A}, \mathbf{B}', \mathbf{B}) \mapsto d(\mathbf{A}, \mathbf{B})$.⁶ As displacement vector $d(\mathbf{A}, \mathbf{B}) \in \mathbb{R}^6$ on SE(3) we use coordinates of $(\log_{SO(3)}(\mathbf{R}), \mathbf{u})$ with respect to a fixed positive orthonormal basis, where \mathbf{u} is a translation and \mathbf{R} a proper rotation such that (uniquely) $\mathbf{u} \circ \mathbf{R} := \mathbf{A}^{-1}\mathbf{B}$. Please note that l becomes independent of \mathbf{T}_{WE} iff d_E is relative because then it cancels out in $\mathbf{B}'^{-1}\mathbf{B}$.

³Traditional B-splines in parameter space are not equivariant with respect to transformations of the world and body frame. Therefore, for a given trajectory the local expressiveness of such a representation typically depends on where the trajectory is in that segment. Furthermore, they can go through ambiguous or unstable regions of the parameter space. The Lie-group valued B-splines we use are bi-equivariant [22] and are neither locally ambiguous nor unstable.

⁴This is equivalent to least squares with a Cauchy loss function.

⁵Approximated with zero-mean Cauchy distributions in the Lie algebra projected to SO(3) using the exponential map.

⁶For absolute d_E, d_H the corresponding first measurements, $i = 1$, are assumed to be dummy variables while the real measurements start with $i = 2$.

4 Datasets

In the scope of this work, we evaluated our calibration framework on three different systems. Firstly, an RGB-D sensor with a visual target in an external tracking system. Secondly, a robot arm with an RGB-D sensor mounted close to the end effector. And, lastly, we have mounted three Tango tablets on a rigid profile, and used it for recording two datasets with different motions.

RGB-D-Sensor in external motion caption system: In the first experiments, we recorded color images from a PrimeSense RGB-D sensor, which was tracked using a Vicon tracking system. We placed a visual target in front of the camera to be able to use the camera pose estimation described in Sect. 3.1.

Robot Arm with RGB-D-Sensor: We recorded two datasets with a UR-10 robot arm equipped with a RealSense SR300 RGB-D sensor, mounted rigidly to a sensor mount close to the end effector. The first dataset is simulated and recorded in the Gazebo robotic simulator [15]. In this dataset, we can extract the ground truth hand-eye transformation from the setup of the robot model. The second dataset is a similar setup, but recorded on a real robot. In both, the simulation and the real world experiment, there is an *AprilTag* target placed on the robot base to estimate the camera motion.

Rig with Three Tango Tablets: The next datasets contain pose estimations from three Google Tango tablets [10] that are rigidly mounted on an aluminum profile. The datasets are recorded in an indoor environment. In order to improve the accuracy of the Tango pose estimation, we used the Tango framework to find loop closures and create optimized localization maps based on the individual trajectories and then exported the self-localized pose estimates of the Tango tablet.

5 Results

Evaluating hand-eye calibrations is inherently difficult, as ground truth values are not available for real systems. In order to evaluate the initial hand-eye calibration results, we use the same evaluation method also employed for the sample evaluation in the RANSAC algorithm, i.e., we transform the pose pairs into a common frame and compute the RMSE of the position and orientation. For the datasets with more than one sensor pair, we further evaluate the accumulation of position/orientation error that occurs when all sensor pair calibrations are combined to form a loop, hence ideally resulting in the identity transform. In order to evaluate the different components of the proposed system we compare the *PF_RC*, *NF_RC*, *PF_RS*, *NF_RS*, *PF_B*, *NF_B*, *PF_EC*, *PF_ES* variants (see Sect. 3.3) on the datasets described in Sect. 4. As a refinement step, we apply the refinement described in Sect. 3.3.2 to the individual initial guesses.

Furthermore, we compare the runtimes of the different algorithms. All non-deterministic algorithms (i.e. *RC* and *RS*) are run 20 times and the results are accumulated using box-plots. For the *Tango* datasets, we additionally accumulated the measurements of all 3 hand-eye calibration pairs.

5.1 Time Alignment

In order to demonstrate the importance of filtering the angular velocity norm prior to the correlation used for time-alignment, we show how the RMSE results of the *ES* algorithm improve, see Table 1 and Fig. 3. For the *PrimeSense* and the *robot arm* dataset we see an improvement of the calibration result. This corresponds with the observation, that there is more noise on the orientation for those datasets. If the time offset, which is a multiple of the discrete time steps of the timestamped poses, is the same with and without filtering, the results are identical, as observed for the datasets: *robot arm sim* and *Tango triplet*.

Table 1 RMSE (position [m]/orientation [deg]) results for the *ES* algorithm with and without angular velocity norm filtering

RMSE (position/orientation)	PrimeSense	Tango triplets short	Robot arm	Robot arm sim
Filtering	(0.0213/0.0213)	(0.0364/0.6389)	(0.0118/0.6619)	(0.0034/0.2677)
No filtering	(0.0227/1.8399)	(0.0364/0.6389)	(0.0120/0.8972)	(0.0034/0.2677)

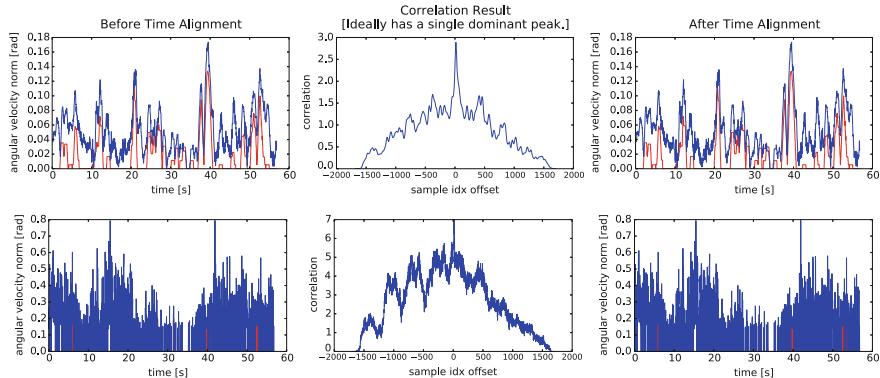


Fig. 3 Time-alignment with (top) and without (bottom) capping and low-pass filtering

5.2 Hand-Eye Calibration

We show evaluations of the runtimes, in Fig. 4, and of the RMSE of position and orientation for every dataset and algorithm in Fig. 5. The two different RANSAC based algorithms (*RS* and *RC*) both outperform the *B* in terms of calibration quality, which is to be expected as the random sampling has a higher chance of finding inliers. Furthermore, both algorithms result in a calibration quality that is very close to the *ES* and *EC* algorithms, which naturally represent the upper bound without using the refinement step. The gap in calibration quality of the RANSAC based algorithms comes at the cost of runtime, i.e. *RS* and *RC* are significantly slower than the *B* algorithm. While intended as a baseline algorithm to provide an upper bound for the calibration quality of *RS*, the *ES* algorithm proves to be efficient and therefore a valid candidate. This is due to the fact, that it only requires a single sample and, therefore, the number of combinations to explore is only the number of input poses, which has been significantly reduced by the selection of informative pose pairs. That is why it is only slightly slower than the RANSAC based algorithms. The *EC* algorithm on the other hand uses a sample size of n ($n >= 3$) and, hence, has to explore a far greater number of combinations, which is reflected in the runtime plot. Surprisingly, the prefiltering of the poses generally had a negligible effect on both calibration quality and runtime, with the exception of the above mentioned exhaustive search, which would not have been feasible without it.

We plot the circular calibration error of the three sensors in the *Tango* datasets, see Fig. 6. After the refinement step we get a mean circular position and orientation error of 4.02 mm and 0.091° for the *Tango 1* dataset, and 7.19 mm and 0.139° for the *Tango 2* dataset, which is a significant improvement over the initial calibration.

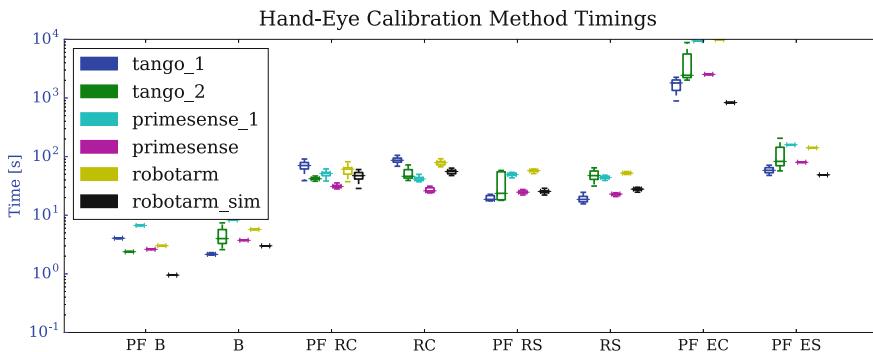


Fig. 4 The timings of the differently filtered algorithms on all datasets

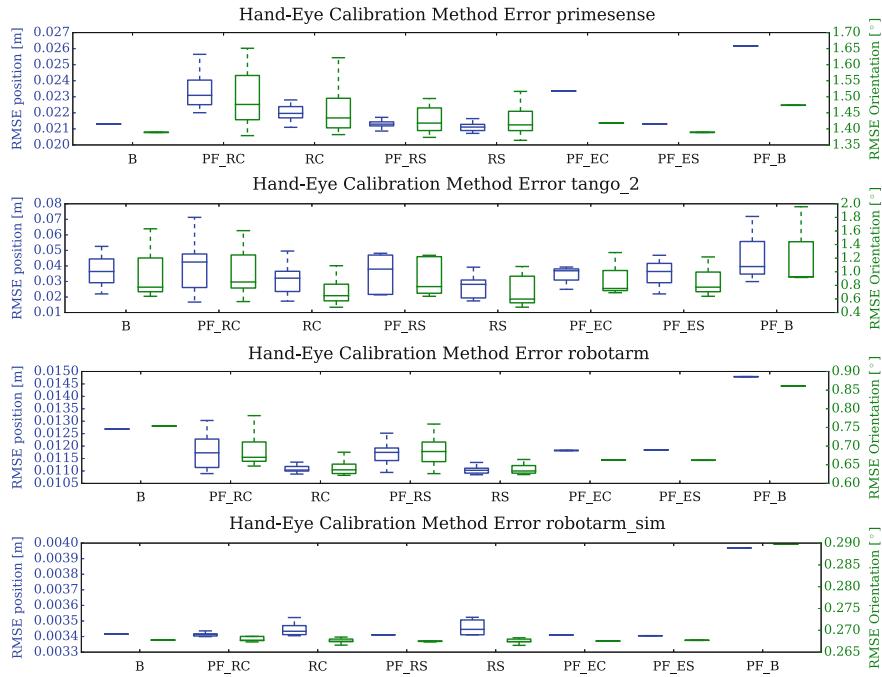


Fig. 5 Evaluation of the different filtering methods on the different datasets

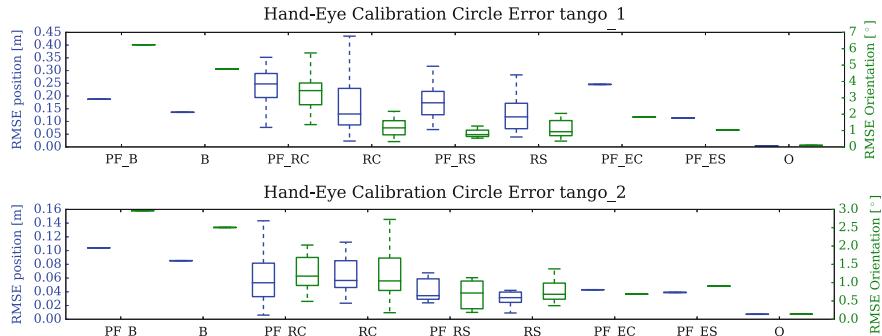


Fig. 6 The circular error of the individual methods and the combined results after applying the refinement step to the other methods, denoted with (O)

6 Conclusion

In this paper we presented a hand-eye calibration system that can easily be used out of the box in a variety of scenarios and environments. In order to substantiate that claim, our system is thoroughly evaluated on different datasets stemming from multiple types of platforms. Taking a holistic view on the hand-eye calibration problem, we consider aspects such as time offset estimation as well as detection and rejection of outliers. All the datasets were made publicly available together with the entire software toolbox, which was designed in a modular way to ensure extendability.

Acknowledgements This work was partially supported by the Swiss National Science Foundation (SNF), within the National Centre of Competence in Research on Digital Fabrication, by Google in the context of the Tango project, and by the European Union's Seventh Framework Programme for research, technological development and demonstration under the EUROPA2 project No. FP7-610603.

References

1. Ackerman, M.K., Cheng, A., Shiffman, B., Boctor, E., Chirikjian, G.: Sensor calibration with unknown correspondence: solving $AX = XB$ using euclidean-group invariants. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1308–1313 (2013)
2. Andreff, N., Horaud, R., Espiau, B.: Robot hand-eye calibration using structure-from-motion. *Int. J. Robot. Res.* **20**(3), 228–248 (2001)
3. Chou, J.C.K., Kamel, M.: Finding the position and orientation of a sensor on a robot manipulator using quaternions. *Int. J. Robot. Res.* **10**(3), 240–254 (1991)
4. Daniilidis, K.: Hand-eye calibration using dual quaternions. *Int. J. Robot. Res.* **18**(3), 286–298 (1999)
5. English, A., Ross, P., Ball, D., Upcroft, B., Corke, P.: Triggersync: A time synchronisation tool. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 6220–6226. IEEE (2015)
6. Fassi, I., Legnani, G.: Hand to sensor calibration: a geometrical interpretation of the matrix equation $AX = XB$. *J. Robot. Syst.* **22**(9), 497–506 (2005)
7. Furgale, P., Barfoot, T.D., Sibley, G.: Continuous-time batch estimation using temporal basis functions. In: 2012 IEEE International Conference on Robotics and Automation, pp. 2088–2095 (2012)
8. Furgale, P., Rehder, J., Siegwart, R.: Unified temporal and spatial calibration for multi-sensor systems. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1280–1286 (2013)
9. Furgale, P., Tong, C.H., Barfoot, T.D., Sibley, G.: Continuous-time batch trajectory estimation using temporal basis functions. *Int. J. Robot. Res.* **34**(14), 1688–1710 (2015)
10. Google. ATAP Project Tango Google (2014)
11. Harrison, A., Newman, P.: Ticsync: Knowing when things happened. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 356–363. IEEE (2011)
12. Heller, J., Havlena, M., Sugimoto, A., Pajdla, T.: Structure-from-motion based hand-eye calibration using l1 minimization. In: 2011 Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3497–3503 (2011)
13. Horaud, R., Dornaika, F.: Hand-eye calibration. *Int. J. Robot. Res.* **14**(3), 195–210 (1995)

14. Kelly, J., Sukhatme, G.S.: A general framework for temporal calibration of multiple proprioceptive and exteroceptive sensors. In: Experimental Robotics, pp. 195–209. Springer (2014)
15. Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), vol. 3, pp. 2149–2154 (2004)
16. Maye, J., Furgale, P., Siegwart, R.: Self-supervised calibration for robotic systems. In: 2013 IEEE Intelligent Vehicles Symposium (IV), pp. 473–480 (2013)
17. Olson, E.: Apriltag: A robust and flexible visual fiducial system. In: 2011 IEEE International Conference on Robotics and Automation, pp. 3400–3407 (2011)
18. Park, F.C., Martin, B.J.: Robot sensor calibration: solving $AX = XB$ on the Euclidean Group. *IEEE Trans. Robot. Autom.* **10**(5), 717–721 (1994)
19. Rehder, J., Siegwart, R., Furgale, P.: A general approach to spatiotemporal calibration in multisensor systems. *IEEE Trans. Robot.* **32**(2), 383–398 (2016)
20. Schmidt, J., Vogt, F., Niemann, H.: Robust hand eye calibration of an endoscopic surgery robot using dual quaternions. In: Vision, Modeling, and Visualization, pp. 21–28 (2004)
21. Shiu, Y.C., Ahmad, S.: Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form $AX = XB$. *IEEE Trans. Robot. Autom.* **5**(1), 16–29 (1989)
22. Sommer, H., Forbes, J.R., Siegwart, R., Furgale, P.: Continuous-time estimation of attitude using b-splines on lie groups. *J. Guid. Control Dyn.* (2015)
23. Strobl, K.H., Hirzinger, G.: Optimal hand-eye calibration. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4647–4653. IEEE (2006)
24. Taylor, Z., Nieto, J.: Motion-based calibration of multimodal sensor extrinsics and timing offset estimation. *IEEE Trans. Robot.* **32**(5), 1215–1229 (2016)
25. Tsai, R.Y., Lenz, R.K.: A new technique for fully autonomous and efficient 3D robotics hand/eye calibration. *IEEE Trans. Robot. Autom.* **5**(3), 345–358 (1989)
26. Wang, C.-C.: Extrinsic calibration of a vision sensor mounted on a robot. *IEEE Trans. Robot. Autom.* **8**(2), 161–175 (1992)

Part III

Inspection

Field Report: UAV-Based Volcano Observation System for Debris Flow Evacuation Alarm

Keiji Nagatani, Ryosuke Yajima, Seiga Kiribayashi, Tomoaki Izu,
Hiromichi Kanai, Hiroyuki Kanasaki, Jun Minagawa and Yuji Moriyama

Abstract Once a volcano erupts, molten rocks, ash, pyroclastic flow, and debris flow can cause disasters. Debris flows can cause enormous damage over large areas. Therefore, a debris-flow simulation is an effective means of determining whether to issue an evacuation call for area residents. However, for safety purposes, restricted areas are set up around a volcano when it erupts. In these restricted areas, it is difficult to gather information such as the amount and permeability of the ash; this information is necessary for precise debris-flow simulations. To address this problem, we have developed an unmanned observation system for use in restricted areas around volcanoes. Our system is based on a multirotor micro unmanned aerial vehicle (MUAV); this system can be used to perform field tests in actual volcanic areas. In this paper, we report the field tests conducted at Mt. Unzen-Fugen during November 2016. The field tests included a demonstration of an unmanned surface flow measurement device and the deployment and retrieval of a small ground vehicle and a drop-down-type ash-depth measurement scale using an MUAV. In addition, we discuss some of the lessons learned.

K. Nagatani (✉) · R. Yajima · S. Kiribayashi
Tohoku University, 6-6-10, Aramaki-Aoba, Sendai, Japan
e-mail: keiji@ieee.org

T. Izu
ENROUTE CO., LTD., 1-3-29, Ureshino, Fujimino, Saitama, Japan
e-mail: izu@enroute.co.jp

H. Kanai · H. Kanasaki · J. Minagawa
Kokusai Kogyo Co., Ltd, 2-24-1, Harumi, Fucyu, Tokyo, Japan
e-mail: hiromichi_kanai@kk-grp.jp

Y. Moriyama
Kokusai Kogyo Co., Ltd, 2-Rokuban, Chiyoda, Tokyo, Japan
e-mail: yuji_moriyama@kk-grp.jp

1 Introduction

Recently, several active volcanoes in Japan have shown an increase in activity. In recent years, eruptions have occurred at Nishino-Shima, Kuchino-Arab, and Ontake. In the near future, there is a real possibility of a new, large-scale volcanic eruption in Japan—this includes Mt. Fuji.

Once a volcano erupts, molten rocks, ash, pyroclastic flow, and debris flow can cause disasters, as shown in Fig. 1. Debris flow is a phenomenon in which rain falls on a slope containing accumulated volcanic gravel and ash, causing them to flow downward. This can result in enormous damage over a large area. For example, in the 1990s, Shimabara City suffered damage from debris flow caused by Mt. Unzen-Fugen's eruption [1].

A debris-flow simulation is an effective means of determining whether to issue an evacuation call for residents. However, for safety reasons, a restricted area is set around a volcano when it erupts. Therefore, in the restricted area, it is difficult to gather the information necessary for a precise simulation, such as the amount and permeability of the ash. In the Mt. Shinmoe eruption in 2011, the mountain was covered with coarse volcanic sediment that made debris flow less likely; however, there was no opportunity to observe it because of the restricted area. Thus, when a small amount of rain fell and no debris flow occurred, conservative evacuation calls were made and area residents gradually disbelief the alarm.

For a precise debris-flow simulation, direct measurements of the (A) topography shape, (B) amount of ash fall, (C) permeability of volcanic ash, and (D) rainfall are required. Therefore, since 2014, our group has been developing an unmanned observation system for use in restricted volcanic areas; the system is based on multirotor micro unmanned aerial vehicles (MUAVs). The MUAV includes a camera system to obtain images and to generate 3D terrain information, as well as a soil sampling device suspended from the MUAV [2]. In this research, we developed devices and

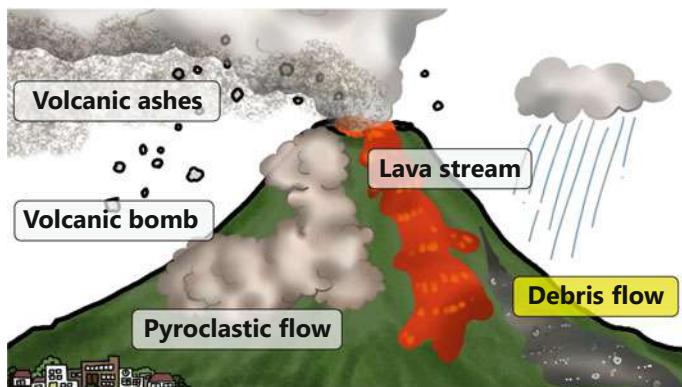


Fig. 1 Types of volcanic disasters

functions, evaluated them in real volcanic environments, and improved them on the basis of the results of the field tests.

In this paper, we report the findings of the field tests conducted at Mt. Unzen-Fugen during November 2016. The tests included demonstrations of the following:

1. A drop-down-type ash-depth measurement scale to obtain the amount of ash fall,
2. A surface flow measurement device to estimate the permeability of volcanic ash, and
3. MUAV-based deployment and recovery of a small ground vehicle that measures the amount of rainfall.

In addition, we discuss some of the lessons learned.

2 Related Works

There have been several attempts to use robotic technology for the remote observation of restricted areas. Many of them used mobile robots. Carnegie Mellon University has conducted volcanic explorations using legged robots named Dante and Dante II [3]. In Europe, a wheeled volcano exploration robot was developed by the Italian-led RoboVolc Project and tested it at Mount Etna and Vulcano Island [4]. In Japan, Tohoku University developed a teleoperated robot called Mobile Observatory for Volcanic Eruption (MOVE) [5]. The above robots were relatively large, and their operation areas were limited because of traversability problems. As a solution, some research institutes considered aerial robots. One famous example was an autonomous helicopter known as Yamaha R-Max [6]. When Mount Usu erupted in Hokkaido in 2000, an unmanned helicopter, equipped with GPS and a video camera, was employed to observe the land features and geological status in the vicinity of the crater. To obtain the advantages of both mobile robots and aerial robots, we proposed a method that combined a mobile robot and an aerial robot to observe restricted areas [7]. In the current paper, we introduce the next version of the mobile robot, which is carried by an MUAV.

3 Drop-Down-Type Ash-Depth Measurement Scale

3.1 *Ash-Depth Measurement Method*

For a precise debris-flow simulation, the direct measurement of the amount of ash fall is very important. However, it is difficult to estimate the amount of ash fall using only the visual information obtained from MUAVs. If there are many vertical scale poles in the target environment as references, it is possible to measure the ash depth by reading the values of those scales. Unfortunately, in most cases, locating such instruments on volcanoes is prohibited in Japan.

To measure ash depth, we propose a simple drop-down-type pyramid-shaped scale. In the initial stage of a volcanic eruption, or when eruption signs have been detected, an MUAV carries the drop-down-type scales to the restricted area and drops them.

After the volcanic eruption, another MUAV flies to the same location and hovers to take a photograph of the scales. Once the size and color of each scale are known, we can estimate the ash depth by detecting the visible scales.

3.2 Prototype of the Ash-Depth Measurement Scales

For our initial tests, we produced scales in three sizes. Figure 2a shows the prototype scales. To eliminate the need to retrieve them later, we used biodegradable plastics to construct the scales. When the ash depth is less than 1 cm, the vision sensor on the robot can see the minimum size of the scales from the sky. When the ash depth is between 1 and 2 cm, the minimum scales are not visible in the photograph, but the other two scales can be seen from the sky, as shown in Fig. 2b. When the ash depth is over 3 cm, only the largest scale can be seen from the sky.

To deploy the scales in a target environment, we developed a drop-down device (i.e., an ash-depth measurement scale). When the MUAV arrives at the target position, the device opens at the bottom and drops the scales from the air. The device has three chambers so that it can deploy the scales to three positions in one flight. A concept image of the deployment is shown in Fig. 2c.

3.3 Initial Tests of the Ash-Depth Measurement Scale and Lessons Learned

We conducted some recognition tests to understand the suitable conditions for the recognition of the scales. Red-, yellow-, and blue-colored 1-cm scales were manually

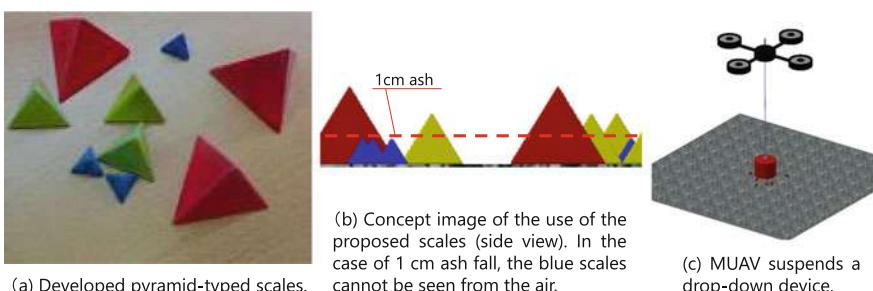


Fig. 2 Prototype of the ash-depth measurement scale, method of operation, and deployment image

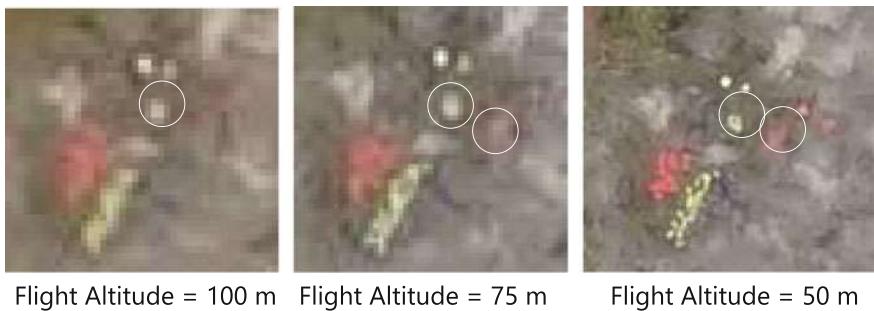


Fig. 3 Scale images from an MUAV obtained at different flight heights

deployed on the ground. The MUAV then took pictures from different flight heights. A 4000×6000 -pixel camera with a resolution of 350 dpi was used.

Test results showed that visual recognition of the scales required a height of 50 m or less. Figure 3 shows scale images obtained at different flight heights. Moreover, the order of the color viewability was (1) yellow, (2) red, and (3) blue. It was difficult to distinguish the yellow scale from other objects, such as vegetation or small waterfalls. On the other hand, blue-colored scales were difficult to recognize from above 50 m. Furthermore, when volcanic ash was spilled on the scales manually, the top of the scales was viewed as we expected.

In addition, we tested the deployment by an MUAV at Mt. Unzen-Fugen. The MUAV suspended the developed drop-down device, which was able to deliver different sizes of scales autonomously.

On the basis of the above tests, several lessons were learned.

1. The pyramid-shaped scales are suitable for measurement of the ash depth. Other shapes, e.g., rectangular shapes, were difficult to observe after the ash fell.
2. We need to consider the relationship between the camera's view angle and the positioning errors of the MUAV. The lower the MUAV flies, the better the resolution of the obtained scale image is. However, because of GPS positioning error, the target scales may be out of range when the MUAV flies lower. It may be better to fly the MUAV in a spiral fashion to obtain images reliably.
3. The scales' color should be chosen on the basis of the color of the target ground. Therefore, we should execute the following procedure for practical use.
 - a. An MUAV with a camera flies to obtain 3D terrain information of the target area. On the basis of this information, the operator determines the scale deployment locations and the optimal scale color.
 - b. An MUAV with a suspended drop-down device flies to deploy pyramid-shaped scales to the planned locations.
 - c. At regular intervals, e.g., every 2 days, an MUAV with a camera flies to obtain images of the deployed scales for measurement of ash depth.

In future field tests, we will consider the above topics and conduct the above procedure as a preparatory exercise.

4 Surface Flow Measurement Device

4.1 Development of the Surface Flow Measurement Device

Debris-flow simulations also require information on the permeability of volcanic ash. However, typical methods of measuring the permeability require manual measurement, a heavy device, an excess amount of water, and time. It is impossible to conduct the same procedure with an unmanned MUAV system. Therefore, we had to change our strategy. To improve the simulation results, we developed a surface flow measurement device to roughly estimate the permeability, instead of measuring the permeability directly.

Figure 4 shows the developed surface flow measurement device and how it works. It consists of a water storage, a controller to detect the landings and to make the water flow, two cameras, and legs. When the device detects a landing, the controller activates a servo motor to move a cutter blade to break the mounted water balloon. Then, water falls to the ground immediately as a simulation of heavy rain. The simulated rain's intensity is equivalent to 1700 mm/h. Finally, two cameras located on opposite sides of the device record the rate of water absorption into the ground. The device is 350 mm in both width and length, and 320 mm in height. The diameter

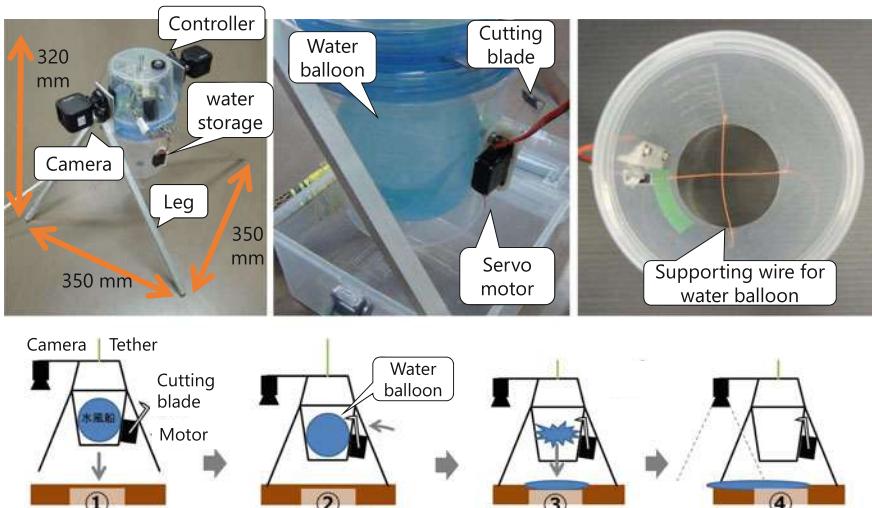


Fig. 4 The developed surface flow measurement device and its method of operation

of the storage of the water balloon is 100 mm. The maximum capacity of the water balloon is approximately 500 mL. A servo motor with a cutter blade for breaking the water balloon is located on the side of the device.

4.2 Initial Tests of the Surface Flow Measurement Device and Lessons Learned

First, we conducted initial tests without the MUAV, using deposited sand from three locations: Mt. Tarumae, Mt. Unzen-Fugen, and Mt. Sakurajima. Figure 5 shows images captured after water had fallen. In the case of volcanic ash with high permeability (Fig. 5a, b), fallen water permeated the soil quickly and the soil formed a crater shape. On the other hand, in the case of volcanic ash with low permeability (Fig. 5c), the initial fallen water let the fine particles dance like a snowstorm and then it temporarily generated a surface flow; finally, it formed an amorphous shape. The above phenomena were observed by wide-angle video cameras (GoPro) mounted on the device; the permeation time could also be measured.

Next, the water permeability coefficients of the target samples were measured by indoor experiments according to a conventional method. The results were as follows:

- Mt. Tarumae — 6.31×10^{-4} (m/s),
- Mt. Unzen-Fugen — 7.96×10^{-5} (m/s),
- Mt. Sakurajima — 1.34×10^{-5} (m/s).

The above results qualitatively matched the results obtained by the proposed surface flow measurement device.

Finally, at Mt. Unzen-Fugen, we tested the deployment of the device by the MUAV. The device was suspended by the MUAV and carried to the target position; it then landed on the ground and obtained a video clip of the surface flow of the water autonomously (Fig. 6).

During the above tests, several lessons were learned.

Soil	(a) Mt. Tarumae	(b) Mt. Unzen-Fugen	(c) Mt. Sakurajima
Images			
Shapes	Crater shape	Crater shape	Amorphous shape

Fig. 5 Images captured after water had fallen onto different deposited sands

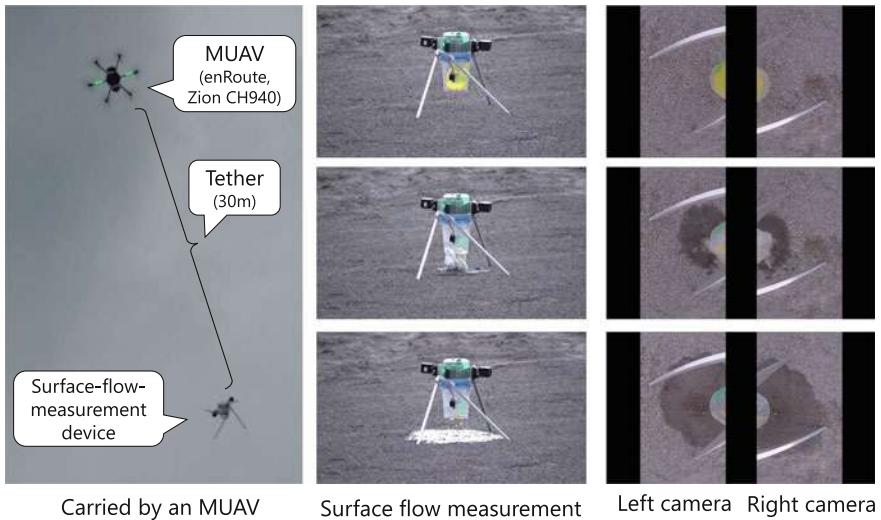


Fig. 6 Flight test of the surface flow measurement device

1. Surface flow measurement devices cannot measure strict permeability. Therefore, we will roughly classify the permeability of the soil by type, e.g., Tarumae type or Sakurajima type. In debris-flow simulations, we will use representative permeability values for each type. For classification, it is necessary to conduct additional experiments on different types of soil.
2. In the above test, we considered the size and shape of the region where water was absorbed. However, we should also consider how long it takes for the water to be absorbed.
3. We need to improve the shape of the device and its components. Once the MUAV became airborne, the tether caught the camera and the posture of the device tilted. Moreover, during the preparations for the test, the water balloon occasionally burst before being set in the device. In general, preparation of the water balloon is very troublesome.

In future works, we will redesign the device and conduct many tests to obtain data for soil classification.

5 Mobile Sensing Device Carried by an MUAV

5.1 Development of a Small-Sized Mobile Robot

The amount of rainfall is also important for precise debris-flow simulations. Continuous measurements in volcanic areas are required to estimate how much water

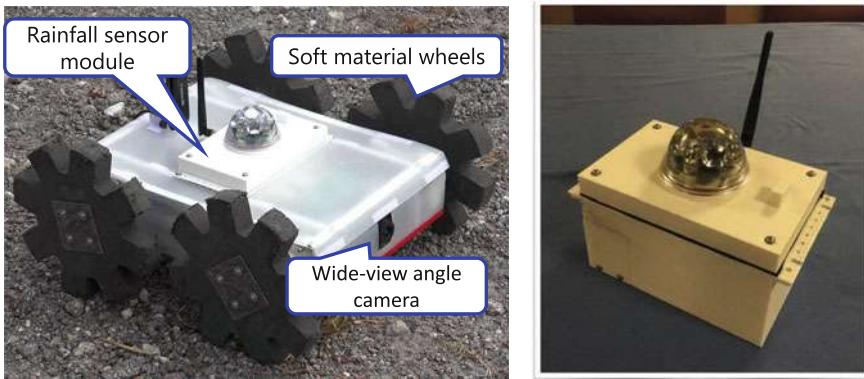


Fig. 7 CLOVER-II (left) and a lightweight rainfall sensor module (right)

the ground absorbs. However, in volcanic eruptions, sensors already installed in such areas may malfunction. In the eruptions that occurred at Mt. Unzen-Fugen in the 1990s, almost all sensors were broken. Therefore, sensors that can be installed after an eruption are needed. To install sensors at suitable positions, we developed a deployment system using a mobile robot carried by an MUAV. The robot has the capability to mount lightweight sensors.

Figure 7-(left) shows the mobile robot called CLOVER-II. It is a next-generation version of CLOVER-I, which was designed for observing volcanic environments. It can be deployed by an MUAV using the sky-crane method [7]. It has space to mount a device at its center. The robot is 400 mm in width, 465 mm in length, and 220 mm in height. The robot's weight is approximately 3.5 kg with batteries and without sensors.

5.2 Development of a Lightweight Rainfall Sensor for a Mobile Robot

In this project, we designed and developed a rainfall sensor module to measure not only the amount of rainfall but also the barometric pressure and frequency of thunder. Figure 7-(right) shows an image of the sensor module. The sensor module includes the following three sensors:

1. Optical rain gauge

To measure the amount of rainfall, we chose an optical rain gauge (GR-11, HYDREON Corporation). Infrared light emissions inside the dome decay when raindrops attach to the surface of the dome. The amount of rainfall can be estimated according to the amount of decay.

2. Thunder sensor

A portable, commercial thunder sensor, whose chip is AS3935 (Austria Micro

Systems), is also installed in the module. It detects a signal generated by thunder or an intercloud discharge.

3. Barometric pressure sensor

A prototype of the MEMS pressure sensor, developed by Murata Manufacturing Co. Ltd., is also installed in the module. The temperature drift of the pressure sensor is smaller than that of other MEMS pressure sensors, and the noise level is relatively low.

5.3 Deployment/Retrieval Sequence of a Small Robot by an MUAV

To carry a small mobile robot in restricted areas, we propose a method for deploying and retrieving small robots using a capturing net suspended from an MUAV. The capturing net is pyramid shaped and is dropped to the ground to allow the deployment/retrieval of the robot. The size of the net is 600×600 mm, and the height is 500 mm. Figure 8 shows the deployment/retrieval sequence for the small robot. After the capturing net lands on the ground, the robot moves off the net and travels to a suitable position based on its teleoperation; it then performs a long-term fixed-point observation. After a certain period has passed, e.g., 2 weeks, the MUAV flies to the position again to retrieve the robot. The retrieval process is the reverse of the process shown in Fig. 8.

5.4 Initial Test of Mobile Robot Retrieval by an MUAV and Lessons Learned

We tested the multirotor MUAV's deployment/retrieval of the mobile robot at a mud control dam at Mt. Unzen-Fugen during November 2016. Figure 9 shows the mobile robot retrieval sequence performed by the multirotor MUAV. The multirotor MUAV

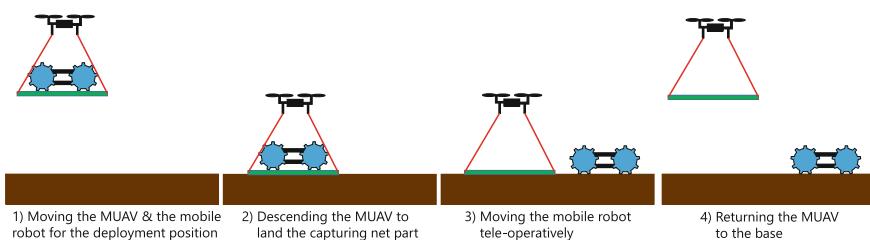


Fig. 8 Mobile robot deployment sequence as performed by the multirotor MUAV. The retrieval sequence is performed in reverse order

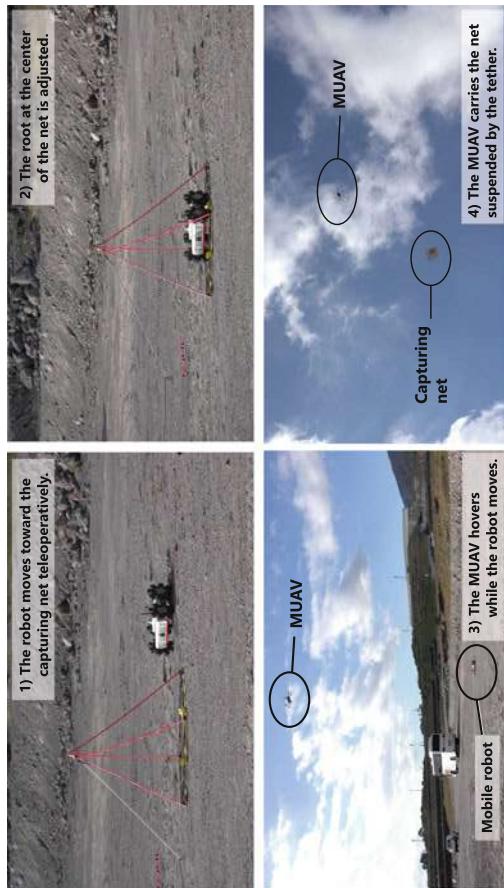


Fig. 9 Mobile robot retrieval sequence as performed by the multirotor MUAV



Fig. 10 A highly compressed rough image with block noise sent from the robot

hovers at the target position based on the GPS information and releases the capturing net so that it falls to the ground (upper left of Fig. 9). The teleoperated mobile robot remains near the landing point of the capturing net. After the net lands, the operator instructs the robot to move onto the center of the capturing net, using the visual information obtained by the front camera of the robot (upper right of Fig. 9). During this procedure, the MUAV hovers in the air (lower left of Fig. 9). After a certain period, the MUAV flies back to the departure point after suspending the capturing net and the robot (lower right of Fig. 9).

According to our results, the success of the above retrieval operation depends on a successful teleoperation to the center of the capturing net within a certain time, and it relies on images from the front camera of the robot. In the above case, wireless communication was the bottleneck. The small robot uses 4G/LTE to transmit its images. In typical communication situations, it sends 640×480 images at approximately four frames per second (fps) without difficulty. However, when the small mobile robot traversed the surface of the mud control dam, its antenna was located in a very low position. Therefore, the position of its electric signal was lowered, and it could send only highly compressed rough images with block noise to the operator at 1 fps, as shown in Fig. 10. As a result, the operator suffered great difficulty during the teleoperation. In this test, the robot was successfully navigated to the center of the capturing net. However, we confirmed that communication near ground surfaces is a serious problem.

6 Conclusion and Future Works

In this paper, we introduced some unmanned technologies to observe restricted areas near volcanoes and reported the results of the field tests conducted at Mt. Unzen-Fugen during November 2016. The field tests included a demonstration of (1) a drop-down-type ash-depth measurement scale for obtaining the amount of ash fall, (2) a surface flow measurement device for estimating the permeability of volcanic ash, and (3) an MUAV-based deployment/retrieval of a small ground vehicle for carrying a sensing module that includes a device for measuring the amount of rainfall. In addition, we reported some lessons learned.

As stated in the Introduction, the aim of this research was to realize a precise debris-flow simulation. Therefore, we are developing software to perform flow rate calculations and data conversions from unmanned observation data. Planned future works include developing a debris-flow simulation based on robotic devices and evaluating the simulation results.

Acknowledgements The New Energy and Industrial Technology Development Organization (NEDO) of Japan supported this work.

References

1. Setsuya, N., Toshitsugu, F.: Preliminary report on the activity at Unzen Volcano (Japan) November 1990–November 1991. *J. Volcanol. Geotherm. Res.*, ORNL/TM-12410:310–333 (1993)
2. Ryosuke, Y., Keiji, N., Kazuya, Y.: Development and field testing of uav-based sampling devices for obtaining volcanic products. In: Proceedings of the 2014 IEEE Int'l Workshop on Safety, Security and Rescue Robotics, October (2014)
3. Bares, J.E., Wettergreen, D.S.: Dante II: technical description, results, and lessons learned. *Int. J. Robot. Res.* **18**(7), 621–649 (1999)
4. Muscato, G., Caltabiano, D., Guccione, S., Longo, D., Coltelli, M., Cristaldi, A., Pecora, E., Sacco, V., Sim, P., Virk, G.S., et al.: Robovolc: a robot for volcano exploration result of first test campaign. *Indust. Robot Int. J.* **30**(3), 231–242 (2003)
5. Keiji, N.: Review: Recent trends and issues of volcanic disaster response with mobile robots. *J. Robot. Mech.* **26**(4), 436–441, August (2014)
6. Sato, A.: The rmax Helicopter uav. Technical Report, DTIC Document (2003)
7. Keiji, N., Kazunari, A., Genki, Y., Kenta, Y., Yasushi, H., Shin'ichi, Y., Tetsuya, I., Randy, M.: Development and field test of teleoperated mobile robots for active volcano observation. In: Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on. pp. 1932–1937. IEEE (2014)

Cooperative UAVs as a Tool for Aerial Inspection of the Aging Infrastructure

Sina Sharif Mansouri, Christoforos Kanellakis, Emil Fresk, Dariusz Kominiak and George Nikolakopoulos

Abstract This article presents an aerial tool towards the autonomous cooperative coverage and inspection of a 3D infrastructure using multiple Unmanned Aerial Vehicles (UAVs). In the presented approach the UAVs are relying only on their onboard computer and sensory system, deployed for inspection of the 3D structure. In this application each agent covers a different part of the scene autonomously, while avoiding collisions. The visual information collected from the aerial team is collaboratively processed to create the 3D model. The performance of the overall setup has been experimentally evaluated in a realistic outdoor infrastructure inspection experiments, providing sparse and dense 3D reconstruction of the inspected structures.

1 Introduction

The annual investments on the infrastructure sector represent a significant percentage of the Gross Domestic Product (GDP) of developed and developing countries e.g. 3.9% of the GDP for the old European states, 5.07% of the GDP for the new European states and 9% of the GDP for China [1]. Towards these inspection tasks, a variety

This work has received funding from the European Unions Horizon 2020 Research and Innovation Programme under the Grant Agreement No.644128, AEROWORKS.

S. S. Mansouri (✉) · C. Kanellakis · E. Fresk · D. Kominiak · G. Nikolakopoulos
Department of Computer, Electrical and Space Engineering,
Luleå University of Technology, Luleå, SE 97187, Sweden
e-mail: sinsha@ltu.se

C. Kanellakis
e-mail: chrkan@ltu.se

E. Fresk
e-mail: emifre@ltu.se

D. Kominiak
e-mail: darkom@ltu.se

G. Nikolakopoulos
e-mail: geonik@ltu.se

of methods and approaches are adopted to address the challenges of infrastructure inspection, where as an example specialized personnel performs visual inspection, nondestructive testing and maintenance tasks using scaffolds, roping or even manned helicopters in order to obtain access to the sites of interest. According to the Helicopter Association International and the Utilities, Patrol and Construction Committee (UPAC) for Safety Guide for Helicopter Operators, 2009 [2] thousands of flight hours are accumulated each day conducting manned aerial work, in support of Utilities Infrastructure (electricity, gas, water), as it is now well-understood that such aerial works bring down the cost and time requirements

In order to decrease the human life risk and to increase the performance of the overall procedure, autonomous ground, aerial or maritime vehicles are employed for executing the inspection tasks. As an example, for these applications it can be mentioned the power-line monitoring using autonomous mobile robots [3], bridge inspection [4], boiler power-plant 3D reconstruction [5], urban structure coverage [6], forest fire inspection [7], aerial manipulation [8] using UAVs, inspection of under-water structures in [9] by the utilization of autonomous underwater vehicles and cooperative sensing [10]. In most of these scenarios, there is an a priori knowledge about the infrastructure, while the 3D or 2D models are available or can be derived using CAD software.

Unmanned Aerial Vehicles (UAVs) equipped with remote sensing instrumentation are emerging in the last years due to their mechanical simplicity, agility, stability and outstanding autonomy in performing complex manoeuvres etc [11]. A variety of remote sensors such as visual sensors, lasers, sonar, thermal, etc. could be mounted, while the acquired information from the UAV's mission can be analyzed and used to produce sparse or dense surface models, hazard maps, investigate access issues, and other area characteristics. However, the main problem in these approaches is to guarantee the full coverage of the area, a fundamental problem that is directly related to the autonomous path planning of the aerial vehicles.

This article demonstrates the novelty of an aerial sensor for the inspection of complex 3D structures with multiple agents. In this approach, the a priori coverage path is divided and assigned to each agent, based on the infrastructure architectural characteristics in order to reduce the inspection time. Furthermore, to guarantee a full coverage and a 3D reconstruction, the introduced path planning for each agent creates an overlapping visual inspection area, that will enable the off-line cooperative reconstruction. Based on the aforementioned state of the art, the major contribution stems from the direct demonstration of the applicability and feasibility of the overall cooperative coverage and inspection scheme with the UAVs for outdoors scenarios without the utilization of any external reference system, e.g. motion capture systems. This demonstration has a significant novelty and impact as an enabler for a continuation of research efforts towards the real-life aerial cooperative inspection of the aging infrastructure, a concept that has never been presented before to the authors best knowledge, in outdoor and with a real infrastructure as a test case. In the outdoors demonstrations, the UAVs have been autonomously operated based on odometry information from visual and inertial sensor fusion and without any other support on localization, which adds more complexity and impact on the acquired

results. The image and pose data on board the platform were post processed to build a 3D representation of the structure.

The rest of the article is structured as follows. First a brief review on related works is presented in Sect. 2. Then the general formulation of the problem is described in Sect. 3. Later on, the proposed method is presented in Sect. 4, which follows with a brief description on the 3D reconstruction from multiple agents. In Sects. 5 and 6 the hardware for autonomous navigation and multiple simulation and experimental results are presented respectively. Finally the article concludes in Sect. 7.

2 Related Work

Navigation of multi-agent systems for infrastructure inspection, is an area of increasing interest both from a research, as well as an application viewpoint. In recent years, multiple approaches have been proposed regarding obstacle free path generation for robotic platforms. The application of potential field-based methods has been explored [12, 13] as a promising research direction for such algorithms. Coverage Path Planning (CPP) is the task of determining a path that passes over all points of an area or volume of interest while avoiding obstacles [14]. The task of coverage is fundamental in many robotic applications, such as, visual inspection of complex structure [15], painter robots [16], wall climbing robots for inspection [17], inspection of complex underwater structures [18], vacuum cleaning robots [19], etc. In [14] a complete survey was presented on CPP methods in 2D and 3D. This article is mainly focused on the application of CPP in aerial robotics for infrastructure inspection, providing simultaneously the required theoretical background.

The task of CPP has received significant attention over the last years in the different application scenarios, however still there are limited CPP approaches in the case of aerial robotics and fewer approaches that addressed coverage of 3D spaces [20]. Especially in the case that the CPP concept is extended in the Collaborative approach (C-CPP), by the utilization of multiple aerial agents, instead of a single one, the overall coverage time has the potential to be dramatically reduced, while it can be achieved realistically by multiple UAVs, when taken under consideration the flying times and the levels of autonomy. Thus, inspired by this vision, the main objective of this article is to establish a C-CPP method that is based on an a priori knowledge of the infrastructure (e.g. a CAD model) and it will have the ability to generate proper way points by considering multiple agents, while ensuring full coverage and the overall collision avoidance among the flying agents. As it will be presented, the proposed novel scheme will create a sub-coverage path planning for cooperative inspection of the whole infrastructure, while having the capability to detect branches of complicated infrastructures, which can be assigned to different agents. In the sequel, supplementary, 3D reconstruction routines can be performed to provide an updated 3D mesh of the structure by using either the monocular or the stereo cameras.

Towards the 3D CPP, Atkar et al. [16] presented an offline 3D CPP method for spray-painting of automotive parts. Their method used a CAD model and the resulting

CPP should satisfy certain requirements for paint decomposition. In [21], the authors presented a CPP with real time re-planning for inspection of 3D underwater structures, where the planning assumed an a prior knowledge of a bathymetric map and they adapted their methodology for the case of an autonomous underwater vehicle, while their overall approach was containing no branches. The authors in [18] introduced a new algorithm for producing paths that cover complex 3D environments. In this case, the algorithm was based on off-line sampling with the application of autonomous ship hull inspection, while the presented algorithm was able to generate paths for structures with unprecedented complexity.

In the area of aerial inspection, [6] presented a time-optimal UAV trajectory planning for 3D urban structure coverage. In this approach, initially the structures to be covered (buildings) were simplified into hemispheres and cylinders and in a later stage the trajectories were planned to cover these simple surfaces. In [22], the authors studied the problem of 3D CPP via viewpoint re-sampling and tour optimization for aerial robots. More specifically, they presented a method that supports the integration of multiple sensors with different fields of view and considered the motion constraints on aerial robots. Moreover, in the area of multi-robot coverage for aerial robotics in [23], a coverage algorithm with multiple UAVs for remote sensing in agriculture has been proposed, where the target area was partitioned into k non-overlapping sub-tasks and in order to avoid collision both different altitudes have been assigned to each UAV and security zones were defined, where the vehicles are not allowed to enter.

3 Problem Statement

Multiple aerial robots will be employed and will address the problem of autonomous, complete, and efficient execution of infrastructure inspection and maintenance operations. To facilitate the necessary primitive functionalities, an inspection path-planner that can guide a team of UAVs to efficiently and completely inspect a structure will be implemented. The collaborative team of UAVs should be able to understand the area to be inspected, ensure complete coverage and an accurate 3D reconstruction to accomplish complex infrastructure inspection. Relying on the accurate state estimation as well as the dense reconstruction capabilities of the collaborative aerial team, algorithms for the autonomous inspection planning should be designed to ensure full coverage.

It should be highlighted that, this work is an extension of our preliminary work presented in [24] which studied the problem formulation in 2D. Let assume $\Omega \subset \mathbb{R}^3$ be a given region that can have multi-connected components (complex structure), while we also consider the finite set

$$\Lambda = \{C_i : i \in I_n = \{1, 2, \dots, n\}\} \quad (1)$$

of cells

$$C_i = \{(x_i, y_i, z_i) \in \mathbb{R}^3 : (x, y, z) \sim \text{camera specification and position}\}. \quad (2)$$

The placement of the cells C_i can be defined by the translation vector $u_i = (x_i, y_i, z_i)$ and orientation vector $o = \{\phi, \theta, \psi_i\}, i \in I_n$, while the set of translated and orientated cells $C_i(u_i, o_i)$ is expressed by $\Lambda(u, o)$, where $u = \{u_1, u_2, \dots, u_n\} \in \mathbb{R}^{3n}$ and $o = \{o_1, o_2, \dots, o_n\} \in R^{3n} : 0 \leq o_i \leq 2\pi$.

The 3D polygonal

$$P(u_i, o_i, n) = \bigcup_{i=1}^n C_i(u_i, o_i) \quad (3)$$

represents the region covered by the union of the cells C_i , while Λ^* is a cover of Ω if there exist a solution such that

$$\Omega \subset P(u_i, o_i, n) = \bigcup_{i \in I_n} C_i(u_i, o_i) \quad (4)$$

Moreover, several cases arise in the interaction between two cells $C_i(u_i, o_i)$ and $C_j(u_j, o_j)$ with $i \neq j$, $u_i = (x_i, y_i, z_i)$, $o_i = (\phi_i, \theta_i, \psi_i)$, $u_j = (x_j, y_j, z_j)$ and $o_j = (\phi_j, \theta_j, \psi_j)$, where mainly determined by:

$$\begin{aligned} C_i(u_i, o_i) \cap C_j(u_j, o_j) &= \emptyset \\ C_i(u_i, o_i) \cap C_j(u_j, o_j) &\neq \emptyset \end{aligned} \quad (5)$$

Additionally the cases of $C_i(u_i, o_i) \subset C_j(u_j, o_j)$ and $C_j(u_j, o_j) \subset C_i(u_i, o_i)$ are not considered when dealing with the coverage problem, because it is contrary to optimality of the path to have a substantial overlapping for visual processing and cover the whole surface of the under inspection object.

4 Methodology

Collaborative UAVs can be deployed, equipped with advanced environmental perception and 3D reconstruction, intelligent task planning, and multi-agent collaboration capabilities. Such a team of UAVs should be capable of autonomously inspecting infrastructure facilities, while this Section presents an experimental setup, towards multi-robot collaboration, path-planning, localization, as well as cooperative environmental perception and reconstruction. Furthermore, the mission-oriented planning algorithms will be integrated with the control and localization components of the platform in outdoor environments. A major component that affects the overall performance of the inspection task, is the Path planning strategy. In this work an extended version of CPP is implemented in a collaborative manner (C-CPP) by

the utilization of multiple aerial agents, instead of a single one. The attributes of this approach are the full coverage of a complex structure and the reduced mission time [25] for the overall coverage, by considering the level of autonomy and flight times for each agent. A full reference on developed C-CPP can be found in [15].

Briefly, the established C-CPP method is based on an a priori knowledge of the infrastructure (e.g. a CAD model) and it has the ability to generate proper way points by considering multiple agents, while ensuring collision avoidance among the flying agents. The implemented method will create a sub-coverage path planning for cooperative inspection of the whole infrastructure, while having the capability to detect branches of complicated infrastructures, which can be assigned to different agents. In the sequel, supplementary, 3D reconstruction routines can be performed to provide an updated 3D mesh of the structure by using various sensors, like cameras or lidars. Additionally, the generated waypoints guarantee enough overlapping Field of View in order to build the structure 3D model from the processed sensor data.

The resulting waypoints are then converted into position-velocity-yaw trajectories, which can be directly provided to the utilized linear model predictive controller cascaded [26] over an attitude-thrust controller. This is done by taking into account the position controller's sampling time T_s and the desired velocity along the path \mathbf{V}_d . These trajectory points are obtained by linear interpolation between the waypoints, in such a way that the distance between two consecutive trajectory points equals the step size $h = T_s ||\mathbf{V}_d||$. The velocities are then set parallel to each waypoint segment and the yaw angles are also linearly interpolated with respect to the position within the segment. The adopted trajectory generation that was used in the experimental realization of the proposed C-CPP.

As stated throughout this article, the C-CPP method addresses the case of autonomous cooperative inspection by multiple aerial UAVs. Each aerial platform is equipped with a camera to record image streams and provide a 3D reconstruction of the infrastructure [11]. More specifically, Monocular mapping is considered to obtain the 3D model of the infrastructure, while the overall aim is to merge the processed data from multiple agents into a global representation. Structure from Motion (SfM) approach is used to provide a 3D reconstruction. While, the aerial agents follow their assigned path around the object of interest the image streams from the monocular cameras of all agents are stored in a database.

The process starts with the correspondence search step, which identifies overlapping scene parts among input images. During this stage, feature extraction and matching algorithms between frames are performed to extract information about image scene coverage. Next, a filtering step is implemented to remove outliers using epipolar geometry [27]. The algorithm requires an initial image pair I_1 and I_2 with enough parallax as the starting point and then to incrementally register new frames. Briefly, image matches are extracted and the camera extrinsics for I_1 and I_2 using the 5-point algorithm [28]. Then projection matrices, including the relative pose between frames, are estimated and used for triangulation of the detected image points, to recover their 3D position X^{3D} . Afterwards, the two-frame Bundle Adjustment refines the initial set of 3D points minimizing the reprojection error.

The aforementioned process consist of the initialization step. The remaining images of the dataset are incrementally registered in the current camera and point sets using Perspective-n-Point (PnP) [29]. The newly extracted points are triangulated and are processed from a global Bundle Adjustment to correct drifts in the process. The absolute scale of the reconstructed object can be recovered by combining the full-pose annotated images from the onboard localization of the camera systems.

5 Setup for Autonomous Navigation

The proposed method has been evaluated with the utilization of the AscTec NEO hexacopter, depicted in Fig. 1. The platform has a diameter of 0.59 m and height of 0.24 m. The length of each propeller is 0.28 m as depicted in Fig. 1. This platform is capable of providing a flight time of 26 min, which can reach a maximum airspeed of 15 m/s and a maximum climb rate of 8 m/s, with maximum payload capacity up to 2 kg. It has an onboard Intel NUC computer with a Core i7-5557U and 8 GB of RAM. The NUC runs Ubuntu Server 14.04 with Robotic Operatic System (ROS) installed. ROS is a collection of software libraries and tools used for developing robotic applications [30]. Additionally, multiple external sensory systems (e.g. cameras, laser scanners, etc.) can be operated in this setup. Regarding the onboard sensory system, the Visual-Inertial (VI) sensor (weight of 0.117 kg.) (Fig. 1) developed by Skybotix AG is attached below the hexacopter with a 45° tilt from the horizontal plane. The VI sensor is a monochrome global shutter stereo camera with 78° FOV, housing an Inertial Measurement Unit (IMU) [31]. Both cameras and IMU are tightly aligned and hardware synchronized. The camera was operated in 20 fps with a resolution of 752 × 480 pixels, while the depth range of the stereo camera lies between 0.4 and 6 m.

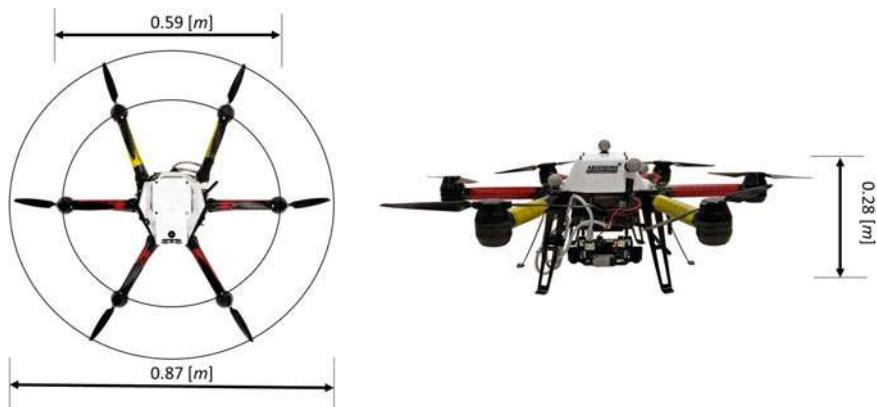


Fig. 1 AscTec NEO platform with the VI sensor attached

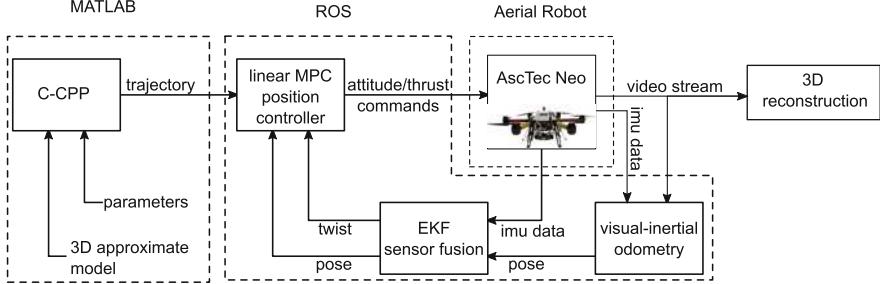


Fig. 2 Software and hardware components used for conducting inspections

The proposed C-CPP method, established in Sect. 4, has been entirely implemented in MATLAB. The inputs for the method are a 3D approximate model of the object of interest and specific parameters, which are the number of agents (n), the offset distance from the object (Ω), the FOV of the camera (α), the desired velocity of the aerial robot (V_d) and the position controller sampling time (T_s). The generated paths are sent to the NEO platforms through the utilization of the ROS framework.

The platform contains three main components to provide autonomous flight, which are a visual-inertial odometry, a Multi-Sensor-Fusion Extended Kalman Filter (MSF-EKF) [32] and a linear Model Predictive Control (MPC) position controller [26, 33, 34]. The visual-inertial odometry is based on the Robust Visual Inertial Odometry (Rovio) [35] algorithm for the pose estimation. It consists of an EKF filter that uses inertial measurements from the VI IMU (accelerometer and gyroscope) during the state propagation and the visual information is utilized during the filter correction step. The outcome of the visual inertial odometry are the position-orientation (pose) and the velocity (twist) of the aerial robot. Afterwards, the MSF-EKF component fuses the obtained pose information and the NEO IMU measurements. This consists of an error state Kalman filter, based on inertial odometry, performing sensor fusion as a generic software package, while it has the unique feature of being able to handle delayed and multi-rate measurements, while staying within computational bounds. The linear MPC position controller [34] generates attitude and thrust references for the NEO predefined low level attitude controller. The image stream from the overall experiment is processed using the discussed method in Sect. 4, while the overall schematic of the experimental setup is presented in Fig. 2.

6 Experimental Results

To evaluate the performance of the method, in a real autonomous inspection task, an outdoors experiment was conducted. For this purpose the Luleå University's campus fountain has been selected to represent the actual infrastructure for the cooperative aerial inspection. The fountain has a radius of 2.8 m and height 10.1 m without

branches. The area is considered rural, while surrounded by multiple buildings and also vegetated in some places. The UAV navigated in a constrained area around the Fountain, for safety purposes 2m away from the structure and the experiment was bounded in a cylindrical area with radius of 10m. In order to achieve a full autonomous flight, the localization of the UAV relied only on the onboard sensory system. Thus, the UAVs followed the assigned paths, based on visual-inertial odometry localization. Before the beginning of the experiment an initialization process was followed to fix the origin of the coordinate frame of UAVs close to the base of the fountain. This initialization step was considered for simplicity purposes due to the robocentric approach of the localization component that fixes the origin at the position where the algorithm is initiated. It should be noted that there was little wind during the described flight tests and the background was mainly static. However, people were passing or standing to observe the experiment, which were considered to be dynamic. Overall, the localization algorithm shows promising results despite to these small variations in environment. This can be resulted from enough texture in the inspected structure. The starting point has 180° difference and the cooperative scheme reduce the flight time from 327 s in case of one agent to 166 s with two agents. The average velocity along the path was 0.2 m/s and the points fed to the agents in a way to guarantee the maximum distance and avoid collision. It should be highlighted that during the experiment all processes were executed on board the aerial platform, while the 3D mesh was build offline from a ground station. During the experiment, the UAVs took-off manually and when they reached specific height switched to autonomous navigation. Similar strategy was followed for the landing of the vehicles. These steps are mainly done for safety reasons of landing and taking off. The actual and reference trajectories followed by both platforms are depicted in Fig. 3.

The 3D model of the structure was build offline using the dataset collected from both aerial agents. The extracted images were combined and processed by the SfM

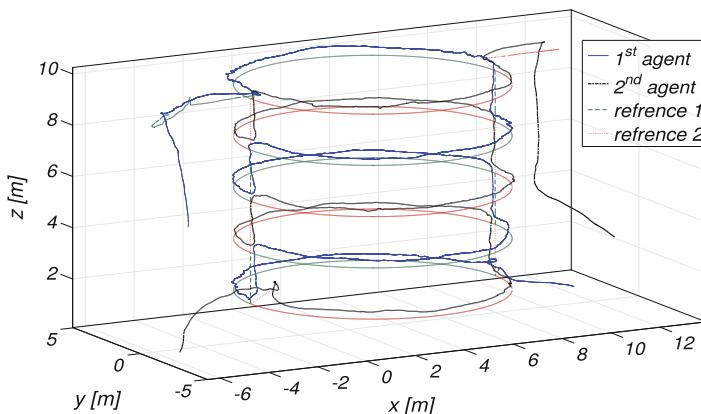


Fig. 3 Trajectories which are followed in outdoor experiment

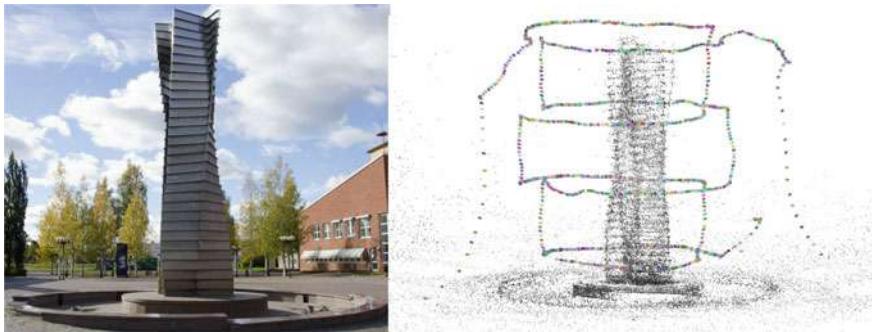


Fig. 4 On the left is the Luleå University outdoor fountain, and, on the right, the cooperative pointcloud of the structure with estimated flight trajectories

algorithm as described in Sect. 4. The fountain and its sparse 3D model are presented in Fig. 4.

In the proposed experiment the same strategy as indoor experiment is followed for two agents. The starting position of each of them has the maximum of distance with 180° difference. The overall flight time is reduced from 370 to 189 s and the average velocity along the path was 0.5 m/s. The sparse reconstruction provides an initial insight regarding the object to inspect, while an extra step is followed to create a 3d mesh. To retrieve the 3D mesh of the structure Autodesk ReCap 360 was used [36]. ReCap 360 is an online photogrammetry software suited for accurate 3D modeling. The reconstructed surface obtained from image data, is shown in Fig. 5. The



Fig. 5 Cooperative 3D mesh of the outdoor structure

results show that the collaborative scheme of the path planner could be successfully integrated for automating inspection tasks (<https://youtu.be/c4q2T5eqYRk>).

7 Conclusions

This article presents an aerial tool towards the autonomous cooperative coverage and inspection of a 3D infrastructure using multiple Unmanned Aerial Vehicles (UAVs). The proposed approach deploys multiple aerial robots and generates collision free trajectories for the inspection of the 3D structure. The aim of this application is to assign different parts of the scene to each agent for complete structure coverage in short time, considering the agents navigate autonomously. The visual information collected from the aerial team is collaboratively processed to create the dense 3D model, which can be used for inspection purposes. The experimental evaluation of the proposed inspection system demonstrated substantial performance in realistic outdoor cases that could act as an enabler for further developments.

References

1. European investment bank. Public and private financing of infrastructure, Evolution and economics of private infrastructure finance (2010)
2. U.P.A.C. Committee. Upac safety guide for helicopter operators. Technical report
3. Fernandes, R.A.: Line-mounted, movable, power line monitoring system. US Patent 4,904,996, 27 Feb 1990
4. Metni, N., Hamel, T.: A UAV for bridge inspection: visual servoing control law with orientation limits. *Autom. construct.* **17**(1), 3–10 (2007)
5. Burri, M., Nikolic, J., Hürzeler, C., Caprari, G., Siegwart, R.: Aerial service robots for visual inspection of thermal power plant boiler systems. In: 2012 2nd International Conference on Applied Robotics for the Power Industry (CARPI), pp. 70–75. IEEE (2012)
6. Cheng, P., Keller, J., Kumar, V.: Time-optimal UAV trajectory planning for 3d urban structure coverage. In: 2008 IEEE RSJ International Conference on Intelligent Robots and Systems, pp. 2750–2757. IEEE (2008)
7. Alexis, K., Nikolakopoulos, G., Tzes, A., Dritsas, L.: Coordination of helicopter UAVs for aerial Forest-Fire surveillance. In: Applications of Intelligent Control to Engineering Systems, pp. 169–193. Springer, Netherlands (2009)
8. Kanellakis, C., Terreran, M., Kominiak, D., George, N.: On vision enabled aerial manipulation for multirotors. In: 2017 IEEE 22nd International Conference on Emerging Technologies and Factory Automation ETFA (2017)
9. Galceran, E., Campos, R., Edifici IV, P., Palomeras, N., de Peguera, P., Ribas, D., Carreras, M., Ridao, P.: Coverage path planning with realtime replanning and surface reconstruction for inspection of 3d underwater structures using autonomous underwater vehicles
10. Kanellakis, C., Mansouri, S.S., Nikolakopoulos, G.: Dynamic visual sensing based on MPC controlled UAVs. In: 2017 25th Mediterranean Conference on Control and Automation (MED) (2017)
11. Kanellakis, C., Nikolakopoulos, G.: Survey on computer vision for UAVs: current developments and trends. *J. Intel. Robot. Syst.* 1–28 (2017)

12. Dimarogonas, D.V., Kyriakopoulos, K.J.: Decentralized navigation functions for multiple robotic agents with limited sensing capabilities. *J. Intel. Robot. Syst.* **48**(3), 411–433 (2007)
13. Pallottino, L., Scordio, V.G., Bicchi, A., Frazzoli, E.: Decentralized cooperative policy for conflict resolution in multivehicle systems. *IEEE Trans. Robot.* **23**(6), 1170–1183 (2007)
14. Galceran, E., Carreras, M.: A survey on coverage path planning for robotics. *Robot. Auton. Syst.* **61**(12), 1258–1276 (2013)
15. Mansouri, S.S., Kanellakis, C., Wuthier, D., Fresk, E., Nikolakopoulos, G.: Cooperative aerial coverage path planning for visual inspection of complex infrastructures. [arXiv:1611.05196](https://arxiv.org/abs/1611.05196) (2016)
16. Atkar, P.N., Choset, H., Rizzi, A.A., Acar, E.U.: Exact cellular decomposition of closed orientable surfaces embedded in R^3 . In: *IEEE International Conference on Robotics and Automation*, ICRA Proceedings, vol. 1, pp 699–704 (2001)
17. Brusell, A., Andrikopoulos, G., Nikolakopoulos, G.: A survey on pneumatic wall-climbing robots for inspection. In: *2016 24th Mediterranean Conference on Control and Automation* (MED), pp. 220–225 (2016)
18. Englot, B., Hover, F.S.: Sampling-based coverage path planning for inspection of complex structures (2012)
19. De Carvalho, R.N., Vidal, H.A., Vieira, P., Ribeiro, M.I.: Complete coverage path planning and guidance for cleaning robots. In: *Proceedings of the IEEE International Symposium on Industrial Electronics* (ISIE), vol. 2, pp. 677–682 (1997)
20. Galceran, E., Carreras, M.: Planning coverage paths on bathymetric maps for in-detail inspection of the ocean floor. In: *2013 IEEE International Conference on Robotics and Automation* (ICRA) pp. 4159–4164. IEEE (2013)
21. Galceran, E., Campos, R., Palomeras, N., Carreras, M., Ridao, P.: Coverage path planning with realtime replanning for inspection of 3d underwater structures. In: *IEEE International Conference on Robotics and Automation* (ICRA), pp. 6586–6591 (2014)
22. Bircher, A., Kamel, M., Alexis, K., Burri, M., Oettershagen, P., Omari, S., Mantel, T., Siegwart, R.: Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Autonomous Robot.* (1–20) (2015)
23. Barrientos, A., Colorado, J., del Cerro, J., Martínez, A., Rossi, C., Sanz, D., Valente, J.: Aerial remote sensing in agriculture: a practical approach to area coverage and path planning for fleets of mini aerial robots. *J. Field Robot.* **28**(5), 667–689 (2011)
24. Mansouri, S.S., Georgoulas, G., Gustafsson, T., Nikolakopoulos, G.: On the covering of a polygonal region with fixed size rectangles with an application towards aerial inspection. In: *25th Mediterranean Conference on Control and Automation* (MED) (2017)
25. Mansouri, S.S., Karvelis, P., Georgoulas, G., Nikolakopoulos, G.: Remaining useful battery life prediction for UAVs based on machine learning. In: *20th World Congress of the International Federation of Automatic Control* (IFAC) (2017)
26. Alexis, K., Nikolakopoulos, G., Tzes, A.: Model predictive quadrotor control: attitude, altitude and position experimental studies. *IET Control Theory Appl.* **6**(12), 1812–1827 (2012)
27. Hartley, R., Zisserman, A.: *Multiple View Geometry in Computer Vision*. Cambridge University Press (2003)
28. Nistér, D.: An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(6), 756–770 (2004)
29. Gao, X.-S., Hou, X.-R., Tang, J., Cheng, H.-F.: Complete solution classification for the perspective-three-point problem. *IEEE Trans. Pattern Anal. Mach. intel.* **25**(8), 930–943 (2003)
30. Robot Operating System (ROS). <http://www.ros.org/>
31. Fresk, E., Mansouri, S.S., Kanellakis, C., Nikolakopoulos, G.: Reduced complexity calibration of Mems Imus. In: *25th Mediterranean Conference on Control and Automation* (MED) (2017)
32. Lynen, S., Achtelik, M., Weiss, S., Chli, M., Siegwart, R.: A robust and modular multi-sensor fusion approach applied to mav navigation. In: *Proceeding of the IEEE/RSJ Conference on Intelligent Robots and Systems* (IROS) (2013)
33. Alexis, K., Nikolakopoulos, G., Tzes, A.: Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances. *Control Eng. Pract.* **19**(10), 1195–1207 (2011)

34. Kamel, M., Stastny, T., Alexis, K., Siegwart, R.: Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system. In: Koubaa, A., (ed.) Robot Operating System (ROS) the Complete Reference, Springer
35. Bloesch, M., Omari, S., Hutter, M., Siegwart, R.: Robust visual inertial odometry using a direct ekf-based approach. In: International Conference on Intelligent Robots and Systems (IROS), pp. 298–304. IEEE (2015)
36. Autodesk recap 360. <http://recap360.autodesk.com/>

Autonomous Aerial Inspection Using Visual-Inertial Robust Localization and Mapping

Lucas Teixeira, Ignacio Alzugaray and Margarita Chli

Abstract With recent technological breakthroughs bringing fully autonomous inspection using small Unmanned Aerial Vehicles (UAVs) closer to reality, the community of Robotics has actively been developing the real-time perception capabilities able to run onboard such constraint platforms. Despite good progress, realistic deployment of autonomous UAVs in GPS-denied environments is still rudimentary. In this work, we propose a novel system to generate a collision-free path towards a user-specified inspection direction for a small UAV using monocular-inertial sensing only and performing all computation onboard. Estimating both the previously unknown scene and the UAV’s trajectory *on the fly*, this system is evaluated on real experiments outdoors in the presence of wind and poorly structured environments. Our analysis reveals the shortcomings of using sparse feature maps for planning, highlighting the importance of robust dense scene estimation proposed here.

1 Introduction

The remarkable agility of small Unmanned Aerial Vehicles (UAVs) has been drawing growing interest in tasks, such as aerial surveying and industrial inspection (e.g. of wind turbines¹). Their employment in open fields can be automated using pre-defined GPS waypoints. However, such methods do not only assume the absence of any obstacles at the pre-specified flight altitude, but also the availability of reliable GPS signals, restricting their applicability. As a result, most missions employing UAVs today, resort to manually driven flights around the structures of interest, albeit

L. Teixeira (✉) · I. Alzugaray · M. Chli
Vision for Robotics Lab (V4RL), ETH Zurich, Zürich, Switzerland
e-mail: lteixeira@mavt.ethz.ch

I. Alzugaray
e-mail: aignacio@student.ethz.ch

M. Chli
e-mail: chlim@ethz.ch

¹<http://aeroworks2020.eu>.

limiting dramatically the inherent agility of UAVs to the pilot's field of view and judgement of clearance to structures.

Combining promising state-of-the-art building blocks, in this paper, we present a novel system capable of generating autonomously collision-free trajectories for aerial inspection in potentially GPS-denied environments. Despite the ability of stereo image processing in providing reliable scene depth estimates, monocular sensing in preferred onboard small UAVs as the stereo baseline most often proves too small to be effective in general missions, such as to provide clearance from structures. Factoring in the possibility of external disturbances, such as wind gusts, a clearance range of 10–20 m is recommended. As a result, the minimal setup of monocular-inertial sensing is typical for Simultaneous Localization And Mapping (SLAM) onboard small UAVs, respecting the platform's constraining payload and computational capabilities.

Starting off without any prior knowledge of the scene, the proposed system builds on monocular-inertial SLAM to obtain a sparse map of the UAV's surroundings and estimate a denser scene representation as in [18]. Employing the Monocular-Inertial SLAM based Planner (MISP) [2], demonstrated to be suitable for aerial inspection in simulation, our system plans the UAV's path to a pre-specified inspection direction using its current estimate of the scene, as shown in Fig. 1. The structure to be inspected is assumed to be a static manifold with enough free space around to safely fly. Exploiting the power of MISP to generate a collision-free path via continuous re-planning, any structures encountered are inspected and used as a source of localization cues. Evaluating MISP in real experiments for the first time, we demonstrate that our denser scene representation provides a far more practical alternative to feature-based maps used in [2].

2 Related Work

The complex task of automating inspection requires that the robot's spatial perception and path-planning capabilities are synchronized and able to cope with the uncertainties arising in a real mission. While there is a vast body of literature addressing either robotic perception or planning, it is due to the inherent difficulty in dealing with uncertainties in both processes at the same time, that there is only a handful of works addressing their combination and employment in real scenarios. On of the first works to demonstrate successful path-planning for UAVs was the framework of [1], which used Rapidly-exploring Random Belief Trees [6] on a pre-acquired map of visual features to predict subsequent scene measurements. The prohibitive cost of this approach, however, requires a base station for off-board computations, while its employment of known maps limits its applicability to general tasks. Tackling path-planning for an incrementally built map using SLAM, the monocular setup of [15] employed a set of heuristic rules to indirectly evaluate the quality of the generated paths, leading to an improvement in performance of the SLAM system in small scale scenarios.

In the most naive approach for collision avoidance, ultrasound range finders are typically used as a last-minute resort to avoid obstacles (e.g. in parking guidance for cars applications). However, it is depth and/or stereo cameras that are mostly often employed to enable more sophisticated obstacle avoidance strategies, as they provide denser scene information. Nonetheless, the limited range of both ultrasound or RGBD sensors as well as their sensitivity to the environment renders them often impractical in general scenarios.

Investigating the applicability of high performing real-time stereo matching algorithms, we tested ELAS [9] and SGBM [10] to estimate a stereo-based scene reconstruction. However, as predicted theoretically, the stereo baseline is relatively too small with respect to the depth of the scene in the mid-range inspection addressed here, producing poor scene estimates, thus rendering such systems unsuitable. As aforementioned, the monocular-inertial setup employed in this work is typical in UAV navigation, however, it poses great challenges in robust and denser scene estimation. The most recent works of monocular ORB-SLAM [16], and monocular-inertial ROVIO [5] and OKVIS [13], demonstrated that robust robot localization using a sparse set of visual landmarks can be performed in real-time even without using cues from a secondary (stereo) camera. ROVIO, for instance, tracks only about 20 visual features per frame, while OKVIS can track about 200. While ROVIO's map is clearly too sparse for meaningful use during path-planning, OKVIS's map often provides a more preferable (i.e. denser) distribution features in space. Denser real-time monocular scene estimation approaches, such as LSD-SLAM [8] offer more complete scene representations, but are too noisy to use for path-planning.

In this work, we employ the low-cost approach for denser scene representation from monocular views of [18]. The outlier removal, smoothing and interpolation of the SLAM landmarks performed by this algorithm overcome the problem of uneven distribution of landmarks and noisy measurements, resulting to a favourable map for path planning. Employing MISP [2], the generated UAV trajectory explicitly considers the motion constraints of a robot with a monocular setup, improving the overall performance of the SLAM system.

3 System

Our system for aerial inspection using visual and inertial sensing cues builds on top of the open-source ETHZ-ASL ROS stack for visual-inertial autonomous flights.² This framework permits the use of the three sensors that we have onboard our UAV; a monocular camera with an Inertial Measurement Unit (IMU) attached to it and another IMU inside the body of the UAV (at its center). The former IMU and the camera are embedded in the same circuit for time-synchronization as in the Intel RealSense ZR300 Camera. The latter IMU is used by the UAV's autopilot system inside the Attitude Controller.

²http://wiki.ros.org/asctec_mav_framework.

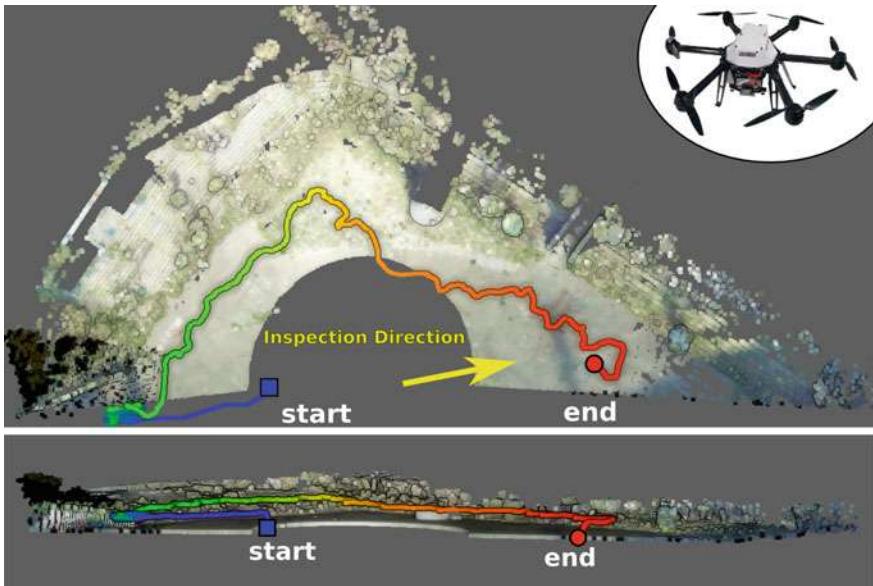


Fig. 1 The top view (top) and the side view (bottom) of a successful run from the start position towards the inspection direction, until the pilot decides to take over (labelled as “end”). Note that the planner chooses to follow the structure, while planning the robot’s path towards the inspection direction as the structure is a rich source of visual cues, and thus ensures the system’s robustness. The color of the trajectory represents the time. The platform used for all the experiments in this paper is the AscTec Neo, shown in the inset

Using the sensor information from all onboard sensors, the proposed system, illustrated in Fig. 2, is able to accurately estimate on the fly and in real-time the UAV’s motion while building a map of the environment. Using such information, the system plans and executes a collision-free path to the user-defined inspection direction, while following closely the structure in front of the UAV (i.e. any structure with visual features).

To this end, the autonomous flight stack, depicted in gray in Fig. 2, is composed of a visual-inertial SLAM system, the keyframe-based odometry algorithm OKVIS [13], the EKF-based Multi-Sensor Fusion algorithm [14], and the linear model-predictive position controller of [12]. This stack is well tested and used in several works, such as [3, 17]. Employing a variant of MISP [2], here we modified the original path-planning algorithm to enable its practicality and feasibility in real-life experiments. Having been only illustrated in simulation so far, the original MISP uses directly the landmarks provided by the visual-inertial SLAM (OKVIS), as illustrated by the red dashed arrow. In this work, we propose to use a robust scene depth estimation module as we demonstrate that the naive use of the pure visual-inertial SLAM landmarks does not provide an accurate enough scene representation in real missions. All of the components of the proposed pipeline run in the onboard UAV’s CPU.

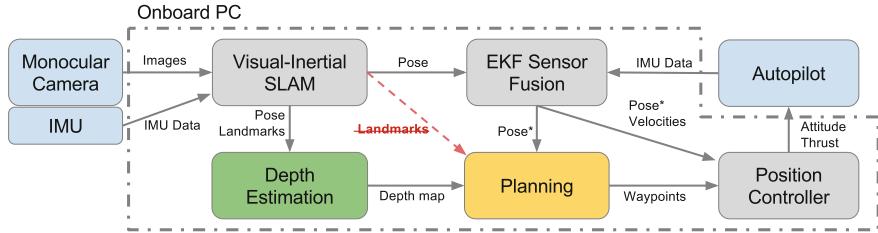


Fig. 2 The architecture of our system. Using both the feeds from the external IMU and monocular camera, and the autopilot’s IMU, the UAV’s onboard computer runs the autonomous flight stack (gray blocks), which estimates the UAV’s pose and a map of landmarks of the UAV’s surroundings. Instead of naively using the landmarks to plan the UAV’s path to the goal (using the red-dashed arrow), our experiments demonstrate that introducing a denser scene estimation module (green block) results to more robust and accurate performance for aerial inspection

In the rest of this section, a brief description of the MISP algorithm is presented, together with our strategy for real-time and robust depth estimation and their integration within the proposed pipeline.

3.1 Path Planning Based on MISP

MISP [2] is a path planning algorithm specially designed for small UAVs with limited computational capacity using monocular-inertial localization and mapping. The algorithm exploits the structure of an *a priori* unknown map to guide the robot towards a known goal position, adapting the navigation as new areas are explored by means of an quick re-planning policy. The efficiency of the algorithm relies on the reuse of most of the underlying information between consecutive planning iterations by limiting its reaction to only local changes in the map.

MISP facilitates the monocular localization by planning around the obstacles in the map, while tracking and orienting the robot towards the visual features on them. This path planning leads to a tangential navigation with respect to the local obstacles that is most suitable for the inspection task proposed in this work. The algorithm is originally designed as point-to-point planner, whereas the mission proposed in this work does not consist of reaching a specific location, but rather inspecting the structure that the UAV is facing at the time that MISP gets initialized. For this reason, here we adapt the original algorithm to be guided by a generic and user-defined inspection direction instead of a goal location as depicted in Fig. 3. In [2], MISP achieves good results in an inspection task in a simulated environment. Here, we address in detail the limitations of original algorithm when it is applied to real scenarios in the following sections.

MISP employs a probabilistic 3D grid map representation [11] to register the obstacles in the map on-the-fly. The obstacles in the map are considered sources of visual features and thus, the accurate tracking of such features has critical impact

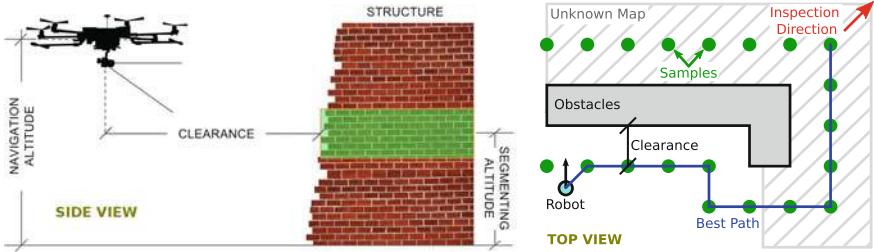


Fig. 3 In an inspection task, the MISP segments part of the inspected structure at a given segmenting altitude h_{seg} . The segmented part of the structure, in green, is then used to generate paths keeping a constant clearance at navigation altitude h_{nav} . On the right, the MISP computes the best path towards the inspection direction. The structure of the environment in the unknown map areas is inferred based on the current map, relying on the efficient re-planning policy to adapt the navigation to upcoming changes

in the performance of the employed vision-based SLAM system. Inspired by Cover et al. [7], MISP generates position samples around to the map obstacles at a given clearance distance ρ enforcing a collision-free navigation.

In this work, we restrict the algorithm to plan in a plane at a fixed navigation altitude h_{nav} , as proposed originally in [2]. For the proposed inspection task, we also define the segmenting altitude h_{seg} at which the structure of the environment is considered of interest and thus any obstacles at this altitude are used to generate the position samples as previously described (See Fig. 3).

The position samples surrounding the obstacles of interest are connected to other neighbouring position samples via collision-free paths generating a graph in which the current robot and goal position are also included. The graph is then searched for the best path connecting the current position to the goal. The weights of the graph are designed so that the resulting best path favours the robot to navigate close to the obstacles at the given predefined clearance distance ρ , even if it implies a larger overall travelled distance towards the goal.

We redefine the “goal” in the original MISP implementation to use an inspection direction instead, guiding the navigation of the robot while avoiding any obstacles in the way. We implement this modification by setting the goal location beyond the reach of the inspection area. The mission is considered completed as soon as the robot leaves the inspection area guided by the inspection direction while flying autonomously.

In each planning iteration, the segments of the best path closer to the current robot position are determined by the known local map, whereas any segments further away are generated by inferring the structure of the unknown map areas subject to changes. As a result, we only execute the path up to a given short travelled distance, usually a couple of meters, and subsequently repeat a planning iteration including any new updates of the map. This receding horizon strategy, inspired by Bircher et al. [4], allows the robot to re-plan its path and adapt its navigation as new areas in the map are explored.

The heading of the robot is defined to be almost perpendicular to the inspected structure’s surface to maximize the parallax of the perceived visual features, ensuring this way to boost the robustness of the SLAM system during the mission. The heading direction deviates slightly (e.g. $\sim 15^\circ$) from the nominal perpendicular heading towards the navigation direction in order to safely detect upcoming changes in the structure. The original MISP implementation defines an additional set of so-called “recovery behaviours” to overcome other endangering situations when navigating close to the obstacles. Since this work focuses on medium-large range inspection, we do not consider such heuristics. Although the structure is assumed to be a manifold, small gaps can be effectively handled by MISP, since it allows the robot to move from one obstacle to another inside of the inspection area, as well as the dense depth estimation, that is able to fill small areas without visual information in the map.

3.2 Robust and Dense Depth Estimation

The original MISP implementation [2] naively assumes that the raw landmarks estimated during SLAM are outlier-free, well distributed across the scene, and dense enough to allow a good enough estimate of the world using a probabilistic 3D voxel grid map at a coarse resolution. While these assumptions may hold in simulated environments, in real scenarios noisy scene estimates and textureless areas are common enough to cause large deviations of the resulting behaviour from the nominal expected performance, as illustrated in the experimental results presented in Sect. 4.

Considering the limited computational capacity and payload restrictions onboard a small UAV, in this work we focus on the already available monocular and inertial sensing cues to generate a denser scene representation, making the most of the onboard sensor suite. Inspired by the mesh-based scene representation of [18], we employ this method, which was specifically designed for aerial inspection to create a mesh out of the landmarks estimated by SLAM and then perform an outlier elimination and smoothing of this mesh. This system assumes that the area in the field of view is a continuous surface without small or thin objects protruding, although the surface can be non-planar and have small discontinuities, which most often holds in reality (see experimental setup of Sect. 4). Note that, although this mesh-based scene representation is also capable of providing surface normals, this feature is not used in this work.

As depicted in Fig. 8, the difference of the map estimation when using only raw landmarks as opposed to the mesh-based depth estimation pipeline [18] is clearly noticeable. This mesh is computed using only the raw landmarks and the camera pose estimates from SLAM. Firstly, the 3D landmarks are checked for neighbourhood support (in their depth estimates) in order to filter out unreliable landmarks, i.e. the ones with much larger uncertainty within their surroundings. The remaining landmarks are projected onto the image plane corresponding to the current viewpoint and a mesh is constructed via a 2D Delaunay Triangulation. An outlier detection algorithm then removes any triangles that do not fulfil the algorithm’s smoothness criteria; very

oblique mesh-triangles and spikes in the mesh are rejected. The remaining mesh-vertices are smoothed with respect to their depth using a Laplacian filter, before interpolation is applied by a rasterization algorithm. The outcome of this pipeline is a smooth mesh representation of the scene, providing one depth measurement per pixel in the image space and resulting into a depth map with the same resolution as the camera image. For all our tests, we use the default parameters provided in the open source implementation of the authors.³ Please refer to the original paper [18] for detailed explanation.

4 Experiments

In order to evaluate the proposed system, we carried experiments out in the ruins of the Hardturm Stadium in Zurich visible in Fig. 4. We choose to inspect the largest structure of the stadium’s stands (*aka* bleachers). This structure is about 8 m tall and forms an almost continuous surface covering a 100 m × 100 m area. Ground-truth data was recorded for the test site by capturing a laser point cloud of the scene using the Leica MS50 Station Theodolite. The ground-truth for the UAV’s trajectory in each run, was generated by post-processing each sequence with OKVIS [13] using unbounded optimization times and extended optimization windows.

During the experiments the algorithms face additional challenges, such as light wind gusts. The wind does not only interfere with the UAV’s trajectory, but also with the environment due to the fact that trees and bushes cover a large area in front of the structure. Additionally, there is a featureless white tent in the area which adds difficulty in performing SLAM robustly as this can occupy a large portion of the UAV’s field of view at times.

Once the algorithm is initialized, the pilot hands over all control of the UAV’s motion to the proposed system and takes back the control either at the end of the route or if the pilot judges that the UAV enters a dangerous situation (e.g. flies too close to a structure). The inspection direction is set as shown in Fig. 7.

Following a few test-flights, we set the clearance to 10m for this setup and the current wind conditions. Making sure that the UAV is facing the structure to be inspected at the beginning of each mission, the origin of the UAV’s coordinate system is set at the position where the SLAM module is initialized. This initial position also determines the segmenting altitude, while the navigation altitude is set at 2 m higher due to the sensor’s slightly downward-tilted configuration as illustrated in Fig. 3. For fairness of comparisons amongst different experiments, the segmenting altitude is set to 2 m measured by the UAV’s barometer and the navigation altitude at 4 mm high. Note that the barometer readings can be affected by the wind, so these settings can only be approximate. In this setup, the rest of this section analyses the capability of our system in respecting the navigating altitude and the clearance. Further on, the quality of the generated map and the UAV trajectories are discussed, on the basis of

³https://github.com/VIS4ROB-lab/mesh_based_mapping.



Fig. 4 Aerial (top) and ground photos (bottom) of the test site. Aerial image source: Swiss Panorama

five different flights with the same starting pose and inspection direction; three of these using the mesh-based robust dense depth estimation and two using only the SLAM sparse landmark map as a representation of the scene.

4.1 Navigation Altitude

A constant navigation altitude is important not only for collision avoidance during the navigation, but also for acquiring images with sufficient overlap, crucial for robust navigation and effective inspection. Figure 5 illustrates the altitude measured during each of the five flights. There is a clear drift tendency of descending UAV altitude, from the SLAM origin and increasing with respect to the travelled distance and the experiment’s duration. As a result, the mesh-based flights have similar descent rate, but larger absolute drift than the sparse-based flights, because the travelled distance is also longer (see Table 1).

4.2 Clearance

Although MISP aims to maintain a constant clearance to the structure, in practice it is a difficult task for the position controller to stabilize the UAV in presence of wind, specially when all computations are based on noisy pose and map estimates. As a result, some variance on the actual clearance is expected in real experiments, but the average clearance achieved should be similar to the specified. Most importantly, the UAV should never fly too close to the structure for safety reasons.

Extracting the same segmented region from the scene ground truth and in order to evaluate the actual clearance during flights, we build a 2D map (i.e. top view)

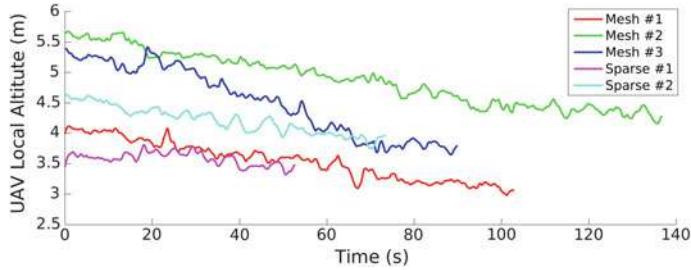


Fig. 5 The ground-truth of the navigation altitude throughout each flight, revealing drift in all trials increasing with travelled distance and flight duration

Table 1 Duration and travelled distance for all five flights used in this analysis. Note that the sparse-based flights are aborted prematurely as they are unable to safely complete the mission

Flight-label	Mesh #1	Mesh #2	Mesh #3	Sparse #1	Sparse #2
Duration (s)	103	137	90	53	74
Distance (m)	120	125	102	62	77

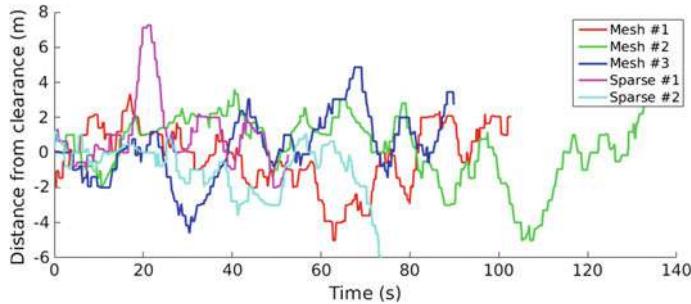


Fig. 6 Distance between the actual UAV's position and the desired clearance set for the trajectory generation. While sometimes this error is large, in the mesh-based flights on average it approaches zero. During flight Sparse #2, the UAV gets too close from the structure and the safety pilot took over

with the same 1m-resolution as the probabilistic 3D voxel grid representation used in the planner, essentially constructing a binary image. Using this image we calculate the distance field to estimate the actual distance of the UAV (from the trajectory's ground truth) to the closest part of the structure. Figure 6 plots the distance of the actual position of the UAV to the desired clearance (here, set to 10 m). The mesh- and the sparse-based approaches exhibit comparable performance, probably due to the fact that the test site was overall sufficiently textured, except the tent region. Self-similar structure, however, e.g. arising from the vegetation around the structure of interest, generates erroneous feature matches. While these can be filtered out to a certain degree during mesh-based flights, for both sparse-based flights this proves fatal as discussed in Sect. 4.3.

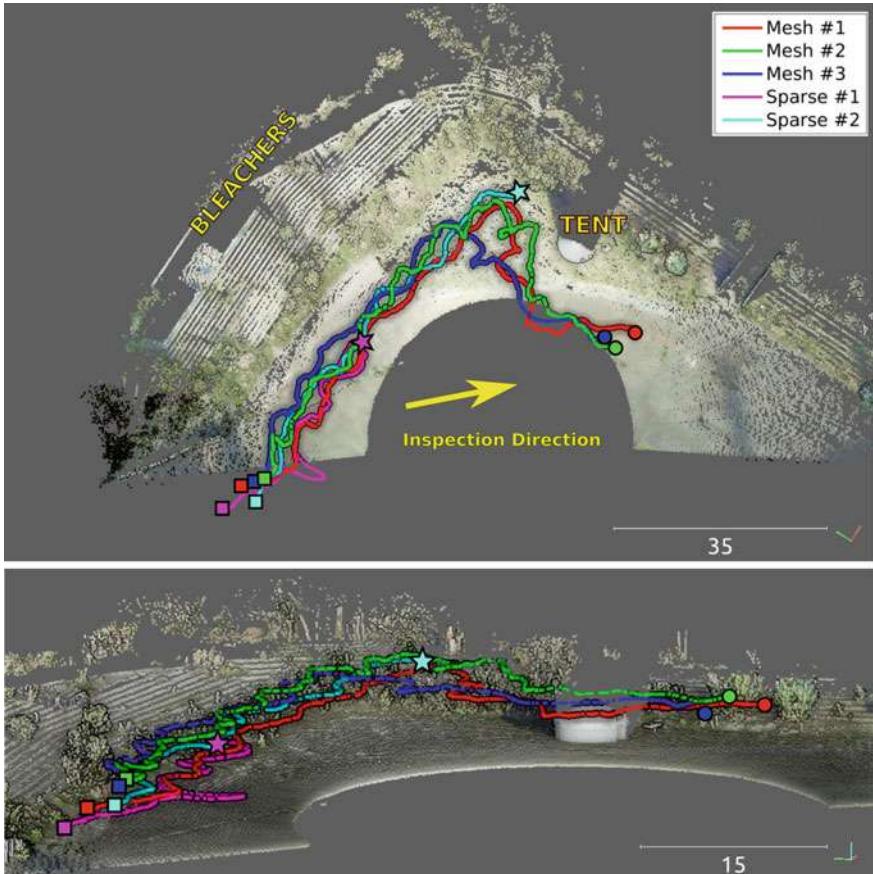


Fig. 7 All the runs of the proposed pipeline with either options of using the sparse landmark map or the mesh-based scene representation, shown in the top view (top) and a side view (bottom). All experiments start from roughly the same position (marked with the square) and have the same inspection direction. All mesh-based runs autonomously reach the boundaries of the experimental area (marked with a disc), while both sparse-based runs end prematurely (marked with a star)

4.3 Trajectory Generation and Scene Estimation

Figure 7 depicts the 3D trajectories of all five flights, as MISP plans a path, such that the UAV inspects and follows the structure in the field of view. Each experiment started at the square marker. An experiment is finished either when the UAV reaches the boundaries of the experimental area marked with a disc in Fig. 7 or when the UAV's trajectory reveals large errors in the estimation processes and the pilot decides to take over (marked with a star) for safety reasons. More specifically, when the UAV loses the structure of interest from its field of view, it is a point of no return revealing that a path is planned on an invalid map containing too many outliers, and this leads to

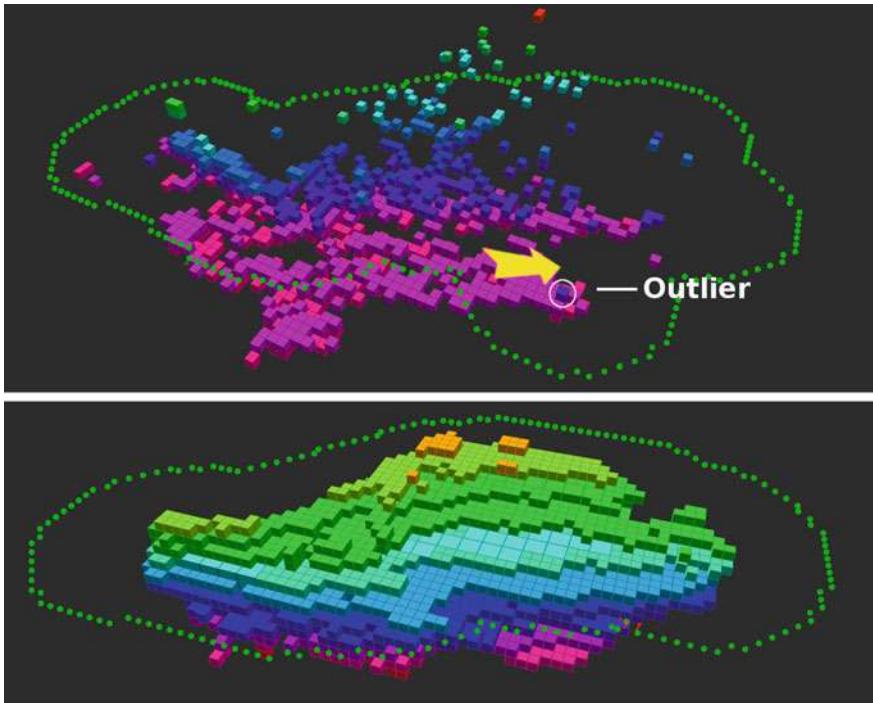


Fig. 8 The sparse scene estimation (top) at the moment when flight Sparse #1 fails due to the falsely perceived outlier very close to the UAV (yellow arrow is UAV’s pose at that moment). As soon as the outlier is encountered a new path is planned (green nodes) around the erroneous obstacle estimate, causing the UAV to lose track of the structure of interest. For comparison, the denser map estimated during the successful flight Mesh #3 is superimposed in the bottom image

the decision of ending the experimental flight prematurely. Another case that leads to premature take-over by the pilot is the segmentation of the ground plane (i.e. containing no free space) due to vertical drift. This is very problematic situation that can be more common in our test setup, where we fly close to the ground, however typically aerial inspection is conducted at high altitudes practically eliminating this failure case.

As clearly depicted in Fig. 7, both experiments relying on the sparse scene estimation have to be interrupted by the pilot, while all flights employing the mesh-based alternative are able to reach the boundaries of the flying area. In fact, mesh-based dense depth estimation exploits the advantage of accessibility to a denser scene estimation accumulated over time (i.e. multiple views of the same area), outliers can be removed effectively resulting to a far less noisy scene estimation than the sparse counterpart. Generally, most of the outliers in the scene estimation arise during inaccurate registration of the current view to the map, following inaccurate pose estimation from SLAM. Such outliers are very unlikely to be replaced by correct measurements later on in the trajectory, when employing sparse depth estimation traditionally used in SLAM.

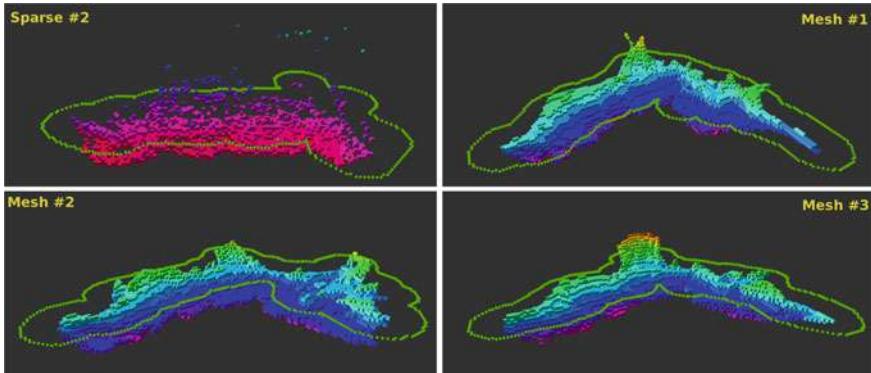


Fig. 9 The final 3D octomaps retrieved during different flights

Figure 8 shows the probabilistic 3D grid map at the moment of the failure of flight Sparse #1. During navigation, an outlier landmark suddenly gets estimated to be very close to the current UAV position. As a result, the UAV is forced to move away from it to maintain the nominal clearance, rotating during this manoeuvre to face to the direction of this new obstacle that does not really exist, losing in the way the track of the actual structure to be inspected. In the bottom image of Fig. 8, the dramatically improved quality of the map when using the mesh-based map is evident.

The top-left image of Fig. 9 illustrates the sparse SLAM map as estimated at the moment of the failure of flight Sparse #2. In this case, this sparse estimation of the test site does not provide enough information to permit continuing the SLAM estimation during the UAV's rotation necessary for turning at the corner visible also in Fig. 7, with the planned trajectory expanding through the white tent—hence if the pilot would not take over, the UAV would collide with the tent.

5 Conclusion

This paper proposes a novel system composed of state-of-the-art vision-based perception and planning able to run onboard a small UAV to perform autonomous aerial inspection. Without any a priori knowledge of the environment, the UAV's surroundings are estimated in real-time and the UAV's trajectory to inspect the structure of interest is constantly re-planned as newly perceived scene information enters the system. The evaluation of this system on real and challenging experiments outdoors reveals the power of the proposed method to deal with noisy estimates in the perception of the UAV's motion and the scene.

Acknowledgements This research was supported by the Swiss National Science Foundation (SNSF, Agreement no. PP00P2_157585) and EC's Horizon 2020 Programme under grant agreement no. 644128 (AEROWORKS).

References

1. Achtelik, M.W., Lynen, S., Weiss, S., Chli, M., Siegwart, R.: Motion and uncertainty aware path planning for micro aerial vehicles. *J. Field Robot. (JFR)* **31**(4) (2014)
2. Alzugaray, I., Teixeira, L., Chli, M.: Short-term UAV path-planning with monocular-inertial slam in the loop. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2017)
3. Bircher, A., Kamel, M., Alexis, K., Burri, M., Oettershagen, P., Omari, S., Mantel, T., Siegwart, R.: Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Auton. Robots* **40**(6), 1059–1078 (2016)
4. Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., Siegwart, R.: Receding horizon “next-best-view” planner for 3D exploration. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1462–1468. IEEE (2016)
5. Bloesch, M., Omari, S., Hutter, M., Siegwart, R.: Robust visual inertial odometry using a direct EKF-based approach. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 298–304. IEEE (2015)
6. 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) (2011)
7. Cover, H., Choudhury, S., Scherer, S., Singh, S.: Sparse tangential network (SPARTAN): motion planning for micro aerial vehicles. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2013)
8. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular slam. In: Proceedings of the European Conference on Computer Vision (ECCV) (2014)
9. Geiger, A., Roser, M., Urtasun, R.: Efficient large-scale stereo matching. In: Proceedings of the Asian Conference on Computer Vision (ACCV) (2010)
10. Hirschmuller, H.: Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)* **30**(2), 328–341 (2008)
11. Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W.: OctoMap+: an efficient probabilistic 3D mapping framework based on octrees. *Auton. Robots* **34**(3) (2013)
12. Kamel, M., Stastny, T., Alexis, K., Siegwart, R.: Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system. In: Robot Operating System (ROS) The Complete Reference Volume 2. Springer (2017)
13. Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., Furgale, P.: Keyframe-based visual-inertial odometry using nonlinear optimization. *Int. J. Robot. Res. (IJRR)* (2015)
14. Lynen, S., Achtelik, M.W., Weiss, S., Chli, M., Siegwart, R.: A robust and modular multi-sensor fusion approach applied to MAV navigation. In: Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS) (2013)
15. Mostegel, C., Wendel, A., Bischof, H.: Active monocular localization: towards autonomous monocular exploration for multirotor MAVs. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). IEEE (2014)
16. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. *IEEE Trans. Robot. (T-RO)* (2015)
17. Oleynikova, H., Burri, M., Taylor, Z., Nieto, J., Siegwart, R., Galceran, E.: Continuous-time trajectory optimization for online UAV replanning. In: Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS) (2016)
18. Teixeira, L., Chli, M.: Real-time mesh-based scene estimation for aerial inspection. In: Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS) (2016)

Sensing Water Properties at Precise Depths from the Air

John-Paul Ore and Carrick Detweiler

Abstract Water properties critical to our understanding and managing of freshwater systems change rapidly with depth. This work presents an Unmanned Aerial Vehicle (UAV) based method of keeping a passive, cable-suspended sensor payload at a precise depth, with 95% of submerged sensor readings within ± 8.4 cm of the target depth, helping dramatically increase the spatiotemporal resolution of water science datasets. We use a submerged depth altimeter attached at the terminus of a 3.5 m semi-rigid cable as the sole input to a depth controller actuated by the UAV's motors. First, we simulate the system and common environmental disturbances of wind, water, and GPS drift and then use parameters discovered during simulation to guide implementation. In field experiments, we compare the depth precision of our new method to previous methods that used the UAV's altitude as a proxy for submerged sensor depth, specifically: (1) only using the UAV's air-pressure altimeter; and (2) fusing UAV-mounted ultrasonic sensors with the air-pressure altimeter. Our new method reduces the standard deviation of depth readings by 75% in winds up to 8 m/s. We show the step response of the depth-altimeter method when transitioning between target depths and show that it meets the precision requirements. Finally, we explore a longer, 8.0 m cable and show that our depth-altimeter method still outperforms previous methods and allows scientists to increase the spatiotemporal resolution of water property datasets.

1 Introduction

Monitoring shallow surface water systems (<10 m) can be a deep challenge for environmental and water scientists. Hampered by limited boat access, scientists are further constrained because boating and wading mix the water, disturbing the water properties under investigation. These water properties include temperature, conduc-

J. -P. Ore (✉) · C. Detweiler

Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, USA
e-mail: jore@cse.unl.edu

C. Detweiler

e-mail: carrick@cse.unl.edu

tivity, dissolved oxygen, and photosynthetically active radiation—all these vary sensitively with small changes in water depth [1]. These stratified properties are linked to ecosystem health and can predict imminent toxic algae blooms that threaten drinking water and fisheries and cost billions of dollars worldwide [2, 3].

Currently, scientists deploy static sensor arrays, often a collection of data-loggers vertically arranged underwater at a static position, and left for days or weeks. Some sensors used by water scientists require settling time (often >3 s), and existing water science datasets from static sensors have ≈ 0.25 m resolution in depth [4]. These static sensors yield datasets with good temporal resolution, but are limited by poor spatial resolution because they have to be installed separately at each location. To increase the spatiotemporal resolution, our prior work presented the first UAV-based water sampler [5], followed subsequently by several efforts [6–12]. The UAV flies above water while connected by cable to a sensor payload below water, as shown in Fig. 1. Our follow-on work [13] explored in-situ water sensing at different depths with a UAV-based submerged sensor payload, and found it does not cause mixing for temperature. However, in this follow-on work the UAV's air pressure altimeter caused the submerged sensor to wander up and down around the target depth, limiting precision.

Fig. 1 UAV-system with a passive cable and sensors for measuring water properties at precise depths



To increase the depth precision of water science datasets, this paper proposes and experimentally evaluates using a depth altimeter as input to a controller that seeks to minimize the target depth error using only the UAVs motors. We assume calm waters ('lentic', not flowing) and that the UAV is stationary. This is challenging because the semi-rigid cable between the UAV and the submerged depth altimeter exhibits non-linear dynamics when bending and descending in water. Fortunately, these depth sensors are fast (50 Hz), light (<5 g), and precise (± 3 mm), and are already included in many underwater sensor payloads. We first explore the feasibility of this approach in simulation including wind and water, then use parameters discovered in simulation to guide implementation, and finally test the implementation of our method in field experiments.

Our contributions are:

- A new method to maintain a precise depth for a submerged sensor payload while passively connected by cable to a UAV. We use only a lightweight submerged depth altimeter, reducing the standard deviation of target depth errors from 16.1 to 4.2 cm compared with the next best method. This method enables increased spatiotemporal resolution of water science datasets without additional payload.
- Field tests of this depth-altimeter method, with comparisons to two previous methods: (1) air pressure altimeter alone; and, (2) air pressure fused with ultrasonic rangefinders. We compare these approaches using a 3.5 m cable for water depths down to 2.5 m, showing an reduction of target depth error by 75%.
- An initial exploration of using an 8.0 m cable, showing sufficient precision for sensing at greater depths.

2 Related Work

Previous efforts relate to our current work in several ways: either a UAV is used to measure water properties, or autonomous surface/underwater vehicles (ASVs and AUVs) are used to measure water properties, or a UAV makes a pose and altitude estimation in unknown environments, or a UAV has a cable-suspended load.

Several efforts seek to use UAVs to monitor water properties [6–10, 12]. These systems use air pressure altimeters to estimate altitude, and like these efforts we seek to sense water properties with a cable-connected payload, but unlike these works we focus on precisely controlling the depth of the sensor payload. Systems like the fixed-wing Flying Fish [14] maintain persistent observation of surface properties, but cannot detect subsurface properties. Some efforts seek an amphibious UAV [15], and like this work we are interested in the advantages offered by UAVs to water monitoring, but unlike this work we seek to minimize water column mixing, that can take hours to settle. Our previous work presents evidence that sensing by a small, submerged sensor payload (≈ 2 cm diameter sensor, 1 cm cable) does not cause a mixing disturbance that impacts water temperature measurements [13].

In environmental monitoring, Dunbabin [16] uses multiple ASVs to sample greenhouse gasses, while underwater the MARES AUV system samples water quality for long duration at depths up to 100 m [17]. Zhang et al. [18] explore water columns with a gliding, fish-like robot, and Higgins and Detweiler [1] WaterBug descends passively in a water column to collect a single water sample at a target depth. Unlike these systems, we want to quickly deploy and redeploy to disconnected or difficult to reach water bodies.

Several approaches seek to estimate altitude for micro UAVs in unstructured outdoor environments, including Jain et al. [19], who autonomously explored rivers using specular laser returns to estimate the plane of the river surface. Unlike this work we seek a method that minimizes the payload devoted to non-water-property sensing. Burri et al. [20] use a stereo camera and IMU to map and estimate a pose in a previously unknown environment, but these methods have not been demonstrated over water, to our knowledge. Our previous work [7] explores the use of fusing downward-facing ultrasonic sensors with an air pressure altimeter, and we use this method for comparison in the current work.

Several efforts model and control cable-suspended payloads from a UAV [21, 22], and assume that the cable can be observed. We likewise use a cable-suspended payload, but do not attempt to observe the cable during flight.

3 Technical Approach

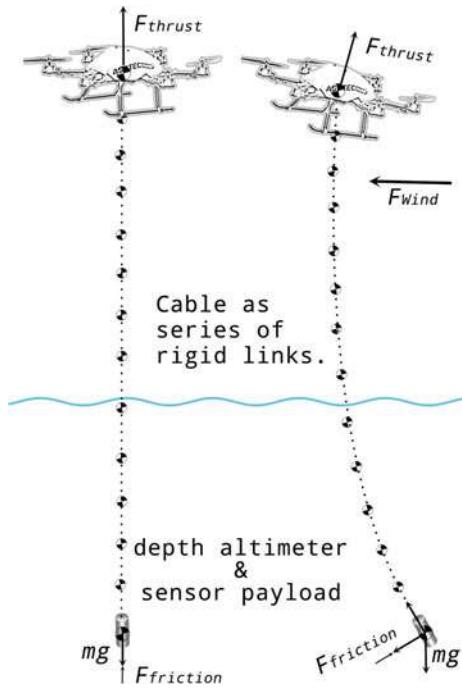
This section describes our approach to the problem of obtaining precise depth control of a submerged sensor payload with minimal system complexity while retaining the benefits of monitoring water properties by UAV. Based on our experience, we assumed that the biggest obstacles would be non-linear dynamics of the partially submerged cable and transient lateral disturbances from wind and GPS drift. But we also wondered about the impact of signal delay in measuring and transmitting depth readings before the controller can use them. We start by developing a model for how the cable-connected sensor moves in the water, and then explore the feasibility and initial parameters in simulation.

3.1 System Model and Simulation to Explore Feasibility

Figure 2 shows the UAV modeled as a point mass and the cable as a series of rigid links, following the approach in [22]. We use Simulink for two configurations, a ideal and perfectly vertical case, and a laterally disturbed case to model wind and GPS drift.

Modeling the Ideal Case. In the ideal case, the cable is vertical and all forces are parallel to gravity, as the UAV, cable, and depth altimeter move up and down. When the UAV moves up, the sensor payload is directly coupled. When the UAV moves

Fig. 2 Model of UAV-system hovering over still water, showing the suspended cable as a series of rigid links. The left side of the figure shows the ideal case, the right shows the system during a lateral disturbance from wind



down, the only forces acting on the payload are gravity and resistance from water. The resistance from water limits the sensor's descent rate to slower than the UAV.

To model the descent rate, we started by experimentally determining the terminal velocity of the sensor in water. We released the sensor at the top of a translucent 3 m vertical water test tank (≈ 25 cm radius by 4 m tall) and measured the time at which the sensor reached markings of depth 1 and 2 m. Using six observations, we estimate the average terminal velocity, v_t , to be 0.54 m/s. Then, using the terminal velocity equation:

$$v_t = \sqrt{\frac{2mg}{\rho AC_d}} \quad (1)$$

we calculate the constant ρAC_d , where ρ is the density of water, A is the area in the direction of the movement, and C_d is the drag co-efficient. Using this constant with the mass of the sensor, m , and gravity, g , we solve for the resistive force as a function of velocity:

$$F_{friction} = \frac{1}{2} \rho A C_d v^2 \quad (2)$$

We use $F_{friction}$ from Eq. 2 to model how the sensor will descend in water. In simulation, $F_{friction}$ limits the descent velocity of the sensor payload as well as making

it drag against lateral disturbances. Decreasing the target depth by <1 m allows the submerged sensor to descend fast enough to match the descent of the UAV. Decreasing the target depth by >1 m requires limiting the descent velocity of the UAV to avoid the UAV outpacing the submerged sensor.

Modeling Lateral Disturbances. Figure 2 also shows the lateral disturbance case from wind or GPS drift. We want to model wind and GPS drift because we believed lateral disturbances could be a critical impediment to our approach.

To model lateral disturbances, we are inspired by the U.S. Military Wind Gust Model [23], that uses a sigmoid function that ‘ramps up’ a gust over time. We model wind as the sum of two smoothed random number sequences that are directly added to the forces exerted on the UAV in x and y . The two sequences operate at different time scales, one changing every second and at a small scale, and the other changing every 15 s at a larger scale. Overall, a maximum force of 0.08 N tends to ‘blow the system off course’ by about 1–2 m, consistent with field observation. Note that during lateral disturbances, $F_{friction}$ acts on the submerged sensor payload and opposes the translation. The simulation indicates that the UAV-cable-sensor system remains stable in response to lateral disturbance while controlling depth using the depth altimeter with the UAV motors. We leave the detailed discussion and analysis of this for future work.

Modeling Signal Delay and Sampling Frequency. We model signal delay because of the multiple network hops for depth readings: from depth altimeter to embedded system to control computer and back to the UAV (as shown in Fig. 3). In simulation, a signal delay of less than 100 ms was required to ensure that the system converges smoothly to changes in target depth of ≈ 1 m, with smaller signal delay yielding only modest improvements.

We model the sampling rate of the depth altimeter to find a baseline requirement, and find that at least 10 Hz is required. During simulation, the system could still converge to a target depth with a slow sampling rate of 1.25 Hz and a long delay of 700 ms, but with significant oscillations.

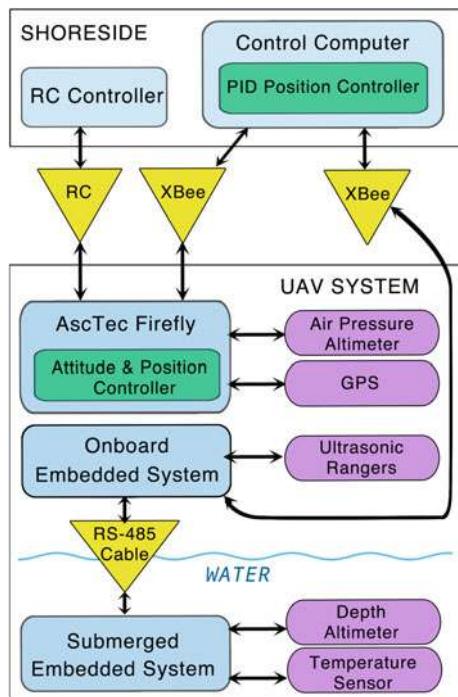
Key simulation results:

- Depth altimeter sampling frequency > 10 Hz.
- Signal delay from the depth altimeter to UAV control input < 100 ms.
- Small transient lateral disturbances do not cause the system to become unstable.
- Limit UAV descent velocity or limit change in target depths to < 1 m.

We now use these simulation results to guide our implementation.

4 Implementation Details

This section describes the system architecture, depth altimeter characterization, and descriptions of the three flight modes under test.

Fig. 3 System architecture

4.1 System Architecture

Figure 3 shows the system architecture used during experiments, consisting of: (1) AscTec Firefly hexrotor UAV; (2) two embedded systems; and, (3) a control computer.

The Firefly includes a GPS and air pressure altimeter as well as an on-board controller for attitude and position. We chose the Firefly for overwater experiments because it can return to shore even after one motor fails. It has a payload capacity of 600 g, flies for 15–20 min per battery, and tolerates winds up to 10 m/s.

The sensor payload installed on the Firefly consists of two embedded systems, the first installed on the vehicle and the second waterproof system attached to the end of the cable that dangles below the UAV. The first embedded system has an NXP-LPC2368 (ARM7TDMI) microprocessor running a control loop at 50 Hz, and has inputs for a variety of water sensors and is used to log and transmit real-time readings over radio. The submerged embedded system contains an ATmega-1284pb microprocessor that reads the depth altimeter at 50 Hz. The embedded system installed on the UAV receives the depth readings and re-transmits them to the control computer. The control computer is an Apple Macbook “Early 2015” running Ubuntu 14.04 with ROS Jade for control and logging.

In this architecture, we estimate the complete signal delay from the depth altimeter to the UAV control input to be 45 ms worst case, within the 100 ms bound identified during simulation and described in Sect. 3.1.

4.2 Depth Altimeter Characterization

For the depth altimeter, we use a Measurement Specialties MS5803-01BA [24] installed on an embedded system, and shown in Fig. 4. By the datasheet, this sensor is water resistant to 100 m with a built-in 24-bit ADC and a 10 ms response time, plus an additional 10 ms to read the onboard thermometer to correct for temperature variation. The 20 ms total means a maximum sampling frequency of 50 Hz, well above the baseline 10 Hz indicated by simulation.

To characterize the depth altimeter's steady-state error, we placed it at a fixed depth in a bucket of 22 °C water. Figure 5 depicts 10 s of readings, and shows steady-state error of ± 3 mm. We characterized and corrected for the depth sensor's bias using the vertical tank described in Sect. 3.1. Because of the MS5803's fast response, accuracy to depths of 10 m, and small steady-state error, we use this sensor for comparison during our field experiments.

Note that changes to ambient air pressure causes pressure changes in water, impacting depth readings. For example, an air pressure difference of 100 mbar across a strong weather front results in 0.6 m pressure difference measured in water depth. In practice, the depth altimeter must be calibrated for the current air pressure every few hours by submerging the sensor to a known depth (we used 5 cm). However, during a 20 min flight, the total air pressure change is likely small, and transient air pressure fluctuations like wind that cause an air altimeter change of ± 2 m cause a

Fig. 4 The waterproof embedded system and sensor payload

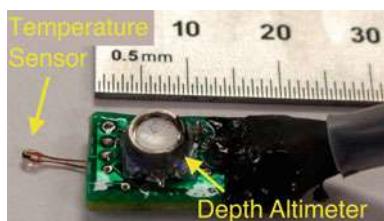
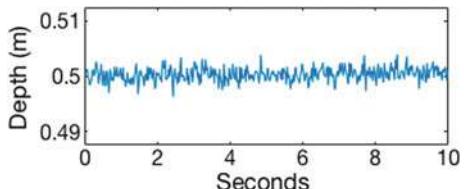


Fig. 5 Depth altimeter steady state error at constant depth showing ± 3 mm error



depth altimeter change of only ± 3 mm. This makes sense since water is ≈ 800 times denser than air.

4.3 Flight Modes: Depth, Air Pressure, Ultrasonic + Air Pressure

We implement three flight modes: depth altimeter, air pressure altimeter, and ultrasonic altimeter. The depth altimeter mode is the new method proposed and implemented in this work, and the air pressure and ultrasonic modes are alternative methods for comparison during experiments. Note that the depth altimeter method controls depth while air pressure and ultrasonic methods control the UAV's altitude as a proxy for sensor depth.

Depth altimeter mode controls only thrust, while the UAV controls roll, pitch, and yaw. On the control computer, the stream of depth readings is passed through a Kalman filter assuming Gaussian noise (characterized in Sect. 4.2), and we assume a linear state transition function during hover. The filtered readings are used in a 50 Hz PID position controller for depth that commands thrust.

Air pressure altimeter mode uses the UAV's onboard controller for everything: roll, pitch, thrust, and yaw. Air pressure mode uses the UAV's air pressure altimeter, GPS, and an onboard IMU together to create a fused pose estimate. This is the basic 'out-of-the-box' controller.

Ultrasonic altimeter mode fuses downward-facing ultrasonic sensors with the air-pressure altimeter to form an altitude estimate. To prevent the ultrasonic sensors from seeing the cable, the readings from each of the two ultrasonic sensors are pre-filtered based on both the variance of recent readings and the proximity of readings to the current altitude estimate, before being passed through a Kalman filter. The details of this pre-filtering are described in previous work [7].

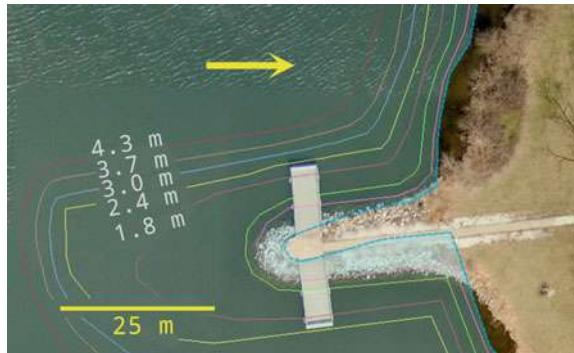
Now that we have an implementation guided by simulation, we discuss experimental validation in the field.

5 Field Experiments

This section describes the setup of field experiments and presents results. To test our approach, we designed three field experiments: (1) a comparison of depth precision for three flight modes with a 3.5 m cable and submerged sensor payload; (2) step responses to changes in target depth for the proposed depth altimeter method; and, (3) an initial exploration of the depth altimeter method using a longer, 8.0 m cable, measuring precision and step response.

All field experiments were conducted during March 2017 at Wildwood Lake near Lincoln, Nebraska, USA, as shown in Fig. 6. In total, we flew 33 missions.

Fig. 6 Wildwood Lake near Lincoln, Nebraska, USA, shown with depth contours. The yellow arrow indicates location of field experiments. Map courtesy of Nebraska Game and Parks



The comparison dataset was collected during three days and the wind average speeds were 3.5 m/s, 2.0 m/s, and 1.3 m/s, respectively with maximum 5 s wind speed of 8.5 m/s, 7.2 m/s, and 7.9 m/s, respectively, indicating occasional strong gusts. We measured wind speed and direction using a weather station recording into a time-synced computer log.

5.1 Comparison of Flight Modes for Maintaining Constant Depth

The purpose of this experiment is to compare three flight modes and quantify how precisely we can maintain the depth of a sensor payload. The three modes are detailed in Sect. 4.3.

We launched the vehicle from a jetty to a sampling location 10 m from shore, indicated in Fig. 6. For these experiments we used a 3.5 m cable affixed below the UAV's center of mass. For each mission, we flew to the sampling location by human pilot, descended until the depth altimeter was at least 1 m deep, then switched to one of the computer controlled flight modes. Once in computer control mode, the system attempts to remain stationary in x and y while recording depth. We flew two modes per flight, each for ≈ 4 min. It can be difficult to compare the results of different outdoor trials, so within a single flight we tested different flight modes successively, and recalibrated the depth altimeter every few hours to adjust for changing air pressure. We swapped the flight mode sequence between trials to ensure that no mode benefited from fresher batteries. In total, we flew 12 min maintaining a specified depth for each mode. We use the depth altimeter reading for comparison, as it is accurate and fast as described in Sect. 4.2, and use this as a basis for evaluating the target depth precision.

Figure 7 show the results of comparing depth precision during the three flight modes. The data are collated from 3–4 flights per mode. We assume a Gaussian distribution of target depth errors and estimate the standard deviation σ with Matlab's `normfit` function. As shown in the figure, the depth altimeter mode is significantly

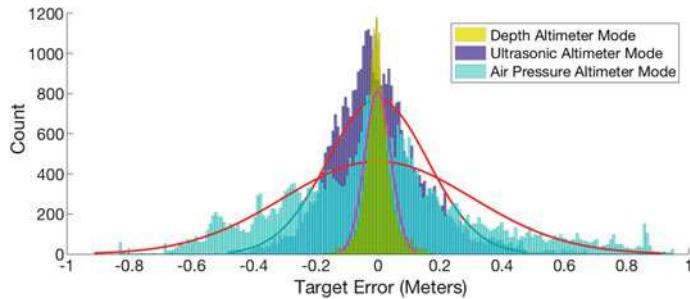


Fig. 7 Distribution of target depth error during three flight modes

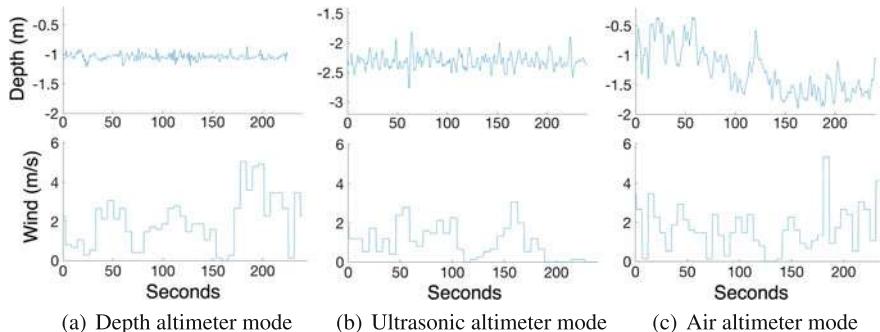


Fig. 8 Comparison of depth precision using three different flight modes with 5-second-average windspeeds. Note that the y-axes of all depth figures span 1.8 m

more precise than either ultrasonic or air pressure modes. Specifically, the statistical dispersion of depth readings in depth altimeter mode keeps 95% of readings within ± 8.4 cm ($\sigma = 4.2$ cm), two standard deviations from the target depth. This is within our goal of being able to obtain readings with a resolution of at least ≈ 25 cm. The ultrasonic-pressure altimeter mode shows $\sigma = 16.1$ cm, or 95% of readings within ± 32.2 cm of the target depth, while the air altimeter has the largest dispersion, with 95% of readings within ± 60.2 cm ($\sigma = 30.1$ cm).

To better show the variation of depth readings in each mode, Fig. 8 shows examples of 4 min of continuous flight for each mode along with the 5-second-average wind speeds. As shown in Fig. 8a, depth altimeter mode stays closest to the target depth with almost no drift and small disturbances, and note that it maintained this target depth during 5 m/s wind gusts. Also note in Fig. 8b how ultrasonic altimeter mode is not as precise and occasionally deviates almost a meter even with calmer winds, but does not drift significantly over time, unlike the air pressure altimeter mode. During the ultrasonic altimeter test, we moved the UAV closer to the water so that the ultrasonic rangefinders could get better readings, resulting in greater depth compared to the other two methods. In Fig. 8c, the air altimeter mode depth readings change by 30–40 cm in a small amount of time, but that within 2 min the depth readings

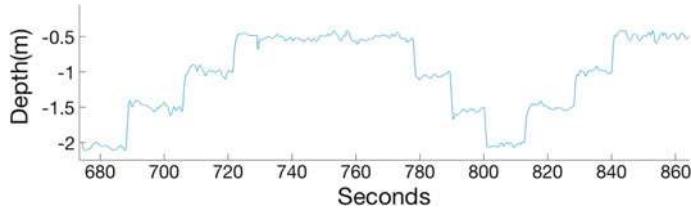


Fig. 9 Step response of depth altimeter mode at 0.5 m increments for a 3.5 m cable

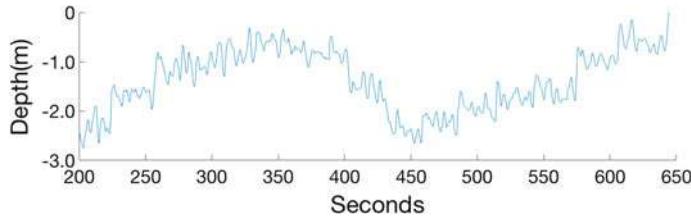


Fig. 10 Step response of air pressure altimeter mode at 0.5 m increments for a 3.5 m cable

can be off by ± 0.75 m. The large drift in air pressure mode is likely caused by larger changes in ambient air pressure, like wind.

5.2 Step Response of Depth Altimeter Mode for a 3.5 m Cable

To determine if the system under depth altimeter control could transition between target depth reference points, we conduct ‘step response’ maneuvers. Figure 9 shows the step responses under depth altimeter control. For each reference depth, we set the system to hold a particular target depth, then adjust the target by 0.5 m. Notice how the reference depths are repeatable both ascending and descending and the depth quickly settles to within ± 5 cm of the target depth. For comparison, Fig. 10 shows the step response for air pressure altimeter mode. We do not show the step response for ultrasonic altimeter mode because it only works well when close to the water. We use 0.5 m changes to avoid the UAV outpacing the submerged sensor toward the target depth, until we implement a descent velocity limit, reserved for future work.

5.3 Longer 8.0 m Cable: Target Error and Step Response

As an additional experiment, we extended the cable length to 8.0 m. The reason we are interested in a longer cable is that in some applications we want to be able to measure water properties across a range of depths, nearly all < 10 m, and we also

Fig. 11 Target depth error comparison between 8.0 and 3.5 m cables both using depth altimeter mode over 4 min of flight

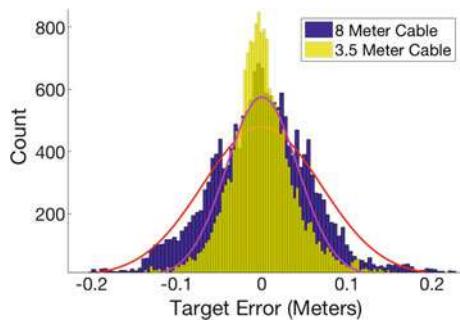
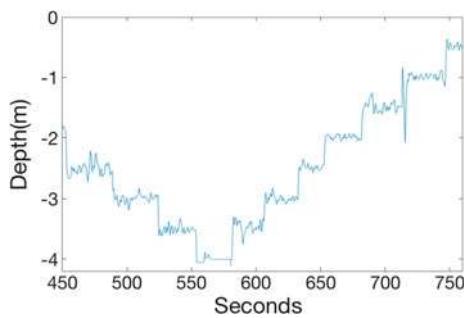


Fig. 12 Step response to target depths at 0.5 m increments for an 8.0 m cable. Note the flat readings near 550 s indicating the water bottom



wanted to see what impact a longer cable would have on the ability to control the depth. Therefore, we field tested the depth altimeter flight mode with the 8.0 m cable. The experimental setup is identical to the depth altimeter mode tests in Sects. 5.1–5.2 other than the longer cable length. We were only able to test the 8.0 m cable down to a depth of ≈ 4 m, due to depth limitations at our field location.

Figure 11 shows the results for holding a target depth, and for comparison the figure shows the distribution of target errors for the 3.5 m cable depth altimeter experiments of Sect. 5.1. As shown in the figure, the 8.0 m cable is less precise than the 3.5 m cable, with 95% of readings within ± 14 cm ($\sigma = 7.0$ cm), still close to the desired ≈ 0.25 m resolution. The dispersion of target errors for the 8.0 m cable still improves on the ultrasonic and air-pressure altimeter modes, even when those modes use the shorter 3.5 m cable.

Figure 12 shows the step response between target depths at 0.5 m increments. Like the system with the 3.5 m cable under depth altimeter control, the configuration with an 8.0 m cable can transition precisely between target depths.

Because the 8.0 m cable allows the sensor payload to remain within ± 14 cm for nearly all readings, this means an 8.0 m cable might be viable for monitoring water properties down to a depth of 7 m, and we intend to explore this in future work.

6 Discussion and Future Work

The precision for the depth altimeter flight mode exceeded our expectations, especially while flying in wind gusts at 80% of the manufacturer’s limit of 10 m/s. We believe that controlling depth with a depth altimeter and the UAVs motors will enable dramatically better vertical resolution with the advantages of quick deployment and redeployment by UAV without additional sensors.

One future challenge includes integrating sensor payloads requested by water scientists. The challenge is that the irregular shapes and configurations of sensors might result in different drag and terminal velocities.

Fusing altitude and depth data might appear to be natural course, but we see little advantage in fusing because other sensors measure the UAV’s altitude, not the depth of the sensor payload. A hybrid controller could switch between flight modes based on violations of invariants inferred during successful flights [25] (like hitting the bed below the water, becoming entangled, or getting encased in muck).

We examined the depth error as a function of lateral disturbance (error in x and y position), like when wind or GPS drift pushes the UAV out of position. However, we saw no relationship between lateral error and depth error. We plan to explore this in flowing water (‘lotic systems’) or while the UAV is translating.

Flying over water with micro UAVs that are not waterproof is inherently risky, and there are a number of ways to increase the reliability of this kind of system: (1) better protection from unsafe descent, including conductivity sensors on the cable near the UAV, and at least one ultrasonic sensor for redundant backup; (2) break-away cable with a buoy; (3) a protective cover for the depth altimeter to prevent fouling; and, (4) buoyancy of the UAV.

7 Conclusions

This work explores a novel method for precisely maintaining the target of a submerged sensor payload passively cabled to a UAV. Our method enables 95% of sensor payload readings within ± 8.2 cm of the target depth, increasing the depth resolution of UAV-based water property datasets. We tested our method in the field moderately windy conditions (4–8 m/s) and the system still quickly and accurately reached and maintained target depths. The non-linear dynamics of a system with a semi-rigid cable are challenging to model as it interacts with wind and water. Still, we used simulation to guide and refine our approach before implementation. We conducted field experiments that validate the approach, resulting in a 75% reduction in standard deviation from a target depth when compared to the previous best method. We also presented initial results for a longer, 8.0 m cable, the longest cabled sensor system yet attempted in UAV-based sub-surface water monitoring, and demonstrated that even at this length the approach allows water monitoring at precise depths from the air.

Acknowledgements Thanks to Dr. Sebastian Elbaum and Dr. Justin Bradley for insightful discussions. Becca Horzewski helped design and build the underwater sensor. For help with field experiments we thank Ajay Shankar, Ashraful Islam, Adam Plowcha, Chandima Fernando, and Nishant Sharma. This work partially supported by NSF NRI-1638099, USDA-NIFA 2013-67021-20947 and USDA-NIFA 2017-67021-25924.

References

1. Higgins, J., Detweiler, C.: The Waterbug sub-surface sampler: design, control and analysis. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 330–337 (2016)
2. Dodds, W.K., Bouska, W.W., Eitzmann, J.L., Pilger, T.J., Pitts, K.L., Riley, A.J., Schloesser, J.T., Thornbrugh, D.J.: Eutrophication of U.S. freshwaters: analysis of potential economic damages. Environ. Sci. Technol. **43**(1), 12–19 (2009)
3. Sanseverino, I., Conduto, D., Pozzoli, P., Dobricic, S., Lettieri, T.: Algal bloom and its economic impact. EUR 27905 EN **660478** (2016)
4. Fischer, H.B., List, J.E., Koh, C.R., Imberger, J., Brooks, N.H.: Mixing in Inland and Coastal Waters. Elsevier (2013)
5. Ore, J.P., Elbaum, S., Burgin, A., Zhao, B., Detweiler, C.: Autonomous aerial water sampling. In: Proceedings of the 9th International Conference on Field and Service Robots (FSR), Brisbane, Australia, vol. 5, pp. 137–151 (2013)
6. Schwarzbach, M., Laiacker, M., Mulero-Pazmany, M., Kondak, K.: Remote water sampling using flying robots. In: 2014 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 72–76 (2014)
7. Ore, J.P., Elbaum, S., Burgin, A., Detweiler, C.: Autonomous aerial water sampling. J. Field Robot. **32**(8), 1095–1113 (2015)
8. Rodrigues, P., Marques, F., Pinto, E., Pombeiro, R., Lourenço, A., Mendonça, R., Santana, P., Barata, J.: An open-source watertight unmanned aerial vehicle for water quality monitoring. In: OCEANS'15 MTS/IEEE Washington, pp. 1–6 (2015)
9. Bae, J.H., Matson, E.T., Min, B.-C.: Towards an autonomous water monitoring system with an unmanned aerial and surface vehicle team. In: 2015 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), pp. 1–2 (2015)
10. Ribeiro, M., Ferreira, A.S., Gonçalves, P., Galante, J., de Sousa, J.B.: Quadcopter platforms for water sampling and sensor deployment. In: OCEANS 2016 MTS/IEEE Monterey, pp. 1–5 (2016)
11. Koparan, C., Koc, A.B.: Unmanned aerial vehicle (UAV) assisted water sampling. In: 2016 American Society of Agricultural and Biological Engineers ASABE Annual International Meeting, p. 1 (2016)
12. DeMario, A., Lopez, P., Plewka, E., Wix, R., Xia, H., Zamora, E., Gessler, D., Yalin, A.P.: Water plume temperature measurements by an unmanned aerial system (UAS). Sensors **17**(2), 306 (2017)
13. Chung, M., Detweiler, C., Hamilton, M., Higgins, J., Ore, J.P., Thompson, S.: Obtaining the thermal structure of lakes from the air. Water **7**(11), 6467–6482 (2015)
14. Eubank, R., Atkins, E., Macy, D.: Autonomous guidance and control of the Flying Fish ocean surveillance platform. In: AIAA Infotech@ Aerospace Conference, pp. 2009–2021 (2009)
15. Bershadsky, D., Haviland, S., Valdez, P.E., Johnson, E.: Design considerations of submersible unmanned flying vehicle for communications and underwater sampling. In: OCEANS 2016 MTS/IEEE Monterey, pp. 1–8 (2016)
16. Dunbabin, M.: Autonomous greenhouse gas sampling using multiple robotic boats. In: Field and Service Robotics, pp. 17–30. Springer (2016)

17. Cruz, N.A., Matos, A.C.: The MARES AUV, a modular autonomous robot for environment sampling. In: OCEANS 2008, pp. 1–6 (2008)
18. Zhang, F., En-Nasr, O., Litchman, E., Tan, X.: Autonomous sampling of water columns using gliding robotic fish: control algorithms and field experiments. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 517–522 (2015)
19. Jain, S., Nuske, S., Chambers, A., Yoder, L., Cover, H., Chamberlain, L., Scherer, S., Singh, S.: Autonomous river exploration. In: Field and Service Robotics, pp. 93–106. Springer (2015)
20. Burri, M., Oleynikova, H., Achtelik, M.W., Siegwart, R.: Real-time visual-inertial mapping, re-localization and planning onboard MAVs in unknown environments. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1872–1878 (2015)
21. Tang, S., Kumar, V.: Mixed integer quadratic program trajectory generation for a quadrotor with a cable-suspended payload. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 2216–2222 (2015)
22. Goodarzi, F., Lee, D., Lee, T., et al.: Geometric stabilization of a quadrotor UAV with a payload connected by flexible cable. In: 2014 American Control Conference (ACC), pp. 4925–4930 (2014)
23. Flying Qualities of Piloted Airplanes, U.S. Military, 11 1980
24. Pressure Sensor MS5803-01BA. <http://www.te.com/usa-en/product-CAT-BLPS0038.html>. Accessed 21 March 2017
25. Jiang, H., Elbaum, S., Detweiler, C.: Inferring and monitoring invariants in robotic systems. Auton. Robots **41**(4), 1027–1046 (2017)

Autonomous and Safe Inspection of an Industrial Warehouse by a Multi-rotor MAV

Alexandre Eudes, Julien Marzat, Martial Sanfourche, Julien Moras and Sylvain Bertrand

Abstract This paper reports field tests of autonomous inspection in an industrial indoor facility by a Micro-Air Vehicle (MAV) with no prior knowledge on the environment. Localization, mapping and safe navigation is achieved using only the embedded sensors (stereo-vision, IMU, laser altimeter) and with the entire perception and control loop running on-board of the MAV. An overview of the algorithmic architecture and design choices is provided and the focus is put on mission and safety capabilities that have been demonstrated via several flight tests defined in association with SNCF (French Railways) in one of their train storage warehouse.

1 Introduction

Building inspection and surveillance can benefit from the use of Micro Air Vehicles (MAVs) to provide additional or complementary data (in terms of nature or point of views) that can be used for diagnosis and maintenance. If outdoor inspection of buildings involves taking measurements at high altitudes, for which the use of MAVs can be helpful, this is also the case for indoor inspection of industrial warehouses. In this case, MAVs can be used to perform autonomous and repeatable inspections inside the warehouse to assess the state of the building (walls, pillars, high altitude structures, ceiling, etc.) or provide information on objects of interest stored or located inside (number and location of goods, for example).

A. Eudes · J. Marzat (✉) · M. Sanfourche · J. Moras · S. Bertrand
ONERA—The French Aerospace Lab, 91123 Palaiseau, France
e-mail: julien.marzat@onera.fr

A. Eudes
e-mail: alexandre.eudes@onera.fr

M. Sanfourche
e-mail: martial.sanfourche@onera.fr

J. Moras
e-mail: julien.moras@onera.fr

S. Bertrand
e-mail: sylvain.bertrand@onera.fr

This paper considers autonomous inspection by a multi-rotor MAV inside a warehouse owned by SNCF (national French railway company) where maintenance of trains and storage of spare parts are performed. Several major issues arise regarding such a task. First the MAV should be able to localize itself in an indoor environment (no GPS) while taking into account possible changes in this environment (stored items moved between two flights). The flight and inspection should be fully automatic, from take-off to landing, to lessen the dependence on a human telepilot and increase repeatability. In addition, robustness with respect to signal loss must be ensured as the vehicles evolve over long distances in a potentially electro-magnetically perturbed environment (motor coils, etc.). Finally, safety must be guaranteed with respect to possible workers entering the area of inspection.

Related work

In [13], Shen et al. present a MAV system able to navigate indoor, inside a multi-floor building. They use a setup composed of one lidar, one camera and an IMU to demonstrate self-localization, mapping, path planning and autonomous control functionalities. In [3], Fraundorfer et al. propose to use a MAV for autonomous exploration and mapping in GPS-denied environment. The MAV is equipped with a stereo camera, an optical flow camera looking downward and an ultrasonic range sensor. The MAV achieves visual odometry using the stereo camera which is fused with the attitude estimated from IMU, the optical flow of the third camera and the ultrasonic range measurement. The system computes a 3D occupancy grid that is used to make a frontier-base exploration of the area. In [11] Omari et al. propose to use a remotely operated MAV for industrial inspection applications. Their platform is equipped with a stereo camera sensor that is used to compute visual odometry and state estimation tightly coupled with an IMU. The MAV also computes an Octomap representation of the environment which helps the pilot by preventing collisions. In [1], Beul et al. use an hexarotor MAV controlled by a 3DR Pixhawk Autopilot. The perception stack uses an omni-directional and heterogeneous sensor system composed of three stereo cameras and two Hokuyo lidar sensors mounted on a moving plate. They achieve localization by combining visual odometry and lidar, as well as environment modeling in order to avoid obstacles.

In [2], Fang et al. address the problem of robust autonomous navigation in difficult environments like inside a ship. Their work propose a robust pose estimation system based on the fusion of RBG-D sensor, downward optical flow camera, downward lidar punctual sensor and an IMU using a Monte Carlo approach. This is coupled with a path planning system based on Receding Horizon Control to navigate between way-points.

In line with these research efforts, the proposed solution is based on the development of a fully embedded software architecture exploiting a stereo-vision system (in association with an IMU and a laser telemeter) which performs on-board all the functionalities required for safety and mission fulfillment: localization and mapping, planning, control, obstacle detection and avoidance, safety supervision. This paper presents this on-board architecture and its validation through field flight experiments in a large-scale indoor facility with operational interest for industrial End-Users.

The paper is organized as follows. In the next Section, a description of the industrial warehouse, of the MAV platform and of the scenarios of inspection are proposed. The overall on-board architecture and the associated algorithms are described in Sect. 3. Section 4 addresses the supervision of the flight and the safety functionalities. Finally, before concluding remarks, results of flight validation experiments are proposed in Sect. 5 for the considered scenarios of inspection.

2 Environment, MAV Platform and Scenarios Descriptions

2.1 Industrial Warehouse

The industrial warehouse in which inspections have to be performed belongs to the French national railway company (SNCF) and is located in Sotteville-lès-Rouen. It is used as a maintenance workshop for trains and as a storage place for spare parts (see Fig. 1). The part of the warehouse dedicated to storage has been used for the flight experiments presented in this paper. The corresponding flight volume is defined by a ground area of 25 m by 50 m with an altitude of 10 m under ceiling.

2.2 MAV Platform

The flight experiments described here are conducted on a Pelican quadrotor base from Asctec (Fig. 2). The proprietary Flight Control Unit (FCU) includes IMU, 3D-magnetometer and processors dedicated to low-level control (attitude stabilization). The FCU is linked to an embedded computer containing an Intel i7 quadcore (2.1 GHz) which handles the complete perception and control chain presented in



Fig. 1 Industrial environment: SNCF infrastructure warehouse in Sotteville, France

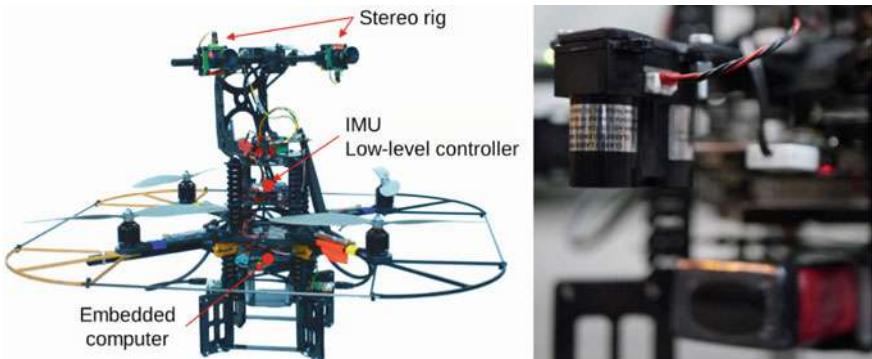


Fig. 2 ONERA MAV (based on an Asctec platform) with stereo-vision sensors, lidar altimeter (right) and embedded processing capabilities

Fig. 3. All processing is achieved on-board so the MAV is fully autonomous and robust to communication loss during the execution of the mission. In addition to the IMU, the sensor setup includes a 22 cm-baseline stereo-rig composed of two USB2 cameras, electronically synchronized, offering a 100° field of view and a laser telemeter pointing downward for altitude measurement (for take-off and landing).

2.3 *Inspection Scenarios*

Typical scenarios of inspection in the warehouse consist in recording pictures or measurements provided by additional on-board sensors (e.g. FLIR camera) of different objects of interest: walls, high altitude structures, spare parts stored in the warehouse. This requires to automatically take-off, fly over long distances to reach areas of inspection, perform the inspection and fly back for automatic landing. During the flight, obstacle avoidance can be performed in case of trajectories conflicting with detected static or dynamic obstacles. The trajectories of inspection are defined by way-points or trajectory coordinates given either in the global frame associated to the warehouse or in the local frame associated to a specific point whose global location is known or detected during flight. In addition, sensor characteristics (range, field of view, orientation) must be taken into account for the inspection as the mission performance is directly impacted by the MAV trajectory (measurement overlap between two parts of the trajectory, relative distance and/or line-of-sight constraints to the object to be inspected, etc.).

Two scenarios representative of realistic inspection tasks have been defined. The first one consists in inspecting objects of interest defined by global coordinates in the warehouse. More precisely, we chose two elements to be inspected, a metallic structure located at an altitude of 7 m that the MAV must follow and inspect and a load located over a platform around which the MAV must turn in order to take

pictures. The second scenario consists in performing the inspection of a wall section whose location is not a priori precisely known, but which can be detected by some specific characteristic (e.g. visual tag). To also validate safety requirements, avoidance situations with respect to static and mobile obstacles are considered in a third scenario.

3 On-Board Navigation System

The embedded navigation system uses ROS to link together the numerous software components of the global architecture from Fig. 3. Besides hardware related components (interface with sensors or other computers), we distinguish 3 main functional blocks described in the following subsections: multi-sensor state estimation, environment modeling and control.

3.1 Multi-sensor State Estimation

The low-level controller provides a reliable estimate of the orientation. Therefore, the vehicle state vector contains only the position and linear velocity in the reference frame aligned with gravity and centered on the IMU-frame origin at the beginning of the mission. The state is estimated by a linear Kalman filter as in [8]. Given the initial pose, the state is predicted by integration of accelerometer and attitude measurements at 100 Hz. The correction is performed at video-rate (20 Hz) thanks to the pose computed from stereo-images by eVO [12], a keyframe-based visual odometry algorithm. eVO builds a map of isolated landmarks automatically initialized from images (no prior map) as in Visual-SLAM. Compared to state-of-the-art SLAM and other odometry algorithms, eVO is oriented towards low computational cost and includes some simplifications concerning the map updates: the landmarks are pruned when they leave the sensor field of view and the landmarks localization is not refined in a multi-view optimization scheme. Nevertheless, the drift on the localization is of the class 1% of the traveled distance. A detection of outliers is performed on the innovation of the Kalman filter which compares the state predicted using only the IMU with the position computed by visual odometry. If this difference is larger than the expected MAV velocity, an alarm is raised and the emergency mode activated (see Sect. 4). A second Kalman filter estimates independently the ground height and vertical velocity from the laser telemeter measurements and IMU attitude. It is only used during automatic take-off and landing or emergency stabilization.

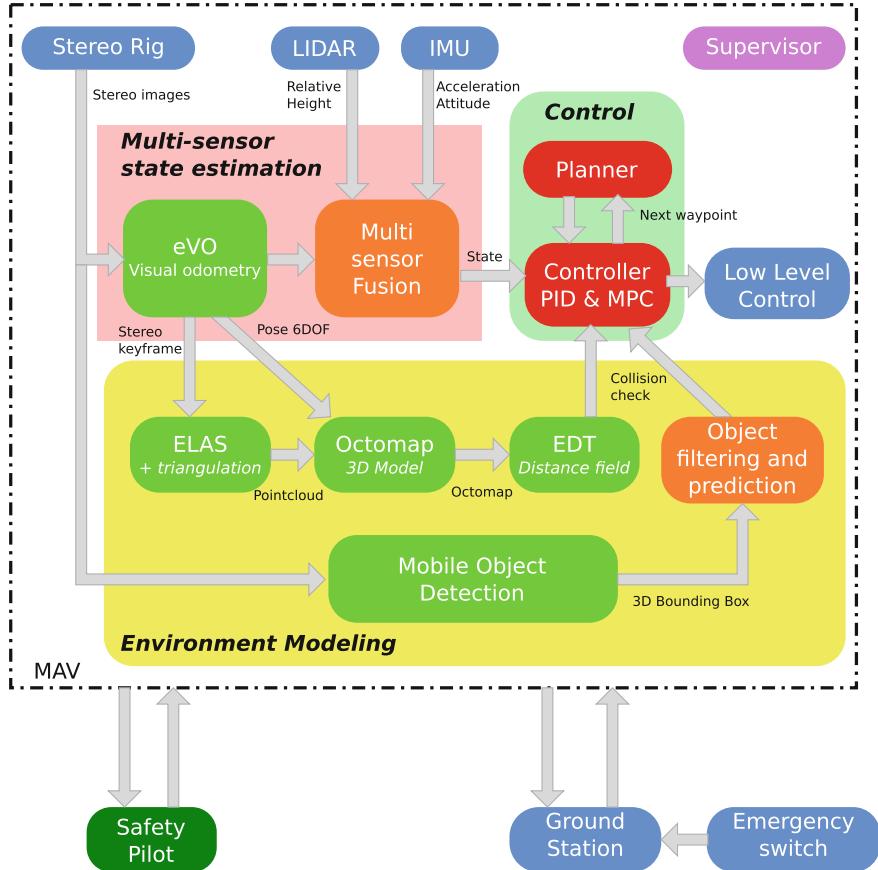


Fig. 3 Perception-control loop for autonomous navigation with obstacle avoidance

3.2 Environment Modeling

In order to be able to avoid collision in such a cluttered environment, mapping is performed on board. To handle both static and mobile objects, we consider a dual model representation. An occupancy grid approach is used to model the static environment while a feature-based (bounding box) approach is used for mobile object detection.

The static mapping module is composed of 3 subtasks as shown in Fig. 3. First, a stereo matching algorithm is used to compute a dense disparity map from a rectified pair of stereo images. Here, we used the ELAS (Efficient Large-scale Stereo Matching) algorithm [4]. This disparity map is used to triangulate a dense point-cloud that is integrated in a well known Octree-based occupancy grid representation named Octomap [6]. This approach models the space with a set of voxels organized into an

Octree structure where each voxel can be either occupied or free. A Bayesian estimation is performed, so each voxel stores its probability to be occupied and the cell then a MAP decision rule is taken when needed. Note that this probabilistic framework ensures that mobile obstacles are not included in this map, since several successive observations of the same point are mandatory. Finally, with the resolution of the map used (0.1 m), an Octomap representation for checking collisions over each potential trajectory would be inefficient, because of the large number of voxels to be tested. A solution to this problem is to additionally compute a distance-field to the closest obstacle during the map update. For this purpose, we apply an Euclidean Distance Transform (EDT) [7] to the Octomap. This mapping updating process is performed at each keyframe.

The detection of mobile objects is based on the analysis of sparse optical flow computed on Harris corners, which are located in 3D by stereo. We assume here that mobile objects are clusters of features which violate two-views and three-views geometrical constraints between two successive stereo-images and spatially consistent with the dimensions of a pedestrian. The first step consists in classifying into moving and static features. The classification is done in a two-tests cascade, firstly by the OpenCV robust RANSAC-based fundamental matrix estimator then, for the inliers, by the robust pose estimator implemented in eVO [12]. In a second step, the candidate moving image features are clustered. A Delaunay triangulation gives access to the rough 3D structure localized in inertial frame thanks to the pose estimated by the multi-sensor state estimator (cf. Sect. 3.1). Triangles are pruned according to their size and orientation (only the most vertical ones are retained). The remaining connected triangles form candidate objects. We finally compute 3D bounding cylinders and select those whose width is inside a specified range. The mobile object raw detection is fed into a Kalman filter tracker in order to remove false alarms and to make prediction of the obstacle position in the next steps. The state vector is composed of the object position, its height and radius. A constant velocity model is used to predict the future positions of the object for avoidance purpose. The entire processing takes less than 100 milliseconds on the MAV embedded CPU.

3.3 Safe Way-Point Navigation and Trajectory Tracking

The control algorithm is designed to track a reference trajectory or to reach a way-point defined by the user (in the way-point case, a straight-line trajectory at constant velocity is generated). Since the environment can be highly cluttered and no prior assumption is made, an avoidance module for static or mobile obstacles has been defined. It relies on a model predictive control (MPC) algorithm which exploits the perception information from the previous subsection. A MPC scheme relies on a prediction of the dynamical model on a time horizon to take into account future MAV behavior and interaction with the environment. The design of the controller

relies on a simplified linear acceleration model, for which an analytical MPC linear quadratic solution has been obtained for trajectory tracking in the nominal case.

To guarantee the safety of the vehicle and the operators, the MPC algorithm is adapted to avoid any obstacle that can be found on the reference trajectory. The distance map information on the static environment and the future mobile object positions are combined to obtain a single distance to the closest obstacle at each position of the MAV predicted trajectory. If there is a risk of collision on the nominal predicted trajectory, an additional control input is then selected in a predefined discretized set so as to optimize a cost function on the prediction horizon which is a weighted sum between tracking the reference trajectory and respecting a parameterized safety distance from any obstacle [9]. The resulting acceleration control input, which is equal to the sum of the nominal and avoidance inputs, is then translated into thrust, roll angle, pitch angle and yaw rate and forwarded to the low-level Asctec controller.

4 Flight Management and Safety Functions

We have implemented a flight manager which supervises the execution of the automatic mission. It includes a mission scheduling editor, a mission scheduler associated to a state machine, the supervision of critical functions and an emergency manager. All of these features contribute to make MAV operation easier and more robust to sensors or software failures, which is of tremendous importance when such automatic functionalities are deployed on the field (where no motion capture system is available, unlike in laboratory).

4.1 Mission Planning and Execution

A mission scheduler allows the operator to define the content of the mission and to configure and execute each task during the mission. The plan is built from a combination of simple tasks such as: way-point goto, viewpoint constraints, waiting steps, user confirmation and more high level tasks like: take-off, land, start wall inspection. A 3D representation of the plan allows an easy mission planning, verification and visualization of the execution (Fig. 7).

The autonomous mission execution is managed by the state machine depicted in Fig. 4. The automatic take-off is triggered by the safety pilot who starts the engine and pushes the thrust stick up to a predefined level, settled near the stability thrust level for easier manual recovery action. It should be noted that after these basic human tasks all the missions are carried out in full autonomy. After the take-off, an on-line calibration permits to adjust some controller parameters. The mission

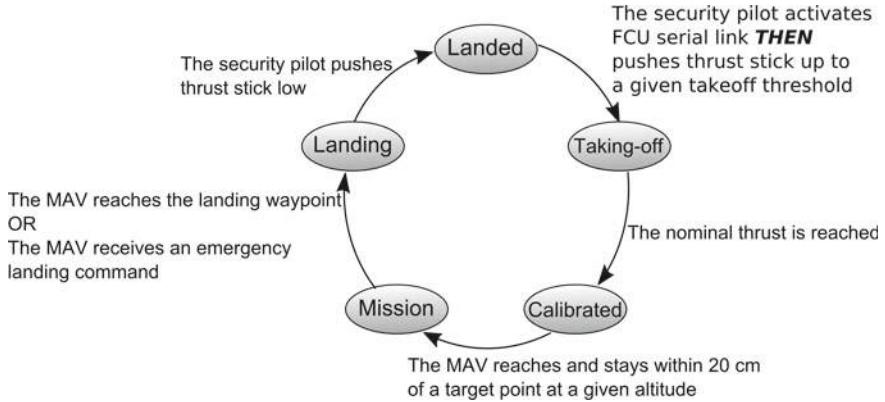


Fig. 4 State machine for flight management

starts and the predefined plan is executed after the MAV reaches and keeps hovering around a target point at a given altitude above the starting point. The landing mode is activated either when the MAV reaches the landing way-point or if the ground station operator triggers an emergency landing. In the nominal case, the landing is done autonomously thanks to relative altitude measurements from laser telemeter. In the emergency case, depending of the state of the MAV, the emergency landing can be fully autonomous or semi-autonomous.

4.2 Supervision and Emergency Manager

In order to handle any issues occurring during the mission, an emergency stack is running on board. This stack is composed of a supervisor monitoring different parameters (module status, sensor activity, etc.). An emergency event is triggered when some parameters are abnormal: loss of a software node, inconsistency between vision and IMU, no safe trajectory found by the MPC module. An emergency event can also be triggered when the ground station operator activates the emergency switch or when the safety pilot pushes any stick outside of a dead zone. Figure 5 presents the different states of the emergency stack. The event sets the MAV in *Emergency Stop* mode. In this mode the MAV suspends its mission and holds its position using all functional sensors and modules. In the case where all critical functions are faulty, the MAV sends a warning signal to the safety pilot and falls back in *Manual* control mode. When the MAV is in *Emergency Stop*, the ground station operator can evaluate the situation in order to check what triggered the emergency event. Finally, he can decide to go out of emergency mode and continue the mission, to activate an *emergency landing* or to ask the safety pilot to recover the MAV in *manual* mode.

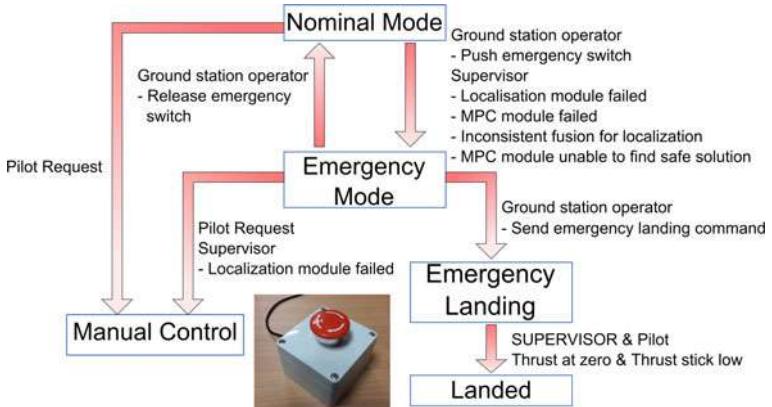


Fig. 5 Emergency management

5 Flight Experiments

Scenario 1

In this first scenario, we demonstrated inspection tasks planned before the mission. This type of mission definition is especially useful for periodic inspection. Different types of trajectory are provided to allow an easy and efficient planning (example in Fig. 6). Figure 7 shows the inspection plan of a bridge and two pillars which is very

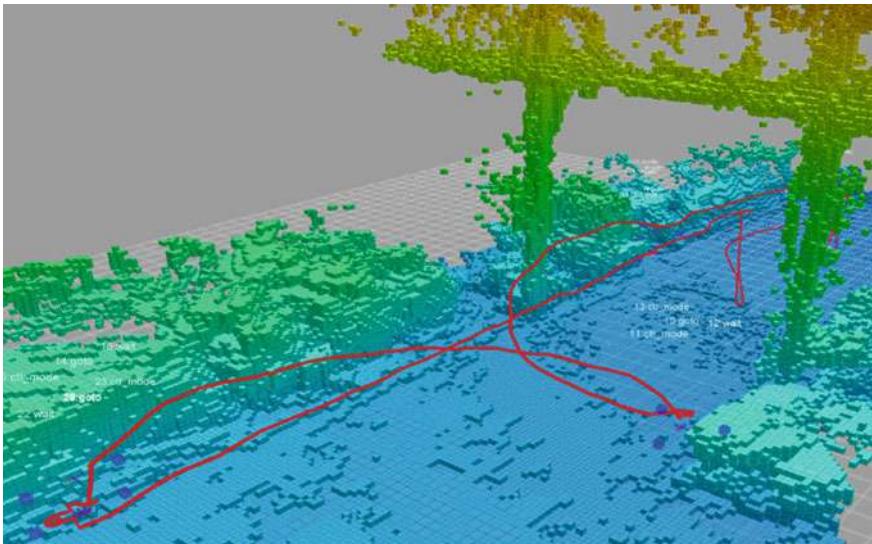


Fig. 6 3D model autonomously built on-board by the MAV while following a long-distance reference trajectory (in red) composed of portions of ellipses and lines

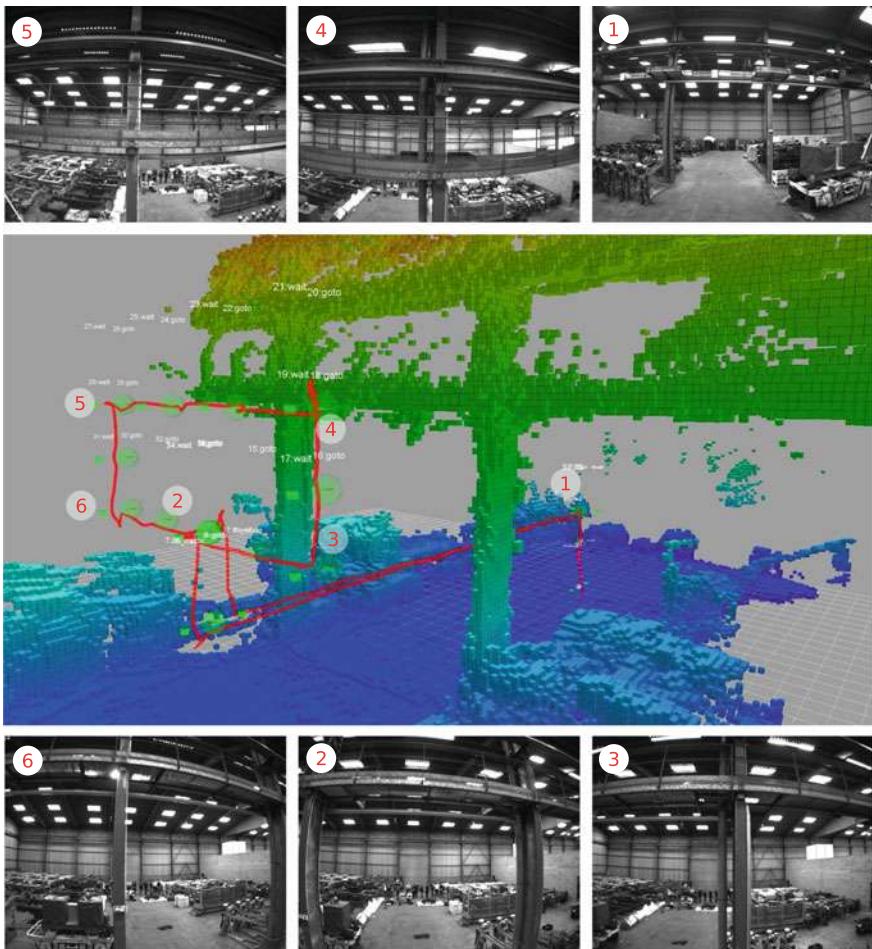
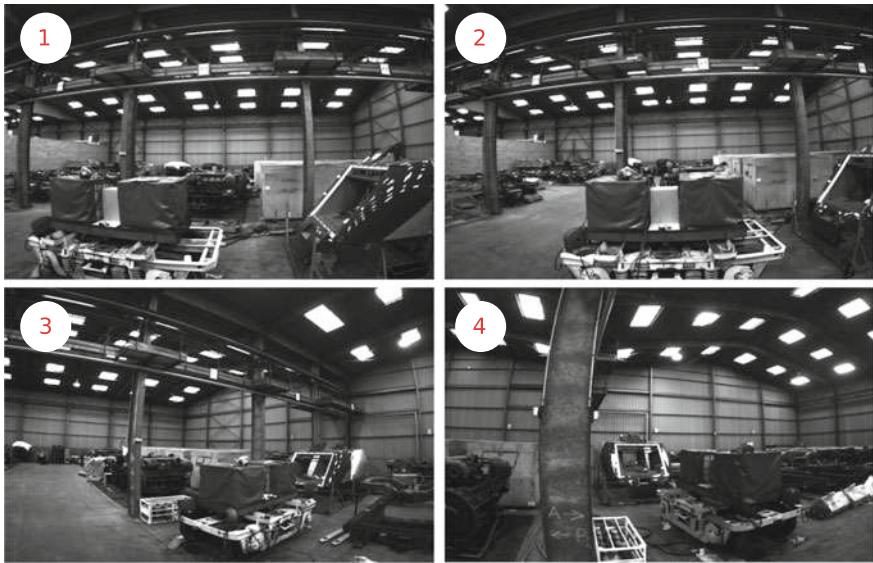


Fig. 7 Trajectory and 3D model built during the inspection scenario. The mission scheduling appears as white text indications over the map

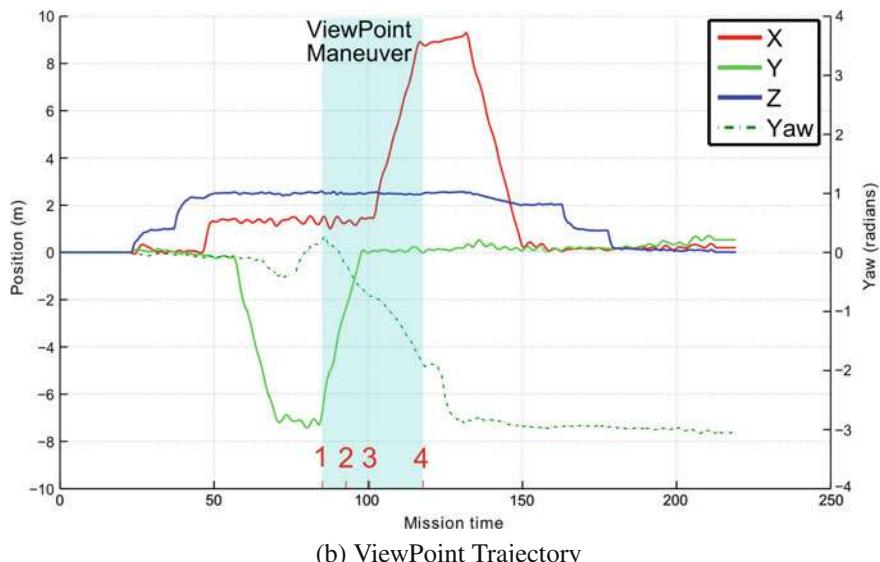
difficult to execute by a human pilot due to height and visibility conditions. We also added the possibility to define a viewpoint function. When this function is active, the yaw is controlled to ensure that a target point is always kept in the field of view. This allows the operator to define trajectories that automatically observe an object from multiple points of view (Fig. 8).

Scenario 2

An automatic wall inspection task has been developed. This task consists in scanning a wall and ensuring its complete coverage by a dedicated sensor. The wall scanning zone is defined by an AprilTag [10] located at the center, predefined dimensions (height and width) and a desired scanning distance. The execution of the task is



(a) ViewPoint Snapshots



(b) ViewPoint Trajectory

Fig. 8 Action of the viewpoint constraint on a L-shape trajectory. The targeted point is located behind the trolley

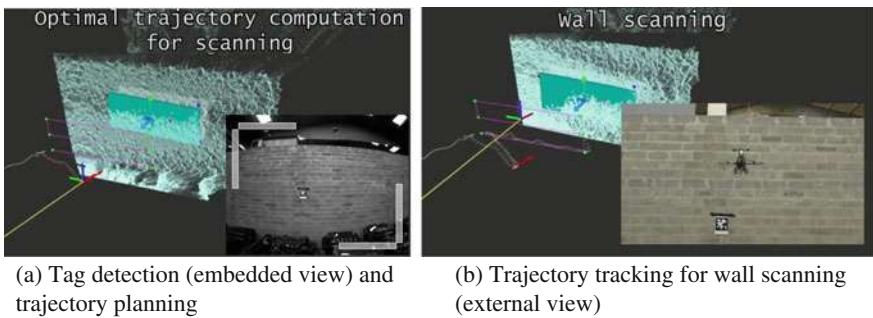


Fig. 9 Automatic wall inspection modus operandi

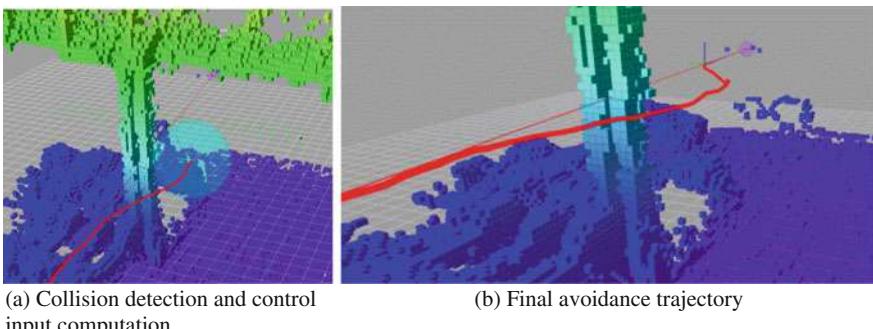


Fig. 10 MPC static obstacle avoidance

carried out as follows. When the tag is detected, the MAV is guided to a position in front of the wall to be able to correctly estimate the AprilTag orientation. The orientation of the wall plane is then refined robustly from the point-cloud computed by ELAS [5]. Assuming this model, the desired scanning distance and the field of view of the payload sensor, a trajectory that covers the wall with the given field of view is planned as a succession of linear segments that are then tracked by MPC (Fig. 9).

Scenario 3

To tackle inaccuracies or mistakes during preflight mission planning, changes in the environment or presence of mobile obstacles, the internal MPC algorithm ensures collision avoidance during the plan execution. Figure 10 shows an example where the original plan (in thin red) passes through a pole. During the execution, the safety distance constraints were not satisfied and the MPC planned reactively an avoidance trajectory (final trajectory in bold red). Figure 11 illustrates the successful avoidance of a mobile obstacle with the same algorithm. These safety situations have been successfully repeated several times.

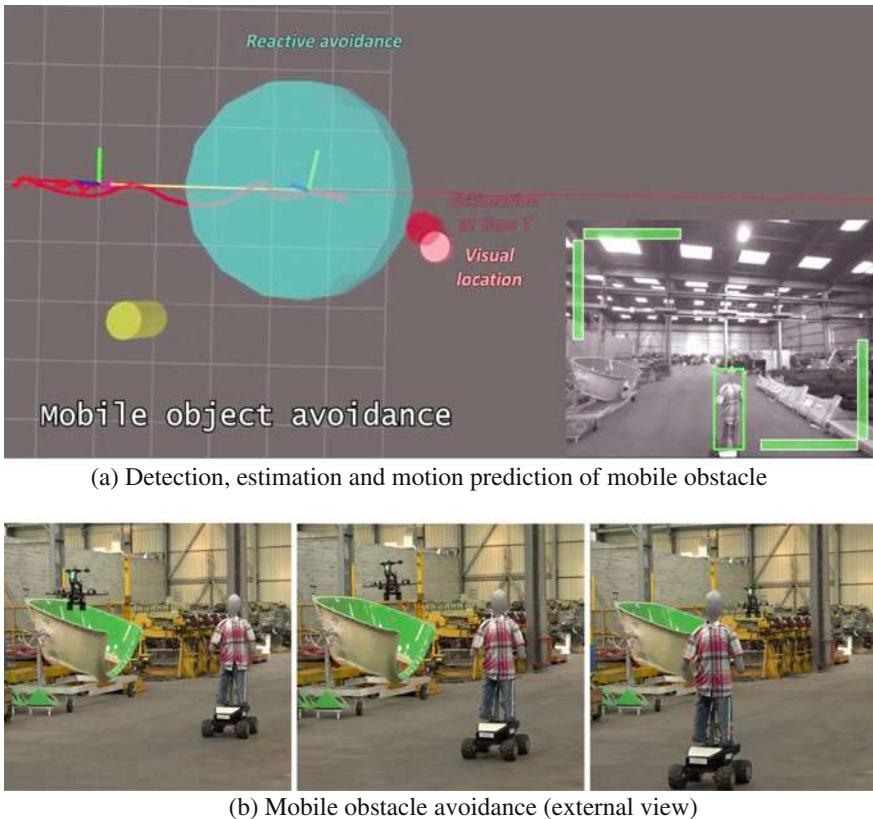


Fig. 11 Detection, localization and avoidance of mobile obstacle

6 Conclusion

In this work, we demonstrated the capability for a MAV to navigate autonomously in an unknown GPS-denied environment in order to achieve different missions in an industrial context. The system presented relies on advanced embedded functionalities such as visual odometry, multi-sensor fusion, environment modeling and path planning and control. The complexity and diversity of the different scenarios tested on the field led us to add a flexible mission manager and several safety functionalities in order to increase the reliability of the system. Field experiments in an industrial facility have shown the soundness of the proposed approach. This is an important step towards the operational deployment of such autonomous MAV solutions for inspection, which will make it possible to relax the constraints on human operators.

Acknowledgements The authors thank SNCF for their support to this work, partially funded by the research partnership between ONERA and SNCF.

References

1. Beul, M., Krombach, N., Zhong, Y., Droseschel, D., Nieuwenhuisen, M., Behnke, S.: A high-performance MAV for autonomous navigation in complex 3D environments. In: 2015 International Conference on Unmanned Aircraft Systems (ICUAS) (2015)
2. Fang, Z., Yang, S., Jain, S., Dubey, G., Maeta, S.M., Roth, S., Scherer, S., Zhang, Y., Nuske, S.T.: Robust autonomous flight in constrained and visually degraded environments. In: Field and Service Robotics (2015)
3. Fraundorfer, F., Heng, L., Honegger, D., Lee, G.H., Meier, L., Tanskanen, P., Pollefeys, M.: Vision-based autonomous mapping and exploration using a quadrotor MAV. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, Portugal, pp. 4557–4564 (2012)
4. Geiger, A., Roser, M., Urtasun, R.: Efficient large-scale stereo matching. In: Proceedings of the 10th Asian Conference on Computer Vision (ACCV), Queenstown, New Zealand, pp. 25–38 (2010)
5. Geiger, A., Roser, M., Urtasun, R.: Efficient large-scale stereo matching. In: Asian Conference on Computer Vision, pp. 25–38. Springer (2010)
6. Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W.: OctoMap: an efficient probabilistic 3D mapping framework based on octrees. Auton. Robots **34**(3), 189–206 (2013)
7. Lau, B., Sprunk, C., Burgard, W.: Efficient grid-based spatial representations for robot navigation in dynamic environments. Robot. Auton. Syst. **61**(10), 1116–1130 (2013)
8. Lynen, S., Achtelik, M.W., Weiss, S., Chli, M., Siegwart, R.: A robust and modular multi-sensor fusion approach applied to MAV navigation. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, pp. 3923–3929 (2013)
9. Marzat, J., Bertrand, S., Eudes, A., Sanfourche, M., Moras, J.: Reactive MPC for autonomous MAV navigation in indoor cluttered environments: flight experiments. In: Proceedings of the 20th IFAC World Congress, Toulouse, France (2017)
10. Olson, E.: AprilTag: a robust and flexible visual fiducial system. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 3400–3407 (2011)
11. Omari, S., Gohl, P., Burri, M., Achtelik, M., Siegwart, R.: Visual industrial inspection using aerial robots. In: 3rd International Conference on Applied Robotics for the Power Industry, Foz do Iguaçu, Brazil, pp. 1–5 (2014)
12. Sanfourche, M., Vittori, V., Le Besnerais, G.: eVO: a realtime embedded stereo odometry for MAV applications. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan, 3–8 November, pp. 2107–2114 (2013)
13. Shen, S., Michael, N., Kumar, V.: Autonomous multi-floor indoor navigation with a computationally constrained MAV. In: 2011 IEEE International Conference on Robotics and Automation (2011)

Part IV

Machine Learning

Online Multi-modal Learning and Adaptive Informative Trajectory Planning for Autonomous Exploration

Akash Arora, P. Michael Furlong, Robert Fitch, Terry Fong,
Salah Sukkarieh and Richard Elphic

Abstract In robotic information gathering missions, scientists are typically interested in understanding variables which require proxy measurements from specialized sensor suites to estimate. However, energy and time constraints limit how often these sensors can be used in a mission. Robots are also equipped with cheaper to use navigation sensors such as cameras. In this paper, we explore a challenging planning problem in which a robot is required to learn about a scientific variable of interest in an initially unknown environment by planning informative paths and deciding when and where to use its sensors. To tackle this we present two innovations: a Bayesian generative model framework to automatically learn correlations between expensive science sensors and cheaper to use navigation sensors online, and a sampling based approach to plan for multiple sensors while handling long horizons and budget constraints. Our approach does not grow in complexity with data and is anytime making it highly applicable to field robotics. We tested our approach extensively in simulation and validated it with real data collected during the 2014 Mojave Volatiles Prospector Mission. Our planning algorithm performs statistically significantly better than myopic approaches and at least as well as a coverage-based

A. Arora (✉) · S. Sukkarieh
Australian Centre for Field Robotics, The University of Sydney, Sydney,
NSW, Australia
e-mail: a.arora@acfr.usyd.edu.au

S. Sukkarieh
e-mail: salah@acfr.usyd.edu.au

P. M. Furlong · T. Fong · R. Elphic
NASA Ames Research Centre, Moffet Field, CA, USA
e-mail: padraig.m.furlong@nasa.gov

T. Fong
e-mail: terry.fong@nasa.gov

R. Elphic
e-mail: richard.c.elphic@nasa.gov

R. Fitch
Centre for Autonomous Systems, University of Technology Sydney, Sydney,
NSW, Australia
e-mail: robert.fitch@uts.edu.au

algorithm in an initially unknown environment while having added advantages of being able to exploit prior knowledge and handle other intricacies of the real world without further algorithmic modifications.

1 Introduction

In robotic information gathering missions, scientists are often interested in variables or phenomenon which cannot directly be measured but must be observed through correlated proxy measurements. Examples include mapping water abundance in remote environments by measuring neutron flux [1], inferring the health of aquatic life by monitoring chemical concentrations [8], and searching for evidence of life on Mars through biomarkers [16]. These proxy measurements often require specialized ‘science’ sensor suites such as spectrometers, subsurface drills and sample processing equipment. These are typically either energetically expensive to use, require the robot to remain stationary or have finite capacity limiting how often they can be used given energy constraints and short life spans of many robotic missions.

Robots are also equipped with sensors that are inexpensive in time and energy, such as navigation sensors like cameras or LIDAR. Learning relationships between underlying scientific phenomena of interest and the different inexpensive sensors on-board will allow scientists to better understand phenomena without incurring the prohibitive cost of exhaustively sampling large environments with specialized sensors. Given the locations to be explored are often remote and mostly unknown, this relationship should be learned or updated *in situ*. Robots that can predict latent science variables at a reduced cost will be able to plan paths and sensor usage more effectively which increases science return, mission productivity and allows the robot to operate at higher levels of autonomy.

In this paper we formulate a sensor planning problem in which a robot equipped with multiple sensors has to learn about a latent scientific variable. The robot must plan paths on a graph representation of the environment and decide when and where to use each sensor, constrained by a sensing budget and a goal position. Sensor correlations are modeled by a Bayesian network (BN) generative model, the parameters of which are learned online as observations are made. Reasoning about the network to plan informative sensing sequences is, however, a challenging optimization problem. We calculate approximate solutions by applying Monte Carlo Tree Search (MCTS) techniques [5]. The combination of BNs and MCTS allows the robot to learn and update sensor correlations recursively in a manner which is constant in the number of samples collected and plan informative sensing sequences in an anytime manner. These two properties make our approach highly applicable for online use in robots with limited computational capabilities.

We apply our general approach to a scenario modeled on the Mojave Volatiles Prospector (MVP) project, conducted by NASA Ames Research Center in the Mojave Desert in 2014 [11]. The purpose of the MVP project was to test high tempo remote operations while attempting to estimate abundance of subsurface water. KReX2,



Fig. 1 KReX2 in the Mojave Desert. KReX carried a neutron spectrometer system (NSS) and a near infrared visible light reflectance spectrometer (NIRVIS). The robot autonomously drove and localized itself at the command of remote scientists. The data that it collected was georegistered and presented to the backroom of a team of 30 scientists who adapted their plans in response to data updates

the robot used in MVP and pictured in Fig. 1, was equipped with several sensors including a downwards facing camera and a Neutron Spectrometer (NSS) which produces measurements that can be correlated with the abundance of subsurface water. The NSS has a small field of view and measurement requires the robot to drive slowly to avoid spatial blurring of readings. NSS is an inexpensive sensor to use, but we use it as a stand-in for more involved subsurface sampling operations.

At the end of the MVP project, the sensor data was analyzed and it was determined that there was a relationship between the visual properties of terrain and the corresponding NSS readings [9]. If this relationship was learned automatically during the mission, scientists could have made more informed decisions regarding where to direct the robot and deploy sensors to maximize understanding of subsurface water distribution. The MVP project was a precursor to the planned Resource Prospector project which aims to deploy a robot with a similar sensor suite on the moon and map the abundance of surface volatiles [1, 11]. Learning sensor correlations online will make the science return of the RP mission much greater, a significant boon given the project is limited to one lunar day of operations.

We illustrate our key ideas both in simulation and with real data acquired from the MVP project. There the robot deduces the water abundance in an environment by autonomously planning paths, sensor placements and simultaneously learning the

relationship between visual properties of terrain and NSS readings. We demonstrate that our approach is statistically significantly better than myopic approaches and comparable to the non-adaptive coverage based planners in initially unknown environments, a result consistent with [13]. Our approach has the added benefit of being generalizable to an arbitrary number of sensors and able to exploit prior knowledge when it's available without further algorithmic modifications.

2 Related Work

Robotic exploration and sensor planning to gain information about the world is an informative path planning problem. Greedy approaches are effective and offer performance guarantees when the problem is submodular [15]. Unfortunately, this property often broken with path dependent rewards often present in field environments. Branch and bound techniques which prune suboptimal branches early in the tree search have shown promise [3, 12] but efficiently calculating tight bounds in problems with unknown environments and multiple sensors becomes non-trivial. There are also various heuristic approaches but they either do not generalize to unknown environments or cannot plan for multiple sensors without significant algorithmic modifications.

In field applications of information gathering, several approaches have been proposed. Thompson and Wettergreen used a greedy algorithm to design maximally informative trajectories for constructing spatial maps of multi-spectral data [18]. Wettergreen et al. extended this in [19] to design trajectories that explore regions of orbital maps that cannot be explained with previous observations—actively solving the spectral unmixing problem. Girdhar and Dudek [10] used a database of observations to detect anomalous data. Similar to our approach, a generative model was learned online by directing the robot towards these anomalies. However these approaches used very short planning horizons and do not make decisions about using expensive secondary sensors to gain information.

Tabib et al. [17] explored a search and rescue application where their robot plans trajectories that maximize the information gained by two different sensors which measure the geometry and temperature of the environment. It is assumed that the instruments are constantly collecting data, instead of actively switched on which simplifies the planning problem. Furthermore, it is assumed that the two sensors are conditionally independent while in our problem, being able to learn and exploit relationships between sensing modalities is fundamental.

Arora et al. used a Bayesian network to model relationships between sensing modalities and the phenomena they are trying to measure [2]. The work assumes the relationship between sensors is known a priori while in this paper we learn this relationship online. Furthermore, the work uses a greedy planner while here we explore long horizon planning and incorporate goal constraints.

Das et al. [7] builds a map of underwater plankton abundance by planning the deployment of a low cost sensor which measures environmental parameters and an

expensive ‘plankton’ sensor. To achieve this, two Gaussian Processes (GPs) are used. The first maps spatial co-ordinates to environmental parameters while the second maps environmental parameters to plankton abundance. The robot samples from locations with high plankton uncertainty, where the uncertainty is propagated through both GPs via the Unscented Transform. However the computational complexity of GPs grows with the number of collected samples. Using this framework in online adaptive planning applications like ours is not amenable to long-term operation with the limited computing resources in field robots.

3 Problem Setup

Like MVP we consider a ground vehicle exploring an open environment searching for subsurface water abundance. The operating environment is discretized into a grid where the robot is required to estimate the abundance of water, W , in each grid cell, n . While the robot can be equipped with an arbitrary number of sensors, for ease of illustration, we consider the case with two sensors: a camera which can be used to classify terrain in a cell and a neutron spectrometer (NSS) which returns counts that are positively correlated with water abundance.

The robot plans action sequences, $a_{1:L}$, to maximize the expected information gained, EI , on the water distribution in each cell. The camera always takes measurements but the robot must actively decide when to use the NSS. The robot must also reach a goal position, x_{goal} , before it exhausts the operating (motion and sensing) budget of the mission, B . The optimization objective is:

$$\begin{aligned} a_{1:L}^* = \arg \max_{a_{1:L} \in A} & EI(a_{1:L}) \\ \text{s.t. } & cost(a_{1:L}) \leq B \\ & x_{end}(x_{start}, a_{1:L}) = x_{goal} \end{aligned} \quad (1)$$

A is the action space of the robot which contains the movements the robot can take in the next time step and the decision of whether on not to use the NSS. We define the action space as the four cardinal directions but any motion models can be used here. Similarly, any general cost function can be used and we define ours in Sect. 5. The expected information gain is given by Eq. 2 where $H(\cdot)$ is the Shannon entropy, W_n is the water abundance in a cell n and N is the total number of cells in the environment.

$$EI(a_{1:L}) = \sum_{n=1}^N [H(W_n) - H(W_n | a_{1:L})] \quad (2)$$

Each action produces some stochastic observation Z_s which reveals information about the water distribution, where $s \in \{\text{Image}, \text{NSS}\}$. The expected information gain

of a sensing sequence is computed over all possible observations that can result from each sensing action in the sequence:

$$EI(a_{1:L}) = \sum_{n=1}^N \left[H(W_n) - \sum_{Z_{1:L}} H(W_n|Z_{1:L}) P(Z_{1:L}|a_{1:L}) \right] \quad (3)$$

$P(Z_{1:L}|a_{1:L})$ is the sensor noise model while the $H(W_n|Z_{1:L})$ term is a function of the robot's belief on the environment and the sensor correlations.

4 Approach

The overall proposed architecture is shown in Fig. 2. Instead of specifying paths and sensing waypoints directly, our framework allows scientists to simply provide a goal position constraint, sensing budget and the variable they are interested in learning about- a useful capability in remote environments with communication constraints. We now describe the two main components of our approach: a generative model for learning sensor correlations online and an anytime, approximate path planner (MCTS) to find approximate solutions to Eq. 1.

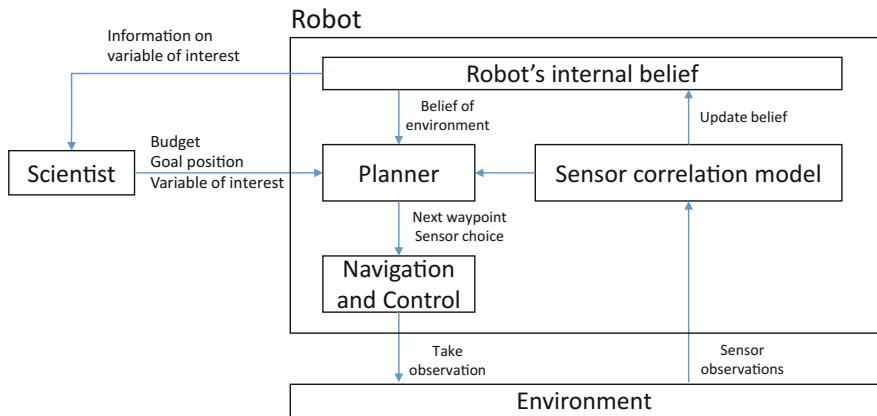
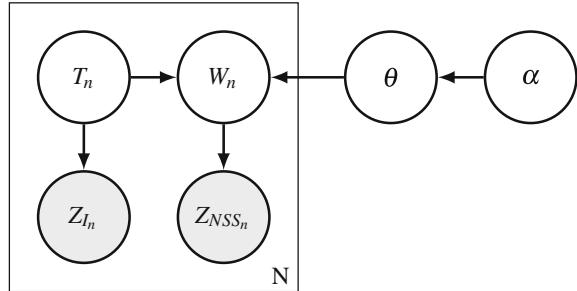


Fig. 2 The overall systems architecture for our approach

Fig. 3 Bayesian generative sensor correlation model.
The model can be extended to an arbitrary number of sensors and adapted to incorporate sensor dependencies that come with specific applications



4.1 Modeling and Learning Sensor Correlations

Loosely inspired from topic modeling literature [4], we structure the dependencies between the NSS observations and the camera with the generative model shown in Fig. 3. The NSS observes the water distribution W in a cell n through observations Z_{NSS} . The camera observation is denoted by Z_I while T is the class of terrain. A conditional probabilistic relationship between terrain and water classes is parameterized by θ and hyperparameters α which are learned during the mission as data is collected. We assume all nodes are discrete variables but the observation nodes can directly handle continuous data as well. The probabilistic mapping from T and W nodes to their corresponding observation nodes is deduced from the sensor/classifier model the robot is using. Unsupervised dimensionality reduction techniques can also be applied here.

In this section we derive the Bayesian update for the beliefs of nodes W_n , T_n and θ as observations are made in a cell. We define:

$$P(W|T = t) \sim \text{Categorical}(\theta_t) \quad (4)$$

$$\theta_t \sim \text{Dirichlet}(\alpha_t) \quad (5)$$

$$\theta = [\theta_1, \theta_2, \dots, \theta_T] \quad (6)$$

By applying Bayes Theorem and exploiting conditional dependencies in the Bayesian network, the beliefs on the water abundance and terrain types can be updated using Eq. 7 where η is the normalization constant. For compactness, we drop the subscript n from the terms W_n and T_n .

$$\begin{aligned}
P(W|Z_I, Z_{NSS}) &= \eta P(Z_{NSS}|W)P(W|Z_I) \\
&= \eta P(Z_{NSS}|W) \sum_T P(T|Z_I)P(W|Z_I, T) \\
&= \eta P(Z_{NSS}|W) \sum_T P(T)P(Z_I|T)P(W|Z_I, T) \\
&= \eta P(Z_{NSS}|W) \sum_T P(T)P(Z_I|T) \int_\theta P(W|T, \theta)P(\theta)d\theta \quad (7) \\
&= \eta P(Z_{NSS}|W) \sum_T P(T)P(Z_I|T) \int_\theta \theta P(\theta)d\theta \\
&= \eta P(Z_{NSS}|W) \sum_T P(T)P(Z_I|T)\mathbb{E}(\theta)
\end{aligned}$$

Similarly, we can iteratively update belief on terrain by evaluating:

$$P(T|Z_I, Z_{NSS}) = \eta P(T)P(Z_I|T) \sum_W P(Z_{NSS}|W)\mathbb{E}(\theta) \quad (8)$$

Since θ is modeled by a Dirichlet distribution, $\mathbb{E}(\theta)$ can be efficiently calculated by normalizing the corresponding hyperparameters. We can update θ using Eq. 9. For compactness we define the full observation vector $Z = [Z_I, Z_{NSS}]$.

$$\begin{aligned}
P(\theta|\alpha, Z) &= \sum_{T,W} P(\theta|\alpha_{init}, Z, T, W)P(T, W|Z) \\
&= \sum_{T,W} P(\theta|\alpha_{init}, T, W)P(T, W|Z) \quad (9)
\end{aligned}$$

Since $P(\theta|\alpha, Z)$ is also a Dirichlet distribution (conjugate prior) we can calculate the posterior by updating the hyperparameters $\alpha_{w,t} = \alpha_{w,t} + P(W=w, T=t|Z)$ for all values of W and T , where $w \in \{1, \dots, |W|\}$ and $t \in \{1, \dots, |T|\}$. When $|W|$ and $|T|$ become large, Gibbs sampling approaches in topic modeling literature can be used to approximate this update [10]. When a terrain cell is observed we also update the terrain beliefs in neighboring cells using a Gaussian kernel.

4.2 Planning

Given the generative BN model, the robot needs to plan paths in an initially unknown environment and decide when to use the NSS to maximize the information gained on the water distribution in the map cells, described in Algorithm 1. As per the optimization objective in Eq. 1, the planned paths and sensing sequences must meet mission budget constraints and arrive at the goal location, x_{goal} .

Algorithm 1 Our algorithm uses MCTS for the $\text{planner}(\cdot)$, which is executed after every action.

```

1: Input: SensingBudget  $S$ , BeliefSpace  $Bel$ , RemainingBudget  $R$ , GoalPosition  $x_{goal}$ 
2: function MAIN
3:    $R \leftarrow S$ 
4:   while  $R > 0$  do
5:      $robotPose \leftarrow getLocalisation()$ 
6:      $a_{opt} \leftarrow \text{planner}(robotPose, R, Bel, x_{goal})$ 
7:      $Z \leftarrow takeObservation(a_{opt})$ 
8:      $Bel \leftarrow updateBeliefSpace(Z, Bel)$ 
9:      $R \leftarrow R - cost(a_{opt})$ 

```

Solving Eq. 1 for large environments, long mission durations and large observations spaces quickly becomes intractable, especially for field robots with limited computational resources. Therefore we explore approximate online planning approaches where in each time step the robot executes the first action in the calculated plan and adaptively updates plans as new observations are taken. To tackle this sequential decision making problem, we employ the MCTS planning algorithm- a best first, anytime algorithm popular in game playing literature, which like our problem requires reasoning about both long horizons and stochasticity [5].

We formulate the MCTS such that each node in the tree is a potential movement or sensing action that can be made. It is a tuple consisting of the robot's x and y position, a binary variable indicating whether the NSS was used and the remaining sensing budget. MCTS then iteratively builds a tree by selecting leaf nodes to expand using a tree policy, estimating terminal rewards associated with the leaf by conducting simulations or ‘rollouts’ in the decision space and back-propagating the reward up the tree. The process is repeated until some computational budget is reached, at which point the root child with the highest average reward is selected as the action to be executed.

We use the Upper Confidence Tree policy to select which leaf nodes to expand, which is a popular approach known to produce good results [14]. For the simulation phase, a random action selection policy is used from the leaf node to the goal position. In this problem instance, the reward of the policy rollout is the expected information gained on water distribution across the map after the policy has been executed, where information gain is defined by Eq. 2. Exact computation of the reward involves averaging over all possible observations that can result from the rollout sequence which quickly becomes intractable. We approximate information gain by sampling observations from the robot's belief of the map and simulating a belief update. As number of iterations increases, the MCTS converges to the optimal sensing action sequence. This formulation gives us a principled approach to incorporate multiple sensors in planning and simultaneously handle long horizons and uncertainty in an anytime manner.

Algorithm 2 MCTS Algorithm

```

function PLANNER(robot Pose, R, Bel, xgoal)
    T  $\leftarrow$  initialiseTree(robot Pose, R)
    currentNode  $\leftarrow$  T.rootNode
    while within computational budget do
        currentNode  $\leftarrow$  treePolicy(T)
        simSeq  $\leftarrow$  defaultPolicy(currentNode, R)
        reward  $\leftarrow$  getReward(simSeq, Bel)
        T  $\leftarrow$  updateTree(T, reward)
    return bestChild(T)

```

5 Analysis

As mentioned in Sect. 2 there are several algorithms in literature for informative path planning [3, 12]. However, these approaches are not suitable for tackling situations in which the robot has to simultaneously decide **when** to activate secondary sensors in addition to planning informative paths which adhere to budget and goal constraints in initially unknown environments. We therefore compare the performance of our approach with the following three baseline algorithms:

Random: At each time step the robot determines the set of actions it can execute in the next step without breaking the goal position and sensing budget constraint. A random action is chosen out of this set. The random policy serves as a baseline for algorithm performance.

Greedy: At each time step, out of the reachable action set, the robot selects the action with the highest expected information gain of the water abundance to sensing cost ratio. This is given by:

$$a_{next}^* = \arg \max_{a \in A} \frac{\sum_z I(z) P(z|a)}{cost(a)} \quad (10)$$

Greedy algorithms are popular in similar field applications [10, 18] due to their simplicity and depending on the problem, submodularity.

Lawnmower: We use a ‘lawnmower’ pattern to get uniform coverage of the environment. Here we arbitrarily allocate 50% of the sensing budget to the path and 50% to using the NSS. A lawnmower-like path which adheres to the initial and final positions and the budget is designed manually and the NSS is used at uniform intervals along the path.

Our approach, **MCTS-50** (50 iterations were used for MCTS) was evaluated against the baseline algorithms on 50 randomly generated 20 by 20 voronoi maps with fixed start and goal positions. Terrain, water, and the observation nodes were categorical variables with three classes. The true correlation between terrain type and water class (initially unknown to the robot) was set to be 0.85. I.e. given the terrain class, the water class could be predicted with 85% accuracy. Sensor noise for the terrain was set to be 10% while the NSS had 5%. All unobserved nodes were given

an uniform prior and the α hyperparameters were initialized to a value of 1. The cost of movement was 1 unit per cell while the NSS required 5 units. Two performance metrics were used: information gain and the average posterior probability of the correct class of water in the cells which we call the recognition score.

Mean and standard deviation is reported and statistical significance is shown with the paired t-test p -value and the effect size using Cohen's d . Negative values of d indicate that the performance of the proposed algorithm is greater than the compared algorithm. The magnitude of d gives the size of the effect, with $d > 0.2$, $d > 0.5$ and $d > 0.8$ being thresholds for small, medium and large effects respectively.

The results are shown in Tables 1 and 2. In terms of average information gain, we statistically significantly outperform random and greedy policies with notable effect sizes (bolded). For the recognition score, the performance improvement is less pronounced. This is because the robot only observes a small proportion of the map and the unseen areas dominate the score.

The performance of the lawnmower is comparable to MCTS in these simulated experiments. In completely unknown and open environments, paths which provide good spatial coverage of the environment are indeed a logical and effective way to gain information. In more realistic environments with obstacles, planning lawnmower paths becomes more complicated. When environmental obstacles are known a priori Choset's approach can be applied [6]. In unknown or partially known environments, however, additional replanning would need to occur as obstacles are discovered, something our approach already does. Further, adapting the lawnmower approach to an arbitrary number of sensors would require a way to split the sensing budget across the different sensing modalities, which the MCTS optimizes automatically in a principled manner. While the 50–50 budget split between paths and NSS produced good results in the simulation setting, there is no guarantee that performance will continue to be competitive in longer missions and large environments.

In robotic missions, there is usually some prior knowledge available such as orbital maps or scientific beliefs on what the robot is likely to see. A key advantage of our approach is that we can easily encode this knowledge in the form of Bayesian priors. Orbital maps can be encoded by biasing the prior distribution of terrain types while scientific knowledge of known sensor correlations can be incorporated by incrementing the α hyperparameters. Unlike the standard lawnmower, our approach will automatically take advantage of this information without algorithmic modifications. To verify this, we ran 50 trials with a sensing budget of 140 where the robot's belief of the correct terrain type was initialized to 0.5 instead of a uniform distribution. MCTS outperformed the lawnmower with p -values and effect sizes of <0.001 and ≈ -0.5 respectively for both information gain and recognition scores.

The computational time of MCTS depends on the remaining sensing budget and problem size. For our MATLAB implementation, 50 iterations took between 0 and 5 s. With more efficient memory management, optimized implementation and parallelization, significant speed boosts can be achieved which will further boost the performance of MCTS as more iterations can be carried out.

Table 1 Information gain for the different algorithms and their performance relative to MCTS-50

Budget	Greedy				Random				Lawnmower				MCTS-50			
	μ	σ	ρ	d	μ	σ	ρ	d	μ	σ	ρ	d	μ	σ		
60	20.7	8.93	0.05	-0.34	15.6	7.16	1e-5	-0.95	22.4	8.99	0.40	-0.17	24.0	10.23		
80	28.4	11.76	0.003	-0.60	20.6	10.92	9e-8	-1.20	31.7	12.62	0.03	-0.35	36.7	15.54		
100	32.3	12.81	0.004	-0.57	27.1	13.42	1e-4	-0.88	39.4	14.50	0.52	-0.12	41.4	18.81		
120	39.3	13.99	0.003	-0.62	29.0	12.93	2e-8	-1.31	46.6	19.24	0.39	-0.14	49.1	17.37		
140	43.4	13.12	3e-5	-0.84	33.3	16.58	5e-9	-1.34	54.8	21.4	0.62	-0.10	56.8	18.50		

Table 2 The average posterior probability of the true water distribution given the maps learned by the different algorithms, larger values are better

Budget	Greedy				Random				Lawnmower				MCTS-50			
	μ	σ	p	d	μ	σ	p	d	μ	σ	p	d	μ	σ	d	
60	0.37	0.02	0.56	-0.08	0.36	0.02	0.004	-0.43	0.38	0.02	0.14	0.25	0.38	0.03		
80	0.39	0.03	0.008	-0.44	0.37	0.03	3e-8	-0.94	0.40	0.03	0.17	-0.19	0.41	0.04		
100	0.41	0.03	0.21	-0.19	0.38	0.03	8e-5	-0.81	0.42	0.03	0.36	0.15	0.41	0.05		
120	0.42	0.04	0.04	-0.35	0.38	0.04	5e-10	-1.37	0.43	0.03	0.84	-0.03	0.43	0.03		
140	0.43	0.04	0.0008	-0.54	0.39	0.04	3e-12	-1.62	0.44	0.04	0.15	-0.26	0.45	0.03		

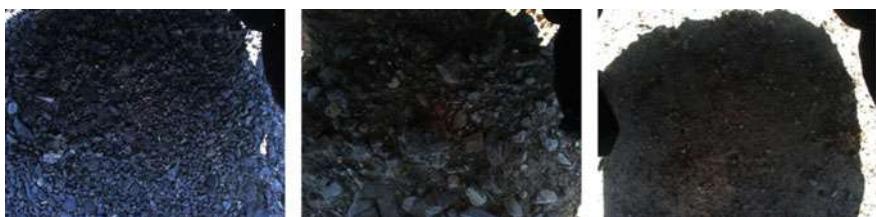
6 Results with Mojave Data

Since much of the data from the Mojave Desert test site was collected in line traverses, we selected 100 pairs of ground camera images and NSS counts from this dataset and redistributed them into a 10 by 10 grid to simulate a field environment. Typical ground camera images are shown in Fig. 4. The images from the MVP dataset are quite noisy with both strong shadows and regions with saturation.

The data now needs to be transformed into a representation that can be fed into the generative model. While any black box classifier model can be used for this, we use a simple example based classifier for illustration. We selected image subsets based on domain knowledge of the terrain classes present and used these to define four cluster centers. Candidate images are then classified based on the closest cluster centre in intensity space. The labels are transformed into soft evidence using a confusion matrix derived from training data. Similarly, k-means clustering with three clusters is used to probabilistically classify NSS counts into water abundance. The probabilistic classifications are fed into the BN as soft evidence. Continuous data can also be directly fed into the proposed generative model as long as the probabilistic mapping from T and W nodes to observations can be determined.

We compare MCTS-50 and the lawnmower algorithms on 20 randomly generated 10 by 10 maps with a sensing budget of 40. We ran two sets of trials with NSS costs of 5 and 2 units. Since the sensing budget of 40 is relatively small, by reducing the cost of NSS, the latter trial artificially increases the planning horizon and intends to show the resulting changes in performance.

The results are shown in Fig. 5. In terms of information gain, MCTS is on average better than lawnmower for this sample and statistically significantly when the NSS cost is 2. There is a larger performance gap compared to the simulations. This is because, doing a 50–50 split in the lawnmower budget allocation is no longer as effective for this map size, sensing budget and sensor model. Like in simulations, we assumed an initially unknown environment and further improvements can be expected with the integration of prior knowledge. In terms of recognition score,



(a) Pavement Terrain Type (b) Transition Terrain Type (c) Wash Terrain Type

Fig. 4 Different types of terrain in the MVP test area. Pavements were found to be associated with high NSS counts, while washes had low NSS counts. The transition terrain was in between washes and pavements and had moderate NSS counts.

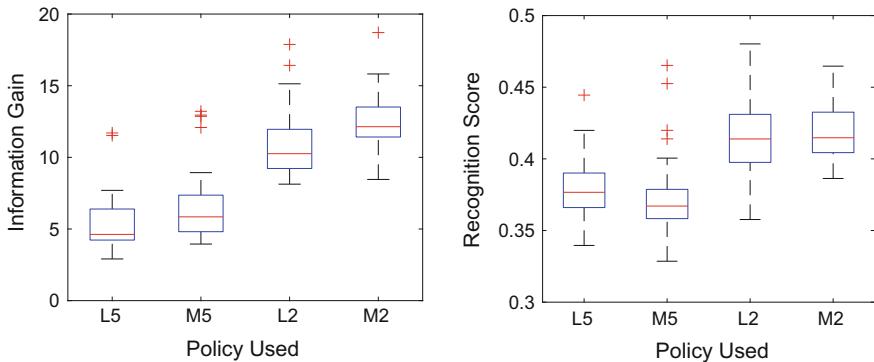


Fig. 5 Comparison of information gain and recognition scores for lawnmower and MCTS for different NSS costs

MCTS is slightly lower than lawnmower in NSS-5 and similar in NSS-2 but remains statistically indifferent like in simulations.

7 Conclusions and Future Work

Being able to reason about scientific latent variables of interest to plan informative sensing sequences is an important problem in field robotics. We have presented a scalable approach to automatically learn sensor correlations online and a sampling based approach to plan long horizon sensing sequences which is anytime and incorporates budget and goal position constraints. Our simulations and real data experiments show we significantly outperform myopic approaches which are popular in similar applications and compete with maximum coverage paths in unknown environments. Our approach can also exploit prior knowledge when it is available without further algorithmic modifications. In future work we would like to incorporate unsupervised approaches to classification and evaluate our approach on different applications such as remote sensing and agriculture.

Acknowledgements We would like to thank NASA's Resource Prospector project and the Australian Center for Field Robotics for funding this work.

References

1. Andrews, D.R., Colaprete, A., Quinn, J., Chavers, D., Picard, M.: Introducing the resource prospector (RP) mission. In: AIAA SPACE 2014 Conference and Exposition, p. 4378 (2014)
2. Arora, A., Fitch, R., Sukkarieh, S.: Extending autonomy of planetary rovers by encoding geological knowledge in a bayesian framework. In: 10th International Cognitive Robotics

- Workshop, IEEE International Conference on Intelligent Robotics and Systems, IROS (2016)
- 3. Binney, J., Sukhatme, G.S.: Branch and bound for informative path planning. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2147–2154. IEEE (2012)
 - 4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *J. Mach. Learn. Res.* **3**, 993–1022 (2003)
 - 5. Browne, C.B., Powley, E., Whitehouse, D., Lucas, S.M., Cowling, P.I., Rohlfschagen, P., Tavener, S., Perez, D., Samothrakis, S., Colton, S.: A survey of monte carlo tree search methods. *IEEE Trans. Comput. Intell. AI Games* **4**(1), 1–43 (2012)
 - 6. Choset, H.: Coverage for robotics-a survey of recent results. *Ann. Math. Artif. Intell.* **31**(1), 113–126 (2001)
 - 7. Das, J., Harvey, J., Py, F., Vathsangam, H., Graham, R., Rajan, K., Sukhatme, G.S.: Hierarchical probabilistic regression for AUV-based adaptive sampling of marine phenomena. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 5571–5578. IEEE (2013)
 - 8. Dunbabin, M., Marques, L.: Robots for environmental monitoring: significant advancements and applications. *IEEE Robot. Autom. Mag.* **19**(1), 24–39 (2012)
 - 9. Foil, G., Fong, T., Elphic, R.C., Wettergreen, D.: Physical process models for improved rover mapping. In: The International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS) (2016)
 - 10. Girdhar, Y., Dudek, G.: Modeling curiosity in a mobile robot for long-term autonomous exploration and monitoring. *Auton. Robots* **40**(7), 1267–1278 (2016)
 - 11. Heldmann, J., Colaprete, A., Cook, A., Roush, T., Deans, M., Elphic, R., Lim, D., Skok, J., Button, N., Karunatillake, S., et al.: Mojave volatiles prospector (MVP): science and operations results from a lunar polar rover analog field campaign. In: Lunar and Planetary Science Conference, vol 46, p. 2165 (2015)
 - 12. Hollinger, G.A., Sukhatme, G.S.: Sampling-based robotic information gathering algorithms. *Int. J. Robot. Res.* **33**(9), 1271–1287 (2014)
 - 13. Hollinger, G.A., Englot, B., Hover, F.S., Mitra, U., Sukhatme, G.S.: Active planning for under-water inspection and the benefit of adaptivity. *Int. J. Robot. Res.* **32**(1), 3–18 (2013)
 - 14. Kocsis, L., Szepesvári, C.: Bandit based monte-carlo planning. In: Proceedings of European Conference on Machine Learning, pp. 282–293 (2006)
 - 15. Krause, A., Guestrin, C.: Near-optimal observation selection using submodular functions. *AAAI* **7**, 1650–1654 (2007)
 - 16. Leshin, L., Mahaffy, P., Webster, C., Cabane, M., Coll, P., Conrad, P., Archer, P., Atreya, S., Brunner, A., Buch, A., et al.: Volatile, isotope, and organic analysis of martian fines with the mars curiosity rover. *Science* **341**(6153), 1238937 (2013)
 - 17. Tabib, W., Whittaker, R., Michael, N.: Efficient multi-sensor exploration using dependent observations and conditional mutual information. In: IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), pp. 42–47. IEEE (2016)
 - 18. Thompson, D.R., Wettergreen, D.: Intelligent maps for autonomous kilometer-scale science survey. In: The International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS) (2008)
 - 19. Wettergreen, D., Foil, G., Furlong, M., Thompson, D.R.: Science autonomy for rover subsurface exploration of the atacama desert. *AI Mag.* **35**(4), 47–60 (2014)

Season-Invariant Semantic Segmentation with a Deep Multimodal Network

Dong-Ki Kim, Daniel Maturana, Masashi Uenoyama
and Sebastian Scherer

Abstract Semantic scene understanding is a useful capability for autonomous vehicles operating in off-roads. While cameras are the most common sensor used for semantic classification, the performance of methods using camera imagery may suffer when there is significant variation between the train and testing sets caused by illumination, weather, and seasonal variations. On the other hand, 3D information from active sensors such as LiDAR is comparatively invariant to these factors, which motivates us to investigate whether it can be used to improve performance in this scenario. In this paper, we propose a novel multimodal Convolutional Neural Network (CNN) architecture consisting of two streams, 2D and 3D, which are fused by projecting 3D features to image space to achieve a robust pixelwise semantic segmentation. We evaluate our proposed method in a novel off-road terrain classification benchmark, and show a 25% improvement in mean Intersection over Union (IoU) of navigation-related semantic classes, relative to an image-only baseline.

1 Introduction

For autonomous vehicles operating in unstructured off-road environments, understanding their environment in terms of semantic categories such as “trail”, “grass” or “rock” is useful for safe and deliberate navigation. It is essential to have robust scene understanding as false information can result in collisions or other accidents.

D. -K. Kim (✉) · D. Maturana · S. Scherer
Carnegie Mellon University, Pittsburgh, USA
e-mail: dkkim@andrew.cmu.edu

D. Maturana
e-mail: dmaturan@andrew.cmu.edu

S. Scherer
e-mail: basti@andrew.cmu.edu

M. Uenoyama
Yamaha Motor Corporation, Cypress, USA
e-mail: mike_uenoyama@yamaha-motor.com

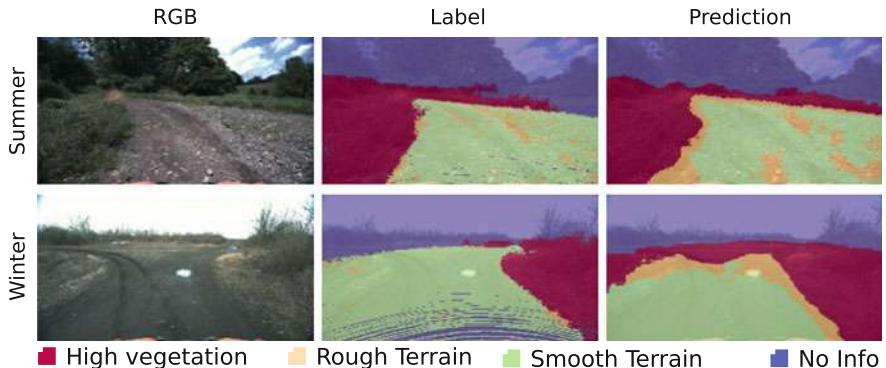


Fig. 1 An image-based CNN [2] trained on a sunny summer dataset (top row) cannot predict robustly when a test dataset has severe appearance variations, such as on a cloudy winter dataset (bottom row)

An important step toward scene understanding is semantic image segmentation, which classifies an image at a pixel level. In recent years, deep Convolutional Neural Networks (CNNs) have achieved the state-of-the-art in semantic segmentation [1–7], surpassing traditional computer vision algorithms. However, we have observed segmentation performance for CNNs suffer when there exist significant appearance variations between the train and testing sets, caused by illumination, weather, and seasons (Fig. 1). A straightforward solution is to add more training data with the relevant variation factors, but this approach is expensive because of the effort required to collect data and label the ground-truth for training.

Instead, an effective approach to address this problem is to use an additional, complementary, sensor, such as LiDAR. Whereas a camera has advantages in the range of vision and the density of data, a LiDAR has an advantage in invariance to appearance variation caused by illumination, weather, and seasons. Thus, a combined approach using an image and 3D point clouds collected by LiDAR creates opportunities for CNNs to take advantage of their complementary characteristics. However, the following questions still remain open: (1) how to jointly use the two sensors for image segmentation, and (2) what features from each modality are useful for robust segmentation.

In this work, we propose a solution in terms of a deep multimodal network, which jointly uses image and 3D point cloud data, and outputs a segmented image. Our main contribution is a framework with projection modules that enable the multimodal network to learn 2D and 3D feature representations, but also combine the features in different domains effectively during training to segment an image robustly. To evaluate the robustness of our method to appearance variations, we assembled a labeled dataset of image and LiDAR captured from a modified All-Terrain Vehicle operating in an off-road location across two different seasons, winter and summer. We show that our proposed approach is highly accurate and significantly more robust to this variation than image-only baselines.

2 Related Work

In general, relevant approaches for semantic scene understanding broadly fall into one of two classes depending on the number of input modalities: unimodal (e.g., only image input) or multimodal (e.g., image and 3D point cloud).

2.1 Unimodal Image-Based Approaches

Semantic segmentation of RGB images is an active research topic. Many successful approaches use graphical models, such as Markov or Conditional Random Fields (MRFs or CRFs) [8–11]. These approaches often start with an over-segmentation of an image into superpixels and extract hand-crafted features from individual and neighboring segments. A graphical model uses the extracted features to ensure the consistency of the labeling for neighboring regions.

Instead of relying on engineered features, CNN-based approaches have achieved the state-of-the-art segmentation performance by learning strong feature representations from raw data [1–3]. The main difference between CNN approaches is the network architecture. Shelhamer et al. [1] introduce the use of skip layers to refine the segmentation produced by so-called deconvolution layers. Badrinarayanan et al. [2] propose an encoder-decoder architecture with unpooling layers. These architectures use the relatively slow VGG [12] architecture. To reduce computational costs, an important goal for robotics, Paszke et al. [3] apply a bottleneck structure, motivated by [13], to build an efficient network with a small number of parameters but similar accuracy to prior models. We base the image-based part of our network on these architectures.

2.2 Multimodal Approaches

Researchers have used image and 3D point clouds for scene understanding. In one of the main inspirations for our work, Munoz et al. [14] train two classifier cascades, one for each modality, and hierarchically propagate information across the two classifiers using a stacking approach. Newman et al. [15] describe a framework that classifies an individual LiDAR data by the Bayes decision rule and support-vector machines, and uses the majority consensus to label superpixels in an image. Cadena and Košecká [16] propose a CRF framework that enforces spatial consistency between separate feature sets extracted from two sensor’s coverage. Alvis et al. [17] extract appearance features from images for CRF and obtain global constraints for sets of superpixels from 3D point clouds.

There are also several CNN-based approaches using RGB and Depth (RGBD) representations, usually from stereo or structured lighting sensors. Couprise et al. [4]

combine feature maps of multiscale CNNs from RGB-D and superpixels obtained from an RGB image to segment an image. Gupta et al. [5] extract CNN features both from the color and the encoded depth to detect objects in indoors. They demonstrate that the augmented features computed based on object detections improve the segmentation performance in [18].

A recent, relevant RGBD approach is that of Valada et al. [6]. In this approach identical 2D CNNs are first learned to segment different modality input. Then features of different modalities are fused by summing up feature maps of each CNN’s output and processed later (late-fusion convolution approach). Whereas their fusion happens at the output of each CNN model (late-fusion), we consider incorporating features hierarchically from the other modality as multiple levels of abstractions learned by CNN have proven beneficial [7].

A critical difference of our approach to methods using RGBD, is that we learn not only 2D features, but also 3D features. 3D features contain useful spatial information, which is hard to learn in 2D.

3 Proposed Approach

Our objective is to predict four semantic classes (“High Vegetation”, “Rough Terrain”, “Smooth Terrain”, “No Info”) for safe navigation in off-roads. Cameras are the most common sensor used for scene understanding because it has advantages in a long range of vision (e.g., obstacles can be detected in the far distance) and dense data. However, the performance of an image-based CNN may suffer when there exists a significant variation between the train and testing image sets caused by illumination, weather, and seasonal variations. On the other hand, 3D information from LiDAR is comparatively invariant to these factors. We additionally use a 3D point cloud data to help CNN learn a more robust set of features to appearance variations.

Our deep multimodal network (Fig. 2) jointly uses an image from a camera and a 3D point cloud from a LiDAR, and outputs a segmented image. Our framework consists of an image network that learns 2D feature representations from an image, a point cloud network that learns 3D feature representations from a point cloud, and a projection module that propagates the learned 3D features to the image network. The propagation of the 3D features enables the image network to combine 2D/3D features and learn a more robust set of features during training. In this section, we describe these major components of our multimodal network in detail.

3.1 Image Network

The goal of an image network is to learn 2D feature representations θ^{2D} from images that minimize the categorical cross-entropy loss. A network should have a good

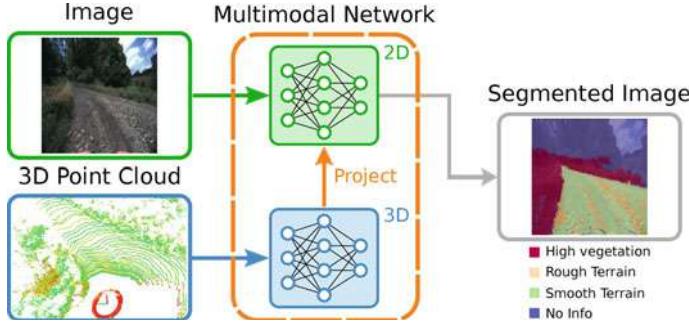


Fig. 2 Our multimodal network takes inputs of an image and a 3D point cloud. Our network learns and combines 2D/3D features; and outputs a segmented image. (point cloud colored by the intensity)

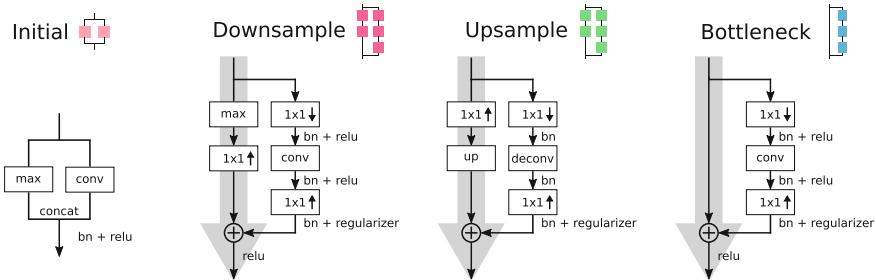


Fig. 3 ENet modules [3] used in our network. *max*: maxpooling layer with non-overlapping 2×2 windows. *up*: upsample layer by a factor of 2. *conv*: either a regular, dilated, or asymmetric convolution layer. *bn*: batch normalization. *regularizer*: spatial dropout. 1×1 with down or up arrow: 1×1 convolution to reduce or expand channels

segmentation performance, but also have fast prediction time and a small number of parameters to be easily embedded in a real-time autonomous system. In this work, we design the network based on ENet [3], which has demonstrated its similar performance to existing models (e.g., SegNet [2]) but with much faster inference time and much smaller number of parameters. ENet has the encoder part (initial, stage 1–3) and the decoder part (stage 4–5), which consist of the initial, downsample, upsample, and bottleneck module described in Fig. 3. The bottleneck module has an architecture of a single main branch and a separated branch with convolutional filters. We use it several times in each stage, which enables the network to be deeper with less vulnerability to the network degradation problem [13]. ENet architecture is described in Fig. 5 (the above network). We refer readers to [3] for more details about the network.

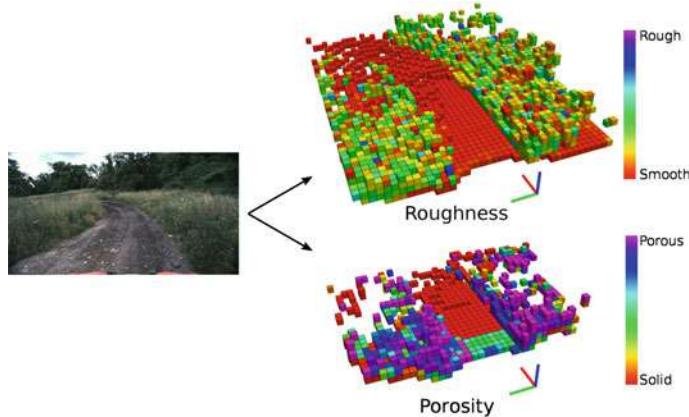


Fig. 4 Visualization of roughness and porosity feature. The terrain area shows a low roughness and low porosity, relatively to the vegetation area. We omit empty voxels for visibility. Axis notation: x-axis (red), y-axis (green), z-axis (blue)

3.2 Point Cloud Network

Similarly to the image network, the point cloud network learns 3D feature representations θ^{3D} that minimize the categorical cross-entropy loss in the 3D modality. For our experiment, we use the image network (Sect. 3.1) but in 3D by using the 3D convolution layer, max-pooling layer, and upsampling layer.¹

We want to predict semantic classes of a high vegetation and a terrain as these commonly appear in off-roads. Intuitively, we would expect that the terrain area to be smoother compared to the high vegetation area; and the space containing vegetation to be relatively more porous compared to the terrain area. Maturana and Scherer [19] use this intuition and train a 3D CNN with the porosity as input to predict a landing zone detection. Similarly, we provide the roughness and the porous feature (Fig. 4) as input to the network, instead of a raw point cloud. Our hypothesis is that these features represent the desired semantic classes better than a raw point cloud.

For each grid voxel² indexed by (i, j, k) , we calculate the roughness feature $R_{i,j,k}^{3D}$ by calculating the mean residual from a fitted plane to each point inside the voxel [20]:

$$R_{i,j,k}^{3D} = \frac{1}{N} \sum_{n=1}^N \frac{|Ax_n + By_n + Cz_n + D|}{\sqrt{A^2 + B^2 + C^2}} \quad (1)$$

¹For performance reasons, we simplify the point cloud network by replacing the dilation layer and asymmetric layer with the regular convolution layer. Also, we replace the deconvolution layer with the upsample layer followed by the $3 \times 3 \times 3$ convolutional layer with stride 1. For simplicity, we use the same term “deconvolution”.

²Point cloud is represented by the 3D voxel grid as a convolutional architecture requires a regular input data format.

where N is the number of points inside each voxel, x, y, z are the position of each point, and A, B, C, D are the fitted plane parameters for N points inside the voxel (i.e., $Ax + By + Cy + D = 0$). For empty voxels (i.e., no points), we assign a constant negative roughness value of -0.1 .

For the porosity feature $P_{i,j,k}^{3D}$, we use the 3D ray tracing [21] to obtain the number of hits and pass-throughs for each grid voxel. Then we model the porosity by updating Beta parameters $\alpha_{i,j,k}^t$ and $\beta_{i,j,k}^t$ for the sequence of LiDAR measurements $\{z^t\}_{t=1}^T$ [19]:

$$\alpha_{i,j,k}^t = \alpha_{i,j,k}^{t-1} + z^t \quad (2)$$

$$\beta_{i,j,k}^t = \beta_{i,j,k}^{t-1} + (1 - z^t) \quad (3)$$

$$P_{i,j,k}^{3D} = \frac{\alpha_{i,j,k}^t}{\alpha_{i,j,k}^t + \beta_{i,j,k}^t} \quad (4)$$

where $\alpha_{i,j,k}^0 = \beta_{i,j,k}^0 = 1$ for all (i, j, k) , $z^t = 1$ for the hit, and $z^t = 0$ for the pass.

3.3 Projection Module

The projection module first projects the 3D features learned by the point cloud network onto 2D image planes. Then the bottleneck module in Fig. 3 is followed so that better feature representations can be propagated to the image network.

In terms of the projection, we map each voxel's centroid position (x, y, z) with respect to the LiDAR onto the image plane (u, v) by the pinhole camera model:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & | & t \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (5)$$

where f_x, f_y, c_x, c_y are the camera intrinsic parameters, R and t are the 3×3 rotation matrix and the 3×1 translation matrix from a camera to a LiDAR, respectively. We sample (x, y, z) for every voxel size from the original point cloud dimension (e.g., $16 \times 48 \times 40$ in Fig. 5). This is to address a problem that the projection becomes sparse due to the 3D maxpooling layers that reduce a dimension of a point cloud. We apply the z-buffer technique to account pixels that have multiple LiDAR points projected onto the same pixel location. Then, we use the nearest-neighbor interpolation to downsample the projected image planes to match the size of the image network's layer that the projection module will be merged to (Sect. 3.4).

We consider a fixed volume of 3D point clouds with regard to a LiDAR (Sect. 4.3). Thus, voxel locations and their corresponding projection locations in the image

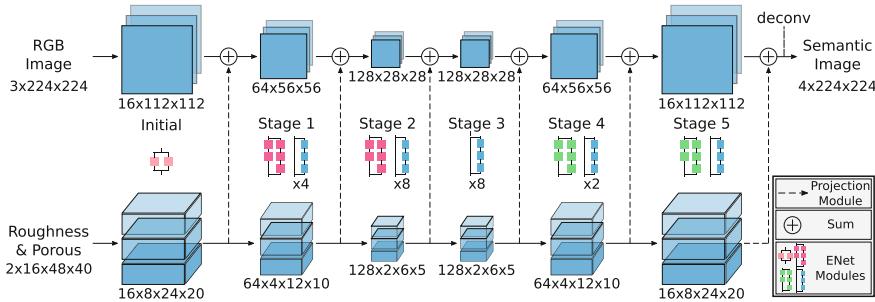


Fig. 5 Our multimodal network architecture. The upper 2D part is the image network, and the lower 3D part is the point cloud network. They are connected via the projection modules. ENet modules refer to the modules in Fig. 3. The number below the bottleneck module indicates a number of times that the module is used

network are constant if the dimensions of a point cloud and an image are same (e.g., projection for stage 1 and 4). In practice, we pre-compute indices of voxel locations and their corresponding pixel indices, and use them inside the network.

3.4 Multimodal Network

Figure 5 summarizes our multimodal network architecture: the point cloud network learns 3D features from the roughness and porous point cloud, the projection module propagates the 3D features to the image network, and the image network combines the 3D features with the 2D features extracted from images. We apply the projection modules to the outputs of the initial and the stage 1–5 because multiple levels of features learned by CNN are beneficial [7].

4 Results

We evaluate our method through a series of experiments. The experiments analyze the ability of our framework to robustly segment images despite the appearance variations caused by illumination, weather, and seasonal variations.

4.1 Dataset

We collected our dataset using a modified All-Terrain Vehicle (Fig. 6a) with a camera and a LiDAR, HDL-64E, mounted. To acquire dataset with a large appearance variation, we collected our data on two separate dates: summer sunny day in July 2016 (24 sessions) and winter cloudy day in January 2017 (2 sessions). Because the amount of winter data collected is considerably small and not enough to train our multimodal network, we only use summer data for training. We divide the dataset

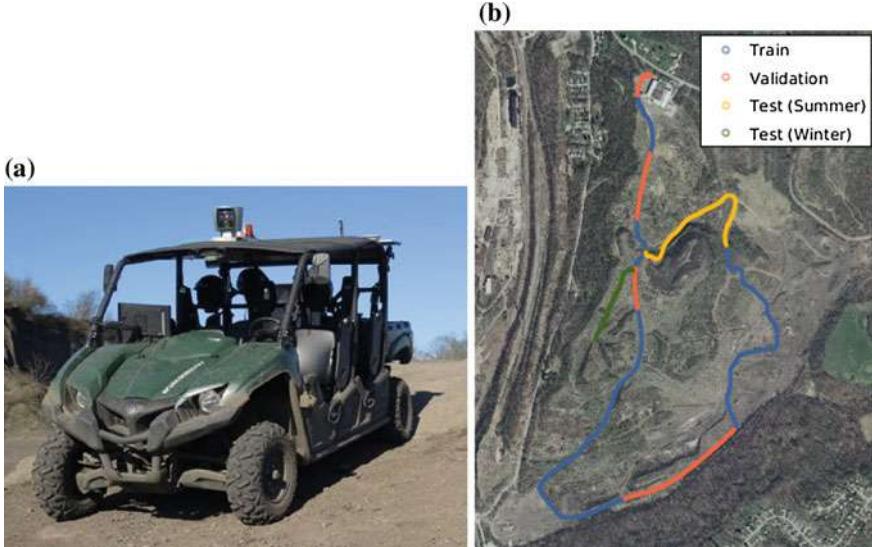


Fig. 6 **a** Our all-Terrain vehicle used for collecting the dataset. The vehicle has a camera on the front and a LiDAR on the top. **b** GPS coordinates overlayed on a geo-referenced satellite map to visualize the data distribution

based on sessions: train (17 summer sessions), validation (4 summer sessions), test summer (3 summer sessions), and test winter dataset (2 winter sessions). For the K-fold cross validation in Sect. 4.4, we set the test datasets, but randomly shuffle train/validation sessions. Data distribution for one of the K-fold cross validations is shown in Fig. 6b. We note that there is no overlap between the train, validation, and test datasets. Among the K-folds, the train data has 7.2k pairs, and the validation data has 1.7k pairs of an image and a point cloud in average. The test data for summer has 1.3k pairs, and the test data for winter has 0.6k pairs.

Our ground-truth semantic labels consist of 4 classes: “High Vegetation”, “Rough Terrain”, “Smooth Terrain”, and “No Info”. To effectively label the ground-truth and minimize the human error, we first construct a registered point cloud by stitching point clouds over time (Fig. 7a). Then we manually label the registered point cloud in the point cloud space between the terrain and high-vegetation class (Fig. 7b). We separately label another cloud with labels between the rough terrain and smooth terrain using the Eq. 1 (Fig. 7c). We merge the two labeled point clouds into one cloud with three classes (Fig. 7d). To get image labels, we project the final labeled point cloud onto an image plane. We consider voxels with no points and pixels with no LiDAR points projected as the no info class.

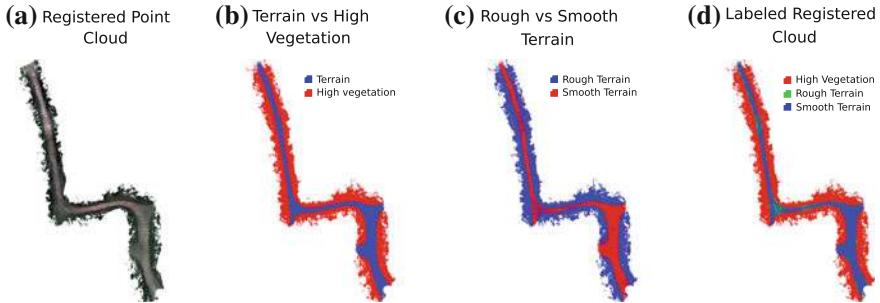


Fig. 7 The point cloud ground-truth generation procedure. **a** Point clouds are first registered. **b** The terrain and high-vegetation class are labeled manually. **c** The rough and smooth terrain class are labeled automatically using Eq. 1. **d** Final labeled point cloud is acquired by merging labeled point cloud (**b**) and (**c**)

4.2 Architectures

We compare the performance of our method (Ours-Proj) against baselines. The first baseline (Mode) classifies each pixel based on a pixelwise mode of the labels in the train dataset. Because off-roads have a general structure of trail on center and vegetation on sides, this baseline is significantly better than chance. The second baseline, SegNet is a popular encoder-decoder image segmentation network [2]. The third baseline, Ours-Image, is the image network of our multimodal network without the point cloud network and the projection modules. The last baseline (Ours-RGBRP) is same as Ours-Image, but its input to the network is 5 channels (RGB, Roughness, Porous) by projecting the point cloud network’s inputs onto the image planes and treating them as additional channels similarly to the color channels. Ours-RGBRP baseline compares the effectiveness of the learning and propagation of the 3D features against learning 2D features.

We also explore options for Ours-Proj with different locations of the projection module. We experiment with a single projection module for each stage, encoder projections (initial and stage 1–3), and decoder projections (stage 4–5).

4.3 Training Details

All input and label images are resized to 224×224 px. With respect to the LiDAR, we have a fixed volume of point cloud: -3.0 to 0.6 m (z-axis), 3.0 – 17.4 m (x-axis), and -6.0 to 6.0 m (y-axis), where the axis corresponds to the one in Fig. 4. The voxel size is 0.3 m, so the input and label point clouds have a dimension of $12 \times 48 \times 40$ (z, x, y-axis). The intrinsic and extrinsic parameters in the projection module are calibrated off-line. To reduce a GPU memory required for training Ours-Proj, we first separately train the point cloud network. Then we remove the deconvolution

Table 1 Quantitative results on summer test (mean and standard deviation)

	Per-class IoU			Average PR		
	Vege.	Rough	Smooth	No Info	Precision	Recall
Mode	0.513 (0.041)	0.000 (0.000)	0.508 (0.015)	0.806 (0.009)	0.572 (0.006)	0.611 (0.010)
SegNet	0.816 (0.008)	0.182 (0.007)	0.670 (0.019)	0.828 (0.010)	0.741 (0.003)	0.767 (0.008)
Ours-Image	0.814 (0.007)	0.183 (0.008)	0.702 (0.059)	0.837 (0.003)	0.742 (0.004)	0.767 (0.008)
Ours-RGBRP	0.833 (0.008)	0.181 (0.019)	0.648 (0.104)	0.858 (0.011)	0.747 (0.007)	0.774 (0.017)
Ours-Proj	0.839 (0.005)	0.179 (0.014)	0.655 (0.072)	0.864 (0.003)	0.747 (0.006)	0.772 (0.015)

Table 2 Quantitative results on winter test (mean and standard deviation)

	Per-class IoU			Average PR		
	Vege.	Rough	Smooth	No Info	Precision	Recall
Mode	0.453 (0.010)	0.000 (0.000)	0.712 (0.012)	0.855 (0.003)	0.589 (0.002)	0.609 (0.004)
SegNet	0.474 (0.067)	0.027 (0.002)	0.660 (0.109)	0.784 (0.059)	0.605 (0.032)	0.630 (0.031)
Ours-Image	0.498 (0.018)	0.017, (0.004)	0.595 (0.120)	0.862 (0.009)	0.623 (0.008)	0.622 (0.020)
Ours-RGBRP	0.582 (0.035)	0.036 (0.008)	0.692 (0.107)	0.881 (0.002)	0.678 (0.010)	0.689 (0.022)
Ours-Proj	0.620 (0.012)	0.040 (0.005)	0.790 (0.061)	0.875 (0.002)	0.688 (0.003)	0.705 (0.010)

and softmax layer in the point cloud network, connect with the image network via the projection modules, and train the image network and projection modules by fixing the point cloud network's weight. Except for SegNet, all learning methods are based on Theano. For SegNet [2], we use its publicly available code. We train all learning methods from scratch. We use the validation data to determine weights for the testing.

4.4 Experimental Results

We report a quantitative performance with the per-class Intersection over Union (IoU) and average precision-recall (PR) in Tables 1 and 2. The numbers correspond to the mean and standard deviation of the K-fold cross validations, where $K = 5$.

Thanks to the off-road's general structure, Mode works reasonably well for both summer and winter. However, there are no pixelwise modes for the rough terrain class, due to a small number of the rough class relative to the other classes. The performances between the unimodality networks (SegNet and Ours-Image) and the multimodality networks (Ours-RGBRP and Ours-Proj) are comparable for summer. But, the multimodal networks outperform the unimodality networks for winter. For instance, Ours-Proj shows a 25% improvement in mean Intersection over Union (IoU) of the navigation-related semantic classes (i.e., semantic classes except the no info class) relative to SegNet. Between Ours-Proj and Ours-RGBRP, Ours-Proj shows improved IoU and PR. Especially, Ours-Proj predicts the smooth terrain class accurately than other baselines. The results imply that the learning and propagation of 3D features help the network learn more robust feature representations. The qualitative results (Fig. 9) support our quantitative results. Videos of the qualitative results can be found at: <http://frc.ri.cmu.edu/~dk683/fsr17/fsr17.mp4>.

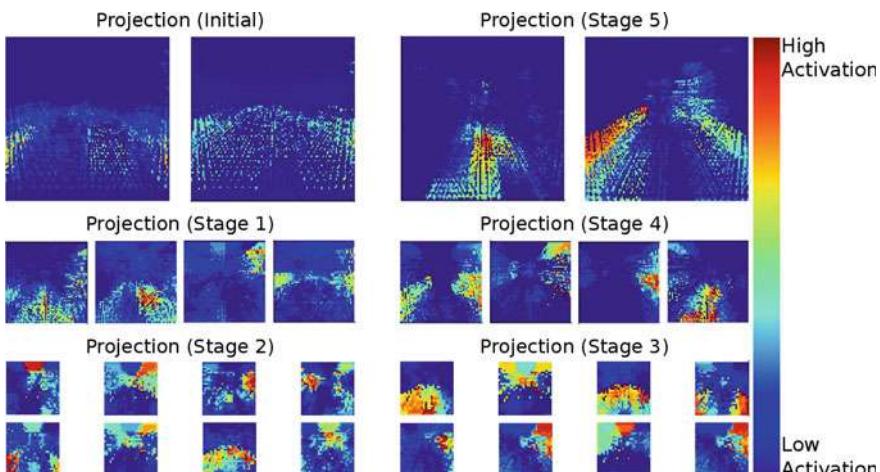


Fig. 8 Feature map visualization for each projection module's output

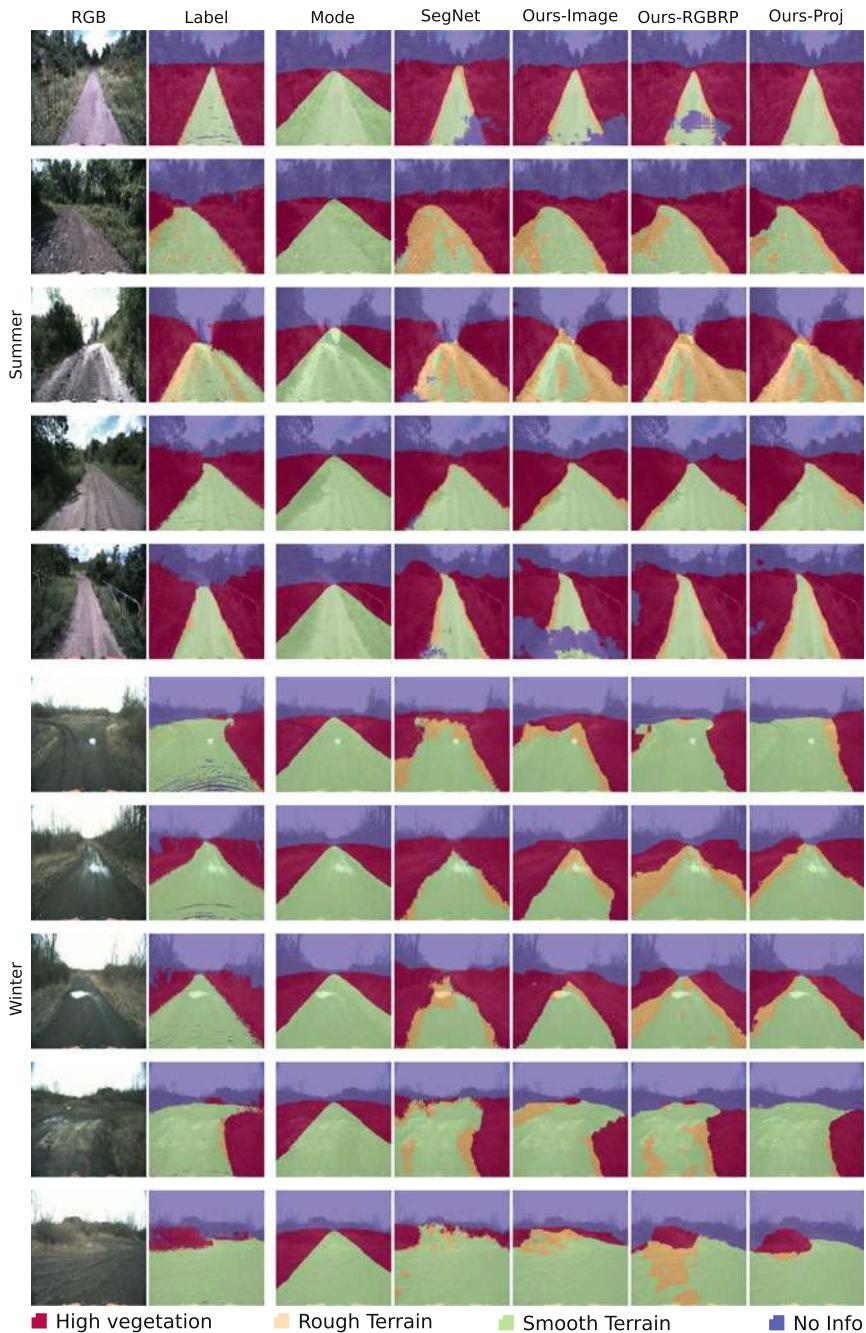


Fig. 9 One of the K-fold cross validation qualitative results

The IoU scores for the winter’s rough terrain class is small due to a little amount of the class in the winter label. We note that the multimodal methods can have advantages in predicting the no info class because the ground-truth for the class is based on the LiDAR projection. However, the multimodal networks still show improved results for the navigation-related classes.

In terms of Ours-Proj with the different projection module locations, empirical results show that the option of the encoder projections (initial and stage 1–3) achieves the best segmentation performance (similar results to the full projections described in Fig. 5). For a single projection module, the early fusion (stage 1 or 2) has better results than the late fusion (stage 4 or 5).

4.5 Network Visualization

Figure 8 shows feature maps for each projection module. Each feature map represents a particular feature on an input that a filter looks at, so it helps understand what 3D features are propagated to the image network and why they improve the results.

The visualization shows that filters focus on lower horizontal planes (e.g., terrain), vertical planes on both side (e.g., high vegetation), or diverse combinations of spatial focus based on height, width, and depth. These are helpful 3D spatial features that are hard to learn in the image domain. Thus, the joint training with 2D and 3D features would explain why Ours-Proj achieves the best performance.

5 Conclusion

We describe a novel deep multimodal network consisting of two streams, a 2D CNN and 3D CNN, which are merged by projecting the 3D features to image space to achieve a robust pixelwise semantic segmentation. We demonstrate the ability to segment robustly despite of the challenge of severer appearance variation caused by seasons. Future works include faster prediction time for a real-time operation.

Acknowledgements We thank the Yamaha Motor corporation for supporting this research.

References

1. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional models for semantic segmentation. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
2. Badrinarayanan, V., Kendall, A., Cipolla, R.: SegNet: A deep convolutional encoder-decoder architecture for image segmentation. [arXiv:1511.00561](https://arxiv.org/abs/1511.00561) [cs.CV] (2015)
3. Paszke, A., Chaurasia, A., Kim, S., Culurciello, E.: ENet: a deep neural network architecture for real-time semantic segmentation. [arXiv:1606.02147](https://arxiv.org/abs/1606.02147) [cs.CV] (2016)

4. Couprie, C., Farabet, C., Najman, L., LeCun, Y.: Indoor semantic segmentation using depth information. [arXiv:1301.3572](https://arxiv.org/abs/1301.3572) [cs.CV] (2013)
5. Gupta, S., Girshick, R., Arbeláez, P., Malik, J.: Learning rich features from RGB-D images for object detection and segmentation. In: Proceedings European Conference on Computer Vision (ECCV) (2014)
6. Valada, A., Oliveira, G.L., Brox, T., Burgard, W.: Deep Multispectral Semantic Scene Understanding of Forested Environments Using Multimodal Fusion. In: Proceedings International Symposium on Experimental Robotics (ISER) (2016)
7. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
8. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* **35**(8), 1915–1929 (2013)
9. Ladický, L., Sturges, P., Alahari, K., Russell, C., Torr, P.H.S.: What, where and how many? combining object detectors and CRFs. In: Proceedings European Conference on Computer Vision (ECCV) (2010)
10. Micusik, B., Košecká, J., Singh, G.: Semantic parsing of street scenes from video. *Intl J. Rob. Res. (IJRR)* **31**(4), 484–497 (2012)
11. Xiao, J., Quan, L.: Multiple view semantic segmentation for street view images. In: Proceedings IEEE Intl Conference on Computer Vision (ICCV) (2009)
12. Simonyan, K., Zisserman A.: Very deep convolutional networks for large-scale image recognition. [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) [cs.CV] (2014)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. [arXiv:1512.03385](https://arxiv.org/abs/1512.03385) [cs.CV] (2015)
14. Munoz, D., Bagnell, J.A., Hebert, M.: Co-inference for multi-modal scene analysis. In: Proceedings European Conference on Computer Vision (ECCV) (2012)
15. Newman, P., et al.: Navigating, recognizing and describing urban spaces with vision and lasers. *Intl J. Rob. Res. (IJRR)* **28**(11–12), 1406–1433 (2009)
16. Cadena, C., Košecká, J.: Semantic segmentation with heterogeneous sensor coverages. In: Proceedings IEEE Intl Conference on Robotics and Automation (ICRA) (2014)
17. Alvis, C.D., Ott, L., Ramos, F.: Urban scene segmentation with laser-constrained CRFs. In: Proceedings IEEE/RSJ Intl Conference on Intelligent Robots and Systems (IROS) (2016)
18. Gupta, S., Arbeláez, P., Malik, J.: Perceptual organization and recognition of indoor scenes from RGB-D images. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2013)
19. Maturana, D., Scherer, S.: 3D convolutional neural networks for landing zone detection from LiDAR. In: Proceedings IEEE Intl Conference on Robotics and Automation (ICRA) (2015)
20. Scherer, S., Chamberlain, L.J., Singh, S.: Online assessment of landing sites. In: Proceedings AIAA Infotech@Aerospace (2010)
21. Amanatides, J., Woo, A.: A fast voxel traversal algorithm for ray tracing. In: Proceedings Eurographics (1987)

StalkNet: A Deep Learning Pipeline for High-Throughput Measurement of Plant Stalk Count and Stalk Width

Harjatin Singh Baweja, Tanvir Parhar, Omeed Mirbod and Stephen Nuske

Abstract Recently, a body of computer vision research has studied the task of high-throughput plant phenotyping (measurement of plant attributes). The goal is to more rapidly and more accurately estimate plant properties as compared to conventional manual methods. In this work, we develop a method to measure two primary yield attributes of interest; stalk count and stalk width that are important for many broad-acre annual crops (sorghum, sugarcane, corn, maize for example). Prior work of using convolutional deep neural networks for plant analysis has either focused on object detection or dense image segmentation. In our work, we develop a novel pipeline that accurately extracts both detected object regions and dense semantic segmentation for extracting both stalk counts and stalk width. A ground-robot called the Robotanist is used to deploy a high-resolution stereo imager to capture dense image data of experimental plots of Sorghum plants. We ground-truth validate data extracted using two humans who assess the traits independently and we compare both accuracy and efficiency of human versus robotic measurements. Our method yields R-squared correlation of 0.88 for stalk count and a mean absolute error of 2.77 mm where average stalk width is 14.354 mm. Our approach is 30 times faster for stalk count and 270 times faster for stalk width measurement.

1 Introduction

With a growing population and increasing pressure on agricultural land to produce more per acre there is a desire to develop technologies that increase agricultural output [1]. Work in plant breeding and plant genomics has advanced many varieties of crop through crossing many varieties and selecting the highest performing. This process however has bottlenecks and limitations in terms of how many plant varieties can be accurately assessed in a given timeframe. In particular plant traits of interest; such as stalk count and stalk width are tedious and error prone when performed manually.

H. S. Baweja (✉) · T. Parhar · O. Mirbod · S. Nuske
The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue,
Pittsburgh 15213, USA
e-mail: harjatis@andrew.cmu.edu

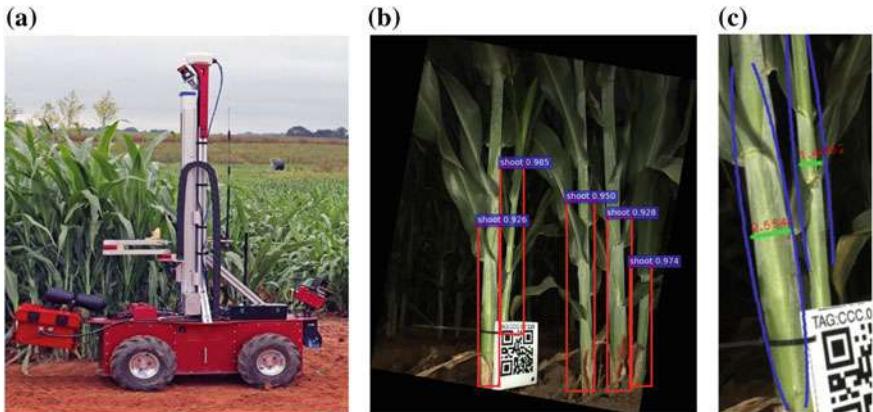


Fig. 1 a The Robotanist [2] mobile ground robot. Stereo imager mounted at back of vehicle (far right of image), b example image of stalk detection c example image of stalk width estimation

Robotics, computer vision and machine learning promise to expedite the process of assessing plant metrics and more precisely identify high performing varieties. This “high-throughput” plant phenotyping has the potential to extract measurements of plants 30 times faster than humans.

We are developing a ground robot called the Robotanist [2] that is capable of navigating tightly spaced rows of plants with cameras and sensors that can assess plant performance. The robot is equipped with machine vision stereo cameras with high-powered industrial flashes to produce quality imagery suitable for extracting robust and precise plant measurements. Figure 1a shows the camera setup where the camera is mounted at bottom right of the robot, b shows sample stalk detections stalk count and c shows stalk width measurement results respectively.

In this paper we present a new image processing pipeline the leverages both recent deep convolutional neural networks for object detection and also semantic segmentation networks together to output both stalk count and stalk width. The combination of networks together provides more precise and accurate extraction of stalk contours and therefore more reliable measurement of stalk width.

We collect data at two separate plant breeding sites one in South Carolina, USA and the other in Puerto Vallarta, Mexico and use one dataset for training our networks and the other dataset we ground truth using manually extracted measurements. We study both the efficacy and efficiency of manual measurements by using two humans to independently measure sets of plants and comparing accuracy and also total time taken per plot to measure attributes. We then compare the human measurements to the robot measurements for validation.

2 Related Work

Machine learning researchers have been making efforts to oust traditional plant phenotyping techniques by adapting neoteric algorithms to work on ground data. A number of such attempts have produced promising results. Singh et al. [3] provide a comprehensive study on how various contemporary ML algorithms can be used as building blocks for high throughput stress phenotyping. Drawing inspiration from traditionally used visual cues to estimate plant health, crop yield etc., Computer Vision has been the mainstay of most Artificial Intelligence based phenotyping initiatives [4]. Our group amongst several research groups is investigating the applications of computer vision to push the boundaries of crop yield estimation [5, 6] and phenotyping. Jimenez et al. [7] provide an overview of many such studies.

Recent advances in deep learning have induced a paradigm shift in several areas of Machine Learning, especially Computer Vision. With deep learning architectures producing state-of-the-art performances in almost all major computer vision tasks such as image recognition [8], object detection [9] and semantic segmentation [10]; it was only a matter of time for researchers to use these for image based plant phenotyping. A broad variety of studies ranging from yield estimation [11, 12], to plant classification [13], to plant disease detection have been conducted [14].

Pound et al. [15] propose using vanilla CNNs (Convolutional Neural Networks) to detect plant features such as root tip, leaf base, leaf tip etc. Though the results achieved are impressive, the images used are taken in indoor environments thus escaping the challenges of field environments such as occlusion, varying lighting amongst several others. Considering work on phenotype metric of interest- stalks, traditional image processing based approaches tailored for a particular task do well on specific data set but fail to generalize [16, 17]. 3D reconstruction based approaches show promising results, however are almost impossible to reproduce in cluttered field environments [18]. Bargoti et al. [19] provide a pipeline for trunk detection in apple orchards. The pipeline uses Hough Transforms on LIDAR data to initialize pixel wise dense segmentation into a hidden-semi Markov model (HSMM). Hough transform proves to be a coarse initializer for intertwined growth environments with no apparent gaps between adjacent plants.

There has not been much work if any, combining deep learning based state-of-the-art object detection and semantic segmentation for high throughput plant phenotyping. Our work uniquely combines the linchpins in object detection (Faster-RCNN) and semantic segmentation (FCN) for plant phenotyping to get an accuracy that is close to that of human at staggeringly high speeds.

3 Overview of Processing Pipeline

The motivation behind the work was to come up with a high throughput plant phenotyping computer vision based approach that is agnostic to changes in the field

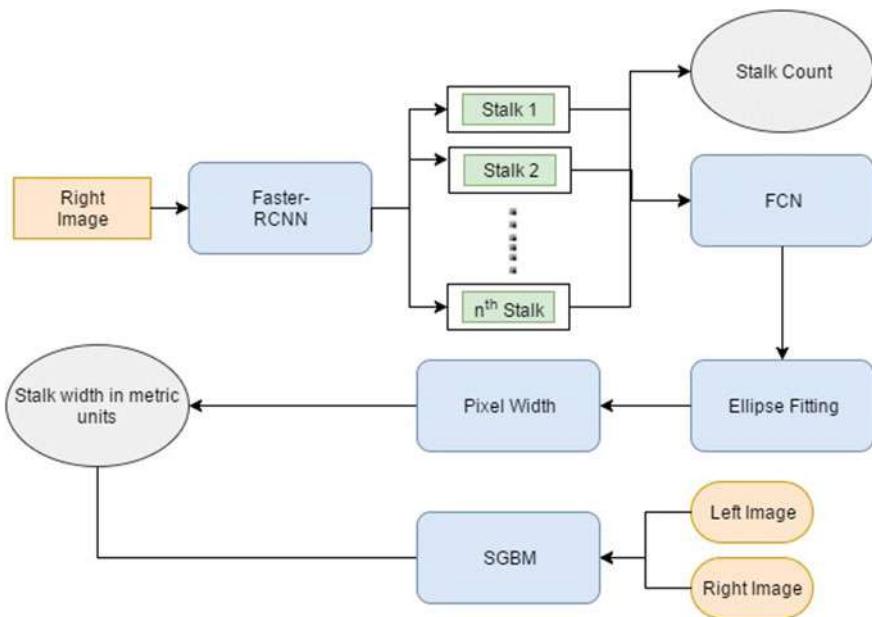


Fig. 2 Overview of the stalk count and width calculation pipeline

conditions and settings such as varying lighting conditions, occlusions etc. Figure 2 shows the overview of the data-processing pipeline, used by our approach. The faster RCNN takes one of the stereo pair images as its input and produces bounding boxes, each for one stalk. These bounding boxes are extracted from the input image (also called as snips) and fed to the FCN, one at a time. The FCN outputs a binary mask, classifying each pixel as either belonging to stalk or the background. To this mask ellipse are fitted, to the blobs in the binary mask, by minimizing the least-square loss of the pixels in the blob [20]. One snip may have multiple ellipses in case of multiple blobs. The ellipse with the largest minor axis is used for width calculation. The minor axis of this ellipse gives us the pixel width of the shoot in the current snip. The corresponding pixels in the disparity map are used to convert this pixel width into metric units.

The whole pipeline takes on an average 0.43 s to process one image, on a GTX 970 GPU. This can make the data-processing on the fly for systems that collect data at 2 Hz.

3.1 SGBM

The stereo pair was used to generate to a disparity map, using SGBM [21] in OpenCV. This was used to get metric measurements from the pixel dimension. It was also used

to calculate the average distance of the plant canopy from the sensor, it was converted into field of view in metric units, so that the estimated stalk count and stalk diameter can be converted into estimated stalk count per meter and stalk diameter per meter respectively.

3.2 Faster RCNN

Fast-RCNN (Fig. 3) by Girshick uses a VGG-16 convnet architecture as feature detector. The network takes pre-computed proposals from images and classifies them into object categories and regresses a box around them. Because the proposals are not computed over the GPU, there is a bottleneck at computing proposals. Faster-RCNN by Girshick et al. is an improvement over the Fast-RCNN, where there is a separate convolution layer that predict object proposals based on the features form the activation of the last layer of the VGG-16 network, called Region Proposal network (RPN). Since the region proposal network is a convolution layer, followed by fully connected layers, it is implemented over GPU, making it almost an order of magnitude faster than Fast-RCNN.

One drawback of the Faster RCNN is the use of non-maximal suppression (NMS) over the proposed bounding boxes. Thus, highly overlapping instances of objects might not be detected, due to NMS rejection. This problem is even severe in highly occluding field images. It was overcome by simply rotating the images by 90° so that the erectness of the stalks may be used to draw tightest possible bounding boxes. We finetuned a pre-trained Faster-RCNN with 2000 bounding boxes. Figure 4 shows sample detections.

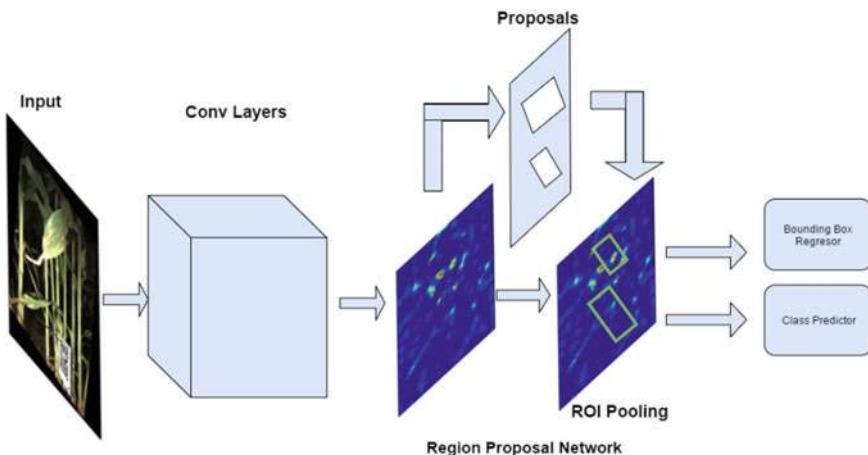


Fig. 3 Faster-RCNN architecture used for stalk detection

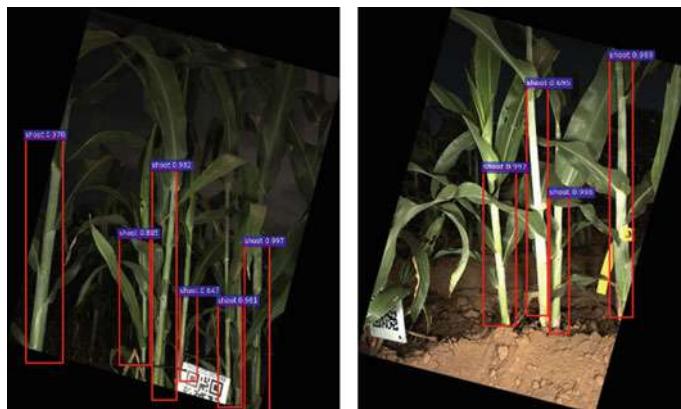


Fig. 4 Example stalk detections using Faster-RCNN

3.3 Fully Convolutional Network (FCN)

FCN (Fig. 5) is a CNN based end to end architecture that uses down-sampling (Convolutional Network) followed by up-sampling (Deconvolutional Network) to take image as input and produce a semantic mask as output.

The snips of stalks detected by Faster-RCNN are sent to FCN for semantic segmentation which by virtue of its fully convolutional architecture can account for different sized incoming image snips. We chose to send Faster-RCNN's output to FCN as input instead of raw image. Output bounding boxes always contain only one stalk and thus FCN is only required to do a binary classification into two classes, namely: stalk and background without having to do instance segmentation also. Our hypothesis was that this would make FCNs job a lot easier and would thus require lesser data to finetune a pretrained version. This hypothesis is validated by results presented in a latter section. We finetuned a pre-trained FCN with just 100 dense

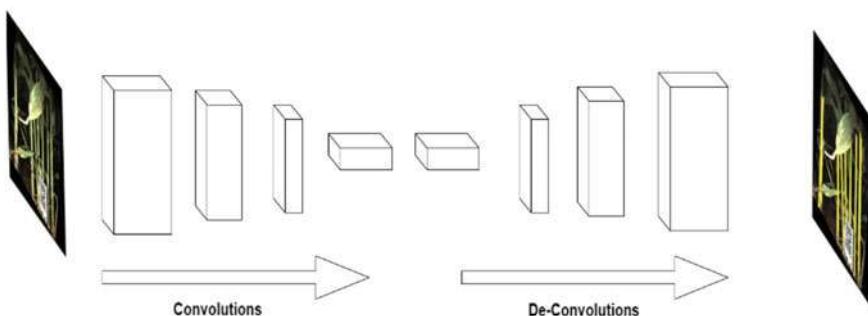
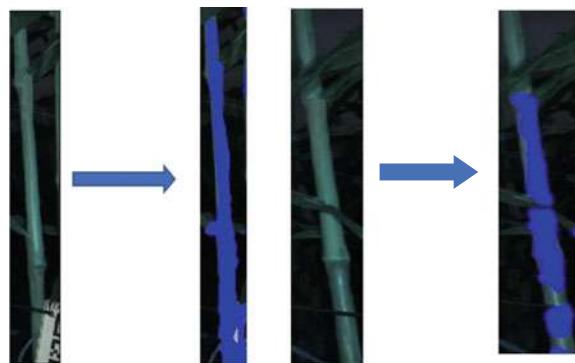


Fig. 5 Fully convolutional network architecture used for dense stalk segmentation

Fig. 6 Sample snipped bounding box input to segmented stalk output



labeled detected stalk outputs from Faster-RCNN. Sample input to output of FCN is shown in Fig. 6.

3.4 Stalk Width Estimation

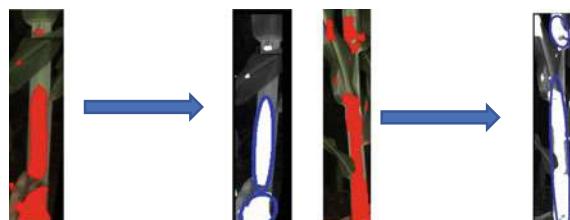
Once the masks have been obtained, for each of the snippets, ellipses are fitted to each blob of the connected contours of the mask. The ellipses are fitted to minimize the following objective:

$$\varepsilon^2(\theta) = \sum_{i=1}^n F(\theta_i, x_i)^2$$

where, $F(\theta; x) = \theta_{xx}x^2 + \theta_{yy}y^2 + \theta_{xy}xy + \theta_x x + \theta_y y + \theta_0$, is the general equation of conics in 2 dimensions. The objective is to find the optimal value of θ such that we get the best fitting conic over a given set of points. We use OpenCV's inbuilt optimizers to find best fitting ellipses. Figure 7 shows the ellipses fitted to the output mask of the FCN.

Ellipse is fitted to the contours of the blob, so that the minor axis can serve as a starting point for width estimation of the stalk. For the same reason, a simple convex hull fitting was not performed. The minor axes of all the ellipses are then trimmed to

Fig. 7 Result of ellipse fitting on mask output of FCN used for estimating stalk width



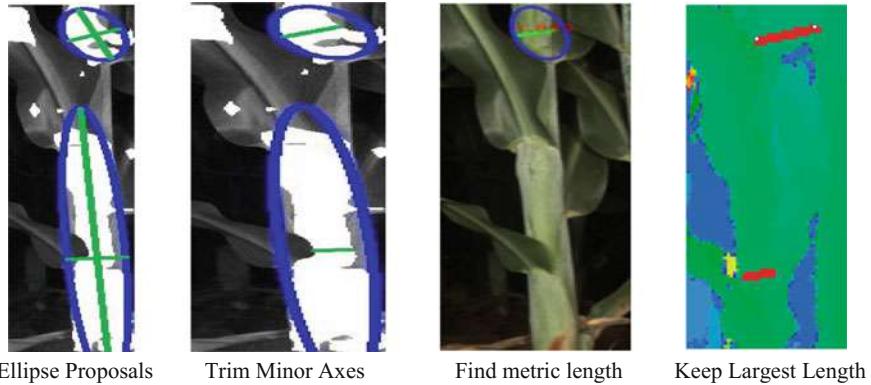


Fig. 8 Stalk width estimation pipeline

make sure they lie over the FCN mask. From the trimmed line segments, any segment that might have a slope of greater than 30° is rejected. The remaining line segments are projected on to the disparity map, so that the pixel width can be converted to the width in metric units, as per Algorithm 1. The line segment with the greatest metric width is selected as the width for the stalk in the current snip. The reason behind choosing max width over others is to get rid of the segments proposals that might have leaf occlusions.

Algorithm 1: Stalk width calculation in metric units

- For each line segment l
 - For each end points (x, y) and (x', y') on l
 - $d = \text{Disparity}(x, y)$
 - $d' = \text{Disparity}(x', y')$
 - $Z - Z' = (f * b) * \left(\frac{1}{d} - \frac{1}{d'}\right)$
 - $X - X' = (x - x') * Z/f$
 - $Y - Y' = (y - y') * Z/f$
 - $\text{width} = \sqrt{(X - X')^2 + (Y - Y')^2 + (Z - Z')^2}$

Hence the overall procedure for width estimation, Fig. 8, can be summarized in the following steps

Algorithm 2: Steps for stalk width estimation

- $\text{widths} := \text{empty list}$
- For all fitted ellipses
 - If $-30^\circ < \text{minor axis slope} < 30^\circ$
 - trim minor axis to fit mask
 - Find the metric length of the line segment as per algorithm 1.
 - Append the metric length to widths list
- $\text{Width} = \max(\text{widths})$

Figure 9 shows the final results for stalk width estimation pipeline.

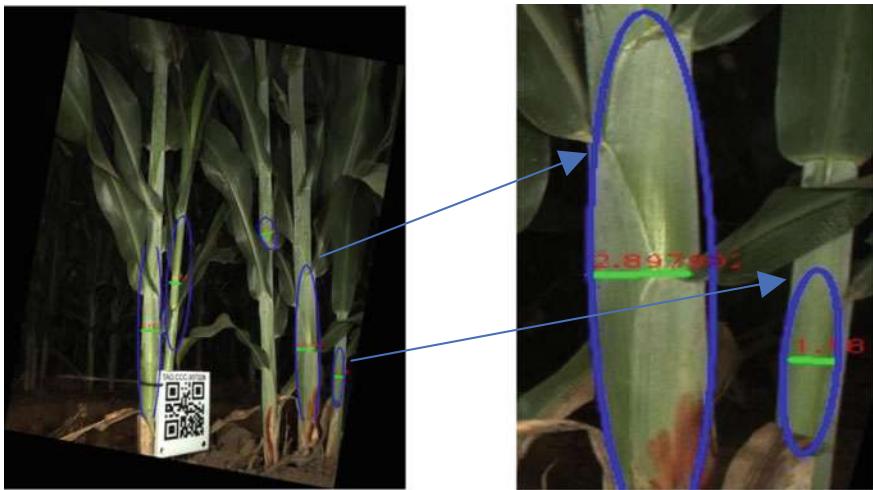


Fig. 9 Stalk width estimation results

4 Results

4.1 Data Collection

Image data was collected in July 2016, in Pendleton, South Carolina using the Robo-*tanist* platform. The algorithms were developed on this data. To test the algorithm impartially, another round of data collection with extensive ground truthing was done in February, 2017 in Cruz Farm, Mexico. The images were collected using a 9 MP stereo-camera pair with 8 mm focal length, high power flashes triggered at 3 Hz by Robot Operating System (ROS). The sensor was driven at approximately 0.05 m/s. Distance of approximately 0.8 m was maintained from the plant growth. Figure 10a shows the *Robo-*tanist** collecting data in Pendleton, South Carolina, Fig. 10b shows the custom image sensor mounted on the robot.

Each row of plant growth at Cruz Farm is divided into several 7 ft ranges separated by 5 ft alleys. To ground truth stalk count data, all stalks were counted in 29 ranges by two individuals separately. The mean of these counts was taken as the actual ground truth. Similarly, for width calculations, QR tags were attached to randomly chosen stalks for ground truth registration in images. The width of these stalks at height of 12 in. (30.48 cm) and 24 in. (60.96 cm) from the ground was also measured by two individuals separately using Vernier Calipers of 0.01 mm precision. Humans at an average took 210 s to count the stalks in each range and an average of 55 s to measure width of each stalk. Each range at an average has 33 stalks, so on an average it takes 33 min to measure stalk widths of entire range.



Fig. 10 **a** The Robotanist collecting data in Pendleton, South Carolina **b** Custom stereo imaging sensor used for imaging stalks

4.2 Results for Stalk Count

Faster-RCNN was trained with 400 images with approximately 2000 bounding boxes using alternate optimization strategy. RPN and regressor are trained for 80000 and 40000 iterations respectively in first stage and 40000 and 20000 iterations in the second stage using base learning rate of 0.001 and a step decay of 0.1 every 60000 iterations for RPN and 30000 iterations for regressor. Best test accuracies were achieved by increasing the number of proposals to 2000 and the number of anchor boxes to 21 using different scaling ratios with NMS threshold of 0.2. Due to inability to get accurate homography for data collected at 3 Hz, we resorted to calculating stalk-count/meter using stereo data and do the same for ground truth stalk counts which were collected from ranges, each of constant length 7 ft (2.134 m). Figure 11 shows the R-squared correlation for results of 0.88.

To put the results into perspective, attempting to normalize counts using image widths from stereo data may induce some error as this data is sometimes biased towards stalk count towards the start and end of each range where the mount vehicle is slowed down. Also, there is a little inherent uncertainty in the count data. There are tillers (stems produced by grass plant) growing at the side of some stalks which are hard to discern from stalks with stunted growth. To better understand this, we observe in Fig. 12 that there is a small variation in ground truth stalk counting between two humans as well. The R-squared of Human1's count versus Human2's count should be 1 in an ideal scenario but that is not the case.

4.3 Results for Stalk Width

We plot the stalk width values as measured by Human1, Human2 and our algorithm at approximately 12 in. (30.48 cm) from the ground. At the time of data collection,

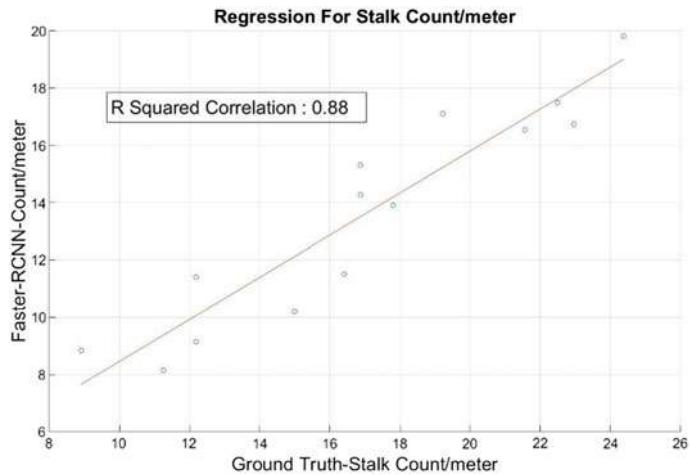


Fig. 11 Linear regression for human stalk count/meter versus Faster-RCNN's count/meter

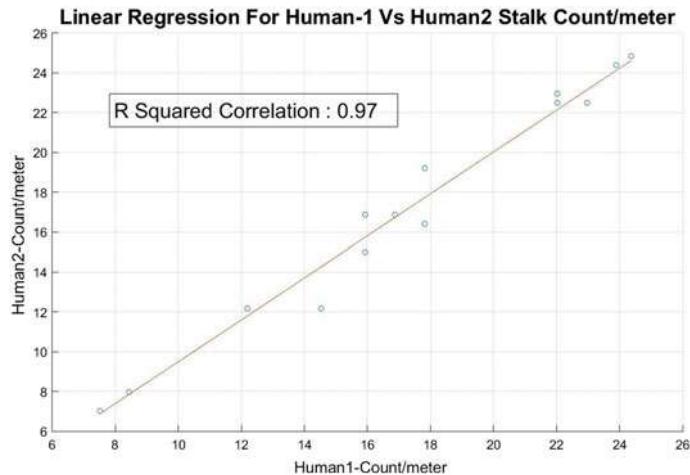


Fig. 12 Linear regression for Human1's stalk count/meter versus Human2's stalk count/meter

it was made sure that a part of ground is visible in every image. This allows us get stalk width at the desired height from the image data. This step is important as there is prevalent tapering in stalk widths as we go higher up from the ground. Figure 13 shows the widths of each ground trothed stalk as per both humans and algorithm.

Since there is a discernible difference in measurements of the two humans, we considered the mean of the two readings as actual ground truth. The mean width of stalks as per this ground truth is 14.354 mm. The mean absolute error between readings of Human1 and Human2 is 1.639 mm and the mean absolute error between readings from human ground truth and algorithm is 2.76 mm. The error can be attributed to

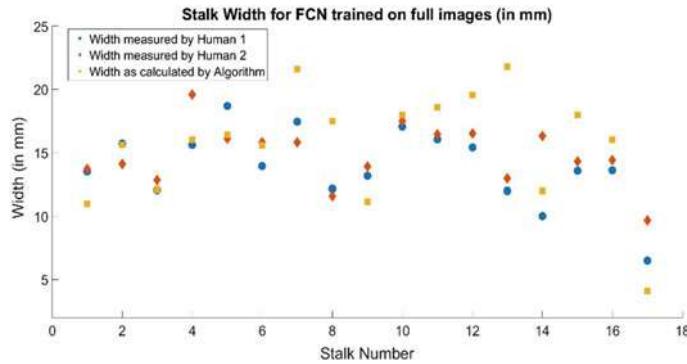


Fig. 13 Width measured by Human1, Human2 and algorithm

rare occlusions that force algorithm to calculate height at a location other than 12 in. (30.48 cm) from the ground. We suspected this as a possibility and thus measure stalk widths at 2 locations during the ground truthing process: at 12 in. (30.48 cm) and 24 in. (60.96 cm) above the ground. Calculations from this data tell us that there was 0.405 mm/in. mean tapering on the measured stalks as we went up from 12 in. (30.48 cm) to 24 in. (60.96 cm).

To validate our hypothesis that providing faster-RCNN’s output bounding boxes as inputs to FCN would require lesser dense labeled data to train it. We trained another FCN on densely labeled complete images. This FCN was trained with more than twice the number of densely labeled stalk data (finetuned with approximately 250 densely labeled stalks) than the previous FCN (finetuned with on approximately 100 densely labeled stalks). Even after assuming perfect bounding boxes around it for instance segmentation, the mean absolute error of this FCN was 3.868 mm for width calculation, which is higher than its predecessor having a mean absolute error of 2.76 mm.

4.4 Time Analysis

Table 1 shows the time comparisons of Humans versus algorithm for an average plot. Each plot has approximately 33 stalks and is about 2.133 m in length. We observe that Algorithm is 30 times faster as compared to humans for stalk counting and 270 times faster than human for stalk width calculation.

Table 1 Time analysis for measuring one experimental plot

	Human1 (min)	Human2 (min)	Robot (s)
Stalk count	3.33	3.66	6.5
Stalk width	29	30	
Total	32.33	33.66	6.5

5 Conclusion

We have shown the strength of coupling deep convolutional neural networks together to achieve a high quality pipeline for both object detection and semantic segmentation. With our novel pipeline we have demonstrated accurate measurement of multiple plant attributes.

We find the automated measurements are accurate to within 10% of human validation measurements for stalk count and measure stalk width with 2.76 mm on average. Ultimately though, we identify that the human measurements are 30 times slower than the robotic measurements for count and 270 times slower for measuring stalk width over an experimental plot. Moreover, when translating the work to large scale deployments, that instead of 30 experimental plots are 100's or 1000's of plots in size, it is expected that the human measurements become less accurate and logically tough to measure in timely fashion during tight growth stage time windows.

In future work we plan to integrate more accurate positioning to merge multiple views of the stalks into more accurate measurements of stalk-count and stalk-width.

References

- United Nations Department of Economic and Social Affairs Population Division.: <http://www.unpopulation.org>. Accessed 10 Oct 2014
- Mueller-Sim, T., et al.: The Robotanist: a ground-based agricultural robot for high-throughput crop phenotyping. In: IEEE International Conference on Robotics and Automation (ICRA), Singapore, Singapore, May 29–June 3 2017
- Singh, A., et al.: Machine learning for high-throughput stress phenotyping in plants. *Trends Plant Sci.* **21**(2), 110–124 (2016)
- Tsaftaris, S.A., Minervini, M., Scharr, H.: Machine learning for plant phenotyping needs image processing. *Trends Plant Sci.* **21**(12), 989–991 (2016)
- Sugiura, R., et al.: Field phenotyping system for the assessment of potato late blight resistance using RGB imagery from an unmanned aerial vehicle. *Biosyst. Eng.* **148**, 1–10 (2016)
- Pothen, Z., Nuske, S.: Automated assessment and mapping of grape quality through image-based color analysis. *IFAC-PapersOnLine* **49**(16), 72–78 (2016)
- Jimenez, A.R., Ceres, R., Pons, J.L.: A survey of computer vision methods for locating fruit on trees. *Trans. ASAE-Am. Soc. Agric. Eng.* **43**(6), 1911–1920 (2000)
- He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
- Ren, S., et al.: Faster r-cnn: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (2015)

10. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015)
11. Sa, I., et al.: On visual detection of highly-occluded objects for harvesting automation in horticulture. In: ICRA (2015)
12. Hung, C., et al.: Orchard fruit segmentation using multi-spectral feature learning. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE (2013)
13. McCool, C., Ge, Z., Corke, P.: Feature learning via mixtures of dcnn's for finegrained plant classification. In: Working Notes of CLEF 2016 Conference (2016)
14. Mohanty, S.P., Hughes, D.P., Salathé, M.: Using deep learning for image-based plant disease detection. *Front. Plant Sci.* **7** (2016)
15. Pound, M.P., et al.: Deep machine learning provides state-of-the-art performance in image-based plant phenotyping. *bioRxiv*, 053033 (2016)
16. Mohammed Amean, Z., et al.: Automatic plant branch segmentation and classification using vesselness measure. In: Proceedings of the Australasian Conference on Robotics and Automation (ACRA 2013). Australasian Robotics and Automation Association (2013)
17. Baweja, H., Parhar, T., Nuske, S.: Early-season vineyard shoot and leaf estimation using computer vision techniques. ASABE (2017) (accepted)
18. Paproki, A., et al.: Automated 3D segmentation and analysis of cotton plants. In: 2011 International Conference on Digital Image Computing Techniques and Applications (DICTA). IEEE (2011)
19. Bargoti, S., et al.: A pipeline for trunk detection in trellis structured apple orchards. *J. Field Robot.* **32**(8), 1075–1094 (2015)
20. Fitzgibbon, A.W., Fisher, R.B.: A buyer's guide to conic fitting. DAI Research Paper (1996)
21. Hirschmuller, H.: Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.* **30**(2), 328–341 (2008)

Learning Models for Predictive Adaptation in State Lattices

Michael E. Napoli, Harel Biggie and Thomas M. Howard

Abstract Approaches to autonomous navigation for unmanned ground vehicles rely on motion planning algorithms that optimize maneuvers under kinematic and environmental constraints. Algorithms that combine heuristic search with local optimization are well suited to domains where solution optimality is favored over speed and memory resources are limited as they often improve the optimality of solutions without increasing the sampling density. To address the runtime performance limitations of such algorithms, this paper introduces Predictively Adapted State Lattices, an extension of recombinant motion planning search space construction that adapts the representation by selecting regions to optimize using a learned model trained to predict the expected improvement. The model aids in prioritizing computations that optimize regions where significant improvement is anticipated. We evaluate the performance of the proposed method through statistical and qualitative comparisons to alternative State Lattice approaches for a simulated mobile robot with nonholonomic constraints. Results demonstrate an advance in the ability of recombinant motion planning search spaces to improve relative optimality at reduced runtime in varyingly complex environments.

1 Introduction

Recent advances in sensors, computing, and algorithms for perception, planning, and control have allowed unmanned ground vehicles (UGVs) to be deployed in increasingly challenging and harsh conditions. Applications such as planetary exploration, nuclear inspection, and resource extraction could require robots to traverse environments that are dangerous or difficult for human operators and situations where

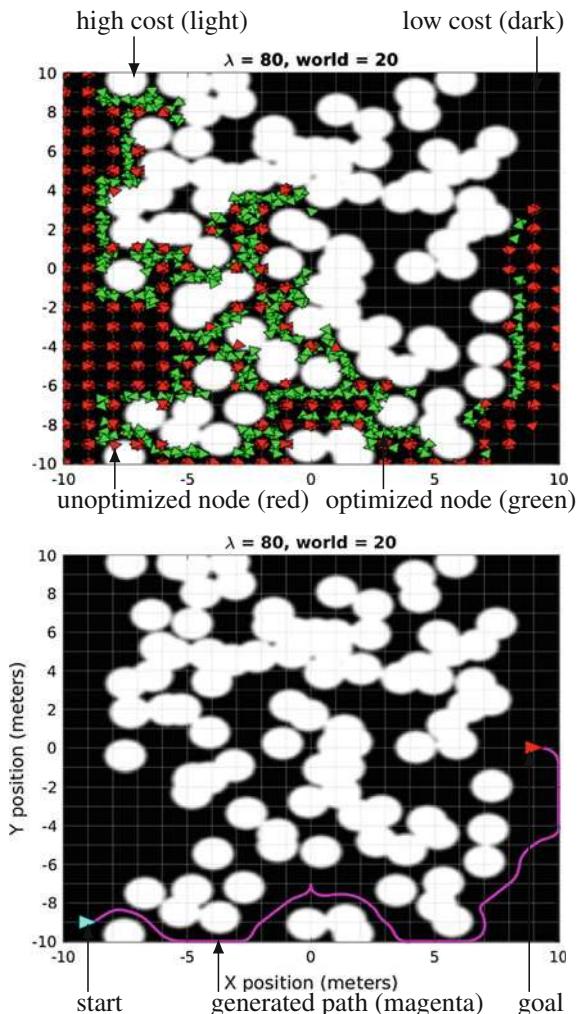
M. E. Napoli (✉) · H. Biggie · T. M. Howard
University of Rochester, Rochester, NY 14627, USA
e-mail: mnnapoli@ur.rochester.edu

H. Biggie
e-mail: hbiggie@u.rochester.edu

T. M. Howard
e-mail: thomas.howard@rochester.edu

assistance (if at all available) may be limited to remote teleoperation of the platform. In these situations, the UGV is dependent on its own autonomy to reliably balance the minimization of vehicle risk with energy consumption and achieve mission success. Often, risk avoidance in these fields is paramount and optimality of achieved planning solutions is strongly desired. In environments with many homotopically distinct classes, graph and sampling approaches have been employed successfully to find near optimal solutions. However, these algorithms provide higher fidelity plans often at the expense of computational runtime. This compromise between optimality and efficiency is fundamental to mobile robot motion planning and many algorithms feature a balance between the two.

Fig. 1 An illustration of optimized and unoptimized nodes expanded during heuristic search (above) and the path generated by the PASL algorithm (below)



An approach to motion planning that considers memory, differential, and environmental constraints involves modeling the continuum of actions and states in a recombinant search space graph structure known as a State Lattice (SL) [1]. Such a formulation converts the motion planning problem into a graph search which can be solved using a variety of existing algorithms. It has also been shown that, in sufficiently complex environments, applying local optimization can improve the optimality of generated solutions [2]. A limitation of this approach, referred to as the Adaptive State Lattice (ASL), is that it indiscriminately optimizes every node that is encountered in the search space. This paradigm results in cycles spent optimizing regions that are not complex or for which optimization is too difficult and are unlikely to impact the resulting trajectory. This paper introduces a novel extension of the ASL, referred to as the Predictively Adapted State Lattice (PASL) which exploits a learned predictive model to adapt the representation of the search space by anticipating the amount of improvement to be obtained when applying optimization over local regions (Fig. 1). The addition of this model allows the PASL to focus optimization on regions where sufficient improvement is expected. The result from the PASL is a feasible solution with comparable optimality and memory requirements to the ASL approach, but with reduced computational runtime.

This paper presents three contributions, the first of which is the algorithm that incorporates a learned model for predictive adaptation in adaptive state lattices (PASL). The second is a statistical evaluation of the performance of PASL in both runtime and relative optimality, a metric which refers to the nearness of a solution to the global optimum, in a selection of randomly generated obstacle fields. This study assesses the PASL against a comparative baseline of SL, ASL and a heuristic based approach to selective adaptation referred to as the Selectively Adaptive State Lattice (SASL) [3]. The final contribution is a qualitative comparison between the solutions obtained with all four algorithms, which visualizes their performance in a random world of nominal complexity.

2 Related Works

Decades of research and advancements have led to a myriad of algorithms which address the motion planning problem in a variety of ways. These algorithms are often classified by the mechanism they exploit to sample the continuum of states and actions. One such classification are probabilistic approaches which leverage iterative random sampling and are often probabilistically complete. The Probabilistic Roadmap (PRM) is a probabilistic approach which randomly samples points in the admissible robot configuration space [4]. Subsequently, the sampled points are connected using a fast local planner. Rapidly-exploring Random Trees (RRTs) are another probabilistic sampling approach, which iteratively sample the state space continuum and expand a tree structure towards these sampled points until a connection to a goal region is made [5]. Extensions of this algorithm that bias sampling towards the goal region and utilize a bidirectional variant were shown to lead to faster

convergence. Further research has been explored to extend the representation within the RRT from a binary (admissible/inadmissible) to continuous representation which considers path optimality [6, 7]. It was proven that the RRT algorithm converges to a suboptimal solution and a new extension was proposed (called RRT*) which almost surely converges to the optimum [8]. Additional methods have been developed to include homotopy aware approaches, bi-directional RRT* variants and application of local optimization techniques which improve global optimality [9–11].

A limitation of approaches based on probabilistic sampling is the memory efficiency of the resulting search space. Another family of motion planning algorithms, referred to as the State Lattice (SL), is a recombinant search space approach that constructs a graph structure by regularly sampling the mobile robot state-action space [1]. These algorithms, which have been applied for navigation in autonomous automobiles [12] and planetary rovers [13], represent samples in the state space as nodes on the recombinant lattice and the edges between them are described by feasible actions that pre-encode system dynamics. The state space utilized in the original SL work consists of position, heading and curvature such that the state vector may be represented as $\mathbf{x} = [x \ y \ \theta \ k]$. Actions used in this work are parameterized functions (such as clothoids or polynomial spirals) of curvature and linear velocity and are generated using constrained optimization techniques [14, 15]. The state lattice is stored compactly in a structure referred to as the *control set* which contains a pre-selected sampling of states and actions describing the transitions from each node. An advantage of the SL is that nodes do not need to be allocated until their parent node is *expanded* (placed on the closed list) during the graph search. This feature reduces the memory requirements to be predominantly the storage of instantiated nodes. Further memory reduction can be obtained while maintaining optimality by utilizing an admissible heuristic in a search algorithm. SLs are resolution complete, meaning that all possible solutions will be considered within the sampling density of the state-action space.

3 Technical Approach

One of the primary limitations of the SL is the rigidity imposed by the regular sampling of the control set, where nodes are chosen irrespective of their associated cost in the robot’s environment, resulting in many unused or unexpanded nodes sampled in high cost regions. To alleviate this limitation, an approach called the Adaptive State Lattice (ASL) was developed to optimize the sampled values of the state space before nodes are represented in an open list during heuristic search [2]. Adaptation of the representation within search has been shown to improve the performance of heuristic search in a State Lattice (shorter trajectories, faster runtime, lower memory utilization) over fixed search with a finer resolution SL in sufficiently complex environments. However, the ASL applies optimization to all nodes regardless of the potential for improvement. This can result in wasteful computations spent attempting to optimize regions where little gain is obtainable. An outline of the general search

process is shown in Algorithm 1 for the PASL and the three variants explored later in Sect. 4. Similarly to traditional graph search, expansion is performed on the top node in the open list generating child (L1) nodes. Instead of directly adding these nodes to the open list, a function is evaluated to predict which L1 nodes may benefit from adaptation. For each of the L1 nodes that exceed a particular threshold, their children (L2) nodes are generated and optimization is performed over that L1 parent. The prediction process for the SL always returns a *false* and conversely a *true* for the ASL. For completeness, the prediction step for the heuristic based SASL is outlined in Algorithm 2. The PASL prediction step is shown in Algorithm 3.

Algorithm 1: Predictive Graph Search

Input : Start node \mathbf{x}_s , goal node \mathbf{x}_g , step length α , line search parameter β , finite difference step Δ , predictive threshold (if applicable) c_{thresh} , predictive model (if applicable) net , training data mean (if applicable) $\hat{\mathbf{X}}$, training data standard deviation (if applicable) σ

Output: Trajectory of nodes $\mathbf{x}(t)$

```

1 Main
2   OPEN ←  $\mathbf{x}_s$ 
3   CLOSED ←  $\emptyset$ 
4   while  $OPEN \neq \emptyset$  do
5      $\mathbf{x}_{next} \leftarrow$  get top from OPEN
6     if  $\mathbf{x}_{next} == \mathbf{x}_g$  then
7       | return  $\mathbf{x}_g$ 
8     end
9      $\mathbf{X}_{L1} \leftarrow$  EXPAND( $\mathbf{x}_{next}$ )
10    foreach  $\mathbf{x}_{L1}$  in  $\mathbf{X}_{L1}$  do
11      if  $\mathbf{x}_{L1}$  is not in OPEN then
12        if predict( $\mathbf{x}_p, \mathbf{x}_{1:N}, m, c_{thresh}, net, \hat{\mathbf{X}}, \sigma$ ) then
13          |  $\mathbf{X}_{L2} \leftarrow$  EXPAND( $\mathbf{x}_{L1}$ )
14          | adapt( $\mathbf{x}_{L1}, \mathbf{X}_{L2}, \alpha, \beta, h$ )
15        end
16        OPEN ←  $\mathbf{x}_{L1}$ 
17      end
18    end
19    closed ←  $\mathbf{x}_{next}$ 
20    sort(OPEN)
21  end
22 end

```

3.1 Local Optimization in State Lattices

The ASL algorithm applies optimization of every instantiated node's state vector [2], where the values of the sampled state space are optimized but trajectories subject to

boundary state constraints of other nodes expressed in the state lattice. This procedure is described mathematically in Eq. 1, where \mathbf{x}_p represents the current (later referred to as the parent) node's state vector containing position, heading, curvature, linear velocity, and/or other quantities of interest. The objective being minimized, denoted as $J_{agg}(\mathbf{x}_p)$ is the aggregate control set cost of the current node expansion.

$$\underset{\mathbf{x}_p}{\text{minimize}} \quad J_{agg}(\mathbf{x}_p) \quad (1)$$

The aggregate control set cost is the sum of each of the edges connecting the current (parent) node to each of its child nodes as shown in Eq. 2 where \mathbf{x}_n denotes the n th child node's state vector and $J(\mathbf{x}_p, \mathbf{x}_n)$ represents the cost of the edge between the parent and the n th child node.

$$J_{agg}(\mathbf{x}_p) = \sum_{n=1}^N J(\mathbf{x}_p, \mathbf{x}_n) \quad (2)$$

The cost of a single edge is the sum of the n th child node's total edge path traversal time $s_{f,n}$ and integrated path cost denoted as $\mathcal{L}(\mathbf{x}_p, \mathbf{x}_n, s)$, as shown in Eq. 3 where $s_{0,n}$ and $s_{f,n}$ represent the n th child node's starting and final time respectively. The integrated path cost represents the cost of traversing a particular region in the robot's environment. In practice, the environment representation is usually defined using a discrete pixel approximation such as a cost map. Consequently, the integration in Eq. 3 is approximated numerically using the cost of pixels intersected by the edge.

$$J(\mathbf{x}_p, \mathbf{x}_n) = s_{f,n} + \int_{s_{0,n}}^{s_{f,n}} \mathcal{L}(\mathbf{x}_p, \mathbf{x}_s, s) ds \quad (3)$$

The optimization of the state vector adapts the parent node's edges to adjust the shape of the control set to conform to features in the local environment, which (in terms of the graph search) reduces the edge's cost and reflects a more optimal set of routes that would be represented by a finer representation of the search space. A visualization of the iterative optimization process applied to a simple cost map is shown in Fig. 2, highlighting how edges no longer cross high risk regions and are therefore better suited to the proximate environment.

The optimization technique in Fig. 2 is gradient descent which uses forward differencing to numerically estimate the aggregate control set cost objective gradient. Numerical estimates of the gradient are performed to enable evaluation of vehicle models that may contain complex models of dynamics and wheel-terrain interaction. When the state-action space is densely sampled, the computational overhead required to compute the gradient is consequential resulting in increased runtime of large searches. For further details about implementation and performance of the ASL method is presented in [2].

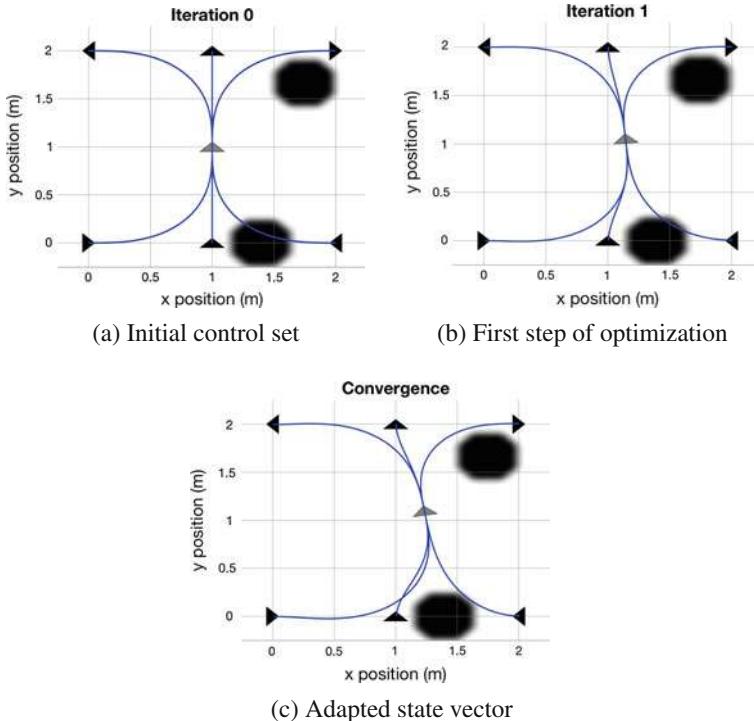


Fig. 2 State vector adaptation using gradient descent. The edges from the center parent node intersect high cost (dark) regions in the cost map. Iterative optimization adjusts the parent node's state vector until all edges traverse safer low cost regions

3.2 Heuristic Based Selective Adaptation

The application of local optimization allows the ASL to achieve higher relative optimality in many cases over the SL [2]. However, a major limitation is the indiscriminate optimization of nodes. In a statistical study, it was shown that a heuristic could be applied to selectively perform adaptation resulting in reduced runtime while providing solutions in the same homotopic class as the ASL [3]. The proposed algorithm, known as the SASL, computes the Normalized Mean Cell Cost (NMCC) of the environment patch spanned by the current node's expansion as a heuristic. This approach is outlined in Algorithm 2 and is included in the experiments discussed in Sect. 4. A threshold was chosen based on statistics collected over training data shown in Fig. 3.

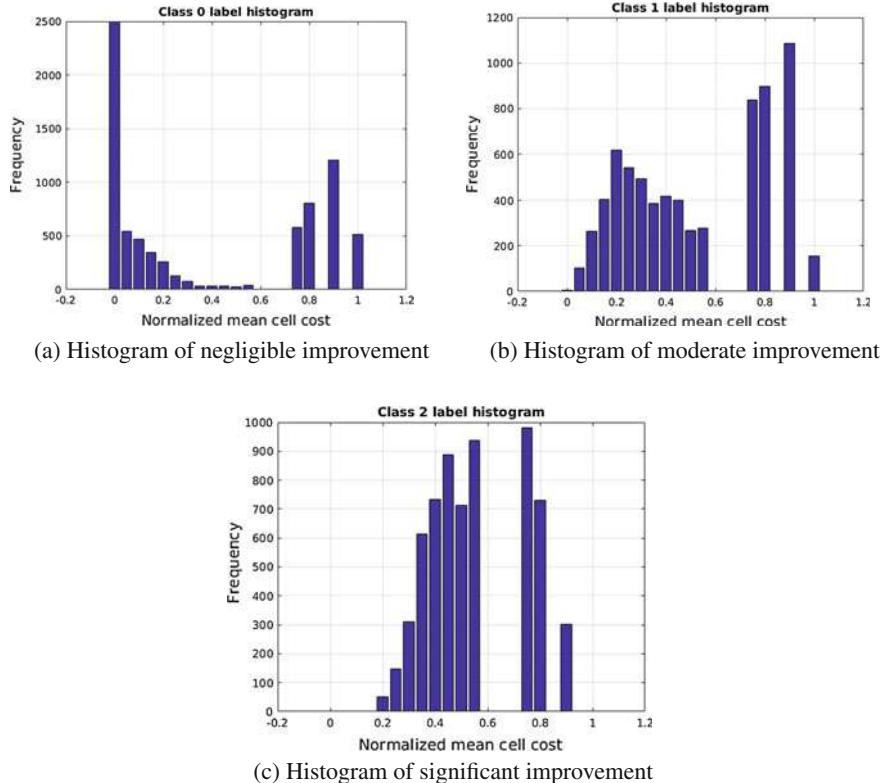


Fig. 3 Histograms of improvement vs NMCC for threshold selection. A large amount of the improvement in class 2 (significant) occurs below $\text{NMCC} = 0.6$. Additionally, this is the case for moderate and negligible (class 1 and 0 respectively) improvement. A threshold of $\text{NMCC} = 0.6$ was chosen for the SASL based on these statistics

3.3 Predictive Adaptation of Search Space Representations

The SASL approach attempts to alleviate some of the computations wasted by the application of optimization to all nodes with the ASL, however experiments demonstrate that it is difficult to design an heuristic that is both accurate and efficient. Instead, we propose the use of a predictive model learned over a selection of training data to serve as the arbiter for predictive adaptation of state vectors in the search space. The model utilized in our proposed PASL method is an artificial neural network [16] where the input layer consists of the vectorized local region pixel values augmented with the parent node's orientation (θ_p) and the free function parameters used to describe all exiting edge curvature profiles. For consistency of the evaluation presented later in Sect. 4, the pixel region is identical to the patch utilized by SASL in Sect. 3.2. An example of the feature vector is shown in Eq. 4 where m_i represents the i th cost map patch index, $k_{j,n}$ the j th parameter of the curvature function for

Algorithm 2: Prediction for Selectively Adaptive State Lattice

Input : Parent node \mathbf{x}_p , child nodes $\mathbf{x}_{1:N}$, costmap m , heuristic threshold h

Output: Boolean prediction

```

1 Predict ( $\mathbf{x}_p, \mathbf{x}_{1:N}, m, h$ )
2    $c \leftarrow 0$ 
3   for  $i$  in patch  $m(\mathbf{x}_p)$  do
4      $| c \leftarrow c + m_i$ 
5   end
6    $c \leftarrow \frac{c}{N_{cells} * MAXCOST}$ 
7   if  $c \leq h$  then
8      $|$  return true
9   end
10  else
11     $|$  return false
12  end
13 end

```

n th child node, and $s_{f,n}$ the path length of the curvature profile for the n th child node. The local regions are 41×41 pixel patches which, when augmented with the additional features, results in an input layer of 1724 features. For every node during planning, the PASL collects all appropriate features and performs adaptation only if the model output is above a chosen threshold. Algorithm 3 outlines the PASL prediction procedure.

Algorithm 3: Prediction for Predictively Adapted State Lattice

Input : Parent node \mathbf{x}_p , child nodes $\mathbf{x}_{1:N}$, costmap m , improvement threshold h , trained artificial neural network net , mean of training data \hat{X} , standard deviation of training data σ

Output: Boolean prediction

```

1 Predict ( $\mathbf{x}_p, \mathbf{x}_{1:N}, m, h$ )
2    $\theta_p \leftarrow orientation(\mathbf{x}_p)$ 
3    $U \leftarrow parameters(\mathbf{x}_p, \mathbf{x}_{1:N})$ 
4    $X \leftarrow [m \; \theta_p \; U]$ 
5    $X \leftarrow X - \hat{X}$ 
6    $X \leftarrow \frac{X}{\sigma}$ 
7    $\hat{y} \leftarrow forward(net, X)$ 
8   if  $\hat{y} \geq h$  then
9      $|$  return true
10   end
11   else
12      $|$  return false
13   end
14 end

```

$$\underline{\mathbf{x}} = [m_1 \ m_2 \dots m_n \ \theta_p \ k_{1,1} \dots s_{1,f} \ k_{2,1} \dots s_{n,f}]^T \quad (4)$$

All inputs are zero centered and normalized to unit variance (calculated exclusively over training data) prior to applying the network with an architecture containing two hidden layers of 50 and 200 nodes respectively and hyperbolic tangent sigmoid activation functions. An output layer consisting of a single node and a linear activation function is used to represent the predicted improvement in objective. All training data was collected over a series of worlds generated using the random process discussed in Sect. 4 and the resulting data was binned into improvement intervals of 50. Subsequently, the data was sub-selected for training using stratified random sampling to prevent overfitting the model. All training targets were scaled by a factor of 10 as it was empirically determined that the model was overfitting to outputs of smaller magnitude.

Prior to employing the model in planning, a five fold cross validation experiment was performed using 70, 15, 15% for training, validation, and testing data respectively to evaluate the chosen network architecture. A series of threshold values were selected and binary classification (adaptation/no adaptation) was performed on each data point using the model's predicted improvement. The True Positive Rate (TPR) of correct classifications for each threshold value was averaged over five folds and are shown in Fig. 4. The results illustrate the model's classification performance of approximately 93–100% over the spanned threshold range.

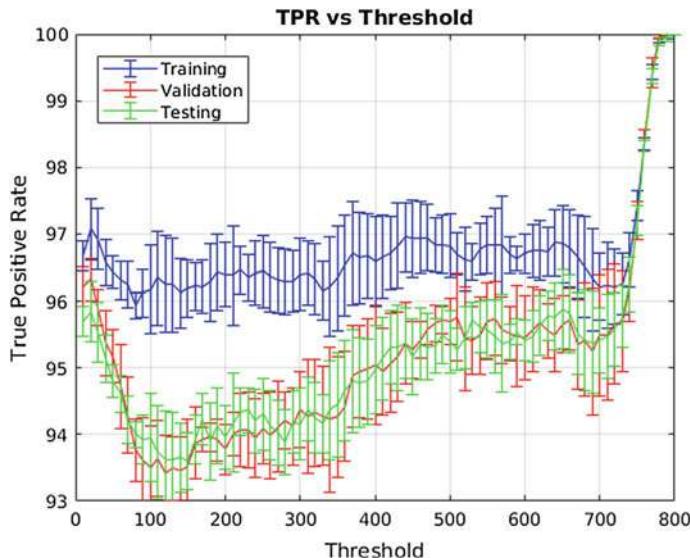


Fig. 4 Five-fold cross validation evaluation of neural network with 95% confidence

4 Experimental Design and Evaluation

Our proposed approach is focused on improving upon the performance of SL and as such, we restrict the scope of our subsequent analysis to variants of this search space representation. A statistical study was performed to evaluate the PASL and compare its performance with the SL, ASL, and SASL. All four algorithms were assessed over simulated random worlds with varying degrees of entropy, representing a mobile robot's environment. Each world contained specified regions where start and goal points were selected. These areas were common for all worlds and each algorithm was tasked with planning for every combination of world, start and goal point. The total number of plans solved for each algorithm was 32100. The simulations used in this evaluation do not include the separate set of random worlds generated strictly for model training from Sect. 3.3.

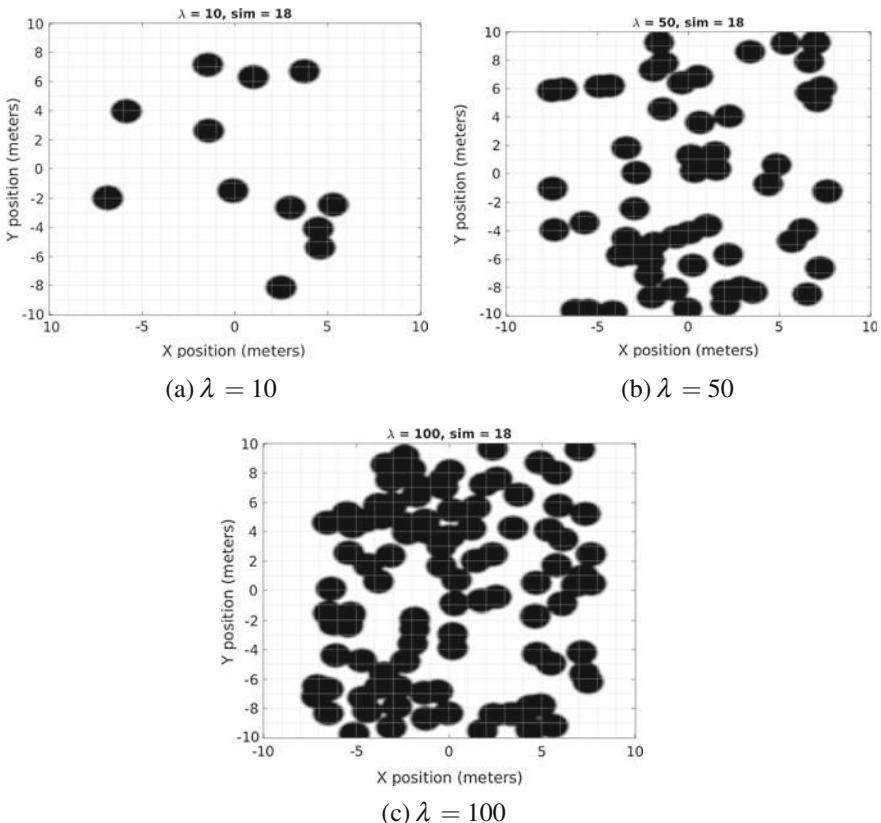


Fig. 5 Sample of random worlds generated for simulation experiments. The number of obstacles (dark regions) increases with the Poisson expected rate of occurrence (λ). The start region for the planners is in the unoccupied space on the left, whereas the goal regions are on the right

Random worlds were created using a Poisson forest procedure similar to the approach described in [17]. The number obstacles were chosen using a Poisson distribution for ten levels of expected rate of occurrence (λ). These ranged from $\lambda = 10 - 100$ and also the free space world $\lambda = 0$. To model a penalty function of proximity to obstacles, the map was blurred using a Gaussian kernel and cropped by half a meter on all sides to avoid edge effects. Each map ranged from $(-10, 10)$ meters in x, y and was sampled at a resolution of 5 centimeters. The search algorithm was forbidden from expanding into regions with maximum cost. A consequence of this is that some worlds do not have solutions in the continuum and the planning algorithms will fail. A small selection of random worlds are shown in Fig. 5 to provide a qualitative representation of the planning difficulty for a range of λ .

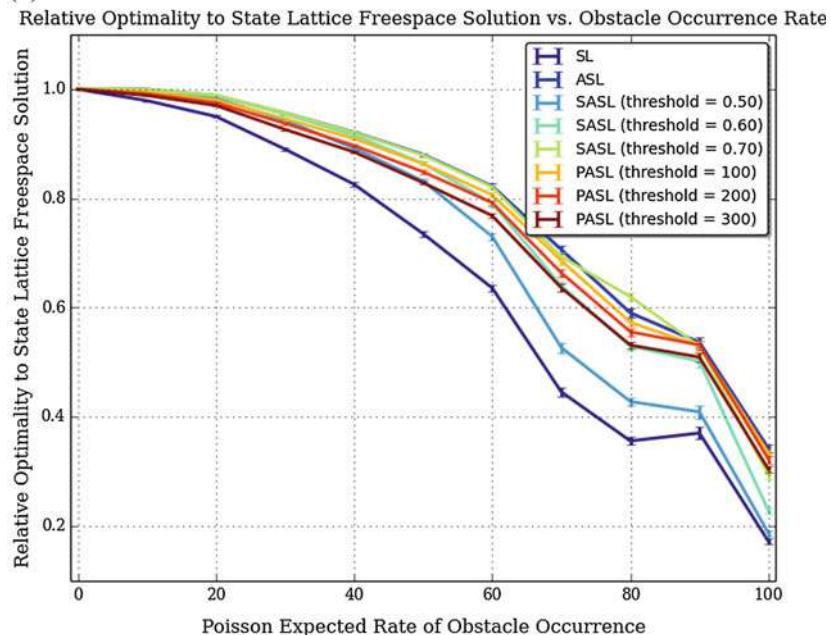
4.1 Statistical Results

The relative optimality is defined as the ratio of the free space solution cost obtained with SL and the solution cost obtained by a planner for a world with a particular λ . Relative optimality and runtime results for all algorithms are shown in Fig. 6. Runtime results were obtained using an Intel Xeon(R) CPU E5-2520 v3 2.40 GHz processor.

The trends in Fig. 6 indicate that the ASL tends to outperform the other algorithms in terms of relative optimality, but also requires the most runtime. A notable exception to this is for SASL with a heuristic threshold of 0.7. Since the adaptation only considers local regions, there is no guarantee that it will improve the global objective. Therefore, performing optimization does not always result in a better solution. For this data point, it is believed that the SASL actually benefited from not performing optimizations in some instances.

The PASL performance is relatively consistent. As the improvement threshold increases, the quality of the solution degrades, however a decrease in runtime is obtained. With an improvement threshold of 200, the PASL runtime is consistently lower than SASL with a heuristic of 0.7. Furthermore, the relative optimalities of the two algorithms are comparable in the more cluttered obstacle fields. At less cluttered obstacle densities, the SASL tends to outperform the pasl in terms of relative optimality, however at significantly increased runtime. The increased runtime for the SASL algorithm is likely due to the difficulty of setting a good threshold when using a simple heuristic. An interesting comparison is between the SASL with a threshold of 0.6 and the PASL with a threshold of 300. Although the runtime is comparable between the two algorithms in the higher obstacle density worlds, the PASL is able to maintain higher relative optimality. In this domain, it appears that the PASL predictive model outperforms the SASL at selecting nodes to optimize given the higher relative optimality. Due to the challenging nature of the planning problems for higher λ , in many domains it is worthwhile to spend computational resources to improve the solution.

(a)



(b)

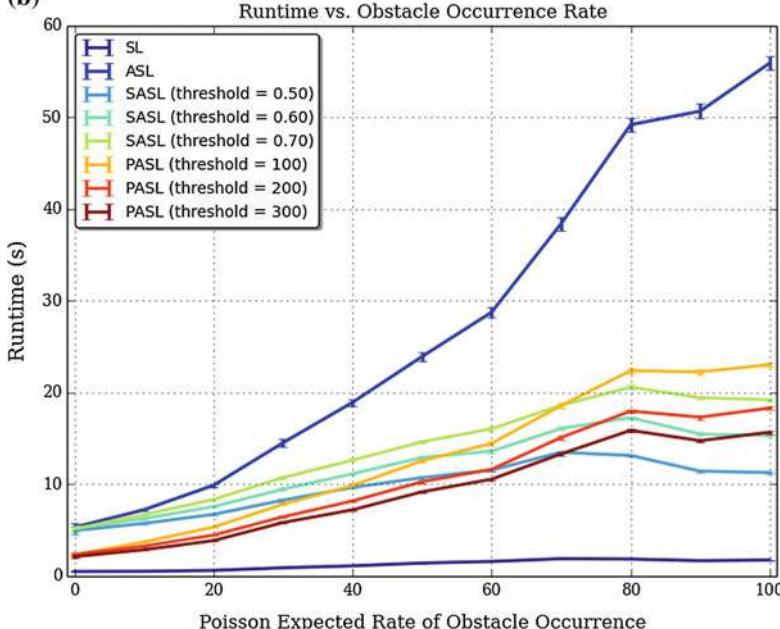


Fig. 6 Simulated random world study results for relative optimality **a** and runtime **b** versus λ presented with 95% confidence intervals

4.2 Comparative Results

To examine the qualitative differences between the four algorithms a sample world is chosen with a nominal amount of clutter and the path representations are visualized in Fig. 7. The map in Fig. 7 is hand selected to represent planning in a moderately complex environment. The solution obtained with the SL is the fastest with a total runtime of 1.07 s, however it has the highest path cost at $J = 37.63$. Due to the regular sampling resolution, the unadapted search is unable to cut through the clutter to reach the goal. The ASL obtains the lowest path cost at $J = 28.00$, but with the highest runtime at 30.08 s. This search is able to optimize to the cost map and weave through obstacles allowing it to achieve a lower cost solution. Similarly to the ASL, the SASL is also able to apply some amount of optimization and achieves a path cost of $J = 33.33$ with a runtime of 15.69 s. The PASL performs similarly but with a lower path cost at $J = 28.54$ and faster runtime at 11.67 s. For this sample, the PASL

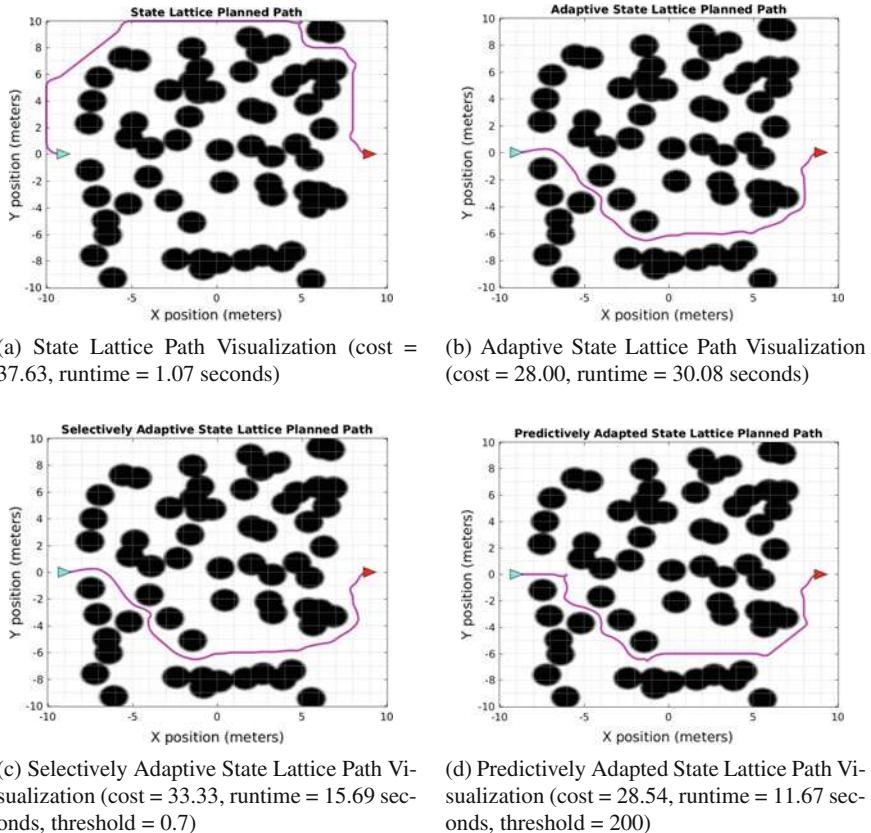


Fig. 7 Qualitative comparison between the solutions obtained by each algorithm tasked with planning from the start node (cyan) to the goal (red) in a randomly generated world with $\lambda = 60$

was able to produce a result comparable to the ASL, with a significantly reduced runtime. An interesting note here is that the ASL, SASL, and PASL solutions all belong to the same homotopic class whereas the SL solution does not. This seems to indicate that the application of predictive adaption can result in solutions of similar quality to fully adapted search spaces, but with large reductions in runtime.

5 Conclusions and Future Work

As the prevalence of UGVs increases, computationally efficient and safe motion planning algorithms become ever more crucial. For applications where memory resources are limited and risk mitigation is paramount, the SL and its extensions are well suited due to their ability to obtain deterministic, resolution optimal solutions that inherently satisfy nonholonomic constraints. Improvements over resolution optimality of the SL is shown to be possible by applying local optimization over samples in the graph. In this paper, we have shown that a learned predictive model can achieve nearly the same optimality as the ASL with significantly reduced runtime requirements and outperform simple hand-coded thresholds for selective adaptation. Statistically significant results are obtained using simulations in random worlds which show an improvement over the SASL and the SL in relative optimality and the SASL and ASL in runtime.

Future work involving the presented algorithm includes optimizations for improving the runtime performance, field experiments in partially observed environments, and adaptation of richer spatial-semantic models of the underlying representation. Although thorough assessment of the algorithm requires examining the performance over many planning scenarios, implementation and validation of these experiments using a physical platform is valuable. The scope of this paper is to improve the performance this particular class of motion planning algorithms, however future work involves comparisons between probabilistic sampling approaches such as RRTss and PRMs.

Acknowledgements This work was supported in part by the National Science Foundation under grant IIS-1637813.

References

1. Pivtoraiko, M., Knepper, R.A., Kelly, A.: Differentially constrained mobile robot motion planning in state lattices. *J. Field Robot.* **26**, 308–333 (2009)
2. Howard, T.: Adaptive Model-Predictive Motion Planning for Navigation in Complex Environments. Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Aug 2009
3. Napoli, M., Biggie, H., Howard, T.M.: On the performance of selective adaptation in state lattices for mobile robot motion planning. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL, USA, Oct 2010

- ference on Intelligent Robots and Systems, Sept 2017
- 4. 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**, 566–580 (1996)
 - 5. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. In: Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C), vol. 1, pp. 473–479 (1999)
 - 6. Urmsen, C., Simmons, R.: Approaches for heuristically biasing rrt growth. *IEEE/RSJ Int. Conf. Intell. Robot. Syst.* **2**, 1178–1183 (2003)
 - 7. Jaillet, L., Cortes, J., Simeon, T.: Transition-based rrt for path planning in continuous cost spaces. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2145–2150, Sept 2008
 - 8. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **30**, 846–894 (2011)
 - 9. Yi, D., Goodrich, M.A., Seppi, K.D.: Homotopy-aware rrt*: Toward human-robot topological path-planning. In: 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 279–286. IEEE (2016)
 - 10. Starek, J., Schmerling, E., Janson, L., Pavone, M.: Bidirectional fast marching trees: an optimal sampling-based algorithm for bidirectional motion planning. In: Workshop on Algorithmic Foundations of Robotics (2014)
 - 11. Choudhury, S., Gammell, J.D., Barfoot, T.D., Srinivasa, S., Scherer, S.: Regionally accelerated batch informed trees (rabbit*): a framework to integrate local information into optimal path planning. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), May 2016
 - 12. Likhachev, M., Ferguson, D.: Planning long dynamically feasible maneuvers for autonomous vehicles. *Int. J. Robot. Res.* **28**, 933–935 (2009)
 - 13. Pivtoraiko, M., Kelly, A.: Differentially constrained motion replanning using state lattices with graduated fidelity. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2611–2616 (2008)
 - 14. Kelly, A., Nagy, B.: Reactive nonholonomic trajectory generation via parametric optimal control. *Int. J. Robot. Res.* **22**, 583–601 (2003)
 - 15. Howard, T.M., Kelly, A.: Optimal rough terrain trajectory generation for wheeled mobile robots. *Int. J. Robot. Res.* **26**(2), 141–166 (2007)
 - 16. Lippmann, R.P.: An introduction to computing with neural nets. *SIGARCH Comput. Archit. News* **16**, 7–25 (1988)
 - 17. Karaman, S., Frazzoli, E.: High-speed flight in an ergodic forest. *CoRR* (2012). [arXiv:1202.0253](https://arxiv.org/abs/1202.0253)

Part V

Mapping

Field Deployment of the Tethered Robotic eXplorer to Map Extremely Steep Terrain

Patrick McGarey, David Yoon, Tim Tang, François Pomerleau
and Timothy D. Barfoot

Abstract Mobile robots outfitted with a supportive tether are ideal for gaining access to extreme environments for mapping when human or remote observation is not possible. This paper details a field deployment with the (TReX) to map a steep, tree-covered rock outcrop in an open-pit gravel mine. TReX is a mobile robot designed for the purpose of mapping extremely steep and cluttered environments for geologic and infrastructure inspection. Mapping is accomplished with a 2D lidar fixed to an actuated tether spool, which rotates to produce a 3D scan only when the robot drives and manages its tether. In order to handle motion distortion, we evaluate two existing, real-time approaches to estimate the trajectory of the robot and rectify individual scans before alignment into the map: (i) a continuous-time, lidar-only approach that handles asynchronous measurements using a physically motivated, constant-velocity motion prior, and (ii) a method that computes visual odometry from streaming stereo images to use as a motion estimate during scan collection. Once rectified, individual scans are matched to the global map by an efficient variant of the ICP algorithm. Our results include a comparison of estimated maps and trajectories to ground truth (measured by a remote survey station), an example of mapping in highly cluttered terrain, and lessons learned from the deployment and continued development of TReX.

P. McGarey (✉) · D. Yoon · T. Tang · F. Pomerleau · T. D. Barfoot
University of Toronto, Toronto, Canada
e-mail: patrick.mcgarey@robotics.utias.utoronto.ca

D. Yoon
e-mail: david.yoon@robotics.utias.utoronto.ca

T. Tang
e-mail: tim.tang@robotics.utias.utoronto.ca

F. Pomerleau
e-mail: francois.pomerleau@robotics.utias.utoronto.ca

T. D. Barfoot
e-mail: tim.barfoot@utoronto.ca

1 Introduction

We are motivated to use tethered mobile robots for mapping extreme environments not suitable for human or remote survey for the purpose of geologic and infrastructure inspection. Deploying an unmanned aerial vehicle to map a steep rock outcrop below the tree canopy would be hazardous, given the challenge of navigating cluttered environments and limited operational time imposed by on-board power storage.

Alternatively, a tethered robot can leverage an attached electromechanical tether, which provides support on steep terrain, continuous off-board power supply, and a reliable wired connection to a remote base station.

McGarey et al. [9] introduced the Tethered Robotic eXplorer (TReX) as a mapping platform capable of advanced mobility on steep terrain. In this paper, we evaluate the TReX platform in a field deployment to an open-pit gravel mine, where the robot was piloted on steep, cluttered terrain to map exposed bed rock as shown in Fig. 1. Our main contributions will be a discussion of mapping results, lessons learned from the deployment, and updates to the TReX system required for deployment.

TReX utilizes a 2D lidar mounted to its rotating tether spool to produce a single 3D scan of the environment as it drives and deploys tether. Thus, we must account for motion distortion in order to produce a global map. We test two existing approaches to estimate the robot's trajectory and handle motion-distorted scans using (i) a continuous-time approach that accounts for asynchronous lidar measurements with a physically motivated motion prior, and (ii) an approach that uses visual odometry to estimate the motion of the robot during the collection of a single 3D scan. Both approaches attempt to reduce distortion prior to scan alignment, which is handled by an efficient Iterative Closest Point (ICP) algorithm. We compare estimated maps and trajectories with ground truth collected by a Leica Total Station, which produces an undistorted point cloud of non-occluded terrain and tracks the position of a marker



Fig. 1 TReX Field Deployment: Our tethered mobile robot navigates extremely steep terrain in order to generate a 3D map of a spanning rock outcrop that is partially occluded by vegetation. The experiment, which was conducted over several days in an open-pit mine in Northern Ontario, Canada, involved just over 1 km of driving on extremely steep terrain. TReX is anchored at the top of the cliff and manually piloted. The attached electromechanical tether (bright green) provides support, off-board power, and wired communications between the robot and base station

attached to the robot if visible. The results show that the odometry-aided approach is best suited for mapping highly unstructured environments when the robot’s motion is complex. We also provide evidence that tethered robots are well suited for mapping steep, occluded environments by highlighting an example map collected in an area hidden from remote view by vegetation.

The paper is structured as follows: Sect. 2 discusses tethered robots and lidar-based mapping, Sect. 3 provides an update on TReX, Sect. 4 details mapping methodologies, Sect. 5 covers the deployment and mapping results, Sect. 6 outlines lessons learned, and Sect. 7 offers concluding remarks and future extensions.

2 Related Work

Tethered robots can be used to explore extreme areas considered dangerous or time consuming for human exploration (e.g., field geology, emergency response, and infrastructure inspection). Tethered mobile robots have been used to explore steep terrain for geologic inspection in the past [7, 13]. McGarey et al. [9] offers an in-depth review of these systems. We are motivated by the idea of lidar-based geologic mapping because it is more efficient than manual survey and also provides a means to investigate rock structure and composition using lidar intensity returns [10].

Since we are interested in producing a map from a rotating lidar mounted to a moving vehicle, we can formulate our problem as a simultaneous localization and mapping (SLAM) problem [12]. The landmark-based simultaneous localization and mapping (SLAM) approach is commonly used to solve this problem, where the state of the robot (i.e., the trajectory and map) is estimated given lidar measurements (i.e., range/bearing to 3D points) [3, 4]. Given that our robot is always in motion while scanning, we require a tool to recover its trajectory in 3D space. Bosse and Zlot [2] use offline, batch scan matching to find a transform between two scans that can be used to approximate the sensor’s trajectory in 3D space without the need for additional odometric measurements. Zhang and Singh [14] expand on this with a real-time, continuous-time lidar odometry and mapping (LOAM) approach that uses high-rate, low-resolution lidar odometry to provide a motion estimate to use in low-rate, fine-resolution scan registration. Without modification, lidar odometry and mapping (LOAM) would not work for TReX because (i) scanning is coupled to the robot’s motion (i.e., lidar odometry fails while stopped), and (ii) its environment is highly unstructured (i.e., lacks planar surfaces and edges). One solution would be to use Vlidar odometry and mapping (LOAM), also from [15], which integrates additional measurements from VO to estimate the robot’s motion during a scan.

In this paper, we test two methods to estimate the robot’s trajectory and produce a global map from lidar measurements. The first is a lidar-only, continuous-time approach from [1], which uses a constant-velocity motion prior for the robot during scan collection and allows for any-time trajectory sampling. As asynchronous lidar measurements arrive, they are associated to a queried transform that can be used to rectify individual 3D scans prior to alignment into the global map. The critical

difference between this approach and lidar odometry and mapping (LOAM), is its use of a physically motivated motion prior, which enables it to work for TReX. Once a scan has been rectified, it is aligned into the global map using an efficient form of the ICP algorithm [11]. The second method also relies on ICP for scan matching and is similar to Vlidar odometry and mapping (LOAM), in that it leverages VO as a motion prior to minimize scan distortion [6]. While the continuous-time method requires the minimum number of sensors (i.e., lidar only), the VO-aided approach is better equipped to capture the robot's complex motion in highly unstructured environments for mapping.

3 Tethered Robotic eXplorer Update

The operational concept for the TReX system is demonstrated in Fig. 2. McGarey et al. [9] first introduced TReX as a system that can both rotate in place regardless of the direction of applied tension due to a passively rotating tether arm, and generate 3D scans of the environment through tether spool rotation. The advanced mobility of the platform allows for turning in place and driving laterally on steep terrain provided sufficient wheel traction. In preparation for field testing, several major updates were made to the TReX system that are not covered in [9], including (i) the integration of an electromechanical tether, (ii) the development of an autonomous, terrain-adaptive tether controller [8], and (iii) the creation of a user interface for use in teleoperation.

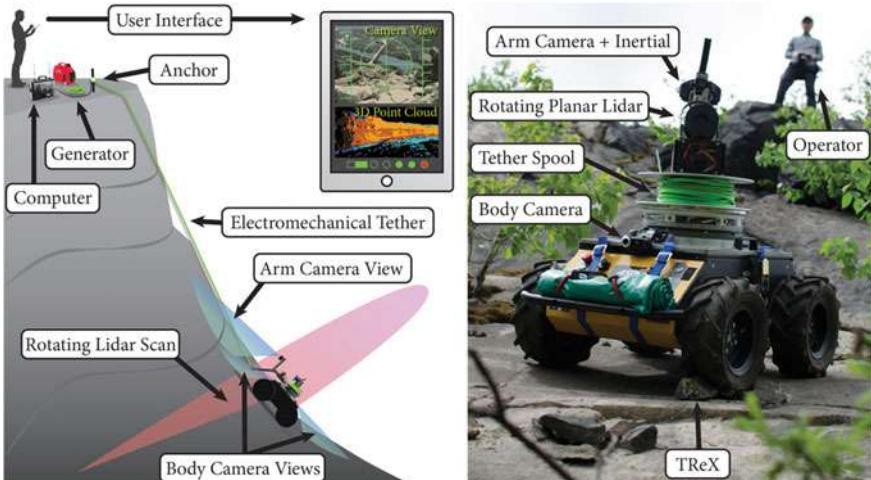


Fig. 2 *Tethered Robotic eXplorer*: The illustration on the *left* is an operational concept for a typical TReX geologic surveying mission. TReX is anchored and deployed near the cliff's edge to map the steep terrain using an on-board lidar. The *right* image shows an actual mapping field deployment



Fig. 3 *User Interface*: A remote user monitors and teleoperates the robot by viewing the live map and cameras, which are transmitted over tether. The tether has been illustrated for reference

(i) *Electromechanical Tether*: Prior to the integration of the electromechanical tether, a static climbing rope was used. As a result, TReX was limited to use on-board power and could only operate for 10–20 mins on steep slopes before a charge was required. The tether¹ allows for simultaneous power and data transmission up to 1200 W and 100 Mb/s respectively, which enables continuous operation and off-board data processing. The tether supports the robot’s 100 kg mass (including the tether), can withstand shocks up to 900 kg, and is constructed of two 18 AWG power and four 24 AWG Ethernet conductors, which are encased in an inner layer of woven Kevlar and an outer layer of polyurethane for strength and resistance to liquid. Its diameter is 9 mm, which limits the spool capacity and range of the robot to 45 m.

(ii) *Tether Controller*: In order to manage the tether, a controller was developed to automatically maintain a set tension regardless of inclination, while assisting the robot to climb on steep terrain [8]. The controller uses both feedback to maintain an inclination-dependent set tension based on the robot’s orientation with respect to gravity (measured by an inertial sensor), and feedforward components to account for the commanded velocity of the robot.

(iii) *User Interface*: The interface shown in Fig. 3 has been developed to allow an operator to monitor multiple camera feeds from the robot while observing the construction of a 3D point-cloud map. The benefit of a ‘wired’ connection is that lidar data can be reliably streamed via tether and processed by the base station computer.

¹Falmat XtremeNet Deep-Water Ethernet Cable—Model: FM022208-03-2K.

4 Mapping Methodology

Figure 4 illustrates how a planar lidar fixed to the robot’s tether spool is used to generate a 3D scan as TRex drives. The slow rotation of the lidar while in motion results in scan distortion analogous to the rolling-shutter effect in passive cameras. In order to complete a single 3D scan, the spool must rotate by 180°, which translates to 0.5 m of distortion along the direction of travel at full speed. For comparison, a car with a Velodyne lidar, scanning at 10 Hz, would only need to travel at 5 m/s (18 km/h) to produce 0.5 m of distortion. However, in our case, the rotational speed of the lidar will not always be constant due to a coupling of spool rotation with vehicle motion, which leads to distortion that is not uniform in time.

Any distorted scan can be rectified by estimating the trajectory of the lidar or robot during collection. Given the rough terrain and slow speed of the robot, wheel odometry and inertial measurements alone cannot be relied on. Alternatively, one approach is to assume a constant-velocity model for the robot during the collection of a single scan in order to handle asynchronous measurements from the lidar. The continuous-time approach uses Gaussian-Process (GP) regression to allow for any-time trajectory querying using GP interpolation, which means that the sensor’s pose can be queried at measurement time. The benefit of this approach is that it only relies on lidar data and uses a physically motivated motion prior (i.e., white noise on acceleration). For more on this continuous-time approach, including detail on the constant-velocity model and implementation, see [1]. In extreme cases, when the constant-velocity model fails to capture the true motion of the robot, the continuous-time approach will not work (see Fig. 12). Accordingly, we test a second method to handle distortion that uses VO as a motion prior for the robot’s trajectory during a single scan; VO is generally accurate over short distances and is better suited to capture complex robot movements, provided that the field-of-view and lighting

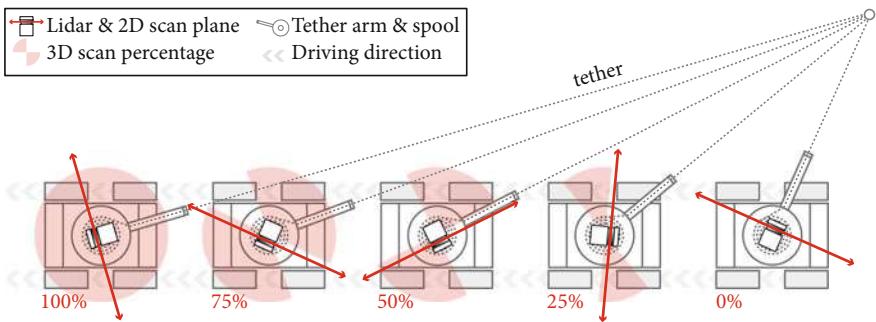


Fig. 4 Generating a 3D Scan: As illustrated, a 3D scan is built from a series of 2D scans only when the robot drives and deploys its tether, which causes 3D scans to be motion distorted. Generally, the robot will need to travel 0.5 m to generate a single scan

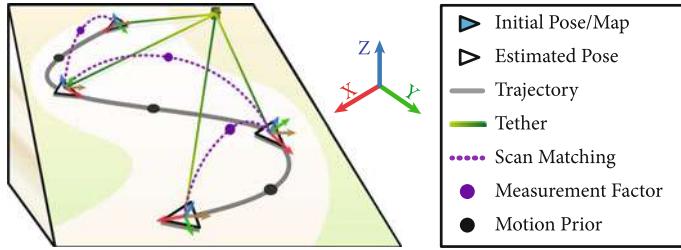


Fig. 5 *Building a Map*: This pose graph represents the scan matching process used to create a global map. Triangles represent the state (i.e., pose and map) being estimated. Prior to matching, the trajectory of the robot is estimated with either a constant-velocity or VO-aided motion prior. Scans can be matched into the global map either sequentially or in a batch, sliding-window setup

conditions are stable. We use an existing VO package² to output pose estimates during scan generation at the frame rate of the camera (10Hz), and rely on a linear-time interpolation function in ROS to associate incoming lidar data to pose estimates. Once scans are rectified, we perform ICP matching³ to align new scans into a global map. Figure 5 illustrates the mapping problem as a pose graph.

5 Field Deployment & Mapping Results

Field Deployment: Mapping experiments were performed in an outdoor, open-pit mine located in Northern Ontario, Canada.⁴ This location features a steep cliff, where bedrock has been exposed by erosion, but is partially hidden by vegetation. Figure 6 provides aerial images of the site that have been annotated with the paths taken to map the terrain. TRex was teleoperated for the entirety of the experiment due to the extreme conditions. Note that much of the cliff is covered by vegetation, meaning that remote survey alone cannot be used to map the contiguous rock outcrop. For more on the deployment, including animated mapping results, see the linked video.⁵

Mapping Results: While driving on Paths C and E, the 3D position of a marker fixed to the robot was recorded by a LTS.⁶ Figure 7 illustrates the ground-truth trajectory overlaid on a dense, ground-truth point cloud. For comparison, we show trajectory estimates from our (CT) and VO-aided ICP pipelines. We note that the full trajectory is not shown because it extends off the visible point cloud. Instead, we plot a portion of the outbound trajectory for clarity. As shown, the VO-aided estimate is closest to ground truth, which is true for all runs. In fact, the CT approach only works when

²Fast Odometry from Vision [5], package available: <https://github.com/srv/fovis>.

³Libpointmatcher [11], package available: <https://github.com/ethz-asl/libpointmatcher>.

⁴Sudbury Ontario, Canada: $46^{\circ}24'33.5''\text{N}$, $80^{\circ}50'27.3''\text{W}$.

⁵Supplemental video: <https://youtu.be/9r10kC7GTmc>.

⁶Model: Leica Nova MS50 MultiStation.

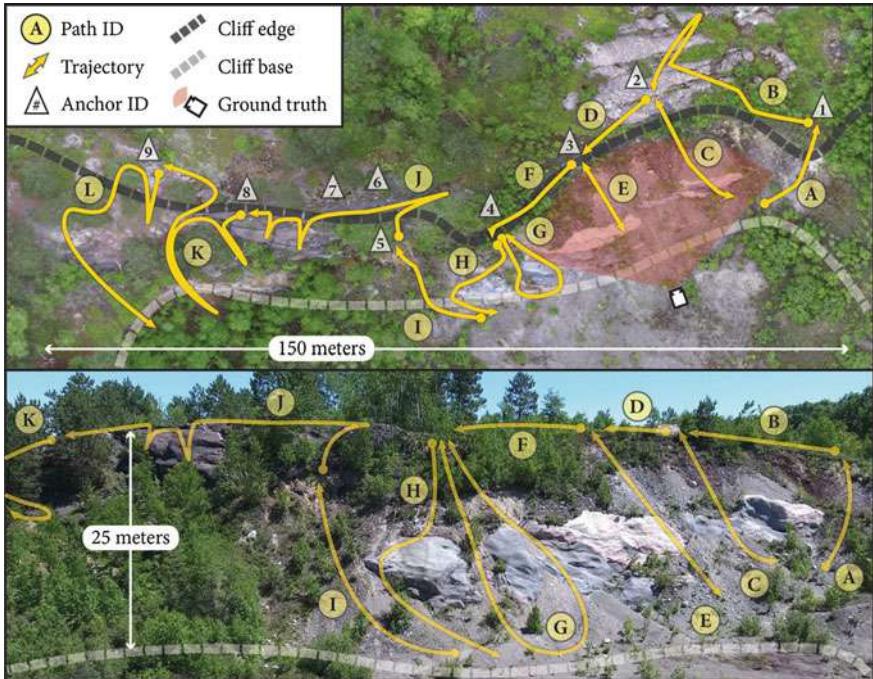


Fig. 6 Experiment Site: The paths driven during the experiment are illustrated on aerial images. The area marked as ‘ground truth’ was mapped by a stationary Leica Total Station. Being that the cliff is covered with vegetation, we use TReX to navigate below the tree canopy for in-situ mapping

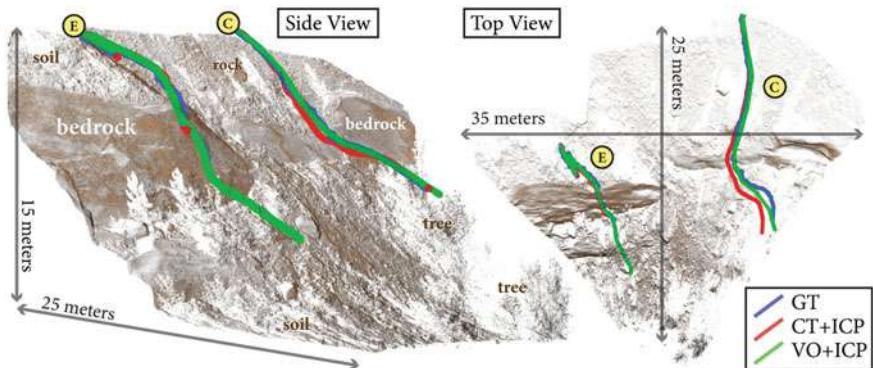


Fig. 7 Trajectory Comparison: Two intensity-colored, ground-truth point clouds are shown. The ‘side view’ shows the slope, which ranges from 30–60° inclination. The position of a marker on robot was recorded to provide ground truth (GT) while driving on Paths C and E only. The VO-aided ICP estimate, VO+ICP, outperforms the continuous-time approach, CT+ICP. We note that GT and CT+ICP are not complete for Path E, which is due to target loss and estimation failure resulting from the robot’s complex motion. Only the outgoing trajectories are shown for clarity

the robot's true motion is smooth and the environment is sparsely vegetated, as later discussed in Sect. 6. On a large scale, the combined VO-aided map (Paths A-E) is comparable to ground truth, and actually covers more area as shown in Fig. 8. Looking closer, Fig. 9 compares a portion of the map produced from Path C. Although the VO-aided map is clearer than CT, both fail to capture the rock outcrop better than ground truth because (i) individual scans collected by TReX are relatively sparse, and (ii) an insufficient number of scans were collected of the target area. Lastly, Fig. 10 provides a map collected by TReX from a densely vegetated, cluttered environment, where the robot navigates below the tree canopy and surveys a rock feature that is not visible remotely.

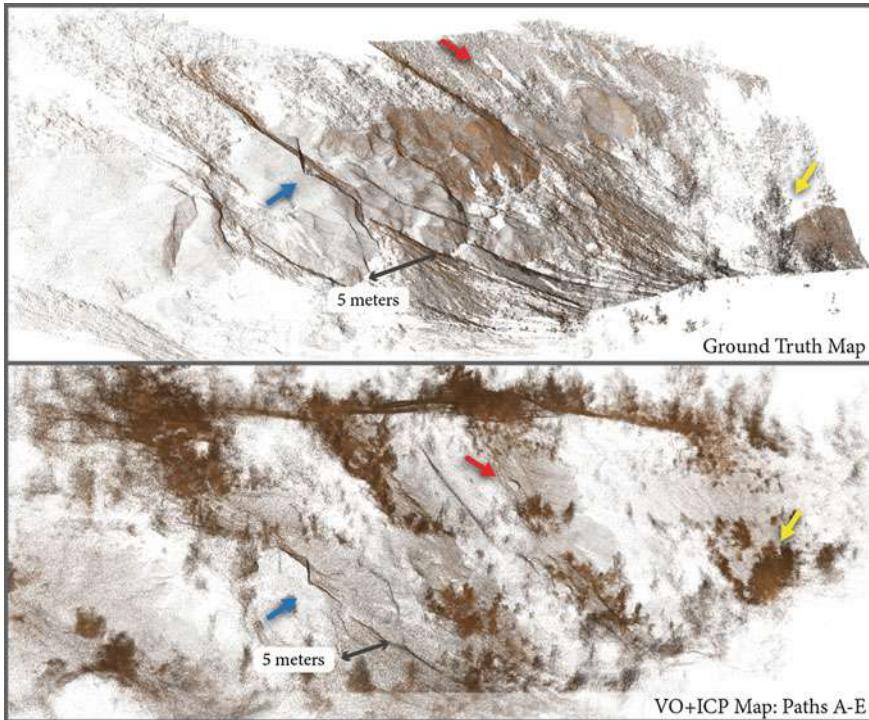


Fig. 8 *Global Map Comparison:* Global point clouds of the unobstructed terrain are compared. The ground-truth map shows the terrain visible to the LTS from a remote position. The VO+ICP map, which encompasses more area than the ground-truth scan, combines estimated point clouds from Paths A-E, which were aligned manually. Common features are indicated by color-coded arrows



Fig. 9 Local Map Comparison: An aerial photo is compared with close-up, virtual images from ground-truth, VO+ICP, and CT+ICP maps colored by point intensity. The rock shown by red arrows in Fig. 8 is marked here as well. While the ground truth is most dense and clearly shows rock features, VO+ICP is our best estimate and is noticeably more detailed than CT+ICP. The estimated maps are sparse due to limited observation time in the target area (e.g., not enough scans were generated). This is also a case where CT+ICP works for a segment of the full trajectory. Generally, the robot's motion is too complex for CT+ICP to work on more challenging trajectories

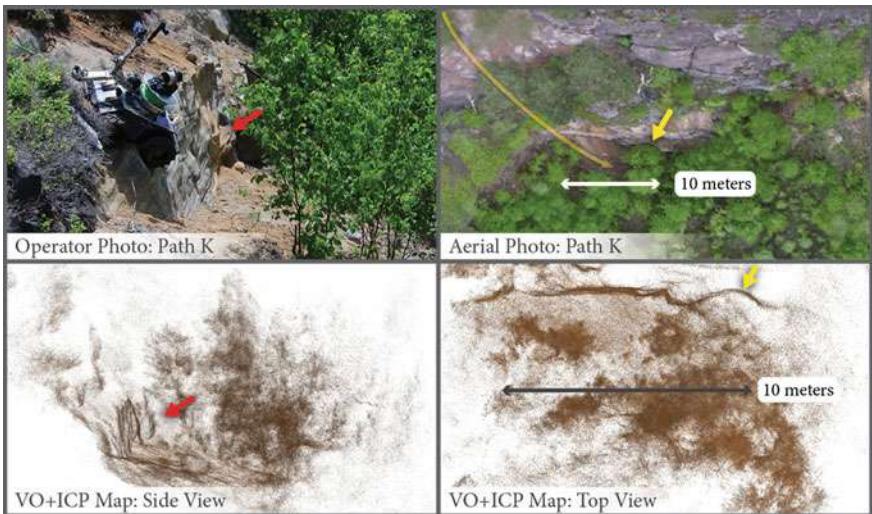


Fig. 10 Mapping Cluttered Environments: This example highlights the benefit of the TReX platform, which can navigate in cluttered environments to explore terrain occluded from remote view. The photos show a heavily vegetated, hard-to-reach area on Path K where bedrock is exposed. Virtual images from the VO+ICP estimated point cloud are compared images. Arrows indicate shared features between the image and map. A portion of Path K is also superimposed on the aerial photo

6 Lessons Learned

We outline lessons learned from the deployment of TReX by addressing mapping challenges, platform mobility, tether management, and perceived design limitations, which impact the robot's ability to explore extreme environments.

Mapping Challenges: Given the mapping performance discussed in Sect. 5, we consider the factors that cause the (CT) approach to perform worse than the VO-aided approach in complex, unstructured environments. One reason is illustrated in Fig. 11, which shows the correlation of spool velocity to 3D scan completion over time. For two sample intervals, we see that the time to complete a scan varies strongly, which implies that TReX, whose spool velocity is coupled to vehicle motion, may violate the constant-velocity model used in CT estimation. One way to deal with this problem is to add additional keyframes (i.e., pose estimates along the trajectory), which would make the velocity assumption arbitrarily better, but also have the effect of slowing down the pipeline. Currently, keyframes are dropped at the completion of each scan (e.g., the robot drives ~ 0.5 m), which allows for running the pipeline in real time. Instead of inserting more keyframes, we set the velocity prior to zero between segments of the trajectory where gaps in measurement time occur. However, we must contend with a greater issue, which is that the velocity prior fails to truly capture the complex motion of the robot driving on extreme terrain. Figure 12 best illustrates this problem; the zoomed-in view shows the estimated trajectory from VO compared to the constant-velocity estimate from CT just as the robot drives over the edge of a cliff. Assuming constant velocity, the robot's motion is estimated to be a smooth arc, when in reality, the robot tilts or rocks at the edge of the cliff before descending. Conversely, VO is able to capture this complex motion. However, VO can fail if the environment is overly cluttered and lighting is too dynamic, which occurs when navigating through dense vegetation (i.e., strong shadows and obstructed vision from hanging leaves).

Another challenge for mapping in the field is that it is difficult to generate dense, minimally distorted point clouds given the operational environment. Figure 13 illustrates this idea by showing how cluttered environments impact scan density and

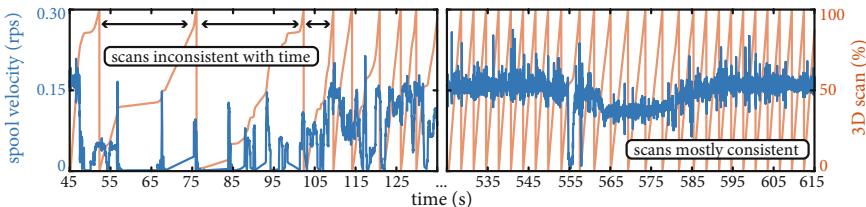


Fig. 11 Scan Completion Over Time: The correlation of spool velocity, measured in half rotations per-second (rps), to percentage scan completion is shown at select time intervals. We note that a full scan only requires a half rotation of the spool. The important take away is that scan completion varies with time, which violates the constant-velocity assumption of the CT method

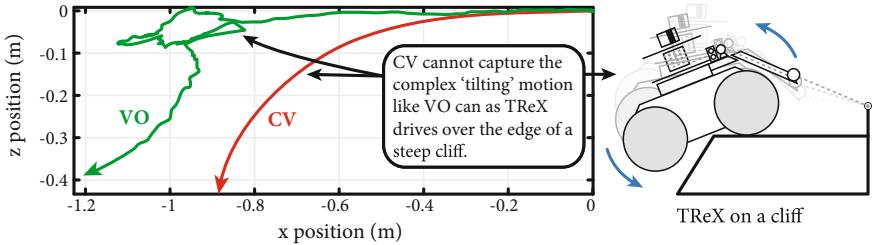


Fig. 12 Motion Prior Comparison: We show the robot's estimated motion for a small portion of the trajectory from the start of a path until just after first descent. The trajectories are plotted as a vertical profile (x - z plane). The constant-velocity motion prior, CV, forces a smooth, arcing trajectory estimate, which fails to capture the complex 'tilting' motion accurately captured by VO. The 'loop' in VO occurs after the robot rocks back and forth in place just after driving over edge

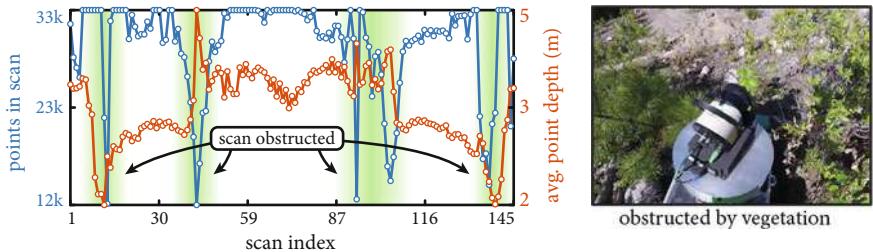


Fig. 13 Scan Quality: The quantity and average depth of 3D points are shown per scan collected on a typical trajectory. The shaded regions represent degenerate scanning conditions that result from navigating through vegetation as shown in the example image. The result is that each 3D scan is generally sparse and shallow in depth, which decreases the likelihood of accurate scan matching

quality; throughout a single traverse, the quantity and average depth of 3D points vary as the robot passes through vegetation. The low average depth indicates that most points lie within 3 m, making it harder to scan distant targets. In terms of point quantity, even the most dense scans are still sparse, with most of the scan being lost to the sky. For comparison, a Velodyne lidar can generate between 100–200 k points per scan, which is an order of magnitude more than TReX. Therefore, it would be beneficial to use a 3D scanning lidar to produce high-quality, detailed maps.

Platform Mobility: In [9], a preliminary mobility evaluation was performed on a steep metal structure to demonstrate lateral movement while under tension. We now provide an in-depth, platform-mobility analysis using examples from the field. Figure 14 provides a collection of images that both highlight the robot's advanced mobility on steep terrain, and demonstrate limitations of the platform. In general, TReX was successful in navigating very steep slopes, regardless of the terrain type (e.g., rock, sand, or grass). With the exception of the inverted slope in Fig. 14.4, the angled tether arm aligns tensional force with the robot's center-of-mass, keeping all tires in contact with the surface. Figures 14.5 and 14.6 show examples of TReX driving laterally under tension, which made it possible to visit and map more terrain



Fig. 14 Platform Evaluation: TReX’s advanced mobility on steep terrain is demonstrated through visual examples during the field deployment. Images 1–3 show TReX successfully navigating steep rock faces. Image 4 shows how the robot’s front tires can lift off the rock due to excessive slope and high tension on the tether. Images 5–6 show the lateral mobility of the robot while under tension. Image 7 shows TReX just after tipping over, which was caused by platform instability when navigating tangent to an angled surface. Image 8 illustrates a common limitation of using a tether; when the taut tether contacts rock, abrasion or severe bending can result in damage

during a single traverse. Despite the success of the deployment, there were several failures, which underline key limitations of the platform. In Fig. 14.7, TReX is shown after tipping over, which occurred while driving parallel to the tether on a steep, angled surface. The fall indicates that (i) the center-of-mass is too high, and (ii) the robot would benefit from some form of passive differential suspension (e.g., rocker-bogie system) to accommodate for uneven terrain.

Tether Management: The prevention of tether damage is a persistent concern when using tethered robots; tether damage is caused by small bend angles, excessive twisting, and or abrasion from rough surfaces. Any one of the above can cause internal wires to break, resulting in power and data loss, or worse, a severed tether, which could damage or destroy the robot and environment. Figure 14.8, shows an example of bending and abrasion during the field deployment, where the robot’s tether became

wrapped around a sharp rock. In practice, we find that the tether must be replaced after each deployment, which results in additional time and cost. To prevent tether damage in future designs, we would need to (i) ensure that the minimum bend radius is not exceeded in the tether management system, (ii) implement better coiling of the tether to prevent twist, and although somewhat impractical, (iii) avoid wrapping the tether around sharp corners or abrasive surfaces in the operational environment.

Size Limitations: The current size of the robot, which encompasses 1 cubic meter and weighs 100 kg, makes it difficult to deploy remotely. During the deployment, the operator would start at the base of the cliff, hike to the top, throw down a rope to an assistant, and manually pull the tether up the cliff to attach to a tree or anchor. The robot would then ascend the cliff, where it could be deployed further. Additionally, the tether-limited range of the robot made it necessary to frequently re-anchor the robot to access new terrain. To cut down on the support and human effort needed to deploy TReX, future iterations of the design should be scaled for use by a single operator, which would allow the entire system, including robot, ground station, and generator, to be backpackable. A reduction in weight would also allow for the use of a thinner, lighter tether and increase vehicle range and overall usability.

7 Conclusion

This paper details the field deployment of the (TReX) to map extremely steep, cluttered terrain in a gravel mine located in northern Ontario. Since TReX produces motion-distorted scans while driving and deploying tether, we evaluate (i) a continuous-time approach that relies on a velocity-based motion prior, and (ii) a VO-aided approach that leverages odometry from a stereo-camera. Both methods are used to compute a trajectory estimate, which can be used to rectify distorted scans prior to ICP alignment into the global map. Each pipeline was evaluated with data collected during the field trial. Our results indicate that the VO-aided approach outperforms the continuous-time method because it best captures the robot's complex motion in dynamic, unstructured environments.

For future work, we would like to deploy TReX to other harsh environments, like caves, mine shafts, and dams. In particular, we would like to test autonomous, route-following techniques [8] that were not developed at the time of the field deployment. We would also like to incorporate other sensors on the robot to use for improving motion estimation, including inertial, inclinometer, and tether measurements (e.g., length and bearing-to-anchor). In the long term, reducing the form factor of TReX design will make it more economical and effective for use in research, inspection, and reconnaissance.

References

1. Anderson, S., Barfoot, T.D.: Full STEAM ahead: exactly sparse gaussian process regression for batch continuous-time trajectory estimation on SE (3). In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 157–164 (2015)
2. Bosse, M., Zlot, R.: Continuous 3D scan-matching with a spinning 2D laser. In: 2009 IEEE International Conference on Robotics Automation (ICRA), pp. 4312–4319 (2009)
3. Cole, D.M., Newman, P.M.: Using laser range data for 3D SLAM in outdoor environments. In: 2006 IEEE International Conference on Robotics and Automation (ICRA), pp. 1556–1563 (2006)
4. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping: part I. *IEEE Robot. Autom. Mag.* **13**(2), 99–110 (2006)
5. Huang, A.S., Bachrach, A., Henry, P., Krainin, M., Maturana, D., Fox, D., Roy, N.: Visual odometry and mapping for autonomous flight using an RGB-D camera. In: International Symposium on Robotics Research (ISRR), pp. 1–16 (2011)
6. Kubelka, V., Oswald, L., Pomerleau, F., Colas, F., Svoboda, T., Reinstein, M.: Robust data fusion of multimodal sensory information for mobile robots. *J. Field Robot.* **32**(4), 447–473 (2015)
7. Matthews, J.B., Nesnas, I.: On the design of the axel and duaxel rovers for extreme terrain exploration. In: 2012 IEEE Aerospace Conference, pp. 1–10 (2012)
8. McGarey, P., Polzin, M., Barfoot, T.D.: Falling in line: visual route following on extreme terrain for a tethered mobile robot. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), (2017)
9. McGarey, P., Pomerleau, F., Barfoot, T.D.: System design of a tethered robotic explorer (TReX) for 3D Mapping of steep terrain and harsh environments. In: 2015 International Conference on Field Service Robotics (FSR). Springer (2015)
10. Osinski, G.R., Barfoot, T.D., Ghafoor, N., Izawa, M., Banerjee, N., Jasiobedzki, P., Tripp, J., Richards, R., Auclair, S., Sapers, H., Thomson, L., Flemming, R.: Lidar and the mobile Scene Modeler (mSM) as scientific tools for planetary exploration. *Planet. Space Sci.* **58**(4), 691–700 (2010)
11. Pomerleau, F., Colas, F., Siegwart, R., Magnenat, S.: Comparing ICP variants on real-world data sets. *Auton. Robots* **34**(3), 133–148 (2013)
12. Smith, R., Self, M., Cheeseman, P.: Estimating uncertain spatial relationships in robotics. In: Autonomous Robot Vehicles, pp. 167–193. Springer (1990)
13. Wettergreen, D., Thorpe, C., Whittaker, R.: Exploring Mount Erebus by walking robot. *Robot. Autonom. Syst.* **11**(3), 171–185 (1993)
14. Zhang, J., Singh, S.: LOAM: lidar odometry and mapping in real-time. In: Robotics: Science and System Conference (RSS), pp. 109–111 (2014)
15. Zhang, J., Singh, S.: Visual-lidar odometry and mapping: low-drift, robust, and fast. In: 2015 IEEE International Conference on Robotics Automation (ICRA), pp. 2174–2181 (2015)

Towards Automatic Robotic NDT Dense Mapping for Pipeline Integrity Inspection

Jaime Valls Miro, Dave Hunt, Nalika Ulapane and Michael Behrens

Abstract This paper addresses automated mapping of the remaining wall thickness of metallic pipelines in the field by means of an inspection robot equipped with Non-Destructive Testing (NDT) sensing. Set in the context of condition assessment of critical infrastructure, the integrity of arbitrary sections in the conduit is derived with a bespoke robot kinematic configuration that allows dense pipe wall thickness discrimination in circumferential and longitudinal direction via NDT sensing with guaranteed sensing lift-off (offset of the sensor from pipe wall) to the pipe wall, an essential barrier to overcome in cement-lined water pipelines. The data gathered represents not only a visual understanding of the condition of the pipe for asset managers, but also constitutes a quantitative input to a remaining-life calculation that defines the likelihood of the pipeline for future renewal or repair. Results are presented from deployment of the robotic device on a series of pipeline inspections which demonstrate the feasibility of the device and sensing configuration to provide meaningful 2.5D geometric maps.

1 Motivation—A Taxonomy of NDT Inspection Techniques

Non-Destructive Testing (NDT) or Evaluation (NDE) is extensively employed by the energy and water industry to assess the integrity of their network assets, particularly their larger and most critical conduits (generally referred to as those larger than 350 mm in diameter), in their decision-making process leading their

J. V. Miro (✉) · D. Hunt · N. Ulapane · M. Behrens
Centre for Autonomous Systems, University of Technology Sydney,
15 Broadway, Ultimo, NSW 2007, Australia
e-mail: jaime.vallsmiro@uts.edu.au

D. Hunt
e-mail: dave.hunt@uts.edu.au

N. Ulapane
e-mail: nalika.ulapane@uts.edu.au

M. Behrens
e-mail: michael.behrens@uts.edu.au

renewal/repair/rehabilitation programs. The key advantage of NDT/NDE is that the structure of the asset is not compromised in estimating its condition.

The sensing modality to use is strongly influenced by the material of the asset. Grey Cast Iron (CI) pipelines remain the bulk of the buried critical water infrastructure in the developed world as that was the material of choice for mass production with the advent of the Industrial Revolution in the middle of the 18th century (alongside its less brittle relative of Ductile Iron since the nineteen fifties), until carbon steel, asbestos cement or plastic pipelines (PVC) amongst other materials made them redundant over the years. The non-homogeneity of the CI produce means that sensing techniques widely employed in the (mild) Carbon Steel networks in the energy pipeline sector, such as ultrasonics or electromagnetic acoustic transducers (EMAT), are inadequate for CI, and the underlying techniques of most commercial propositions for CI are instead based on either magnetics (e.g. Magnetic Flux Leakage (MFL), Pulsed Eddy Currents (PEC) and Remote Field Eddy Currents (RFEC)), or the study of the propagation of pressure waves in the pipeline and/or fluid.

NDT techniques produce results that tend to be a trade-off between deployment costs and information gain. Local inspection techniques (i.e. 1–3 m) can provide dense measurements but are time-consuming and generally costly per unit-length as significant preparatory civil works are required (excavations, network re-routing for guaranteed supply, traffic control, etc.). Moreover inspections can only be undertaken at locations which are accessible from the surface. An example of these tools can be seen in Fig. 1b.

On the other hand, the taxonomy of long-coverage tools can be broadly split into techniques that provide average pipe wall measurements over longer distances (generally from a few to 100s of meters, even kilometers), and in-line intrusive (ILI) devices (“smart pigs”) deployed inside the pipeline to inspect in higher detail

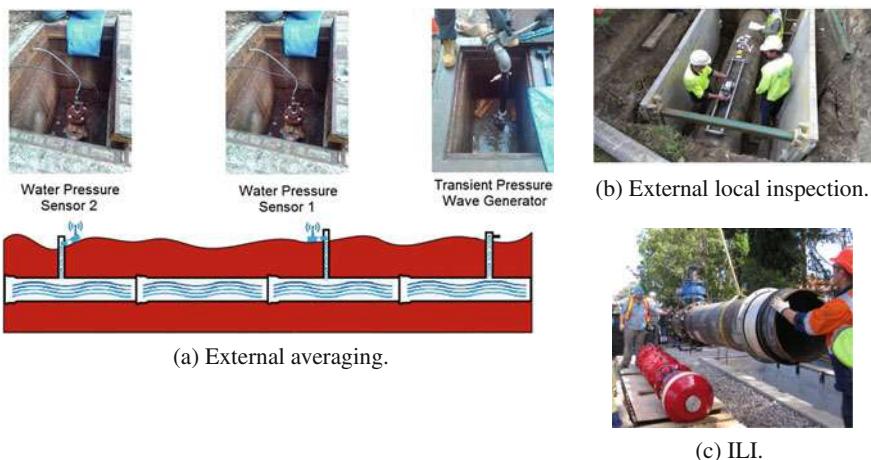


Fig. 1 Example of various configurations of NDT tools

over longer distances (generally 100s of meters to kilometers to make it more cost-effective), while propelled by the operating pressure of the fluid.

The former are generally deployed by accessing the external pipe wall or water column at a few access points spread over the length of the pipeline, either through small key-hole excavations or through external access points such as valves or hydrants. As such they tend to have low or no impact in the continuing operation of the pipeline and are more affordable alternatives for condition assessment. An example can be seen in Fig. 1a. However given the averaging nature of their results, these tools are aimed at providing an initial screening of the condition of an asset, and lack the ability to provide the type of detailed geometry information needed to ascertain likelihood of pipe failure.

Flow-driven ILI tools, on the other hand, are inserted into the charged water column either through standard large appurtenances present in critical mains, or more often than not via dedicated launch and retrieval mechanisms, as depicted in Fig. 1c. While these tools are able to provide direct measurements related to the pipe wall condition over long distances, they do so at the expense of higher disruption to the utilities and combined costs from the substantial civil engineering support from the utility prior, during and post inspection. Moreover, the effectiveness of these techniques has not been fully established within the industry given the consequential validation investment required to do so in a statistically meaningful way.

ILI tools present additional shortcomings in the pursuit of attaining an accurate depictions of the condition of a pipe wall:

- they are at the mercy of the pressure of the fluid driving them (both in the tethered and free-flowing case).
- should the tools were to be operated in de-watered conditions, they necessitate complicated winch mechanisms between entry and exit points.
- operating parameters need to be closely controlled (e.g. tool velocity), meaning that discriminating flow controls need to be in place, not necessarily an easy feat to achieve in a complex interconnected network.
- they lack the ability to do fine control and adjustments for mapping (e.g. ensuring tight tolerances in sensor lift-off, repeatability, rectify missed measurements).

Driven by the needs of the water industry the work hereby presented describes the development and field testing of a novel internal NDT inspection robotic vehicle able to:

1. undertake localised, controlled inspections.
2. generate dense NDT mapping suitable for condition assessment and failure prediction.
3. tightly control inherent lift-off during sensing (as induced by the presence of non-magnetic cement lining and pipeline wall irregularities)
4. access arbitrary (within tethered range) pipeline spools from a single point of entry, hence reducing costs to utilities and allowing inspection of inaccessible sections from the surface (e.g. under a rail pass) and minimising disruption to customers (e.g. a pipeline under a driveway).

While the proposed solution requires pipes to be de-watered for deployment, this serves a clear mandate from the utility sector that necessitates a robotic NDT inspection vehicle that can be deployed in an opportunistic manner to ascertain the condition of a particular pipeline, specifically when a mains break occurs, or on the back of a valve inspection or repair program when pipelines are inevitably taken off-line. The remainder of this paper describes such an NDT robot for the inspection of buried network infrastructure and the novelties behind its inception.

2 NDT Pipeline Wall Inspection

Recent research in the space of stress analysis and failure prediction of critical CI water mains has revealed that over and above pit depths, as traditionally provided during condition assessment of a critical asset, there is a need to ascertain the presence and geometries of large corrosion patches in the pipe walls [1], such as those depicted in Fig. 12d. There exist a wide range of NDT technologies developed for the purpose of material characterisation for CI [2], yet the provision to build dense 2.5D maps of remaining wall geometries for lined water mains has driven the need to design an internal inspection tool around Pulsed Eddy Current (PEC) sensing technology, as a proven technique typically used in the NDT sector for ferromagnetic material thickness estimation [3, 4], resilient to sensor lift-off.

Fig. 2 enables interpreting 2.5D maps of remaining wall thickness produced through PEC sensing, and the conventions shown in Fig. 2 hold for all thickness maps presented herein. The axial location indicates the distance along the pipe's longitudinal axis, while the circumferential location represents rotational degrees around the pipeline. 0° and 360° coincide on the top (crown) of the pipe. In the field inspection results presented in Sect. 4, longitudinal locations are in reference to the origin set at the robot's entry point to the pipe. The colour bar to the right of the thickness maps is a legend representing thickness in mm, between 0 and 30.

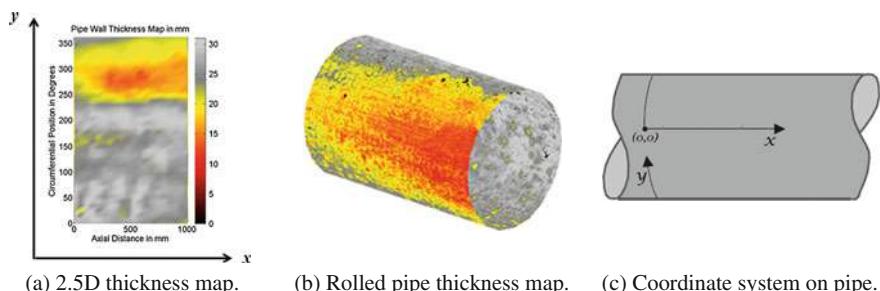


Fig. 2 Axial x and circumferential y coordinates of a 2.5D pipe thickness map

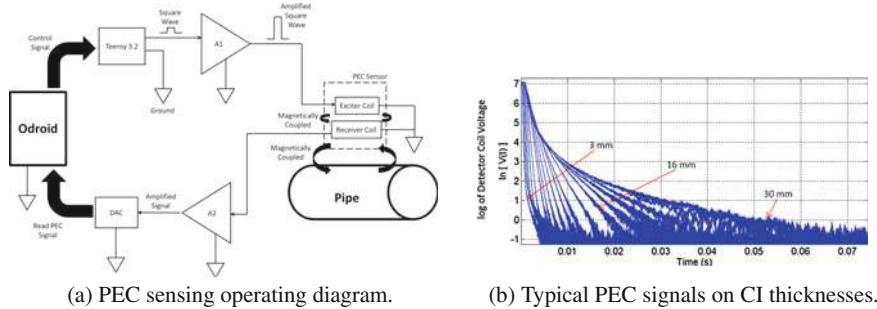


Fig. 3 PEC sensing setup embedded in inspection robot, and typical PEC signals

Developed PEC Sensing System A typical PEC sensing system developed for ferromagnetic materials consist of an exciter coil, a detector coil, a voltage pulse generator for excitation and an amplifier for the detected signal. A block diagram of the PEC sensing set up developed for this work is shown in Fig. 3a. Given the size of the pipes of interest the footprint of the sensor used was 50 mm, indicating that it measures the average thickness of a 50 mm \times 50 mm area under the sensor. Signals captured from the system on different CI thicknesses are shown in Fig. 3b and as reported in the literature features can be extracted from such signals which can be directly linked to material thickness [5, 6].

Validation of PEC Robot Sensor Setup The validity of the sensor arrangement was first assessed by comparing results on the exhumed CI pipe in Fig. 4 with intact cement lining. The objective was to evaluate how well the measurements agree if a section of the pipe is scanned externally and internally via cement lining. External measurements were performed on known locations with the aid of the grid pattern marked in Fig. 4a. The same locations were scanned internally as shown in Fig. 4b with the aid of the robot localized with reference to the pipe's edge. Measurements were recorded at 50 mm distance increments along rings in the circumferential direction, whilst distance between consecutive rings was set to 100 mm to speed-up the inspection process, since thus generated thickness maps can be then upsampled with minimal information loss as shown in Fig. 12d. The rationale and methodology for this will be further elaborated on in the following two Sections. Strong agreement between both measurements was notable as depicted in Fig. 5; the error histogram in Fig. 5e, calculated by subtracting internal thickness estimates from external ones hints at a small positive bias in the error, with a mean and standard deviation of 0.323 mm and 0.417 mm respectively. This is an expected result since marginally better sensitivity can be expected when scanning externally (particularly for higher thickness), as the sensor touching the pipe wall can achieve stronger penetration than from the inside given the lift-off effect induced from the cement lining layer. The errors are indicative of acceptable agreement between internal and external measurements confirming the sensor's suitability for internal assessment of pipes via cement lining. In another sensor verification experiment, Fig. 5c shows results of a repeat-



(a) Exhumed pipe on which internal and external measurements were performed.
 (b) Pipe assessment robot performing internal measurements.

Fig. 4 Laboratory setup with exhumed pipe for internal and external PEC validation

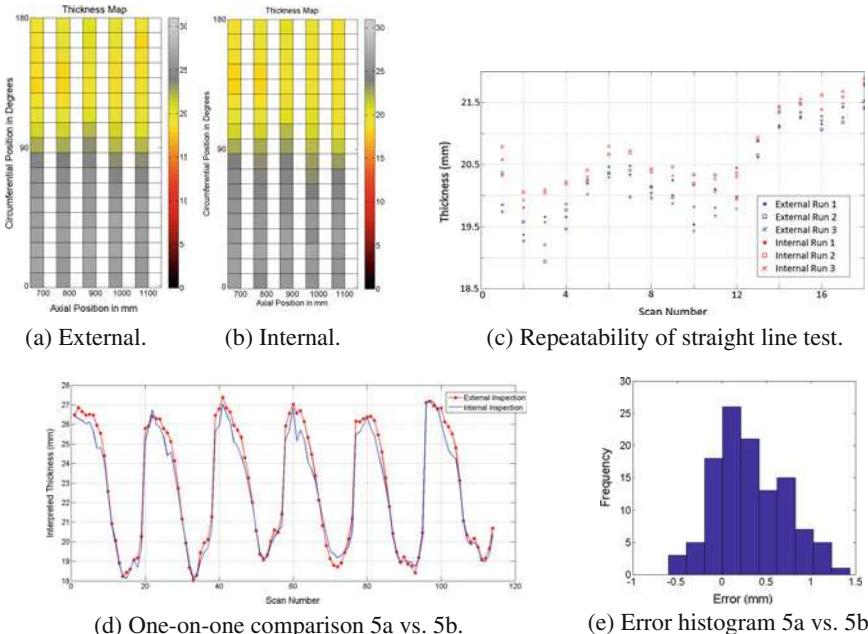
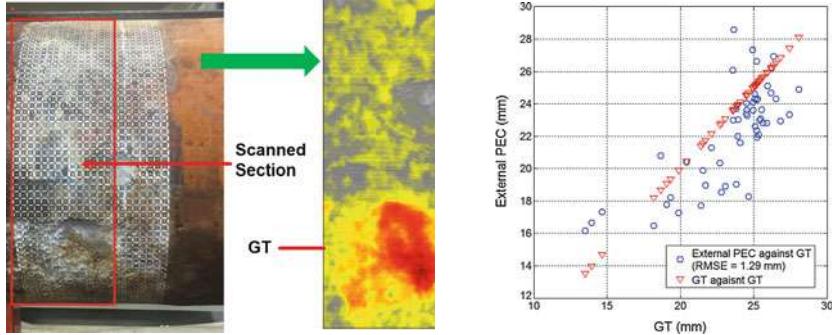


Fig. 5 Thickness maps from internal and external measurements

bility test carried by measuring a straight line along the pipe six times (three times internally and three times externally, with each line having 18 measurements); the average variation on a location was less than 1 mm, indicating appreciable measurement repeatability.

Further to the comparison of internal versus external deployment, the PEC sensor arrangement was also validated on a pipe whose actual wall thickness ground truth (GT) had been previously obtained, with the results collected in Fig. 6. Attaining the GT is a destructive process, whereby the pipes are first exhumed, then both internal and external pipe surfaces are grit-blasted to remove rust and graphitization,



(a) Pipe section and high resolution GT map. (b) Estimates against GT.

Fig. 6 External PEC thickness estimates against GT

the by-products of the corrosion process inflicted on a buried pipeline, leaving only the bare metal—the target of the PEC sensor measurement. Both surfaces are then reconstructed with a high-resolution 3D laser scanner and ray-tracing performed on the collocated upsampled internal and external pipe surface point clouds to derive the GT thickness maps at a resolution of 0.6 mm [7]. This high resolution GT can then be downsampled to the sensor's 50 mm footprint by means of averaging so as to match the PEC sensor measurements in order to compare. RMS error of 1.29 mm was observed between external PEC measurements and GT, indicating reasonable agreement even when challenged by significant defects as evident from the testing pipe depicted in Fig. 6a, selected to better capture variability in the remaining wall thickness.

3 NDT Robot Kinematics, Locomotion and Control

The robotic NDT mapping unit was designed to allow accurate positioning of sensors internally on the pipe surface, in a robust and repeatable manner. To achieve this, a mechanism designed to self-align inside the pipe while providing circumferential and longitudinal control with a single actuation to place sensors against the pipe inner wall was developed, shown in Fig. 7.

Mechanical Design Mecanum wheels were selected for the robot locomotion. In planar applications they enable holonomic robot motion as they allow control in all three degrees-of-freedom (DoF) available to the robot [8]. For this application it is only necessary to control two degrees-of-freedom, longitudinal and circumferential motion. By applying a non-standard wheel configuration it is possible to exploit the unique geometry of the operating environment to passively align with the central pipe axis, automatically tracking the pipe should minor changes in direction occur. Fig. 8a and b demonstrate the layout designed to achieve these requirements. In this

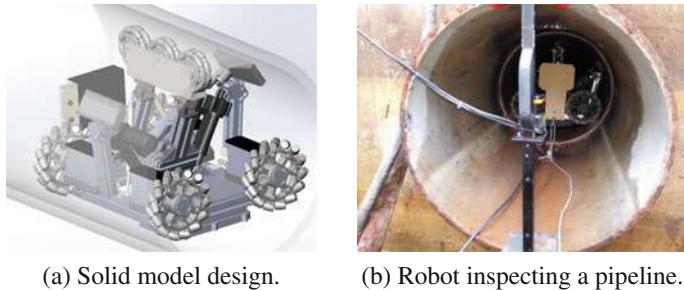


Fig. 7 NDT inspection robot design, and during field deployment in a pipeline

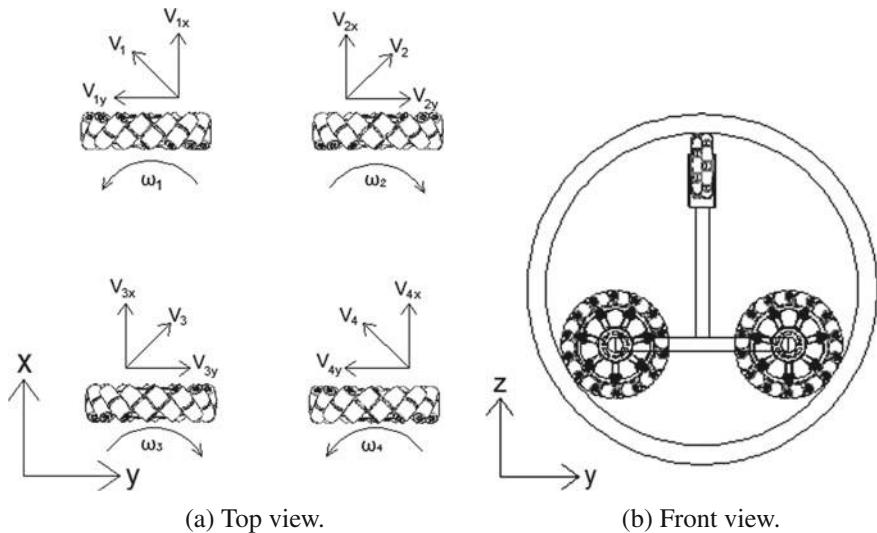


Fig. 8 Mecanum wheel layout

configuration, the axis of rotation of the pipe contacting rollers all pass through a single point allowing the robot to rotate freely about this point in response to an external force. When resting on a cylindrical surface, such as a pipe wall, an external restoring force is generated in response to angular disturbances which acts to return the robot to the aligned position.

Control in the longitudinal direction and rotation about the circumferential direction are achieved by controlling wheel velocities using the kinematic relations derived in Eq. 1, which follow standard forwards kinematic equations in simplified form [9], where $v_x(t)$ reflects the longitudinal velocity (m/s), $v_y(t)$ is the circumferential velocity (m/s), ω_i ($i = 1 \dots 4$) is the wheel rotation speed (rad/s) and r is the wheel radius (m).

$$\begin{aligned} v_x(t) &= (\omega_1 + \omega_2 + \omega_3 + \omega_4) \times \frac{r}{4} \\ v_y(t) &= (-\omega_1 + \omega_2 + \omega_3 - \omega_4) \times \frac{r}{4} \end{aligned} \quad (1)$$

It is essential that the angular velocities of diagonally opposite wheels are matched to prevent excessive motor loads as the robot is constrained in the z-axis. Driving each pair of diagonally opposite wheels with a single motor would achieve this requirement, however, the required drivetrain is complex and in the proposed designed control of each separate motors is implemented in software, as discussed below as part of the system overview.

To maintain stability during circumferential rotations, a set of free-wheeling omni-wheels are mounted on a parallel four bar linkage shown in Fig. 9a. This assembly is pressed against the pipe wall with a preload of approximately twice the robot weight, maintaining control authority regardless of orientation while simultaneously compensating for variation in pipe diameter. A linear actuator is included to retract the omni-wheels from the pipe surface during insertion. This actuator features a spline so that it does not affect the self correcting behaviour of the parallel linkage during normal operation.

The PEC sensors are coupled to actuated lever arms using a stiff rubber joint. This joint allows the sensor to conform to the pipe surface in the presence of minor irregularities while maintaining a precise placement. The actuators drive until a stall condition is detected, allowing the sensor to be reliably placed on the pipe surface regardless of pipe variations or actuator drift.

System Overview The system uses two computers, one on-board the robot for data acquisition and actuator control, and one outside the pipe for the user interface. The entire system runs from a generator on the surface with power delivered to the robot with a power over ethernet (PoE) connection. The user interface can receive data and issue control commands in real-time over the local area network (LAN) connection provided by the ethernet tether. This approach limits operational range

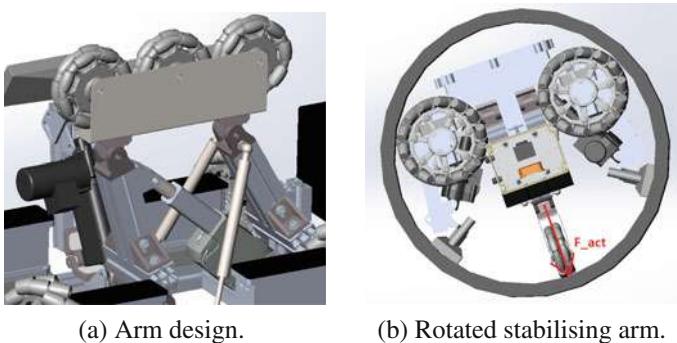


Fig. 9 Stabilising arm

Table 1 Core component specifications

Computing	Odroid XU4—arm based single board computer
Sensors	Xsens Mt-10 IMU w/Gyro 450°/s, acc 50 m/s ² Odroid USB-Cam 30FPS, FOV: 68° 3D Structure sensor w/HFOV: 58°, VFOV: 45°
Control	Maxon motor DCX26L, gear ratio 231:1, sensor 500 counts/turn Sensor actuator—linear actuator, max force: 1000 N Stability actuator—linear actuator, max force: 2500 N
Power	PoE injector/splitter, 60 W, 70 m cable reel

but provides for significantly longer operation times. To overcome power limitations of PoE, an ultracapacitor bank and bespoke charger was developed to supply bursts of high power while ensuring that the PoE equipment maintains an optimal power delivery rate. This setup provides a steady power supply for the overall system on the condition that high power maneuvers are not sustained for extended periods.

The on-board Odroid, running Linux and the Robotic Operating System (ROS), receives data from the sensor suite through a powered USB hub and controls on-board actuators via digital input/outputs pins. Each sensor has its own monitoring node to manage incoming data and publish to the communication layer. Custom task allocation/behaviour nodes then subscribe to the data streams, processing and publishing control commands as required to the motor and actuator nodes. System control is accomplished using a state machine, which allows both user and autonomous control modes for consistent data retrieval and safe user override. When switched into automatic scanning mode, the circumferential angle and longitudinal position are managed using independent set-point control loops. This simplifies both the kinematics and the algorithms required for control. Controlling circumferential angle is achieved using the on-board IMU and a standard PID control algorithm. The IMU publishes attitude data to ROS at a fixed rate of 100 Hz. As each data packet is received, the attitude data is transformed into the local coordinate frame to maintain consistency even when the pipe is not level. Similarly, longitudinal control is achieved using odometry calculated using encoder readings published at a 100 Hz and filtered to detect wheel stalls. In addition, an overriding human in the loop (HITL) input allows direct control of the longitudinal position. This is used to recover when odometry fails due to motor stalls or excessive wheel slip during the ring-to-ring transitions. A laser distance sensor is utilised to confirm longitudinal position when conditions are safe to deploy in the excavation pits. Details of the system are collected in Table 1.

Motion Validation Consistency during automated scanning was verified at onsite trials, with angular repeatability and translational slip during rotation being the key metrics providing confidence in the sensor placement accuracy. Data was collected from an IMU to validate robot orientation, and a laser distance sensor was used to confirm longitudinal positions. Fig. 10a shows the measured rotation angle error during repeated scanning cycles of 180° using 10° increments. The observed offset of 0.53° in relation to the set-point was found to be originated by the pressing action

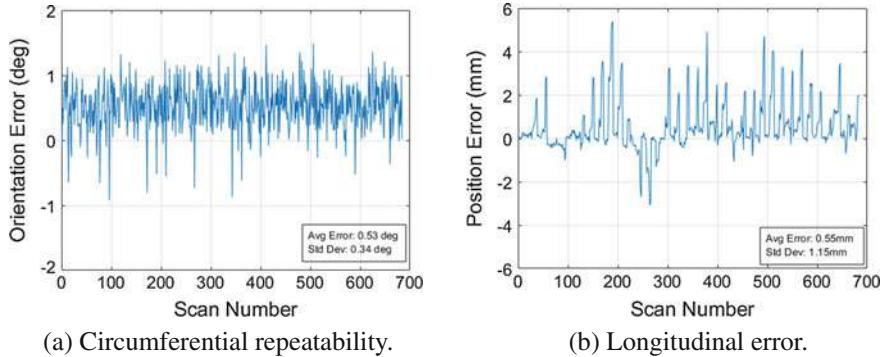


Fig. 10 Circumferential and longitudinal consistency

of the sensor against the pipe wall. Fig. 10b shows longitudinal drift during these scans. An average error of 0.55 mm was produced (maximum 5.4 mm). Since PEC sensing produces a result averaged over a 50×50 mm area, and scan rings are spaced at increments of 100 mm, a 5 mm drift in the longitudinal direction is well within the safety margins generally assumed for failure prediction analysis in civil infrastructures. Given overall sensing and mechanical constraints, the system operates at slow speeds: deployment runtime metrics showed an average spool completion time of 166 min, or 1.3m/h. This includes 216 s of automated scanning for each ring, with 30–60 s dedicated to motion from one ring to the next.

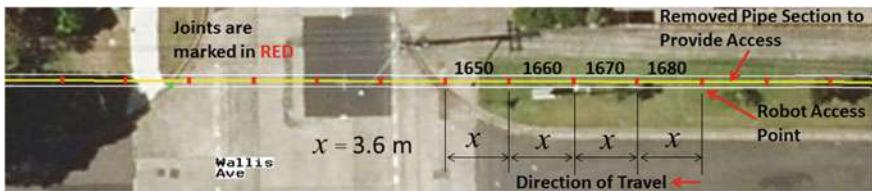
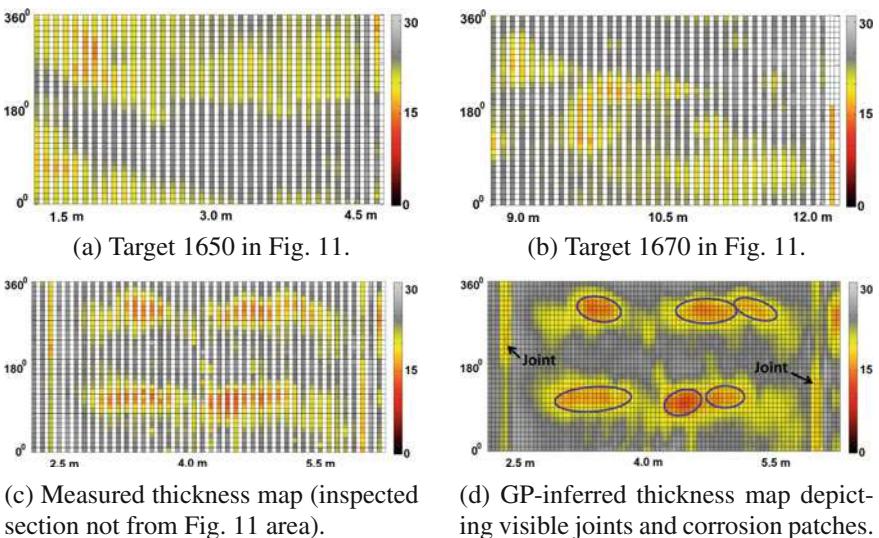
4 Field Pipeline Inspection Results

The proposed robotic device has been extensively deployed in a buried 1 Km live CI Cement Lined (CICL) pipeline provided by a utility in Sydney, Australia, in what effectively constitutes a unique worldwide opportunity for the advancement of NDT sensing and automation research in the field [10]. The pipeline has been decommissioned and is therefore no longer part of the utility's live network. However a connection point to an adjacent 600 mm water main and various scour valves and hydrants allow for the pipeline to be pressurised and discharged as needed. Details of the pipeline are collected in Table 2. Pipe sections between 3 and 4 m in length were targeted for scanning by inserting the inspection robot through a removed pipe section, be that a previously replaced section, as shown in Fig. 1c, or a new cut-out. An example of an inspection plan is shown in Fig. 11.

As mentioned in Sect. 1 the salient novelty of the proposed robotic integrity assessment is the ability to carry out internal detailed inspections that enable dense mapping where identification of the geometry of wall loss patches can be confirmed. An example of the final outcome achieved is shown in Fig. 12d, where measurements indicative of the lead run joints are also shown.

Table 2 Test-bed specifications, adapted from [10]

Year installed	1922
Nominal pipe diameter	600 mm
Internal pipe diameter	579–590 mm (with cement lining)
External pipe diameter	662–666 mm
Nominal wall thickness	27 mm
Material	Pit cast iron
Internal liner	Cement (installed in 1964)
Cement lining thickness	9.5–16.5 mm

**Fig. 11** A typical inspection plan**Fig. 12** Various examples of remaining pipe wall thickness maps as measured by the robotic wall inspection during field deployment on buried critical water mains

To achieve this outcome various inspection patterns were studied to mitigate the slow robot examination speed reported in Sect. 3, and it was proven that circumferential rings 100 mm apart in axial distance were able to reconstruct detailed dense

maps by means of Gaussian Process (GP) [11] spatial data dependences from limited NDT inspection data [12]. Given the 50 mm sensor footprint this effectively meant skipping every other ring with considerable time savings yet inconsequential information loss in relation to map quality and sizing of critical patches.

Robot localisation with respect to an entry point while travelling towards a section targeted for inspection was done by means of robot odometry, measurement of tether release and accounting for spool joints traversed as seen by the robot camera. Validation from an external laser scanner mounted at the entry point as seen in figure was also used when it was deemed safe to be deployed in the field excavation pit. Moreover, discontinuity on spool joints also reveals a characteristic PEC signal comparable to a crack that was also exploited in case of ambiguity about spool length. After reaching the target spool, circumferential and longitudinal ring inspections were undertaken as described in Sect. 3 to generate maps such as those depicted in Fig. 12. Where wall loss is present the spread of the reduction is clearly evident and can be identified and measured. Such patches are modelled as ellipses (see Fig. 12d), and their defining parameters can then be incorporated for stress calculation and remaining life prediction of the asset [1].

Prior to using the robotic tool for extensive measurements, repeatability tests were also carried out on pipe sections at the test-bed to ascertain the performance of the robotic inspection unit in-situ. Results from one of the tests are shown in Fig. 13. The error histogram in Fig. 13c suggests a close to zero-mean Gaussian (0.112 mm mean, 0.869 mm standard deviation). Information such as minimum, maximum and average thickness of the inspected pipe section are key parameters of interest to water

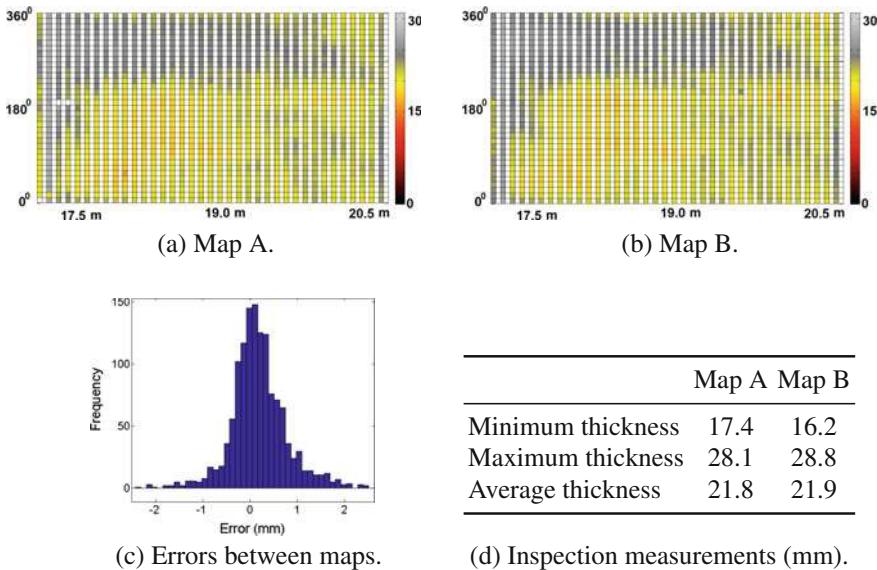
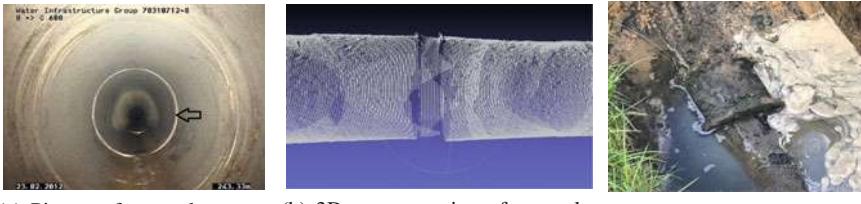


Fig. 13 Robotic inspection repeatability tests on a single pipe section



(a) Picture of anomalous narrowing (white ring at pointer). (b) 3D reconstruction of anomaly captured by RGBD robot sensor. (c) Outer clamp found after excavation.

Fig. 14 Uncharted pipe anomaly found during inspection; verification excavation

utilities for stress analysis and asset management in general. Figure 13d collects the most typical quantitative information being currently reported with the robotic device on the two inspections shown—Map A (Fig. 13a) and Map B (Fig. 13b).

Pipe Inner Surface Profiling In addition to PEC measurements, perceptual information from video streaming and point clouds of the pipe inner surface (cement lining) can also be recorded with the RGB camera and the 3D structure sensor mounted at the front of the robot. The latter in particular allows mapping the geometry of the pipe inner surface in order to evaluate the surface unevenness, variation in the nominal pipe diameter and mapping the structure of in-pipe features (chainage, off-takes, valves). Moreover, reconstructing the inner surface profile has the advantage that it enables identifying and locating uncharted coarse anomalies present on the cement lining surface which may impede motion of ILI tools. An example of the latter was apparent during one of the inspections where the robot encountered an abnormality in the form of pipe narrowing during the experiment is shown in Fig. 14b. The mean diameter of the narrow region was observed to be 579 mm while the expected nominal diameter of the cement lined inner surface is expected to be close to 600 mm. A posterior excavation found an uncharted outer clamp, shown in Fig. 14c, whose existence was not known to the utility.

5 Concluding Remarks

An in-line robotic solution for the inspection of buried critical water mains and its evaluation during field deployments has been presented in this paper. A singular kinematic locomotion design that optimizes mobility in such tubular environments has been coupled with an embedded NDT sensing solution based on PEC for measurements unsusceptible to sensor lift-off, as typically found in cement lined water pipelines. The device addresses a utility sector need for an automatic NDT inspection vehicle that can report dense pipe wall thickness discrimination as prescribed by failure prediction analysis, and that can be deployed in an opportunistic manner—e.g. when a mains break occurs, or during valve inspection or repair programs when pipelines are discharged and access made available. Extensive results have proven

the validity of the solution on laboratory tests and field pipeline inspections which demonstrate the feasibility of the device and sensing configuration to provide meaningful 2.5D geometric maps.

References

1. Ji, J., Robert, D., Zhang, C., Zhang, D., Kodikara, J.: Probabilistic physical modelling of corroded cast iron pipes for lifetime prediction. *Struct. Saf.* **64**, 62–75 (2017)
2. Liu, Z., Kleiner, Y.: State of the art review of inspection technologies for condition assessment of water pipes. *Measurement* **46**(1), 1–15 (2013)
3. Huang, C., Xinjun, W., Zhiyuan, X., Kang, Y.: Pulsed eddy current signal processing method for signal denoising in ferromagnetic plate testing. *NDT E Int.* **43**(7), 648–653 (2010)
4. Xu, Z., Wu, X., Li, J., Kang, Y.: Assessment of wall thinning in insulated ferromagnetic pipes using the time-to-peak of differential pulsed eddy-current testing signals. *NDT E Int.* **51**, 24–29 (2012)
5. Huang, C., Wu, X.: An improved ferromagnetic material pulsed eddy current testing signal processing method based on numerical cumulative integration. *NDT E Int.* **69**, 35–39 (2015)
6. Ulapane, N., Alempijevic, A., Vidal-Calleja, T., Miro, J.V., Rudd, J., Roubal, M.: Gaussian process for interpreting pulsed eddy current signals for ferromagnetic pipe profiling. In: Conference on Industrial Electronics and Applications, pp. 1762–1767 (2014)
7. Skinner, B., Vidal-Calleja, T., Miro, J.V., De Brujin, F., Falque, R.: 3D point cloud upsampling for accurate reconstruction of dense 2.5D thickness maps. In: Australasian Conference on Robotics and Automation, p. 7 (2014)
8. Xie, L., Scheifele, C., Xu, W., Stol, K.A.: Heavy-duty omni-directional mecanum-wheeled robot for autonomous navigation: System development and simulation realization. In: International Conference on Mechatronics, pp. 256–261. IEEE (2015)
9. Taheri, H., Qiao, B., Ghaeminezhad, N.: Kinematic model of a four mecanum wheeled mobile robot. *Int. J. Comput. Appl.* **113**, 6–9 (2015)
10. Miro, J.V., Rajalingam, J., Vidal-Calleja, T., de Brujin, F., Wood, R., Vitanage, D., Ulapane, N., Wijerathna, B., Su, D.: A live test-bed for the advancement of condition assessment and failure prediction research on critical pipes. *Water Asset Manage. Int.* **10**(2), 3–8 (2014)
11. Rasmussen, C.E., Williams, C.K.: Gaussian Processes for Machine Learning. The MIT Press, MA, USA (2006)
12. Shi, L., Miro, J.V.: Towards optimized and reconstructable sampling inspection of pipe integrity for improved efficiency of NDT. In: World Water Congress, IWA, p. 8 (2016)

Real-Time Semantic Mapping for Autonomous Off-Road Navigation

Daniel Maturana, Po-Wei Chou, Masashi Uenoyama
and Sebastian Scherer

Abstract In this paper we describe a semantic mapping system for autonomous off-road driving with an All-Terrain Vehicle (ATVs). The system’s goal is to provide a richer representation of the environment than a purely geometric map, allowing it to distinguish, e.g., tall grass from obstacles. The system builds a 2.5D grid map encoding both geometric (terrain height) and semantic information (navigation-relevant classes such as *trail*, *grass*, etc.). The geometric and semantic information are estimated online and in real-time from LiDAR and image sensor data, respectively. Using this semantic map, motion planners can create semantically aware trajectories. To achieve robust and efficient semantic segmentation, we design a custom Convolutional Neural Network (CNN) and train it with a novel dataset of labelled off-road imagery built for this purpose. We evaluate our semantic segmentation offline, showing comparable performance to the state of the art with slightly lower latency. We also show closed-loop field results with an autonomous ATV driving over challenging off-road terrain by using the semantic map in conjunction with a simple path planner. Our models and labelled dataset will be publicly available at <http://dimatura.net/offroad>.

1 Introduction

The last few years have seen enormous progress in the 3D sensing capabilities of autonomous vehicles. Mature and robust LiDAR and INS technologies give self-driving vehicles an accurate and real-time sense of the geometric structure around them, immensely simplifying navigation-related tasks.

However, we have observed that relying primarily on geometric information leads to disappointing results for autonomous navigation in off-road environments. The

D. Maturana (✉) · P. -W. Chou · S. Scherer
Robotics Institute, Carnegie Mellon University, Pittsburgh, USA
e-mail: dimatura@cmu.edu

M. Uenoyama
Yamaha Motor Corporation, Cypress, USA
e-mail: mike_uenoyama@yamaha-motor.com

Fig. 1 Our autonomous all-terrain vehicle (ATV). The two main sensors, a spinning 3D LiDAR and a stereo camera, can be seen on the vehicle roof



main reason is that geometric structure, by itself, fails to provide many important distinctions for wheeled All-Terrain Vehicles (ATVs) such as ours, shown in Fig. 1. For example, tall grass may be perceived as an obstacle, but our ATV may traverse it if desired. Similarly, leaf litter may appear as rocky terrain, or puddles may appear as either holes or smooth surfaces. All of these may lead to suboptimal, even dangerous, decisions in path planning. Similar observations have been made many times before in the context of off-road robotics, e.g., [11, 12, 17, 24].

In this paper, we describe a system to counter this problem by building a *semantic map*, a representation of the vehicle's surroundings encoding both geometric (e.g., height, roughness) and semantic information (navigation-relevant classes such as trail, grass, obstacle, etc.). The map is stored as a 2.5D grid centered on the vehicle frame and is continuously updated as new sensor data is acquired. Using this representation, a motion planner can create semantically-aware trajectories.

Our key contribution is a simple yet effective system coupling a custom Convolutional Neural Network architecture, based on Fully Convolutional Networks [16], and a 2.5D vehicle-centered semantic grid map that fuses the geometric and semantic measurements as the vehicle moves and acquires more data. We show the effectiveness of the semantic segmentation CNN in offline benchmarks. By using a simple planner with the semantic map, we show qualitative examples of our system being successfully used to navigate challenging off-road terrain.

As additional contributions, the labeled dataset of off-road imagery used to train our network, as well as our semantic segmentation code, will be made publicly available. See the project website, <http://dimatura.net/offroad> for links.

2 Related Work

Our system is heavily inspired by the rich literature on semantic approaches to off-road navigation tasks, going as far back as 1990 [7].

A decade later, various practical systems showed impressive results with this paradigm, usually with a combination of LiDAR and images [12, 17, 24, 27, 28].

The LAGR program [11] featured various highly relevant systems such as [2, 13, 20, 26], which performed semantic classification with hand-engineered vision pipelines. An exception is [9], featuring an early deep neural network system for semantic segmentation.

In more recent work, [23] demonstrate autonomous navigation featuring a light-weight semantic segmentation system. Unlike our system, they use traditional visual feature engineering, leading to noisy pixel-wise predictions that they smooth with a novel regularization method. In contrast, our architecture, based on Fully Convolutional Networks (FCNs) [16], incorporates spatial context that naturally smoothes the output. Another relevant work is [25], which uses an encoder-decoder network architecture that is similar to FCNs. They explore modalities beyond RGB and achieve impressive segmentation results. However, they do not build a metric map or demonstrate closed-loop navigation.

3 Approach

Overview. This system architecture is outlined in Fig. 2. There are two primary sensor streams, RGB imagery and LiDAR point cloud data. The RGB images are fed into the semantic segmentation module, which uses a CNN to generate a pixel-wise labeling. Concurrently, the LiDAR point clouds are used to update a 2.5D grid map in the semantic mapping module. The semantic mapping module also receives the pixel-wise prediction images from the semantic segmentation module and projects

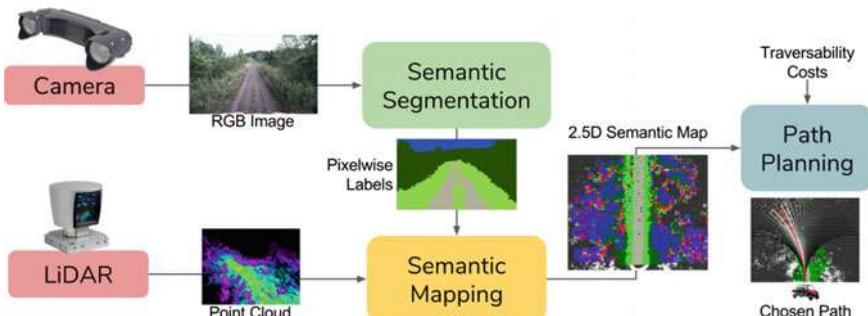


Fig. 2 Overview of our semantic mapping system

them onto the 2.5D grid map, which fuses the semantic predictions over time. The result is a vehicle-centered 2.5D grid map encoding continuously updated estimates of relevant geometric and semantic information for off-road navigation. Finally, the map is used for semantically-aware path planning. For our initial testing, we used a simple receding horizon path planner that assigns a traversal cost to each semantic class and continuously chooses a path to minimize the cost. The whole system runs at 10Hz, a rate dictated by the speed at which the semantic mapping module processes images.

Hardware Platform. Our vehicle is shown in Fig. 1. It is a commercial All-Terrain Vehicle modified and instrumented for experiments in autonomous off-road driving. The sensor suite includes an INS/GPS system, a 64-line Velodyne LiDAR and an RGB stereo camera with a 21 cm baseline manufactured by Carnegie Robotics. Note that the system in this paper does not currently use the stereo depth information. All computation is performed onboard with two COTS laptops, connected through high-speed ethernet. The laptop for semantic mapping includes an NVIDIA GT980M GPU, used to achieve real-time execution of the CNN classifier.

Software Platform. All computers run Ubuntu Linux. The different system modules run concurrently as ROS nodes and communicate through ROS messages. The nodes are implemented in C++ and Python, using CUDA (generated via the Theano library [3]) to make effective use of the GPU.

3.1 Semantic Segmentation

The goal of 2D semantic segmentation is to assign one of K predefined classes to each pixel in an image. Like many tasks in computer vision, the state of the art for this task has been recently revolutionized by Deep Learning, and in particular Convolutional Neural Networks.

For this task, the most successful neural networks architectures are Fully Convolutional Network (FCNs) [16]. The key idea in these networks is to take advantage of convolutional structure to label all the pixels simultaneously with a very similar network to more traditional CNNs. Due to pooling, this results in low-resolution outputs; to reverse this, so-called “deconvolution” layers are added to upsample the output. In order to preserve high-frequency detail, skip layers connecting early layers to upsampled feature maps are added. Encoder-Decoder architectures [1, 19], of which UpNet [25] is an example, are similar but omit skip layers.

At the start of the project, we found state of the art architectures to be relatively slow, as they were optimized for accuracy over speed. Thus we implemented and trained our own architectures, using the Theano [3] and Lasagne [6] libraries. We have found various possible architectures to show very similar accuracy for our off-road semantic segmentations tasks, differing mostly in time cost, which in turn is largely driven by details of the architecture and input/output resolution. We believe this is due to the relatively small datasets used.

Name	Units	Size	Stride	Input	Name	Units	Size	Stride	Input	Name	Units	Size	Stride	Input
conv1	96	7	2	RGB Image	conv01	32	3	1	RGB Img	conv16	1024	3	1	
norm1		2	2		pool01	2	2			conv17	512	1	1	
pool1	2	5	1		conv02	64	3	1		conv18	1024	3	1	
conv2	256	2	2		pool02	2	2			conv18nin	64			
pool2	2	2	2		conv03	128	3	1		up1	64	4	2	
conv3	512	3	1		conv04	64	1	1		pool04min	64			pool04
conv4	512	3	1		conv05	128	3	1		fuse1				up1 + pool04nin
conv5	512	3	1		pool03	2	2			up2	64	4	2	
pool5	3	3	1		conv06	256	3	1		pool03nin	64			pool03
convfc6	4096	6	1		conv07	128	1	1		fuse2				up2 + pool03nin
convfc7	4096	1	1		conv08	256	3	1		up3	64	4	2	
ninfc7	6				pool04	2	2			pool02nin	64			pool02
up1	6	4	2		conv09	512	3	1		fuse3				up3 + pool02nin
conv5min	6			conv5	conv10	256	1	1		up4	64	4	2	
fuse1				conv11	512	3	1		up4nin	8			up4	
up2	6	4	2		conv12	256	1	1						
pool1nin	6				conv13	512	3	1						
fuse2					conv14	1024	3	1						
up3	6	5	3		conv15	512	1	1						
conv1nin	6				...									
fuse3														

Fig. 3 Our two network architectures. “conv” denotes a convolutional layer; “pool”, a pooling layer; layers ending in “nin” are 1×1 convolutional layers; “fuse” is an elementwise sum layer; “up” is an upsampling deconvolution layer; “norm” is a local response normalization layer. The input is assumed to be the layer above, unless otherwise specified. For convolutional layers, “size” is the kernel size; for pooling layers, it is the pooling receptive field. Note that for **dark-fcn** we split the table due to space constraints

We use two architectures. The first, **cnns-fcn**, is based on our “convolutionalization” of VGG-CNNs from [4], and has 227×227 input size with 109×109 output size. The second, **dark-fcn**, is based on our convolutionalization of the Darknet architecture [21], which in turn is similar but more efficient than VGG16 [22]. For **dark-fcn** both the input and output are 300×300 , in order to facilitate comparison with UpNet. Despite the higher resolution **dark-fcn** is faster than **cnns-fcn**: 21 ms on a GT980M, compared to 37 ms. The authors of UpNet [25] describe a 50 ms with Caffe on a GTX Titan X, which in our experience has similar speeds to the GT980M. This leads us to believe our model should be faster or at least comparable. Figure 3 shows both of our architectures. Code and trained models will also be made available.

3.2 Semantic Mapping

The output of the semantic mapping step is in 2D image space, but it is far more natural for vehicles to plan in a 3D, metric space. In our case, we adopt a 2.5D grid with each grid cell containing estimated height, i.e., a height map. This suffices for many environments, but would potentially have issues with overhanging trees or tunnels.

To keep an up-to-date height map of the vehicle’s surroundings, we use a scrolling grid data structure, which has been reinvented multiple times in the literature. This structure is a generalization of ring-buffers to two dimensions, and its main feature is that it can be shifted (translated) without copying its data, and instead updating the

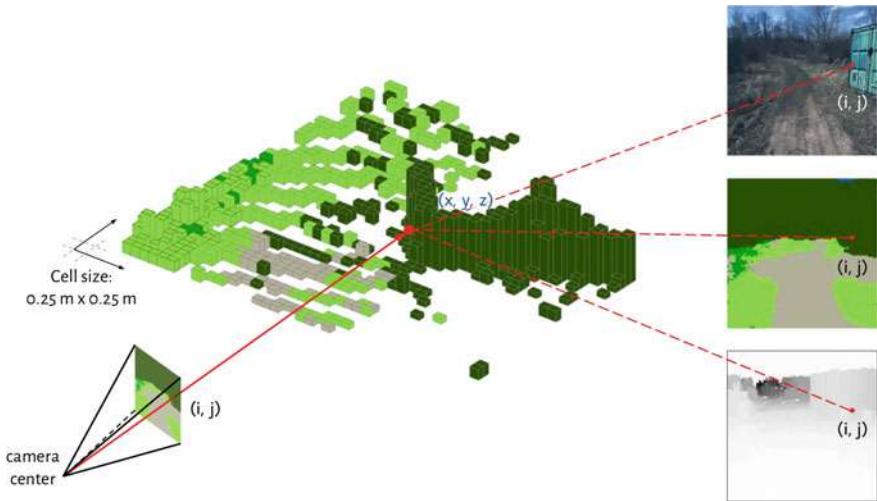


Fig. 4 Mapping the semantic segmentation to the 2.5D map

variables indicating its limits. This is a speed optimization; logically, the grid behaves like a finite 2D array centered around the vehicle, with each grid cell containing various properties about the terrain. In our paper the grid cells are $0.25 \text{ m} \times 0.25 \text{ m}$ each, and the map has 400×400 cells. Each grid cell maintains a running estimate of the minimum and maximum height in that grid cell, computed by using occupancy and free-space constraints derived from LiDAR rays, similar to [8, 30]. For each point in the point cloud, we raytrace on our grid using Bresenham’s algorithm in 3D; cells that are passed through, and above, are considered empty, and cells where the beam stops, and below, are considered occupied.

The semantic map also integrates semantic measurements, as its name indicates. To project the output of the 2D semantic segmentation into a height map representation, we follow a straightforward process depicted in Fig. 4.

Given that we know the relative position of the camera and the LiDAR, and the camera intrinsics, we can project the 2D semantic predictions onto the 2.5D grid cells using simple geometry. However, for added robustness, we fuse measurements over time. To this end we adopt a scheme inspired by the sequential filtering process of occupancy maps [18], but generalized to K classes.

For this, we use the probabilistic (softmax) pixel-wise output of the classifier. We maintain a running sum of the log odds of the K classes projected to each grid cell. While this soft multiclass representation could be used directly, for simplicity when interfacing with other systems, we use the argmax of the K classes as our current best estimate of the semantic class for each grid cell. Note that this representation assumes a single class per cell, which may be a limitation in certain environments.

An example cumulative output of the semantic map in a live field run is shown in Fig. 5.

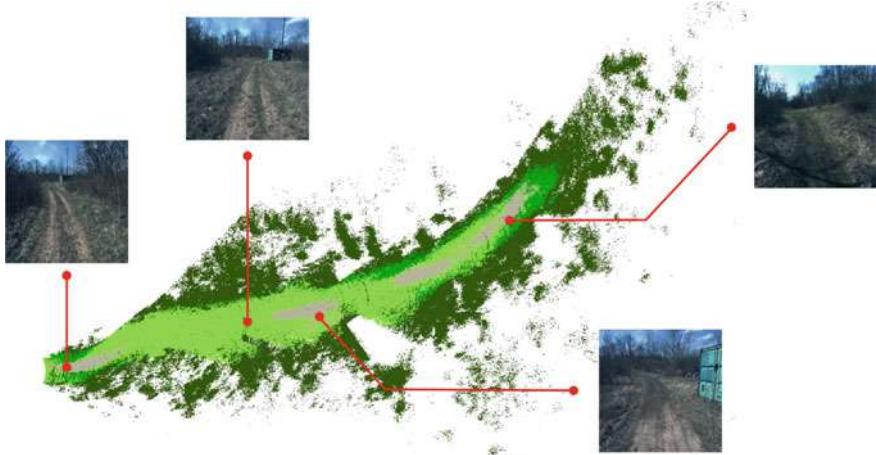


Fig. 5 Example output of semantic map in a field run

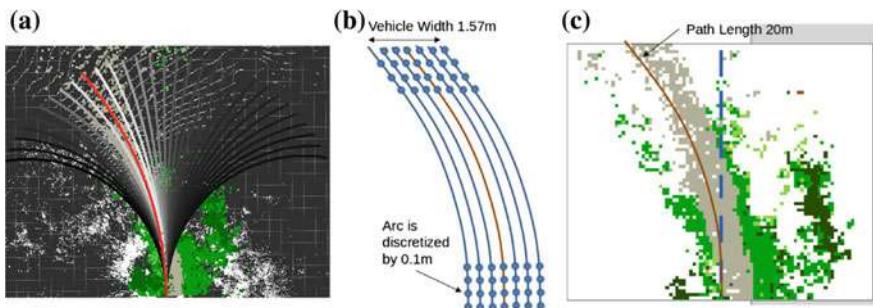


Fig. 6 Path planning. **a** Library of candidate paths, overlaid on top of the semantic map. Red indicates feasible paths. **b** Illustration of how we account for vehicle width. For each trajectory, we compute the cost (or reward) over seven shifted versions of the trajectory, covering the vehicle footprint. **c** An example of a chosen trajectory, chosen according to the traversability score of the semantic classes it covers

3.3 Path Planning

In order to demonstrate autonomous operation, we implement an extremely simple receding horizon path planner. The planner has a library of 30 trajectories corresponding to yaw rates of $-15^{\circ}/\text{s}$ to $15^{\circ}/\text{s}$, discretized at $1^{\circ}/\text{s}$, and at constant velocity of 9 km h^{-1} ; see Fig. 6a).

Each time the map is updated, which happens at 10 Hz, a trajectory is chosen from the library. The choice of trajectory maximizes a reward function derived from the semantic map as follows. Cells labeled as “smooth” or “rough” trail have a reward of 1, and cells labeled as “grass” have a reward of 0.1. All other classes have zero

reward. The total reward of a trajectory is the sum of rewards over a 20 m trajectory length, originating from the vehicle. To account for vehicle width, we slightly modify this calculation, as shown in Fig. 6b).

The advantage of this planner is that in its extreme simplicity, its performance depends largely on the output of our semantic mapping, with no interference from other factors that will be present in a more complex, multi-layered system. However, our system was also used as an additional input to a more deliberative proprietary planner, for which the main representation was a geometric map built with LiDAR. In this planner, our semantic predictions were used primarily to avoid treating grass on and near trails as obstacles, enabling operation on narrow trails.

4 Experiments

Overview. We evaluate our system in two ways. First, we run offline benchmarks of the semantic segmentation module in two datasets. Second, we demonstrate the whole system operating autonomously live field experiments.

4.1 Offline Benchmarks

In order to evaluate our semantic segmentation module, we use two datasets, the DeepScene dataset from Valada et al. [25] and our own dataset, the Yamaha-CMU Off-Road Dataset.

DeepScene Dataset. This dataset consists of 233 training images and 139 validation images of off-road imagery densely labeled with six semantic categories: void, road, grass, vegetation, tree, sky, and obstacle. While this dataset shows some interesting variety in appearance due to the time of day, it is fairly small and seems to lack diversity in terms of weather and location. A key feature of this dataset is various modalities (depth, NIR), but we do not currently make use of them.

Yamaha-CMU Off-Road Dataset. In order to train and evaluate our method we have collected our own dataset, which we call Yamaha-CMU-Off-Road, or YCOR. It consists of 1076 images collected in four different locations in Western Pennsylvania and Ohio (Fig. 8), spanning three different seasons (Fig. 7). The dataset was labeled using a polygon-based interface with eight classes: sky, rough trail, smooth trail, traversable grass, high vegetation, non-traversable low vegetation, obstacle. The polygon labels were post-processed using a Dense CRF [15] to densify the labels; the output of the CRF was manually inspected, and in some cases corrected, to ensure no wrong labels were created.

We believe our dataset is more diverse and challenging than DeepScene. In Fig. 8, we show the mean RGB image and pixel-wise label mode of each dataset. The DeepScene dataset shows a left-right bias and more predictable structure than ours; if



Fig. 7 Montage of frames from our dataset collection

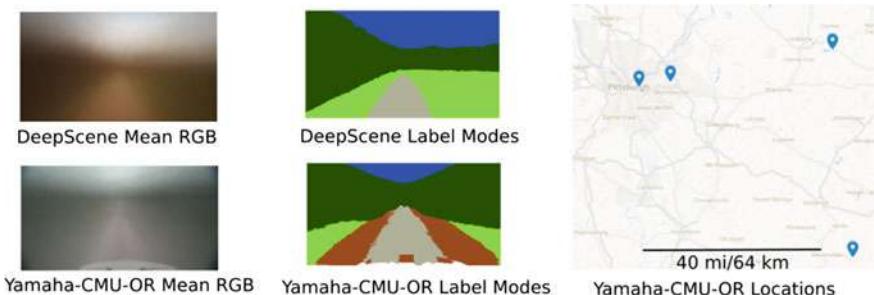


Fig. 8 First two columns: a comparison of dataset statistics. We show the mean RGB frame and the pixel-wise mode for the labeled frames in the training sets of each dataset used. Last column: a map with locations where YCOR was collected

we used the pixel-wise mode as a baseline classifier, we would obtain 0.30 pixel-wise error-rate in DeepScene, but 0.51 in ours. However, we acknowledge that compared to recent efforts, both datasets are relatively small; cf. CityScapes [5], with 25000 labeled images.

Our current split has 931 training images, and 145 validation images. This split was generated randomly, ensuring there was no overlap in data collection session between images in the training and validation split. However, there is overlap in

locations used. We will provide location and time of acquisition metadata to enable further evaluation regarding generalization across these factors.

Quantitative Results. We evaluated our models on the two datasets. In each case, we train our models from scratch on the predefined training set until convergence with SGD, dividing by the initial learning rate (0.0001) by a factor of 10 three times. We use a standard pixel-wise cross-entropy loss with a small L_2 regularization factor (0.0005). Training takes around two days on a GT980Ti GPU. We use crop, rotation and color augmentations at training time, and none at test time. We use per-class intersection over union (IoU) as the evaluation metric, the most common metric for semantic segmentation.

Table 1 shows results for DeepScene and Table 2 shows results for YCOR. In both, we include a variant of the dark-fcn model with 448×448 resolution, in addition to the standard 300×300 . We report the numbers from their paper [25], where we denote by frequency-weighted IoU (fw-IoU) what they denote as IoU, and add mean IoU (mIoU), calculated by ourselves. As we can see, both our models perform comparably, with dark-fcn having a slight advantage. In the DeepScene dataset we can also compare the two models with the RGB UpNet. We see that our models have a slight edge in fw-IoU, though they display dramatically worse performance for obstacles, which severely skews the mIoU metric. We note that the number of obstacle pixels in the dataset is three orders of magnitude less than for the other classes, so the network tends to ignore it. A similar situation occurs with puddles in YCOR. Nonetheless, it is an important class, and we are investigating how to detect it more accurately. Finally, we see that increasing the input resolution gives a slight boost in performance.

Qualitative Results. We show some qualitative labelings of the cnns-fcn architecture for each dataset in Fig. 9. As can be seen, the results are generally accurate. For the YCOR, most of the confusions come from smooth vs. rough trail, a distinction that is hard for humans to make consistently.

Table 1 Per-class, mean IoU and frequency-weighted IoU of UpNet (RGB) and our models in DeepScene dataset. The first three rows use a 300×300 image size, as in UpNet; the last row uses 448×448

	Road	Grass	Veg./tree	Sky	Obstacle	mIoU	fw-IoU
Upnet (RGB) [25]	85.03	86.78	90.90	90.39	45.31	79.68	85.30
cnns-fcn	85.95	85.34	87.38	90.53	1.84	58.51	87.47
dark-fcn	88.03	86.65	89.21	93.17	5.03	60.35	89.41
dark-fcn-448	88.80	87.41	89.46	93.35	4.61	60.61	89.85

Table 2 Per-class, mean IoU, and frequency-weighted IoU of our models in the YCOR dataset

	Smooth	Grass	Rough	Puddle	Obstacle	Low veg.	High veg.	Sky	mIoU	fw-IoU
cnns-fcn	46.70	64.03	38.29	0.0	32.74	24.32	79.15	88.01	46.66	61.31
dark-fcn	46.24	71.25	41.35	0.0	29.74	28.17	80.15	91.45	48.54	63.62
dark-fcn-448	52.46	72.11	39.61	0.0	35.56	24.61	82.51	92.69	49.82	65.18

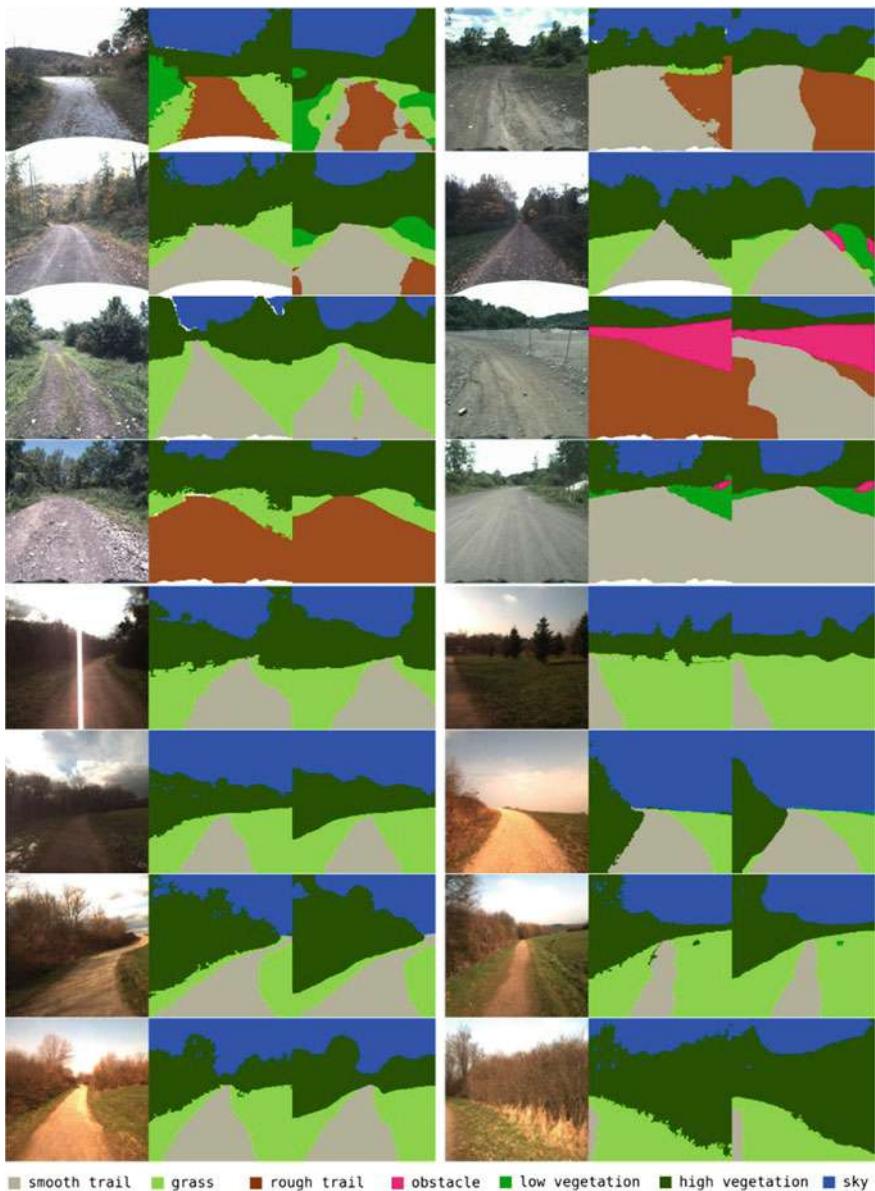


Fig. 9 Montage of predictions from cnns-fcn in the YCOR dataset (top four rows) and Deep-Scene (bottom four rows). In each case, we show three images: input, ground truth labels, and predicted labels

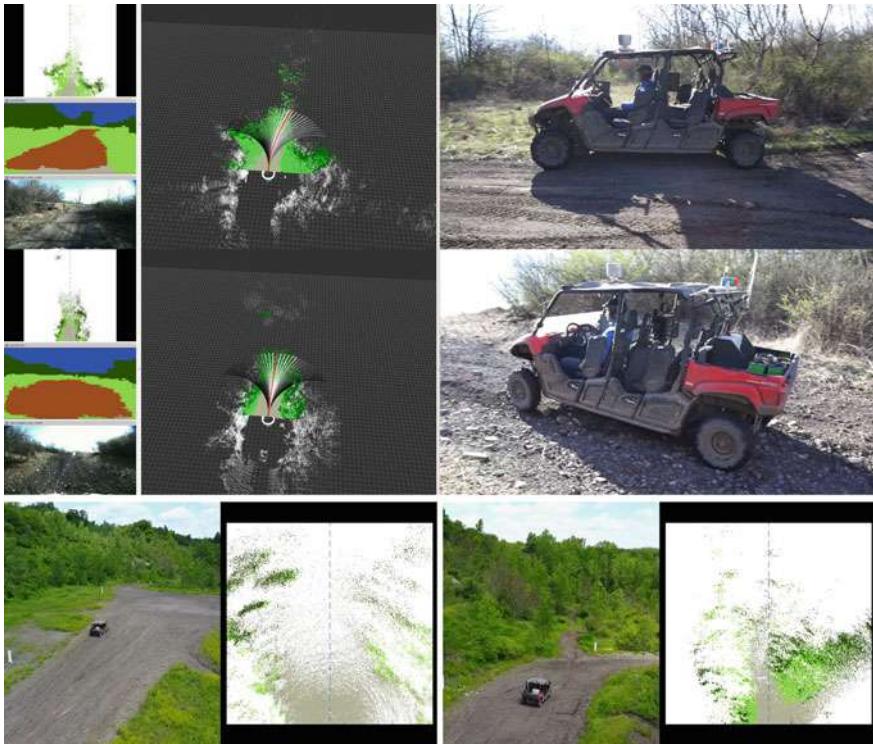


Fig. 10 Action shots of autonomous off-road driving in our testing site. In the first two rows, the left-hand side shows screenshots of the sensor data and map as seen from the vehicle. The last row shows aerial action shots with the right-hand showing the semantic map. Videos will be made available on the project page

4.2 Field Experiments

We performed various self-driving experiments in March and July 2017, in various locations around our testing site near Pittsburgh, PA. The terrain traversed including steep slopes, rocky and muddy terrain, puddles, and vegetation of various heights surrounding and covering the trails. Despite the simplicity of our planner, the vehicle managed to successfully traverse various trails that were too challenging for a previous LiDAR-only system. These include locations with puddles, grass in the middle of the trail, and narrow trails. Video is available in the project website. Figure 10 shows the vehicle in autonomous operation.

On the other hand, we observed some limitations of our current system. Many of the limitations were due to the simplicity of the receding horizon planner, which often swerved from side-to-side in wider trails.

Some of the failures were also due to our semantic classification system. For example, it sometimes failed to detect sparse grass alongside the trail, resulting in

the vehicle veering off-trail. In one occasion, it also confused a large non-traversable bush with traversable grass, forcing us to manually intervene.

While we maintained a nominal speed of 9 km h^{-1} , the velocity in practice varied by a kilometers per hour; when traveling at more than 12 km h^{-1} , we occasionally observed the map would not update in time to make correct planning decisions, again resulting in failures to react appropriately.

More extensive testing was performed with the proprietary deliberative planner. In trials traversing more than 100 km, we observed far more stable operation.

Timing. We run all computation onboard the vehicle using an i7 laptop with a 6GB GT980M GPU. The bottleneck of the system is in the raytracing operation of semantic mapping, with semantic segmentation taking approximately 35 ms per image and the label projection taking around 60 ms per image. These steps occur sequentially, leading to the roughly 10 Hz rate operation of the system. This is sufficient for medium-speed operation, but there is ample space to optimize performance further to support faster driving and/or more limited computing platforms.

5 Conclusions

We have introduced an efficient and robust semantic mapping system for off-road navigation featuring a state-of-the-art CNN classifier. To train the CNN, we have collected and labelled a new dataset of off-road imagery. We have evaluated it in offline benchmarks with results comparable to the state of the art with lower latencies. We have also demonstrated closed-loop operation in challenging off-road terrain.

In future work, we are interested in incorporating recent advances from the state of the art in semantic segmentation, such as Dilation layers [29], pyramid spatial pooling [10]. We are also evaluating the contribution of multiple input modalities, including an approach jointly using LiDAR and imagery for segmentation [14].

Having verified firsthand the difficulty of accurately labelling large amounts of data, in future work we are interested in alternatives to manual labelling, such as self-supervision and inverse reinforcement learning.

Acknowledgements We gratefully acknowledge Yamaha Motor Corporation for their support, the Yamaha-CMU team for building and maintaining the autonomous ATV, and Mesh Robotics for field testing data.

References

1. Badrinarayanan, V., Handa, A., Cipolla, R.: SegNet: a deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling (2015). [arXiv:1505.07293](https://arxiv.org/abs/1505.07293)
2. Bajracharya, M., Howard, A., Matthies, L.H., Tang, B., Turmon, M.: Autonomous off-road navigation with end-to-end learning for the lagr program. *J. Field Robot.* **26**(1), 3–25 (2009)

3. Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., Bengio, Y.: Theano: a CPU and GPU math expression compiler. In: SciPy (2010)
4. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: delving deep into convolutional nets. CoRR (2014). [arXiv:1405.3531](https://arxiv.org/abs/1405.3531)
5. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
6. Dieleman, S., Schlüter, J., Raffel, C., Olson, E., Sonderby, S.K., Nouri, D., et al.: Lasagne: first release (2015). <https://doi.org/10.5281/zenodo.27878>
7. Fujimori, T., Kanade, T.: An approach to knowledge-based interpretation of outdoor natural color road scenes. In: Vision and Navigation: The Carnegie Mellon Navlab, pp. 39–81 (1990)
8. Hadsell, R., Bagnell, J.A., Huber, D., Hebert, M.: Non-stationary space-carving kernels for accurate rough terrain estimation. IJRR **29**(8), 981–996 (2010)
9. Hadsell, R., Erkan, A., Sermanet, P., Scoffier, M., Muller, U.: Deep belief net learning in a long-range vision system for autonomous off-road driving. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 1(1), pp. 628–633 (2008)
10. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. CoRR (2014). [arXiv:1406.4729](https://arxiv.org/abs/1406.4729)
11. Jackel, L.D., Krotkov, E., Perschbacher, M., Pippine, J., Sullivan, C.: The DARPA LAGR program: goals, challenges, methodology, and phase i results. J. Field Robot. **23**(11–12), 945–973 (2006)
12. Kelly, A., Stentz, A.T., Amidi, O., Bode, M.W., Bradley, D., Diaz-Calderon, A., Happold, M., Herman, H., Mandelbaum, R., Pilarski, T., Rander, P., Thayer, S., Vallidis, N.M., Warner, R.: Toward reliable off road autonomous vehicles operating in challenging environments. Int. J. Robot. Res. **25**(5–6), 449–483 (2006)
13. Kim, D., Oh, S.M., Rehg, J.M.: Traversability classification for UGV navigation: a comparison of patch and superpixel representations. In: IEEE International Conference on Intelligent Robots and Systems, pp. 3166–3173 (2007)
14. Kim, D.K., Maturana, D., Uenoyama, M., Scherer, S.: Season-invariant semantic segmentation with a deep multimodal network. In: FSR (2017)
15. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. CoRR (2012). [arXiv:1210.5644](https://arxiv.org/abs/1210.5644)
16. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
17. Manduchi, R., Castano, A., Talukder, A., Matthies, L.: Obstacle detection and terrain classification for autonomous off-road navigation. Auton. Robots **18**(1), 81–102 (2005)
18. Moravec, H., Elfes, A.: High resolution maps from wide angle sonar. In: International Conference on Robotics and Automation (ICRA) (1985)
19. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. CoRR (2015). [arXiv:1505.04366](https://arxiv.org/abs/1505.04366)
20. Procopio, M.J., Kegelmeyer, W.P., Grudic, G.: Terrain segmentation with on-line mixtures of experts for autonomous robot navigation. Image. Rochester, N.Y., pp. 385–397 (2009)
21. Redmon, J.: Darknet: open source neural networks in c. <http://pjreddie.com/darknet/> (2013–2016)
22. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR (2014). [arXiv:1409.1556](https://arxiv.org/abs/1409.1556)
23. Suleymanov, T., Paz, L.M., Pinis, P., Hester, G., Newman, P.: The path less taken: a fast variational approach for scene segmentation used for closed loop control. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3620–3626 (2016)
24. 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.: Winning the DARPA grand challenge. J. Field Robot. (2006)

25. Valada, A., Olivera, G.L., Brox, T., Burgard, W.: Deep multispectral semantic scene understanding of forested environments. In: Proceedings of the 2016 International Symposium on Experimental Robotics (ISER) (2016)
26. Vernaza, P., Taskar, B., Lee, D.D.: Online, self-supervised terrain classification via discriminatively trained submodular Markov random fields. In: 2008 IEEE International Conference on Robotics and Automation, pp. 2750–2757. IEEE (2008)
27. Wellington, C., Courville, A., Stentz, A.: A generative model of terrain for autonomous navigation in vegetation. *Int. J. Robot. Res.* **25**(12), 1287–1304 (2006)
28. Wolf, D.F., Sukhatme, G.S.: Semantic mapping using mobile robots. *IEEE Trans. Robot.* **24**(2), 245–258 (2008)
29. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. *CoRR* (2015). [arXiv:1511.07122](https://arxiv.org/abs/1511.07122)
30. Zhou, S., Xi, J., McDaniel, M.W., Nishihata, T., Salesses, P., Iagnemma, K.: Self-supervised learning to visually detect terrain surfaces for autonomous robots operating in forested terrain. *J. Field Robot.* **29**(2), 277–297 (2012)

Boundary Wire Mapping on Autonomous Lawn Mowers

Nils Einecke, Jörg Deigmöller, Keiji Muro and Mathias Franzius

Abstract Currently, the service robot market mainly consists of floor cleaning and lawn mowing robots. While some cleaning robots already feature SLAM technology for the constrained indoor application, autonomous lawn mowers typically use an electric wire for boundary definition and homing towards to charging station. An intermediate step towards SLAM for mowers is mapping of the boundary wire. In this work, we analyze three types of approaches for estimating the boundary of the working area of an autonomous mower: GNSS, visual odometry, and wheel-yaw odometry. We extended the latter with orientation loop closure, which gives the best overall result in estimating the metric shape of the boundary.

1 Introduction

Autonomous lawn mowers are on the verge of a major market in the lawn and garden segment. The segment is still small with an installation volume of 103k units in 2015 [1]. However, the market for autonomous mowers is growing quickly (25% in 2015 [1]), mostly because the robot mowers increase their owner's leisure time, while maintaining good cutting results.

N. Einecke (✉) · J. Deigmöller · M. Franzius
Honda Research Institute Europe, GmbH, Carl-Legien-Strasse 30,
Offenbach/Main 63073, Germany
e-mail: Nils.Einecke@honda-ri.de

J. Deigmöller
e-mail: Jorg.Deigmoller@honda-ri.de

M. Franzius
e-mail: Mathias.Franzius@honda-ri.de

K. Muro
Honda R and D Co., Ltd. Power Products R and D Center, 3-15-1 Senzui, Asaka-shi,
Saitama 351-0024, Japan
e-mail: keiji.muro@h.rd.honda.co.jp

Currently, autonomous vacuum cleaning robots [2–4] and autonomous lawn mowers [2, 5] are the most promising entry points for robots at home. While high-end research robots for household tasks are typically too expensive and not robust enough for 24/7 application, autonomous vacuum cleaning and autonomous lawn mowers have become a robust and stable platform. For vacuum cleaners, there is an active research especially for vision SLAM [6–8]. However, autonomous lawn mowers still lack intelligent functions known from state of the art research like mapping, object recognition, obstacle avoidance, localization, dynamic path planning or speech recognition.

Almost all autonomous lawn mowers use a boundary wire emitting an electromagnetic signal for limiting the working area and for homing to the charging station. Typically, robotic lawn mowers move randomly within the boundary wire. Such a system is simple and reliable and does not require localization during operation. However, finding the charging station, for example, requires searching and following the wire. Direct navigation to the charging station would obviously be more efficient.

In this work, we compare different sensors and methods for estimating a boundary wire map (see Fig. 2), which is an intermediate step towards localization. Given a map as produced by our approach, and measurement of wire signals, localization is a much easier problem than full SLAM, e.g. using Particle Filtering [9]. Localization is the basis for more intelligent behavior of the robot mower, like direct start (move from the base station directly to a user defined position), direct home (directly move to the base station without following the wire), or mowed area bookkeeping (mower keeps track of mowing time in certain zones like front yard).

2 Related Work

Publications specifically on autonomous lawn mowers are scarce. In [10, 11] an omni-directional position estimation system for an autonomous lawn mower was introduced. The authors used an off-the-shelf robotic mower and extended it with an omni-camera and an additional compute unit. In their latest work [11] the equipment was nicely integrated in a near-product fashion. Unfortunately, the outdoor results were not compared with state of the art approaches. Another example for localization on a lawn mower uses odometry, a gyroscope, and an additional RFID system [12]. While detection of the RFID tags can compensate for drift, this system requires additional infrastructure in the garden.

Besides the few mower-specific work there is a large base of research on outdoor navigation. In the automotive domain, the online KITTI benchmark [13] has become a standard tool for the comparison of state of the art visual odometry (VO) methods. The different approaches are evaluated against ground truth trajectories generated by a fusion of GPS and IMU data. In contrast to the KITTI VO benchmark, our focus is on the robot navigation in garden environments which have a different characteristic.

Odometry accumulates drift, which can be corrected by external measurements, such as visual outdoor localization. The robustness of visual localization depends

on invariance against appearance changes on different time scales (e.g. lighting, weather, seasons). Popular local descriptors like SIFT and SURF are prone to fail in these conditions [14]. However, recently there has been increased interest to develop long-term robustness for visual localization (e.g., [15, 16]).

In this publication, we concentrate specifically on garden environments, which are typically relatively small and with poor GNSS reception. Our test gardens were equipped with border wires that allowed method comparisons under precise path repetitions. While camera-based systems have high potentials, we will show that a low-cost wheel odometry system can be competitive or better in these specific settings.

3 Approach

As explained above our target is to accurately estimate the garden layout. Our idea is to let the autonomous mower follow the wire while recording sensor data.

3.1 GNSS

Global navigation satellite systems (GNSS) like GPS can be an easy out-of-the-box solution for many localization problems. However, GNSS localization quality strongly depends on the visibility of satellites. At least four satellites must be visible simultaneously in order to compute the receiver's location. Localization quality also depends on the angular distribution of satellites in the sky, and it can strongly decrease if signals are not received directly but as reflections from nearby objects. Thus, the operating conditions for GNSS in a typical garden environment can be extremely bad, given that trees and buildings often shadow or reflect the direct line-of-sight. We considered three improvements for the accuracy: RTK, an additional base station for differential mode, and IMU integration.

In April 2016 we commissioned a study from the institute of Physical and Satellite Geodesy at Technical University Darmstadt for one of our test gardens to compare a high-end reference GNSS system with a low-cost system. The high-end system was based on two Novatel ProPak-V3 and the cheap system on one u-blox ANTARIS AEK-4T. Average accuracy of the reference system in this garden was just below 0.5 m, while the u-blox ANTARIS AEK-4T often yielded position errors more than 2 magnitudes higher. Only after the end of the study in April 2016, the new cm-precision low-cost differential RTK system u-blox C94M8P became available. We found that this system outperformed the reference system of the earlier study. Hence, we only show results for the new u-blox system.

3.2 Wheel Odometry

In general, the movement of an autonomous lawn mower is described by differential drive kinematics [17]. This means autonomous lawn mowers typically have two independent drive wheels located on one axis. Using the principle of instantaneous center of curvature the movement of the mower system can be computed from just the velocities of the two wheels:

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} \cos(\omega\Delta t) & -\sin(\omega\Delta t) & 0 \\ \sin(\omega\Delta t) & \cos(\omega\Delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R\sin(\theta) \\ -R\cos(\theta) \\ 0 \end{bmatrix} + \begin{bmatrix} R\sin(\theta) \\ R\cos(\theta) \\ \omega\Delta t \end{bmatrix}, \quad (1)$$

where

$$R = \frac{l}{2} \frac{v_r + v_l}{v_r - v_l} \quad (2)$$

$$\omega = \frac{v_r - v_l}{l}. \quad (3)$$

In these equations v_l and v_r are the left and right wheel velocity, respectively. The distance along the axis between the two wheels is l . The actual wheel velocities are calculated from the wheel turning rates.

It is well known that the wheel odometry heavily suffers from wheel slip which is in particular true for autonomous lawn mowers as grass tends to be slippery when wet. Hence, in addition to the wheel rate, we also record the yaw sensor data of the autonomous lawn mower. Since the orientation estimation in the differential drive Eq. (1) is independent of the position estimation, the computed orientation changes $\Delta\theta$ can be replaced by measured orientation changes $\Delta\theta_{yaw}$. In the remainder, we will refer to this as yaw sensor enhanced wheel odometry (yWO).

Unfortunately, the standard differential drive kinematics are defined for 2D movements. However, not all gardens are flat. Thus, we additionally record data from the mower's accelerometer. Please note that in Europe autonomous mowers require to have a tilt sensor in order to automatically switch off the mowing blades in case of a turn over. Typically, this tilt sensor is an accelerometer. Using this data it is possible to rotate the 2D movement computed by means of the differential drive Eq. (1) in 3D such that actually 3D movements can be estimated.

3.3 Visual Odometry

Simply put, visual odometry (VO) is odometry estimation using cameras. The basic idea is to use the apparent image motion in order to derive the movement of the camera. In the single camera case the estimation struggles with a scale problem, i.e. the algorithms have to estimate metric distances by clever assumptions and integration

over time. Using stereo cameras, this problem is directly solved by available depth information. We decided to concentrate on VO based on stereo cameras as the correct metrics are very crucial for our boundary wire estimation target.

We use two different VO approaches in this work: the stereo version of ORB-SLAM [18] and NOTF [19]. ORB-SLAM is a full blown algorithm for simultaneous localization and mapping. VO is just one piece in this complex algorithm. ORB-SLAM also encompasses a key-frame based pose graph construction, local and global refinements using bundle adjustment, 3D map building and place recognition. For our needs mainly the estimated camera path is interesting, however, the other parts of the algorithm indirectly contribute to this path estimation thereby potentially improving it.

In contrast to ORB-SLAM, NOTF is a pure VO algorithm. It does neither involve any kind of temporal filtering like bundle adjustment nor does it build up a map of the environment. Essentially, NOTF is a simple frame by frame estimation framework. Nevertheless, it ranks very high in the KITTI benchmark [13]. Due to its concentration on pure VO and simple nature but high performance on KITTI we decided to include NOTF in our comparison.

3.4 Weighted Loop Closure

Position estimated by wheel odometry or visual odometry both drift over time due to error accumulation in the single movement estimation steps. Full blown SLAM algorithms like ORB-SLAM tackle this problem by relocalization using features stored along with the map data. Once a position is detected as being revisited, a loop closure is done. The idea of loop closing is to realign the position estimates between the two visits of the same point such that the estimated position of the two visits is the same. This leads to a strong improvement of the position estimation for visual SLAM methods compared to pure VO methods. For wheel odometry this is more difficult as the detection of revisiting the same position is very difficult without additional data as from a camera.

The situation in our scenario of boundary wire layout estimation is more favorable. Since the autonomous lawn mower starts and ends in the base station the start and end point are known to be the same. This makes it possible to also compute a loop closure for the wheel odometry.

The naïve approach for loop closing (LC) is to distribute the position error equally to all n estimated positions $p = (x, y)$ within the loop:

$$p_{err} = p_{end} - p_{start} \quad (4)$$

$$\Delta p_i^{corr} = \Delta p_i - \frac{1}{n} p_{err} \quad (5)$$

In our experiments, we found this correction to work but having some issues especially at points where the position of the mower does not change much, e.g. during

turning or standing at the base station. Hence, we propose a weighted error distribution based on the movement distance:

$$d_i = |p_i - p_{i+1}|^2 \quad (6)$$

$$d_{all} = \sum_i d_i \quad (7)$$

$$\Delta p_i^{corr} = \Delta p_i - \frac{d_i}{d_{all}} p_{err} \quad (8)$$

We dubbed this technique weighted position loop closure (wPLC). The advantage of wPLC is a better handling of minor movements and especially of periods of standstill which in the naïve equal distribution lead to a generated drift where actually no movement takes place.

Another advantage of the base station is that, for charging, the mower needs to stand in a defined orientation in the base station. We use this property to also apply an orientation loop closure as we know that the start and end orientation of the mower should also be the same. Similar to wPLC we propose to weight the correction of the error. For the orientations we use the absolute rotation as weighting:

$$\theta_{err} = \theta_{end} - \theta_{start} \quad (9)$$

$$\theta_{all} = \sum_i |\Delta\theta_i| \quad (10)$$

$$\Delta\theta_i^{corr} = \Delta\theta_i - \frac{|\Delta\theta_i|}{\theta_{all}} \theta_{err} \quad (11)$$

We refer to the combined application of weighted orientation and weighted position loop closure as wOPLC. Since the rotation estimation in the differential drive Eq. (1) is independent of the position estimation, it is favorable for the wheel odometry to first calculate the orientation loop closure and only afterwards calculate the position loop closure.

4 Results

4.1 Hardware Setup

For recording realistic data we use an off-the-shelf autonomous lawn mower and extend it with a stereo camera module that also hosts a mass storage for recording the camera data as well as CAN bus data. Figure 1 shows the mower with the camera module attached. The module replaces the maintenance lid and thus seamlessly integrates into the lawn mower body. For stereo image capturing and recording we use a phyFLEX-i.MX6 (Fig. 1, right image, top board), with an industrial grade



(a) Test Unit

(b) Embedded Board

Fig. 1 **a** Autonomous lawn mower with camera module for image capturing and data recording. **b** Internal embedded board phyFLEX-i.MX6 on top of a base board providing interfaces to the cameras and the mower CAN

Freescale i.MX6 quad-core processor running Linux OS, manufactured by Phytec Messtechnik GmbH (Mainz, Germany). The reason for the industrial grade is the expected high burden on the electronics due to heat and moisture in the garden environment. Especially, in high summer heat can be a major issue as active cooling is not possible because the cut grass snippets would clog the air intake or the fan after a short time.

For seamless integration of the computing board with the autonomous lawn mower, we use a custom base board (Fig. 1, right image, bottom board also by Phytec) that provides a CAN interface to the phyFLEX-i.MX6 board and that allows a direct connection to the battery power line of the autonomous mower. The overall power consumption of the whole module is 5–7 W, and the direct connection to the mower power line allows a fully autonomous operation of our prototype as the mower recharges autonomously without user interaction.

To capture stereo images we use two Phytec camera boards that feature an Aptina MT9V024 automotive CMOS chip with a resolution of 752×480 pixels and global shutter at 20 Hz.

4.2 Test Setup

Our primary target is to achieve a good estimation of the outline of the garden shape the autonomous mower is working in. The idea is to let the autonomous mower drive along the boundary wire (in “border cutting mode”) and simultaneously record sensor data for estimating the position of movement of the mower. As explained in Sect. 3, we compare three different methods: wheel odometry, visual odometry and RTK-GPS.



Fig. 2 Aerial images of the five test gardens with the boundary wire layout in white. *Source* Google map

In order to be able to compare the methods, we need representative test gardens and ground truth position data of the boundary wire. To this end, we selected five different gardens in the Frankfurt area that had already an autonomous lawn mower (and accordingly a boundary wire) installed. Figure 2 shows aerial images of the five selected test gardens.

We tried to have test gardens of different size and shape but also of different GNSS occlusion in order to get representative measures for all methods. Furthermore, by selecting gardens with already installed mower and wire we made sure that the wire was placed without being biased by our experiment.

For ground truth generation of the boundary wire layout we had a land surveyor measure the exact position of the wire. We manually marked between 16 and 183 positions on the wire using pegs. In curves we put more pegs than on the straights in order to get a good approximation of the continuous shape of the wire. The position of the pegs were then accurately measured by the land surveyor with a theodolite. The origin of the local position coordinate system was put in the center of the base station and the orientation of the coordinate system was aligned with the heading of the base station (the direction the mower has to enter for charging). Figure 3 shows the ground truth layout of all five test gardens as determined by the land surveyor.

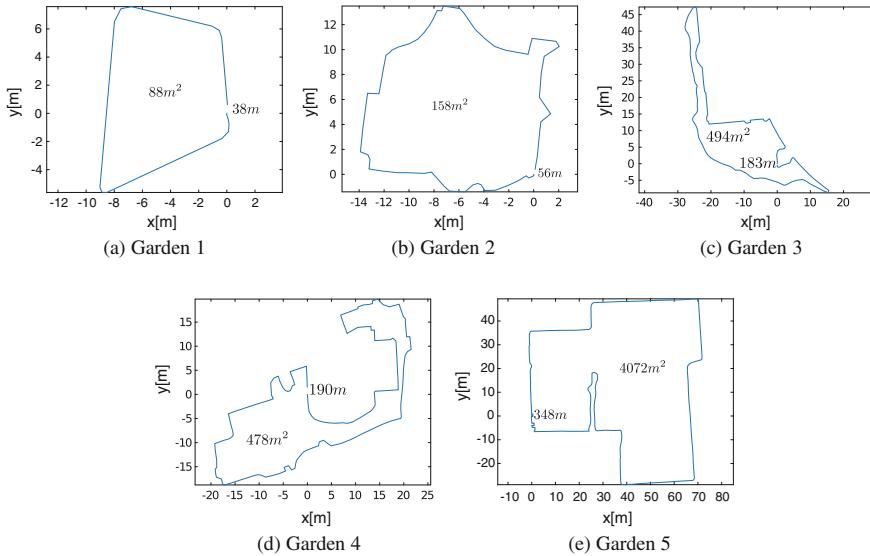


Fig. 3 Boundary wire layout in the five test gardens, as determined by the land surveyor

As can be seen, the size of the gardens varies from 88 to 4072 m^2 and the shape encompass elongated and roundish structures. The length of the wire ranges from 38 m in the smallest garden to 348 m in the largest garden.

The error of the ground truth accuracy is governed by two things. Firstly, it depends on the accuracy of the actual measurement with the theodolite, which is very high ($<1\text{ cm}$). Secondly, there is an error introduced by placing the pegs. Since in some gardens the wire was buried the placing error is about 3 cm as we had to use the mowers internal wire sensor for locating the wire in these cases. Thus, in summary the maximum position error of the ground truth boundary wire layout is 4 cm .

4.3 Results

In each of the five test gardens we did three types of recordings. Firstly, we recorded wheel rate, yaw sensor and accelerometer data for the wheel odometry. This was done by reading the CAN data from the mower, i.e. we used the internal sensor data of the off-the-shelf mower. The recording rate from the CAN bus was 10 Hz . Secondly, we recorded stereo image data for the visual odometry with the Phytec stereo camera. The resolution of the cameras is 752×480 pixels and the images were recorded with 20 Hz . Thirdly, we recorded GPS data with the RTK-GPS system, which was just mounted to the top of the autonomous lawn mower. The recording rate was set to 1 Hz . Please note that the three types of recording were done in separate runs.

For measuring the accuracy of the different methods we compare the estimated trajectory of the mower driving along the boundary wire to the ground truth data. Although we normalized the ground truth data in position and direction according to the position and direction of the base station, we realized that there are minor differences in orientation of the mower when standing in the base station. These minor orientation differences can amount to gross position differences in the large gardens. To tackle this problem, we apply an Iterative Closest Point algorithm (ICP) [20, 21] for all methods between the estimated trajectory and the ground truth wire layout prior to the error calculation. Note that this has no influence on the goal of our evaluation as we are only interested in the shape of the layout and not its correct global position and orientation.

The actual computation of the accuracy is based on point to line distances. Given a trajectory, output by one of the evaluated methods, we first compute the minimal point to line distance for each trajectory point to the polylines of the ground truth data. We then compute the average minimal point to line distance for all points of the ground truth to the polyline of the trajectory. Finally, we take the maximum of both measures. In the discussion below we will just refer to this measure as average distance.

The results of the comparison for all evaluated methods (ORB-SLAM, NOTF, WO and RTK-GPS) are summarized in Table 1. For ORB-SLAM we tested two different variants, the original ORB-SLAM and ORB-SLAM with deactivated global bundle adjustment (BA). Since ORB-SLAM is doing a loop closure and a global BA upon loop detection, we analyzed how much the additional BA is contributing.

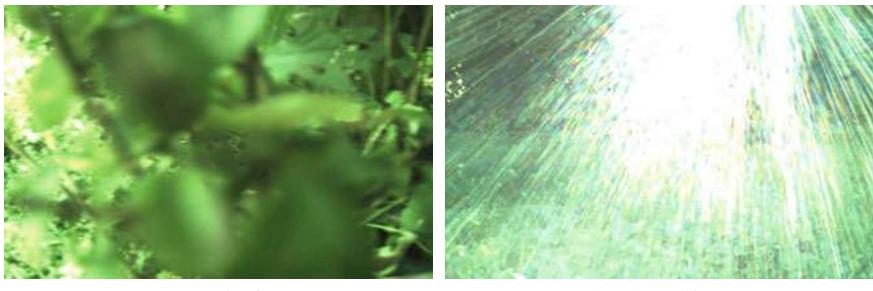
From the results it can be seen that the additional BA has only a major influence in garden 4. The reason for this is that garden 4 posed a particular difficult setting for the visual odometry. The boundary wire in garden 4 was very close to some bushes, leading to heavy occlusion as the mower drives so close that the camera touches the leaves of the bushes (see Fig. 4a). This of course leads to high estimation errors. As the result in Table 1 shows both NOTF and ORB-SLAM without global BA have a very high error in garden 4. BA significantly reduces the error for ORB-SLAM from 2.37 to 0.91 m because the global BA can recover gross misestimations of key frame positions if the surrounding key frames have a good position estimation. Another problem encountered mainly in garden 4 were heavy sun glares (see Fig. 4b). Similar to the occlusions the sun glares lead to wrong feature movements which lead to failure in camera motion estimation.

It is interesting to see that given its simple nature, NOTF is not much worse than ORB SLAM. This means that the internal map building does not yield much more information for the visual odometry than the selection strategy of feature points in NOTF. This is in line with the observations in [19] that even without temporal filtering and map building a high accuracy of VO can be achieved. It should be noted though that we extended the NOTF algorithm with our proposed weighted position loop closure (wPLC) and that the results presented here might not generalize to other application areas where loop detection is not straightforward.

Next we evaluated the RTK-GPS system. The obvious advantage of GPS is that it does not drift with time. However, it is strongly influenced by the 3D layout of the

Table 1 Comparison of visual odometry methods ORB-SLAM and NOTF; plain wheel odometry (WO), yaw sensor enhanced WO with standard loop closure (yWO+LC) and yaw sensor enhanced WO with our proposed weighted orientation and position loop closure (wOPLC); as well as RTK-GPS with altitude information and without. The table entries are average distances between the boundary wire layout estimated by the evaluated methods and the ground truth data. Gardens 1–5 correspond to the gardens seen in Figs. 2 and 3

Method	Garden 1 38 m 88 m^2 error (m)	Garden 2 56 m 158 m^2 error (m)	Garden 3 183 m 494 m^2 error (m)	Garden 4 190 m 478 m^2 error (m)	Garden 5 348 m 4072 m^2 error (m)	Avg. error
ORB SLAM	0.26	0.13	0.38	0.91	1.14	0.56
ORB SLA w/o global BA	0.28	0.21	0.33	2.37	0.95	0.83
NOTF+wPLC	0.22	0.20	0.47	1.46	1.45	0.76
RTK-GPS	0.61	0.57	3.90	1.65	0.20	1.39
RTK-GPS w/o altitude	0.18	0.33	0.68	0.82	0.24	0.45
WO	2.72	1.38	4.19	2.84	12.57	4.74
yWO+LC	0.20	0.15	0.76	0.45	4.08	1.13
yWO + wOPLC	0.19	0.15	0.38	0.24	0.85	0.36



(a) Occlusion

(b) Sun Glare

Fig. 4 Encountered issues with the visual odometry. **a** Occlusion in camera view when the mower drives close to a bush in garden 4. **b** Heavy sun glare caused by direct sun

environment. High structures like buildings or trees block the view to some satellites, which reduces the position accuracy. This is directly reflected in the accuracy of the boundary wire layout estimation. Garden 5 has no higher structures which leads to a very high accuracy of 0.2 m. Gardens 1 and 2 have only some obstruction while gardens 3 and 4 have many high structures leading to some areas in the garden where position estimation is extremely difficult. Looking at the actual position estimations reveals that the error for RTK-GPS is mainly in the altitude, i.e. the height estimates are very noisy. Thus, we also calculated the average distance for the RTK-GPS with

a fixed height (starting height). This leads to a major improvement in almost all gardens.

The last three entries in Table 1 show our results with plain wheel odometry (WO), yaw sensor enhanced WO with standard loop closure (yWO+LC) and yaw sensor enhanced WO with our proposed weighted orientation and position loop closure (yWO+wOPLC). As expected the results with plain WO are not very accurate, mainly due to wheel slip. Using additional yaw sensor data and standard loop closing strongly improves the estimations as the orientation estimation is not influenced by wheel slip anymore. However, in the larger gardens 3–5 the drift of the yaw sensor has a significant impact on the overall accuracy. A visual comparison of the three different WO methods is shown with the resulting maps of garden 5 in Fig. 5.

In the end, we achieved the best overall results with our proposed wOPLC method applied on the yaw sensor enhanced wheel odometry (yWO), which was surprising to us. The evaluation demonstrates that our proposed wOPLC is very capable of correcting the drift error. Of course the error still increases with driven distance as the higher average layout distance in garden 5 indicates. Nevertheless, it shows that todays autonomous lawn mowers could build a map of the boundary wire with just their internal sensors which opens the way to many functions improving the efficiency and perceived intelligence of the autonomous lawn mowers.

Figure 6 shows the trajectories estimated by yWO+wOPLC for all test gardens.

The plots show the high similarity between the layout estimated by yWO+wOPLC trajectories and the ground truth boundary wire layout. In all gardens a small arc is visible close to the origin of the coordinate system that deviates from the ground truth. This is the location of the base station and the arc in the trajectory is caused by the autonomous lawn mower driving around the base station in order to enter it from the correct direction (or to reach the wire after exiting the station).

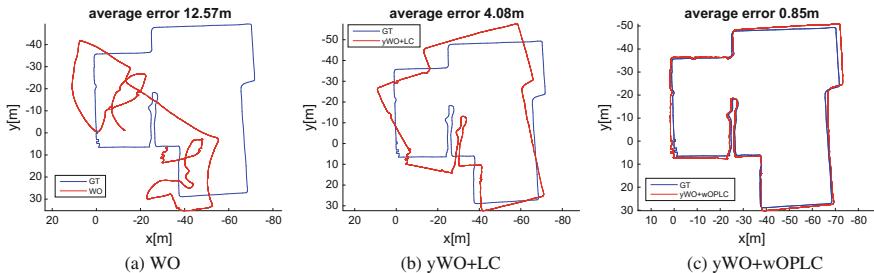


Fig. 5 Comparison of plain wheel odometry (WO), yaw sensor enhanced WO with standard loop closure (yWO+LC) and yaw sensor enhanced WO with our proposed weighted orientation and position loop closure (wOPLC) on the data of garden 5. The blue line is the ground truth layout

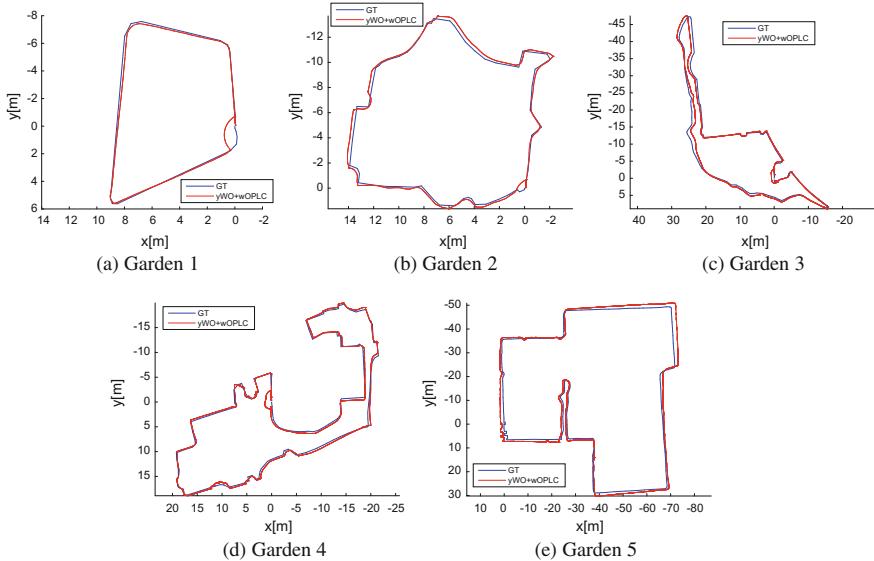


Fig. 6 Results for our proposed yWO+wOPLC method for all five test gardens overlayed with ground truth (GT) layout measured by the land surveyor

5 Summary

We presented a comparison of visual odometry (VO), wheel odometry (WO) and RTK-GPS for estimating the layout of the boundary wire of autonomous lawn mower installations. For the evaluation five test gardens were selected and the wire was measured by a land surveyor for ground truth position data.

The result of our comparison shows that WO together with our proposed weighted loop closure shows the best overall results. VO methods struggle with occlusion of the camera when the mower comes very close to objects like bushes, and with heavy sun glare in certain conditions. Furthermore, VO methods would require artificial lighting in night time operation. RTK-GPS also showed good results when only few high structures are located near the garden. Otherwise the accuracy is lower than VO or WO. In particular the altitude is very noisy and not suitable to estimate the slopes in most gardens.

The good performance of WO may be surprising but also means that current off-the-shelf mowers are capable to estimate a boundary wire map with their already equipped internal sensors. Such maps make localization feasible, which in turn opens the way to intelligent functions like automatic start point definition, mowing schedule optimization, direct homing or mowed area bookkeeping.

In the future, we want to investigate the possibility to fuse multiple wire following trajectories in order to improve the accuracy. Also a fusion of the different methods has the potential for canceling out respective weaknesses. In particular, the fusion of

the WO with GPS would allow a map with global position data, which in turn would help facilitate the localization of the lawn mower within the garden, especially after a user intervention.

Acknowledgements We would like to thank Nico Steinhardt for supervising the GNSS prestudy. Furthermore, we want to thank Hideaki Shimamura and Makoto Yamamura for supporting us with the autonomous lawn mower.

References

1. Hägele, M. (ed): World robotics 2016 service robots. In: International Federation of Robotics Statistical Department (2016)
2. Şahin, H., Güvenç, L.: Household robotics: autonomous devices for vacuuming and lawn mowing. *IEEE Control Syst.* **27**(2), 20–96 (2007)
3. Fink, J., Bauwens, V., Kaplan, F., Dillenbourg, P.: Living with a vacuum cleaning robot. *Int. J. Soc. Robot.* **5**(3), 389–408 (2013)
4. Prassler, E., Munich, M.E., Pirjanian, P., Kosuge, K.: Domestic robotics. In: Springer Handbook of Robotics, pp. 1729–1758 (2016)
5. Hicks, R.W., Hall, E.L.: Survey of robot lawn mowers. In: Intelligent Robots and Computer Vision XIX Algorithms, Techniques, and Active Vision (2000)
6. Jung, M.-J., Myung, H., Hong, S.-G., Park, D.-R., Lee, H.-K., Bang, S.: Structured light 2D range finder for simultaneous localization and map-building (SLAM) in home environments. In: Micro-nanomechatronics and Human Science, and The Fourth Symposium Micro-nanomechatronics for Information-Based Society, pp. 371–376 (2004)
7. Lee, H.-K., Choi, K., Park, J., Myung, H.: Self-calibration of gyro using monocular SLAM for an indoor mobile robot. *Int. J. Control Autom. Syst.* **10**(3), 558–566 (2012)
8. Lee, S., Lee, S., Baek, S.: Vision-based kidnap recovery with SLAM for home cleaning robots. *J. Intell. Robot. Syst.* **67**(1), 7–24 (2012)
9. Thrun, S., Fox, D., Burgard, W., Dellaert, F.: Robust monte carlo localization for mobile robots. *Artif. Intell.* **128**(1–2), 99–141 (2001)
10. Yang, J., Chung, S.-J., Hutchinson, S., Johnson, D., Kise, M.: Vision-based localization and mapping for an autonomous mower. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3655–3662 (2013)
11. Yang, J., Chung, S.-J., Hutchinson, S., Johnson, D., Kise, M.: Omnidirectional-vision-based estimation for containment detection of a robotic mower. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 6344–6351 (2015)
12. Levрatti, A., Secchi, C., Fantuzzi, C.: A low cost localization algorithm for an autonomous lawnmower. In: IEEE 9th Workshop on Robot Motion and Control (RoMoCo), pp. 276–281 (2013)
13. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3354–3361 (2012)
14. Valgren, C., Lilienthal, A.J.: SIFT, SURF & seasons: appearance-based long-term localization in outdoor environments. *Robot. Auton. Syst.* **58**(2), 149–156 (2010)
15. Metka, B., Besetzny, A., Bauer-Wersing, U., Franzius, M.: Predicting the long-term robustness of visual features. In: International Conference on Advanced Robotics (ICAR), pp. 465–470. IEEE (2015)
16. Stone, T., Differt, D., Milford, M., Webb, B.: Skyline-based localisation for aggressively manoeuvring robots using UV sensors and spherical harmonics. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 5615–5622. IEEE (2016)

17. Dudek, G., Jenkin, M.: Computational Principles of Mobile Robotics. Cambridge university press, Cambridge (2010)
18. Mur-Artal, R., Tardos, J.D.: ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras (2016). [arXiv:1610.06475](https://arxiv.org/abs/1610.06475)
19. Deigmöller, J., Eggert, J.: Stereo visual odometry without temporal filtering. In: German Conference on Pattern Recognition (GCPR), pp. 166–175 (2016)
20. Besl, P.J., McKay, N.D.: Method for registration of 3-d shapes. In: Robotics-DL Tentative, pp. 586–606 (1992)
21. Chen, Y., Medioni, G.: Object modelling by registration of multiple range images. Image vis. comput. **10**(3), 145–155 (1992)

A Submap Joining Based RGB-D SLAM Algorithm Using Planes as Features

Jun Wang, Jingwei Song, Liang Zhao and Shoudong Huang

Abstract This paper presents a novel RGB-D SLAM algorithm for reconstructing a 3D surface in indoor environment. The method is a submap joining based RGB-D SLAM algorithm using planes as features and hence is called SJBPF-SLAM. Two adjacent keyframes, with the corresponding small patches and planes observed from the keyframes, are used to build a submap. Then the current submap is fused to the global map sequentially, meanwhile the global structure is recovered gradually through plane feature associations. The use of submap significantly reduces the computational cost during the optimization process, without losing any information about planes and structures. The proposed method is validated using publicly available RGB-D benchmarks and obtains good quality trajectory and 3D models, which are difficult for existing RGB-D SLAM algorithms.

1 Introduction

The indoor environment is one of the most common scenarios in robotic applications, where building a high quality map of the unknown environment in real-time is still a challenging task for Simultaneously Localization and Mapping (SLAM). Point-based SLAM algorithms [1, 2] have achieved much success in various environments, but it still cannot deal with environments with a large area of texture-less planes, for example, walls, floors, and ceilings. These areas actually provide useful information

J. Wang (✉) · J. Song · L. Zhao · S. Huang
Centre for Autonomous Systems,
University of Technology, Broadway, P.O. Box 123,
2007 Sydney, NSW, Australia
e-mail: Jun.Wang@uts.edu.au

J. Song
e-mail: Jingwei.Song@student.uts.edu.au

L. Zhao
e-mail: Liang.Zhao@uts.edu.au

S. Huang
e-mail: Shoudong.Huang@uts.edu.au

of the structure, however they are regarded as challenges in point-based SLAM. In plane-based SLAM, the planes can help to improve the estimation of both robot poses and surfaces. The high quality dense maps can be used in various applications, such as robot navigation, real estate and base maps for virtual reality. While various algorithms have been developed to make use of planes in SLAM and most of them can produce a relatively accurate trajectory, it is still difficult to obtain a high quality model in a challenging indoor environment, such as long corridors including ceilings, floors and other walls.

In this paper, we proposed a Submap Joining Based RGB-D SLAM algorithm using Planes as Features (SJBPF-SLAM), which is more efficient than batch off-line 3D reconstruction and more accurate than existing RGB-D SLAM algorithms. We compared our method with the state-of-the-art approaches on several publicly available challenging datasets, which demonstrated the improvements of our method regarding the quality of both trajectory and surface estimation.

The paper is organized as follows: some related work is discussed in Sect. 2. The proposed method is described in Sect. 3. Specifically, this comprises the local map building in Sect. 3.1 and local map joining in Sect. 3.2. Section 4 presents experiments and results. Section 5 concludes the paper and gives some future work about this method.

2 Related Work

The exploitation of planes in SLAM or 3D reconstruction has been studied for years in robotics, computer vision and computer graphics.

RGB-D SLAM. Kinect-style depth sensors provide the 3D model of the environment in a straightforward way. CPA-SLAM [3] modeled the environment with a global plane model, which reduces the drift along with direct image alignment. In their global model, two types of residuals are defined, one is the distance between points to points, and the other is the distance from points to the global plane, when the points are detected on a plane. This kind of residuals may suffer from the noise of depth images. Kintinuous [4] and Elastic fusion [5] did not use planes explicitly. Instead, they use a frame-to-model strategy to overcome drift. Though providing large scale dense mapping, these two methods still have drift when recovering global structures because the plane structure is not utilized directly, as shown in Sect. 4. Dense planar SLAM [6] used planes as a representation of the model, however, the planes are not included in the state vector, and thus the planes are not optimized during map building. [7] proposed a fast place recognition algorithm based on plane-based maps.

Visual Odometry and Monocular SLAM. Planes are explored for different applications and with various sensors. Plane-primitives are applied to provide a robust odometry for RGB-D cameras [8]. Recent research use planes to improve the accuracy in monocular SLAM. Pop-up SLAM [9] used deep convolutional networks to

detect planes in images and demonstrated that this kind of structures could improve both state estimation and dense mapping, especially in texture-less environments. In [10], a structured learning algorithm was proposed to fit the reconstructed model to the “box” structure of the room following the perspective cues. Parallel and orthogonal walls were applied to correct the odometry of the camera [11]. In monocular scenery, the plane structure of the environment is difficult to detect, and these approaches can only be used in limited scenes under restrictive conditions.

Off-line 3D reconstruction. [12] proposed a fine-to-coarse global registration method. In particular, the plane association windows increase gradually during iterations and cover the whole sequence at the end of the algorithm, which is called hierarchy optimization. As all the cost functions are computed in each step, this method is only suitable for off-line applications. [13] made use of the labeled objects in the environment to guide the optimization and thus obtain a better global registration. In the method, using the limit extend of one object as constraint can significantly reduce drift and improve the global registration, however labeling objects manually for video sequences is exhausting.

Submap joining algorithm. Submap based approaches have attracted the interests of researchers in recent years [14–16]. Sparse local submap joining filter (SLSJF) [15] presented a canonical and efficient submap joining algorithm that makes use of consistent local submaps to build large scale feature-based maps. They explored the sparse structure together with a novel state vector and applied a covariance submatrix recovery technique. [16] proved that the submap joining problem can be formulated as a linear least square problem, and thus can be solved with a closed form solution. All these methods used points as features.

In this paper, we use planes as features to build precise local submaps [15], and exploit the plane structures in the global map during submap joining.

3 Methodology

There are two main stages of our framework. First, we build the local submaps from each pair of adjacent keyframes. Second, all the local submaps are joined by map joining algorithm in a sequential manner. As shown in Fig. 1, we apply a rough visual odometry between consecutive frames. The most time consuming computations, i.e. extracting small patches and detecting planes, just occur on the keyframes, and they can be easily distributed to parallel processes. In a local submap, small patches improve the estimation of the pose and obtain a better image alignment. The output of the local submap is one end pose and planes observed from the first keyframe, which are defined in local frame of the first keyframe. In the bottom part of Fig. 1, several planes in different submaps are associated as one unique plane in global model.

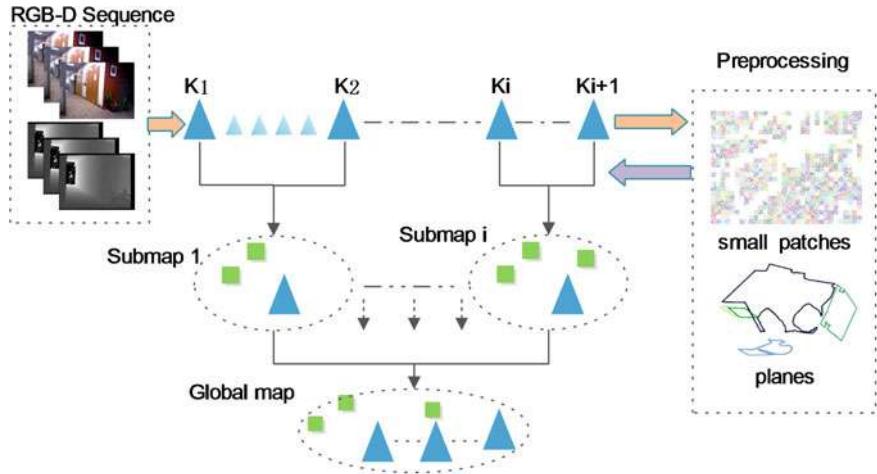


Fig. 1 Framework of the proposed algorithm. K_i denotes the keyframes, the triangles denote the poses, and small squares denote planes

3.1 Build Local Submaps

A local submap is built on two adjacent keyframes. The input of this stage is the keyframes K_i and K_{i+1} : including the coefficients of planes observed by keyframe K_i , small patches on the two keyframes, and a transformation from visual odometry. The output of this stage is a local submap: the information matrix, the optimal solution of the state vector including the pose of the keyframe K_{i+1} , and the coefficients of the planes observed from keyframe K_i . Note that, the pose of first keyframe K_i is not in the state vector, as it defines the local frame for the submap.

3.1.1 Preprocessing

Before building the submap, small patches and planes are extracted on each keyframe. Figure 2a presents the original color image and depth image, Fig. 2b shows the extracted small patches, the detected planes are shown in Fig. 2c.

Small patch extraction. Small patches are detected from the depth images. First we divide the depth image into n by n grids uniformly. Then a RANSAC fitting algorithm is applied to extract the plane coefficients for each grid cell separately, with a threshold \mathcal{T}_{pa} controlling the minimal number of inliers. In our algorithm, a small patch is denoted by $\omega = \{a, b, c, d, p_c\}$, where $\{a, b, c, d\}$ is the coefficients of plane, and p_c is the anchor point of the patch.



Fig. 2 Preprocessing: small patches detection and plane extraction from RGB-D images. **a** Color and depth images. **b** Detected small patches overlaid on color image. **c** Extracted planes overlaid on point cloud

Plane detection. We apply a two-round clustering method to extract planes from the depth image. First, the normal vectors of each pixel in the depth image are computed efficiently by utilizing the integral image [17], then they are classified based on the angles between them. Second, the distances between every point to the origin along the normal direction are calculated, and all pixels are clustered using these distances. The points lying in a common region of the two-round clustering results are supposed to be on a same plane. Finally, the initial values of plane coefficients can be estimated by fitting the points on the plane.

3.1.2 Local Relative Pose

In this part, we utilized the fact that visual odometry is accurate locally although it suffers from drift in long sequences. Fortunately, our submaps are exactly locally defined, so the pose in a local submap should be close to the visual odometry. We incorporate the visual odometry as information and define an error term as follows:

$$E_L = \sum_{m=1}^{p_{max}} \|\hat{R}p_m + \hat{t} - Rp_m - t\|^2 \quad (1)$$

where rotation matrix \hat{R} and translation vector \hat{t} are measurements from visual odometry, R and t are rotation matrix and translation vector from state vector, points p_m are constants and uniformly selected from a unit-radius sphere and p_{max} is the number of the points selected. In this way, we can avoid the direct measurements of angles in poses and make all the errors under Euclidean distance [12, 18]. In our study the

visual odometry is provided by a bundle adjustment on adjacent frames using residuals of pixels and 3D feature positions, however any other proper visual odometry algorithms should work for this task.

3.1.3 ICP-style Registration on Small Patches

The basic idea of this ICP-style registration is to only use small patches to do a 3D registration, with a rough visual odometry providing initial values. This kind of registration utilizes the small patch coefficients fitted from the depth values, rather than the point clouds transformed from the depth images, which usually suffer from much noise from the sensor itself.

Small patches association. Patches observed on keyframe K_{i+1} is transformed to the keyframe K_i using the current estimated pose (visual odometry is used as initial guess in the first iteration). Then we can find the closest patches in terms of angles and distance in the same coordinate frame. To make it clearer, suppose ω_{i+1}^i is a small patch from keyframe K_{i+1} , and transformed to keyframe K_i coordinates. ω_i is a small patch from keyframe K_i . We check two criterions to decide whether the two patches are correspondence: one is the angle between the two normals, and the other is the distance between the two patches. To boost the speed of finding correspondences, a KD-tree is created using the center point of each patch. We can easily find several pairs of closest patches, and compare the normals within the small set of patches, instead of computing all the distances exhaustively. The keyframe K_i defines the local coordinates and its pose remain unchanged during iterations, so we create a KD-tree on the patches from this keyframe. The patch observed on keyframe K_{i+1} can be used to query in the KD-tree. In this way the KD-tree will be created only once during optimization, despite that the pose of keyframe K_{i+1} will be adjusted.

Cost function. We defined an error term as follows to minimize the distance between two corresponding patches, aiming at obtaining coplanar patches.

$$E_S = \sum_{k=1}^{k_{max}} \|(R p_{c(i+1)}(k) + t)^T \cdot n_i + d_i\|^2 + \sum_{k=1}^{k_{max}} \|(R^{-1}(p_{ci}(k) - t))^T \cdot n_{i+1} + d_{i+1}\|^2 \quad (2)$$

where rotation matrix R and translation vector t are from the estimated state vector, $\{n_i, d_i\} = \{a_i, b_i, c_i, d_i\}$ are the coefficients of the patch on keyframe K_i , $\{n_{i+1}, d_{i+1}\}$ are the coefficients of the patch on the keyframe K_{i+1} , $p_{c(i+1)}(k)$ is the k -th anchor point of the patch observed on keyframe K_{i+1} , $p_{ci}(k)$ is the k -th anchor point from the patch on keyframe K_i . k_{max} anchor points are selected around center point on the patch. Briefly, this term measures the point-to-plane distance between the anchor

points from left patch and the corresponding plane of right patch, and vice versa. During iterations, the correspondences between patches from left and right frames will be updated according to the estimated pose of right frame. Ideally, the center points should be exactly on the corresponding planes, and can be expressed as equation $ax + by + cz + d = 0$, where $\{a, b, c\}$ is the unit normal of a patch, d is the distance from the patch to the origin and $\{x, y, z\}$ is the location of one anchor point. Note that in this cost function, the relative poses are the only variables and the coefficients of patches on both frames are considered as constants.

3.1.4 Plane to Small Patches

In this section, we describe the relationship between planes and small patches. For the local submap built by keyframe K_i and K_{i+1} , only the planes observed from keyframe K_i are included (planes only observed from keyframe K_{i+1} are incorporated in the next submap, in which K_{i+1} is the start pose). These plane coefficients are optimized both in submap building and submap joining stages.

Association between planes and small patches. The planes are associated with small patches on the first keyframe of submap, i.e. keyframe K_i . As they are defined under the same frame, we can compare their normals and distance without any coordinate transformations. Similar to small patches associations, we compute the angle between the two normals, and the distance between plane and small patch. When the angle and distance are under some threshold, the plane is associated with the small patch. By this way, all the small patches on the plane are found.

Cost function. We defined another error term to measure the relationship between planes and small patches. The following geometric error is minimized, aiming at obtaining the plane coefficients.

$$E_{PS} = \sum_{m=1}^{m_{max}} \sum_{k=1}^{k_{max}} \|p_c^T(k) \cdot n_m + d_m\|^2 \quad (3)$$

where $p_c(k)$ is k -th anchor point from small patches, $n_m = \{a_m, b_m, c_m\}$ and d_m are the coefficients of the plane Ω_m , and m_{max} is the number of planes.

As the parameterization of planes is not a minimal representation, we define an error term to constrain the normal of the plane to make the representation unique.

$$E_C = \sum_{m=1}^{m_{max}} \|a_m^2 + b_m^2 + c_m^2 - 1\|^2 \quad (4)$$

3.1.5 Optimization and Output of Submap

The objective function for the local submap is defined as follows:

$$E_{LM} = W_L E_L + W_S E_S + W_{PS} E_{PS} + W_C E_C \quad (5)$$

where E_L, E_S, E_{PS}, E_C are the cost functions defined in previous sections, W_L, W_S, W_{PS}, W_C are the corresponding weights for each term. The function is optimized using Levenberg-Marquardt (LM) methods. The output of submap is the optimal solution of the state vector and information matrix as defined by (\hat{X}^L, I^L) , where \hat{X}^L (the superscript “L” stands for the local submap) is an estimate of the state vector $X^L = (X_c^L, X_1^L, X_2^L, \dots, X_n^L)$ and I^L is the corresponding information matrix, which is computed using the estimated variables in the last iteration. The state vector contains the camera final pose X_c^L (the subscript “c” stands for the camera) and the plane coefficients X_1^L, \dots, X_n^L . Since only one pose and several plane coefficients are included in a submap, the convergence of the optimization is very fast and stable.

3.2 Local Submap Joining

The input of this stage is the local submap built in the previous section. The local submap joining algorithm merge all the local submaps in a sequential way. The output is a global map, containing all the poses, the coefficients of the planes, and the corresponding information matrix. The major difference between this algorithm and the SLSJF [15] is that we explore the global structure by associating the planes in local submap to the structure, which utilizes planes as features not points as features.

3.2.1 The State Vector in Global Map

The global map starts from the first local submap and expands with the fusing of them. After the fusion of 1 to $j - 1$ local submaps, the global map can be denoted by $(\hat{X}_{j-1}^G, I_{j-1}^G)$, where \hat{X}_{j-1}^G is an estimate of state vector X_{j-1}^G , I_{j-1}^G is the corresponding information matrix, and the superscript “G” stands for the global map. The following equation gives the detail of the state vector X_{j-1}^G

$$X_{j-1}^G = (X_{c1}^G, X_{c2}^G, \dots, X_{c(j-1)}^G, X_{p1}^G, X_{p2}^G, \dots, X_{pm}^G) \quad (6)$$

where X_c^G are the robot poses, and X_p^G are the plane coefficients. When fusing a local submap into the global map, the end pose and new planes in local submap are added

to the global map as new variables, and the planes associated with existing planes in global map will also be incorporated. The global map will cover all the planes and poses at the end of the fusion stage.

3.2.2 Fusion as a Least Squares Problem

We formulate the fusion of local submap with global map as a least squares problem, which takes two part of measurements as input: global map from the previous step and current local submap to be fused.

For local submap (\hat{X}_j^L, I_j^L) , we believe that the estimation of the end pose and plane coefficients are locally accurate and can be regarded as a measurement of the true values, with a Gaussian noise. The covariance matrix can be given by the information matrix (the inverse of information matrix is the covariance matrix)

$$\hat{X}_j^L = H_j(X_j^G) + w_j \quad (7)$$

where H_j is a transformation function, which transforms the variables from global frame to current local frame using the start pose of the local submap, which is the last pose of global map, w_j is the zero-mean Gaussian ‘‘observation noise’’, whose covariance matrix is

$$P_j^L = (I_j^L)^{-1} \quad (8)$$

Similarly, the global map $(\hat{X}_{j-1}^G, I_{j-1}^G)$ can also be regarded as measurements of the true values of poses and plane coefficients, with a Gaussian noise. The covariance matrix can also be given by the information matrix. That is

$$\hat{X}_{j-1}^G = X_{j-1}^G + W_{j-1} \quad (9)$$

where W_{j-1} is the zero-mean Gaussian ‘‘observation noise’’, whose covariance matrix is $P_{j-1}^G = (I_{j-1}^G)^{-1}$.

From the above, the fusion of the j -th local submap to the $(j-1)$ -th global map can be formulated as a least squares problem, using all the information from the global and the local submap. The equation below shows this weighted least squares problem

$$\min_{X_j^G} (\hat{X}_j^L - H_j(X_j^G))^T I_j^L (\hat{X}_j^L - H_j(X_j^G)) + (\hat{X}_{j-1}^G - A_j X_j^G)^T I_{j-1}^G (\hat{X}_{j-1}^G - A_j X_j^G) \quad (10)$$

where A_j is a matrix that extract corresponding variables from X_j^G , which already exist in \hat{X}_{j-1}^G . This problem can be solved using Gauss-Newton algorithm.

Algorithm 1: Find plane correspondence

Input: planes in local submap, planes in global submap, end pose in global map

Output: plane correspondences

- 1: Transform planes in global map from global to local frame.
 - 2: Compute the covariance matrices for both features under the same frame.
 - 3: Compute Mahalanobis distance d_{gl} between each pair of planes.
 - 4: If $d_{gl} < d_{min}$, the two planes are regarded as associated.
 - 5: Otherwise, a new plane is added as new variables in global map.
-

3.2.3 Data Association

The data association in Algorithm 1 is applied to find the plane correspondences between local and global map. Let Ω^G be a plane feature in the global map, P^G is the corresponding covariance matrix. Let Ω^L be the plane feature in local submap, P^L be the corresponding covariance matrix. The plane feature in global map Ω^G is transformed to local frame of the current local submap $\bar{\Omega}^G$ using

$$\begin{aligned}\bar{\Omega}^G &= H(\Omega^G) \\ H(\Omega^G) &= ((R_e)^{-1}n^G, t_e n^G + d^G)\end{aligned}\tag{11}$$

where $\Omega^G = \{n^G, d^G\}$, $p_e^G = \{R_e, t_e\}$. $H(\Omega^G)$ transforms plane coefficients Ω^G from the global frame to the local frame using the end pose of camera p_e^G in global map (“e” stands for end pose in global map). We then define a Mahalanobis distance:

$$d_{gl} = (\bar{\Omega}^G - \Omega^L)^T (\bar{P}^G + P^L)^{-1} (\bar{\Omega}^G - \Omega^L)\tag{12}$$

where \bar{P}^G is the covariance matrix of transformed global map. Thus a proper threshold value d_{min} can be selected based on χ^2 distribution, such that the null hypothesis that the two features are the same one is not rejected under some confidence level. This procedure is repeated at every iteration.

4 Experiments and Evaluation

In this section, we evaluate our SJBPF-SLAM algorithm and compare it with the state-of-the-art methods. Three publicly available datasets are used in the assessment: ICL-NUIM synthetic scenes [19] and Princeton University SUN3D dataset [13] and TUM datasets [20]. The ICL-NUIM synthetic scenes have ground truth, while the SUN3D dataset provides challenging scenes with distinct geometric structure that can be used to compare the reconstructed surfaces.

The building of local submaps is parallel in nature and thus can be distributed to different computing units. For the submap joining algorithm, even though the state vector grows linearly with local submap joining, the number of plane features are limited in common scenes. At present, the entire SLAM algorithm is implemented in MATLAB, and evaluated with an Intel Core i5-6300, 2.30GHz and 8G RAM. For the computation time, it costs about 2 s for each frame in preprocessing, about 4 s to build a submap. The submap joining is running in real-time.

We set $p_{max} = 8, k_{max} = 6$ for sample point selection on sphere and plane. Keyframes are selected by measuring the movement or the number of frames reaches $m = 5$. The weights in local submap building stage are set to: $W_{PS} = 1, W_L = 2, W_S = 1, W_C = 10$. The threshold \mathcal{T}_{pa} is set to 0.95 in our datasets. The LM algorithm converges in 3–4 iterations for each submap. While in submap joining stage, we don't need to set the weights manually, the information matrix provides the information of the uncertainty and are used as the weights in the optimization. The map joining algorithm using Gauss-Newton converges quickly in just 2 iterations.

4.1 Trajectory Estimation

In the first experiment, we evaluate the performance of our SJBPF-SLAM by comparing the results with ground truth using noisy ICL-NUIM synthetic scenes. The trajectories from global map are compared to the ground truth trajectories. Table 1 presents the results of comparison with absolute trajectory error (ATE). We compare our SJBPF-SLAM algorithm with other state-of-the-art RGB-D SLAM systems, including Elastic Fusion [5], Kintinuous [4], RGB-D SLAM [2], MRSMap [21], DVO-SLAM [22], and some planar based methods: CPA-SLAM [3] and dense planar SLAM [6]. Table 1 shows the results of comparison with these methods. Many systems performed not very well on some datasets, for example, the maximum error of DVO SLAM is 0.191 (on dataset kt2), RGB-D SLAM 0.433 (on dataset kt3), MRSMap 1.090 (on dataset kt3), Kintinuous 0.355 (on dataset kt3), Elastic Fusion 0.106 (on dataset kt3), Dense planar SLAM 0.246 (on dataset kt1), CPA SLAM performs similar as ours, however it utilizes GPU for its computation. As our SJBPF-SLAM explore the global structure and benefits on the structure, the maximum error is below most of the previous SLAM systems.

4.2 Surface Estimation

To evaluate the surface estimation performance of our approach we compared the global map to the output of other state-of-the-art methods. As shown in Fig. 3a, in the output of batch Bundle Adjustment (BA) of [13], which does not use any plane information, there is a gap between two walls. We evaluated our SJBPF-SLAM on

Table 1 The RMSE of the absolute trajectory error (m) of our SJBPf-SLAM algorithm in comparison to previous methods on synthetic datasets of [19]

Systems/Datasets	DVO SLAM	RGB-D SLAM	MRSMap	Kintinuo-us	Elastic Fusion	Dense Planar SLAM	CPA SLAM	Ours
lr kt0	0.104	0.026	0.204	0.072	0.009	0.246	0.007	0.035
lr kt1	0.029	0.008	0.228	0.005	0.009	0.016	0.006	0.017
lr kt2	0.191	0.018	0.189	0.010	0.014	N/A	0.089	0.088
lr kt3	0.152	0.433	1.090	0.355	0.106	N/A	0.009	0.067

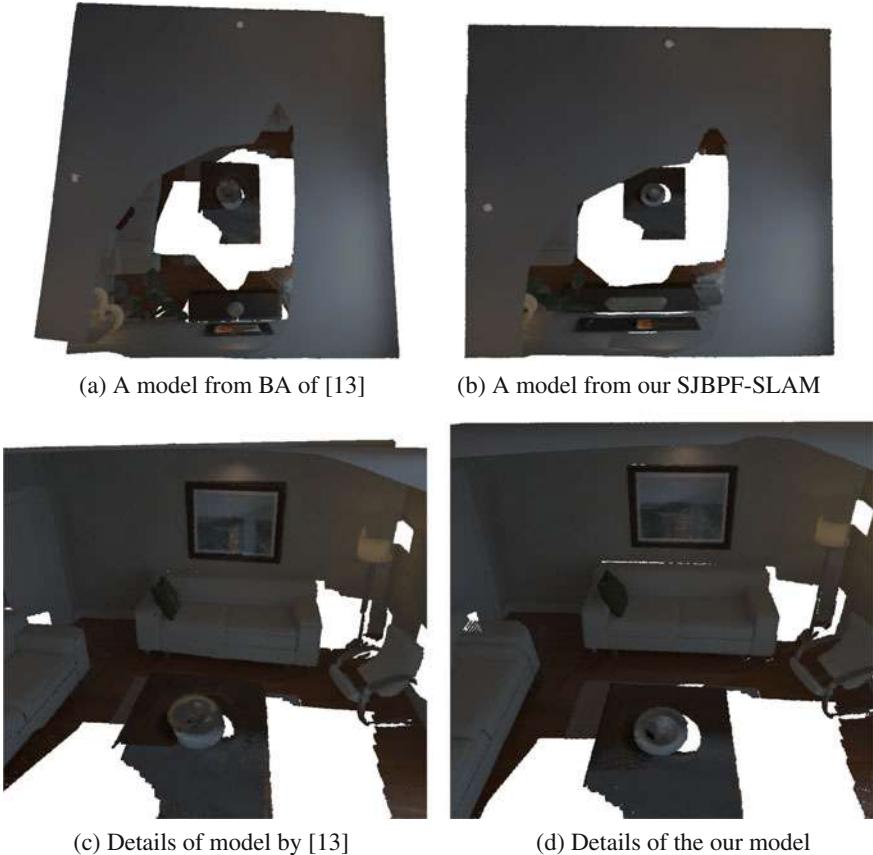
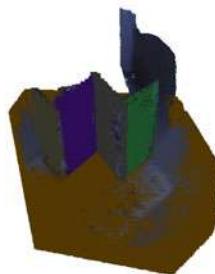


Fig. 3 Surface comparison: **a** the reconstructed model of batched Bundle Adjustment from [13], **b** the reconstructed model of our SJBPF-SLAM, **c** some details from (a), **d** some details from (b)

the same dataset [19], it can be seen from Fig. 3b that the global structure is well reconstructed. Figure 3c, d show more details of the two models, which marked the remarkable improvements of our method. We also tested our method on TUM datasets [20], as shown in Fig. 4. The models are shown with planes highlighted. The big planes on the floor and some other artificial objects, such as checkerboard, desks, are detected and maintained in our plane model. Figure 5a shows the result of Kintinuous [4] on the dataset from SUN3D [13], which was captured along a long corridor. Kintinuous failed on this challenging scene because of tracking errors. Elastic Fusion [5] preforms well on the most part of the dataset, but also drifts near the end of the corridor, as show in Fig. 5b. Figure 5c shows the output of our SJBPF-SLAM methods. With a global structure maintained in our algorithm by submap joining, it helps a lot to reconstruct a precise 3D model of the environment.



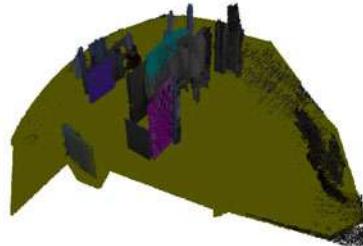
(a) A sample image from fr3/texture.



(b) 3D model of our result on fr3/texture



(c) A sample image from fr2/pioneer.



(d) 3D model of our result on fr2/pioneer.

Fig. 4 Result of our method on some TUM datasets [20]

(a) Results of Kintinus



(b) Results of Elastic Fusion



(c) Results of our SLJEP SLAM.

Fig. 5 Comparison with Kintinuous and Elastic Fusion. Top: the result from Kintinuous on dataset SUN3D. Middle: the result of Elastic Fusion. Bottom: the output of our method

5 Conclusion and Future Work

In this paper, we present a SJBPF-SLAM algorithm which takes advantage of planes in man-made indoor scenes, thus could be used to build a high quality 3D model. The algorithm is very efficient as it applied submap joining technique. The global structure is gradually recovered by joining new submaps, which are built on local frames and thus very precise. The experiments demonstrate that the 3D model produced by our algorithm is much better than most of the state-of-the-art RGB-D SLAM algorithms, and our algorithm is more efficient than off-line 3D reconstruction.

In the future, we will improve the algorithm such that it can robustly handle very large loop closures and some dynamic objects. We will also extend the work to environments with non-planar regions and common objects in man-made scenes, such as chairs and desks. The active perception for robust RGB-D SLAM in dynamic environments is also our future research topic.

References

1. Mur-Artal, R., Tardos, J.D.: Orb-slam2: an open-source slam system for monocular, stereo and rgb-d cameras (2016). [arXiv:1610.06475](https://arxiv.org/abs/1610.06475)
2. Endres, F., Hess, J., Sturm, J., Cremers, D., Burgard, W.: 3-d mapping with an rgb-d camera. *IEEE Trans. Robot.* **30**(1), 177–187 (2014)
3. Ma, L., Kerl, C., Stückler, J., Cremers, D.: Cpa-slam: Consistent plane-model alignment for directrgb-d slam. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1285–1291. IEEE (2016)
4. Whelan, T., Kaess, M., Fallon, M., Johannsson, H., Leonard, J., McDonald, J.: Kintinuous: spatially extended kinectfusion. Advanced reasoning with depth cameras. In: RSS Workshop on RGB-D (2012)
5. Whelan, T., Leutenegger, S., Salas-Moreno, R.F., Glocker, B., Davison, A.J.: Elasticfusion: dense slam without a pose graph. In: Robotics: science and systems, vol. 11 (2015)
6. Salas-Moreno, R.F., Glocker, B., Kelly, P.H., Davison, A.J.: Dense planar slam. In: 2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp. 157–164. IEEE (2014)
7. Fernández-Moral, E., Mayol-Cuevas, W., Arévalo, V., Gonzalez-Jimenez, J.: Fast place recognition with plane-based maps. In: 2013 IEEE International Conference on Robotics and Automation (ICRA), pp. 2719–2724. IEEE (2013)
8. Raposo, C., Lourenço, M., Antunes, M., Barreto, J.P.: Plane-based odometry using an rgb-d camera. In: British Machine Vision Conference (BMVC) (2013)
9. Yang, S., Song, Y., Kaess, M., Scherer, S.: Pop-up slam: semantic monocular plane slam for low-texture environments. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1222–1229. IEEE (2016)
10. Hedau, V., Hoiem, D., Forsyth, D.: Recovering the spatial layout of cluttered rooms. In: 2009 IEEE International Conference on Computer Vision (ICCV), pp. 1849–1856. IEEE (2009)
11. Stuckler, J., Behnke, S.: Orthogonal wall correction for visual motion estimation. In: 2008 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–6, May 2008
12. Halber, M., Funkhouser, T.A.: Structured global registration of RGB-D scans in indoor environments (2016). [arXiv:1607.08539](https://arxiv.org/abs/1607.08539)
13. Xiao, J., Owens, A., Torralba, A.: Sun3d: A database of big spaces reconstructed using sfm and object labels. In: 2013 IEEE International Conference on Computer Vision (ICCV), pp. 1625–1632. IEEE (2013)

14. Ni, K., Steedly, D., Dellaert, F.: Tectonic sam: Exact, out-of-core, submap-based slam. In: 2007 IEEE International Conference on Robotics and Automation (ICRA), pp. 1678–1685. IEEE (2007)
15. Huang, S., Wang, Z., Dissanayake, G.: Sparse local submap joining filter for building large-scale maps. *IEEE Trans. Robot.* **24**(5), 1121–1130 (2008)
16. Zhao, L., Huang, S., Dissanayake, G.: Linear slam: A linear solution to the feature-based and pose graph slam based on submap joining. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 24–30. IEEE (2013)
17. Holz, D., Holzer, S., Rusu, R.B., Behnke, S.: Real-time plane segmentation using rgbd cameras. In: Robot Soccer World Cup, pp. 306–317. Springer (2011)
18. Pulli, K.: Multiview registration for large data sets. In: 1999 Proceedings of the Second International Conference on 3-D Digital Imaging and Modeling, pp. 160–168. IEEE (1999)
19. Handa, A., Whelan, McDonald, T. J., Davison, A.J.: A benchmark for rgbd visual odometry, 3d reconstruction and slam. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 1524–1531. IEEE (2014)
20. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgbd slam systems. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 573–580. IEEE (2012)
21. Stückler, J., Behnke, S.: Multi-resolution surfel maps for efficient dense 3d modeling and tracking. *J. Vis. Communun. Image Represent.* **25**(1), 137–147 (2014)
22. Kerl, C., Sturm, J., Cremers, D.: Dense visual slam for rgbd cameras. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2100–2106. IEEE (2013)

Mapping on the Fly: Real-Time 3D Dense Reconstruction, Digital Surface Map and Incremental Orthomosaic Generation for Unmanned Aerial Vehicles

Timo Hinzmann, Johannes L. Schönberger, Marc Pollefeys
and Roland Siegwart

Abstract The reduced operational cost and increased robustness of unmanned aerial vehicles has made them a ubiquitous tool in the commercial, industrial and scientific sector. Especially the ability to map and surveil a large area in a short amount of time makes them interesting for various applications. Generating a map in real-time is essential for first response teams in disaster scenarios such as, e.g. earthquakes, floods, or avalanches or may help other UAVs to localize without the need of Global Navigation Satellite Systems. For this application, we implemented a mapping framework that incrementally generates a dense georeferenced 3D point cloud, a digital surface model, and an orthomosaic and we support our design choices with respect to computational costs and its performance in diverse terrain. For accurate estimation of the camera poses, we employ a cost-efficient sensor setup consisting of a monocular visual-inertial camera rig as well as a Global Positioning System receiver, which we fuse using an incremental smoothing algorithm. We validate our mapping framework on a synthetic dataset embedded in a hardware-in-the-loop environment and in a real-world experiment using a fixed-wing UAV. Finally, we show that our framework outperforms existing orthomosaic generation methods by an order of magnitude in terms of timing, making real-time reconstruction and orthomosaic generation feasible onboard of unmanned aerial vehicles.

T. Hinzmann (✉) · R. Siegwart
Autonomous Systems Lab, ETH Zurich, Zurich, Switzerland
e-mail: hitimo@ethz.ch

R. Siegwart
e-mail: rsiegwart@ethz.ch

J. L. Schönberger · M. Pollefeys
Computer Vision and Geometry Group, ETH Zurich, Zurich, Switzerland
e-mail: jsch@inf.ethz.ch

M. Pollefeys
e-mail: pomarc@inf.ethz.ch

1 Introduction

A fast and precise overview of an area is important for first aid teams in disaster scenarios such as earthquakes, floods, or avalanches. In particular, digital surface models (DSM) and orthomosaics are essential tools to support the human operator in quick decision-making. An orthomosaic gives a broad overview of the surroundings and helps the human operator to find regions of interest. Furthermore, orthomosaics enable every agent with a camera to infer its own absolute pose by employing feature extraction or image matching. The orthomosaic can therefore be used to localize the robot and other unmanned aerial vehicles (UAVs) while solely relying on an image stream [1]. An orthomosaic image is obtained by correcting aerial images for perspective and camera distortion using the information about the camera intrinsics and camera poses such that the generated image is true to scale and corresponds to a map projection throughout the image. The task of true orthorectification requires a three-dimensional model of the scenery. This is necessary in order to appropriately map intensities observed by the perspective camera to their location with respect to the orthographic camera. The DSM represents the three-dimensional model in form of a height map and furthermore helps to detect changes in elevation or to plan robot or human missions. The literature distinguishes between a DSM and a digital terrain model (DTM). The DSM includes the earth's surface and all objects such as buildings and trees on top of it. In contrast, the DTM models the bare earth's surface. In this publication, we are only interested in generating DSMs.

2 Related Work

The literature for creating overview images can be roughly categorized into panorama and mosaic generation where we utilize the distinction from [2, p. 12]: “Panorama is an extension of field of view (FOV) while mosaic is an extension of point of view (POV)”. The mosaic generation can be divided into forward projection, using e.g. homographies or dense point clouds, and backward projection, using e.g. ray tracing in combination with grids or triangle meshes. An overview of the categories is given in Fig. 1. In this publication, we describe and compare a homography-based and point cloud-based forward projection, as well as a batch, and incremental grid-based backward projection approach by analyzing the advantages and disadvantages in particular with respect to their real-time capabilities.

All of the approaches above are incorporated in our end-to-end mapping framework (cf. Fig. 2) that tightly couples IMU odometry, GPS position and visual cues in a smoothing-based estimator and thus does not detach state estimation from orthomosaic generation. In summary, we claim the following contributions:

- A real-time incremental end-to-end dense reconstruction and orthomosaic generation framework for UAVs that tightly couples state estimation and seamless mosaic generation.

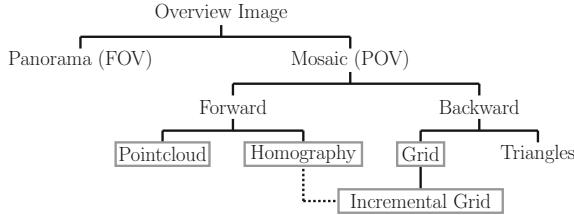


Fig. 1 Categories for generating an overview image. In this paper, we analyze a homography-based and point cloud-based forward projection, as well as a batch and incremental grid-based backward projection approach

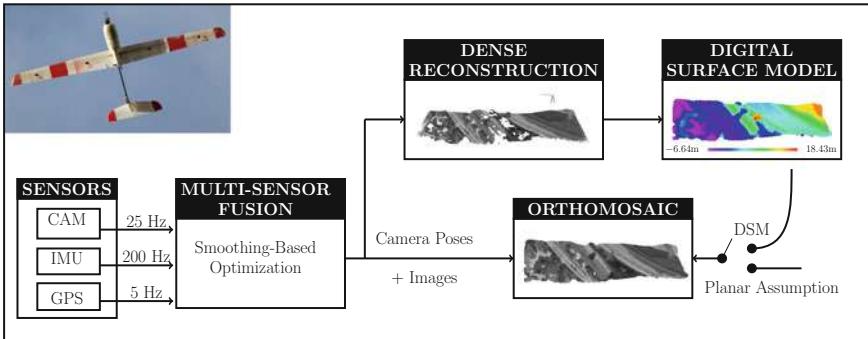


Fig. 2 System overview: The IMU, camera, and GPS measurements are fused in a smoothing-based optimizer. The optimized camera poses and images are used as input for the dense reconstruction and orthomosaic generation. The DSM is updated incrementally from the 3D dense georeferenced point cloud. The orthomosaic is computed via an incremental backward grid-based approach while employing the DSM or a planar assumption and considering the optimal viewing angle

- Most importantly, we propose an incremental grid-based orthomosaic generation algorithm that is suitable for real-time applications in arbitrary terrain by considering the surface model and best viewing angle. We validate its performance on a synthetic and real-world dataset with respect to homography-based, point cloud-based, and batch alternatives.
- We open-source our framework `aerial_mapper` consisting of all described DSM and orthomosaic generation approaches. Our framework augments the efficient and modular `grid_map` library [3] with utilities for georeferenced mapping from aerial views.

2.1 Panorama Generation

Many approaches exist to generate a panoramic view given a set of images by applying a homography. Brown et al. presents in [4] an approach to robustly stitch a

set of unordered images to a seamless panorama assuming rotations only around the optical axis. The main steps consist of feature extraction, matching in feature space, applying RANSAC and then computing the homography and applying bundle adjustment. Steedly et al. [5] build up on [4] and predict overlapping images more efficiently by utilizing the fact that the video stream is not unordered. Agarwala et al. [6] generate a multi-viewpoint panorama of a street using a homography and Markov Random Field (MRF) optimization. Laganière et al. [7] use homographies to generate bird-eye views for teleoperation of a robot. All of the approaches have in common that they focus on obtaining seamless and visually appealing panoramas or bird-eye views and are not concerned about georeferencing or georeferencing errors. However, stitching using only feature correspondences leads to error accumulation and distorted maps when directly applied to UAVs as demonstrated e.g. in [8, p. 20]. The same is true when only the first image is georeferenced and the subsequent images are incrementally stitched to this reference image.

2.2 *Mosaic Generation*

2.2.1 **Forward Projection**

In UAV applications, where we are rather interested in generating a seamless and georeferenced mosaic, additional sensor measurements are used to obtain camera pose measurements or estimates: Hemerly et al. [9] describe the process of obtaining a single georeferenced image using a UAV. Olawale et al. [10] recover the camera intrinsics and extrinsics using GPS and manually collected ground control points in combination with the commercial photogrammetric software (*Agisoft*) and generate an orthomosaic. Yahyanejad et al. present in [2, 8] the results of homography-based image mosaicing from sensor data recorded on board of a rotary-wing UAV with a down-looking camera.

As presented, many approaches use a camera pose estimate and an image as input and then apply a robust but costly feature detection and matching algorithm. For instance, [2, 8] assume noisy IMU and GPS measurements and deal with this by designing a quality function that finds a trade-off between geo-referencing error and seamless stitching. In contrast, we do not detach state estimation and orthomosaic generation but fuse GPS and IMU measurements as well as feature tracks in a consistent smoothing-based state estimation. The small offset between images in combination with gyroscope measurements enables fast and subpixel-accurate Lucas-Kanade feature tracking (KLT) [11]. The use of KLT is also supported by the findings in [12] claiming that KLT achieves the best quantitative results in the context of orthomosaic generation. Our philosophy is that accurate and efficient estimation of the camera poses is the backbone of consistent dense 3D reconstruction and seamless orthomosaic generation. Furthermore, the literature was previously not

concerned about presenting runtime results and [2, 12] deplore lack of quantitative performance measures. We tackle this absence of information by presenting the runtime of all methods and an open-source Gazebo-based HIL environment [13] capable of generating synthetic datasets.

2.2.2 Backward Projection

Note that none of the homography-based forward projection approaches presented in the previous section employ a DSM as input. In contrast, in order to generate true orthomosaics, [14] employ triangle-based backprojection, also known as ray tracing, in combination with a DSM. Our backprojection approach is very similar to [14] but we utilize a grid of squares to simplify the raytracing process. Furthermore, we present a novel incremental grid-based orthomosaic generation approach to speed up the computation.

3 Methodology

The methodology section follows the data flow illustrated in Fig. 2: Sect. 3.1 presents the smoothing-based GPS-IMU-Vision fusion. Given the input images and corresponding optimized camera poses, a dense georeferenced point cloud can be generated using planar rectification, as demonstrated in Sect. 3.2. Section 3.3 presents how this dense point cloud can be used to generate a DSM by employing inverse distance weighting (IDW). Finally, Sect. 3.4 presents our approaches to generate an orthomosaic from a stream of images, optimized camera poses, and DSM using (a) forward projection and (b) backward projection.

3.1 Multi-Sensor Fusion

In this section, we present the core elements of our proposed multi-sensor fusion framework. We distinguish three coordinate systems: the global frame \mathcal{F}_G , the camera frame \mathcal{F}_C , and the body frame \mathcal{F}_B . To avoid unnecessary conversions due to the vision-based fusion, we choose the Universal Transverse Mercator (UTM) coordinate system where \mathbf{p}_B^G expresses easting, northing, and elevation. We seek to estimate the robot states \mathbf{x}_R as well as the set of landmarks \mathbf{x}_L . We define the robot state as: $\mathbf{x}_R := [\mathbf{p}_B^G \ \mathbf{q}_B^G \ \mathbf{v}_B^G \ \mathbf{b}_a \ \mathbf{b}_g]$ where the orientation, position and velocity of the body frame expressed in global coordinates are denoted with \mathbf{q}_B^G , \mathbf{p}_B^G and \mathbf{v}_B^G . The remaining state vector consists of accelerometer bias \mathbf{b}_a and gyroscope bias \mathbf{b}_g .

3.1.1 Vision Front-End

FAST features [15] are extracted from every input image and tracked from frame to frame using KLT with subpixel refinement. To speed up the tracking process and to avoid outliers, we use the gyroscope of the IMU to predict the location of the pixel in the subsequent image. Furthermore, we employ feature bucketing to guarantee uniformly distributed features across the image for improved vision-based motion estimation.

3.1.2 Smoothing-Based State Estimation

For sensor fusion and pose estimation we use the incremental smoothing and mapping algorithm *iSAM2* [16]. For the employed reprojection residual, we refer to [17]. Every reprojection factor has a Cauchy M-Estimator associated with it to reduce the influence of outliers.¹ The IMU measurements are preintegrated and summarized in a single relative motion constraint connecting two time-consecutive poses as described in [18]. The residual and Jacobian of the GNSS position factor is calculated by “lifting” the residual: $\mathbf{e} = \tilde{\mathbf{t}}_B^G - \mathbf{t}_B^G, \frac{\partial \mathbf{e}}{\partial \delta t} = -\frac{\partial}{\partial \delta t} (\mathbf{t}_B^G + \mathbf{R}_B^G \delta t) = -\mathbf{R}_B^G$ where $\tilde{\mathbf{t}}_B^G$ is the measured position transformed to UTM coordinates.² All measurements are inserted into the factor graph once they become available. For every measurement, the factor graph is augmented by a state node. To estimate the initial position, orientation as well as accelerometer biases, at the beginning of every experiment, the plane is kept level for few seconds. During this time, the GPS position measurements are averaged to determine the initial position. The averaged accelerometer readings are used for coarse gravity alignment and bias estimation. After take-off is detected, the vision measurements are incorporated into the factor graph. Note that in this publication only open-loop SLAM was employed, i.e. no loop closures or inter-matches were included in the factor graph. Albeit we did not experience any inconsistencies in the generated dense reconstruction or orthomosaics we consider to integrate an online loop-closure or map-tracking module in future work to guarantee the global consistency of the map.

3.2 Dense Reconstruction

Given the optimized camera poses of our monocular camera rig, a virtual stereo-pair is generated using planar rectification [19]. The dense point cloud is then computed by applying efficient stereo block matching. Note that planar rectification assumes that the epipoles of a virtual stereo-pair are outside the field of view which is fulfilled for our fixed-wing UAV with down-looking camera due to the approximately fronto-parallel motion with respect to the ground.

¹The Cauchy weight is $k^2/(k^2 + e^2)$, where e is the residual and k is a constant set to 3.0.

²Note that we neglect the translational offset between GNSS antenna and IMU since for our setup this corresponds to few centimeters.

3.3 Digital Surface Map Generation

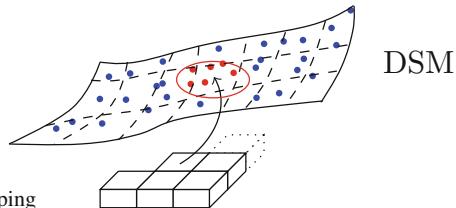
The georeferenced dense point cloud serves as input for the digital surface map. The algorithm consists of a for-loop that iterates over all affected cells in the grid.

Algorithm 1 Grid-Based DSM

p_r : Radius of the kd-tree used for interpolation.
 λ : Factor to increase the interpolation radius.

```

1: function DSM(POINT CLOUD,
CAMERA POSES)
2:   cells  $\leftarrow$  identifyAffectedCells( $T_C^G$ )
3:   for  $c$  : cells do
4:     while  $N = \{\}$  do
5:        $p_r \leftarrow \lambda \cdot p_r$ 
6:        $N \leftarrow$  kd-tree( $x_c, y_c, p_r$ )
7:     end while
8:     Apply interpolation methods
9:     (Optional:) Height-to-color mapping
10:   end for
11: end function
```



A fast kd-tree³ implementation returns the set of nearest points N found within the interpolation radius p_r . Next, inverse distance weighting (IDW) is applied as interpolation method. IDW intuitively determines the cell's height by using a linearly weighted combination of the nearest neighbors, where the weight corresponds to the inverse distance to the cell center, thus giving higher weight to points that are closer to the cell center. An adaptive interpolation radius is utilized (cf. Algorithm 1) that is guaranteed to return an interpolated height value in sparse regions and still keeps a high level of detail in dense regions.

3.4 (Ortho-)Mosaic Generation

In this section, we present the implemented approaches for computing an (ortho-)mosaic while focusing on the proposed incremental grid-based orthomosaic generation.

3.4.1 Homography-Based Mosaic (Forward Projection)

A perspective homography \mathbf{H} is computed which relates the border pixel coordinates of the image to points on the ground surface.

³*nanoflann*: nano fast library for approximate nearest neighbors.

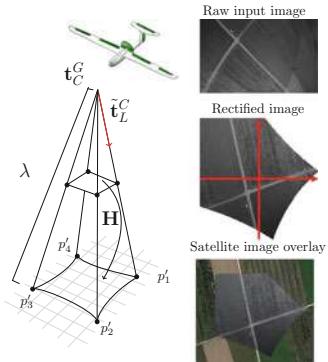
Algorithm 2 Homography-Based Mosaic

w : Image width in pixel. h : Image height in pixel.

```

1: function MOSAICHOMOGRAPHY(IMAGE, CAMERA
   POSE, CAMERA INTRINSICS)
2:    $p_1 \leftarrow (0, 0)$ ,  $p_2 \leftarrow (w, 0)$ ,  $p_3 \leftarrow (w, h)$ ,  $p_4 \leftarrow (0, h)$ 
3:   undistort(image)
4:   // Obtain ground points.
5:   for  $i = 1 : 4$  do
6:     // Computing the scale.
7:      $\lambda_i \leftarrow -(z_C^G - h_{ground}) / (\mathbf{R}_C^G \tilde{\mathbf{t}}_L^C)_z$ 
8:     // Computing the ground position [UTM].
9:      $p'_i \leftarrow \mathbf{t}_C^G + \lambda_i \mathbf{R}_C^G \tilde{\mathbf{t}}_L^C$ 
10:    end for
11:    $H \leftarrow \text{computeHomography}(\mathbf{p}, \mathbf{p}')$ 
12:    $image_{transf.} \leftarrow \text{applyHomography}(H, \text{image})$ 
13:   mosaic  $\leftarrow \text{iterativeBlending}(image_{transf.})$ 
14: end function

```



The homography is then applied to the undistorted input image and the transformed single rectified image is blended with the overall mosaic using feathering. The pseudo code of the algorithm and the results are presented in Algorithm 2 and Fig. 6, respectively.

3.4.2 Grid-Based Orthomosaic (Backward Projection)

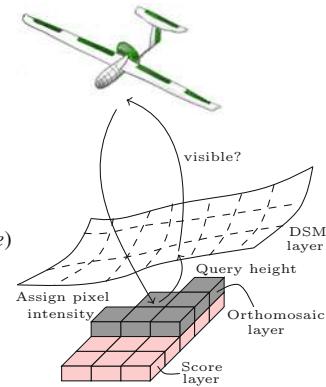
The grid-based orthomosaic generation in *batch* formulation iterates over all cells and, for every cell, queries the corresponding height from the DSM layer. An additional for-loop iterates over all images and, given the corresponding camera pose and camera intrinsics, checks if the cell is within the visible camera cone. Since every cell is usually observed from several camera frames, the question poses which is the ideal pixel intensity value to be assigned to the cell of the orthomosaic. Various mosaic strategies exist [14]. We propose to extract the pixel intensity from the image where the corresponding camera pose is the closest to nadir. This elevation angle is defined as the observation vector from the camera to the cell center. Instead of performing these operations on all cells in the grid, our proposed *incremental* formulation (Algorithm 3) identifies the subset of cells that needs to be updated, as illustrated in green in Fig. 6. The cells are identified by projecting the border-pixels of the current image onto the plane defined by the minimal elevation obtained from the DSM. All cells which are potentially visible given the current camera pose are obtained by min/max operation. Depending on the image distortion, a tighter approximation could be achieved using e.g. the Bresenham algorithm [20]. Only those cells which show a higher elevation angle than currently stored and which are visible from the current camera configuration are updated.

Algorithm 3 Incremental Grid-Based Orthom.

```

1: function           INCREMENTALORTHOMOSAIC-
2:   GRID(IMAGE, CAMERA POSE, CAMERA INTRINSICS)
3:   cells  $\leftarrow$  identifyAffectedCells( $T_C^G$ )
4:   for  $c : cells$  do
5:      $z_c \leftarrow dsm(c)$ 
6:     if visibility( $x_c, y_c, z_c, T_C^G$ ) then
7:       score  $\leftarrow$  computeScore( $x_c, y_c, z_c, T_C^G$ )
8:       if score > score( $c$ ) then
9:         ( $u, v$ )  $\leftarrow$  backproject( $x_c, y_c, z_c, T_C^G, image$ )
10:        ortho( $c$ )  $\leftarrow$  pixelIntensity( $u, v, image$ )
11:      end if
12:    end if
13:   end for
end function

```

**3.4.3 Point Cloud-Based Orthomosaic (Forward Projection)**

In contrast to the approach described in Sect. 3.4.2 one can directly use the dense 3D reconstruction of the environment (cf. Sect. 3.2) to generate an orthomosaic view and hence avoid the costly backprojection step. The proposed point cloud-based orthomosaic generation approach closely follows Algorithm 2 but instead of the height we compute the IDW of the pixel intensity.

4 Platform and Sensors

For our experiments, we use *Techpod* (cf. Fig. 2), a small unmanned research plane with a wingspan of 2.60 m. The IMU *ADIS16448*, and the grayscale camera *Aptina MT9V034* of the sensor pod are hardware-synchronized using a VI-Sensor [21] and run at 200 Hz and 25 Hz, respectively. The camera *Aptina MT9V034* has a focal length of 2.8 mm, a sensor diagonal of 1/3 inch, and a resolution of 752×480 pixels. The *ublox LEA-6H* GPS receiver and the pressure sensors are connected to the *Pixhawk* autopilot running a real-time EKF [22].

5 Simulation Experiments

The Gazebo-based HIL environment was used to validate the DSM and orthomosaic generation is illustrated in Fig. 3. The aerodynamic coefficients and mounted sensors closely model our UAV *Techpod*. Figure 4 shows the results from a simulated single scan line at a relatively low altitude of 50 m above the mesh of *Pix4D's cadastre* [23] dataset. For this experiment, 429 images are rendered at a frame rate of 20 Hz and



Fig. 3 Gazebo-based HIL environment for fixed-wing UAVs

each image is associated with the ground truth pose. Figure 4a shows the coordinate system of the last camera pose and the point cloud generated by the planar rectification algorithm using every 10th image. In this experiment, we deliberately do not use every frame for the dense reconstruction to underline the framework’s potential to handle sparse regions or holes in the point cloud. The DSM layer, which is given in Fig. 4b, is generated by applying IDW with an initial radius of 5 m to the dense point cloud. Figure 4c depicts the incremental grid-based orthomosaic in which the pixel intensity is queried from the first camera that is in line of sight of the respective cell. In contrast, Fig. 4d shows the incremental grid-based orthomosaic where the pixel intensities are obtained from the camera frame with the view closest to nadir. The corresponding elevation angles between selected camera pose and orthomosaic cell are shown in Fig. 4f, g. In particular in regions with a small altitude to terrain height ratio (e.g. tree in center) one can observe that the nadir-view approach renders an improved orthorectified view and avoids double object mapping. As Fig. 4e illustrates, the result of our nadir-view approach is in accordance with the orthomosaic generated by *Pix4D*, for which we used the same georeferenced images as input. The homography approach is not shown since the underlying flat plane assumption results in the predicted large orthomosaic distortions.

6 Real-World Experiments

In this section, we present the results obtained from the semi-autonomous flight at an altitude of 100 m above ground (cf. Fig. 5). The dense point cloud and orthomosaics are presented in Fig. 6. In contrast to the previous experiment, we present the output of the incremental grid based on a flat DSM. Due to the high altitude to terrain height ratio in this experiment, the assumption does not introduce measurable orthomosaic inconsistencies with respect to *Google* imagery (cf. Fig. 6b). The homography-based orthomosaic with applied feathering, shown in Fig. 6a, can handle an image stream of up to 57 Hz. Given the measured runtime in Table 1, the combination of dense

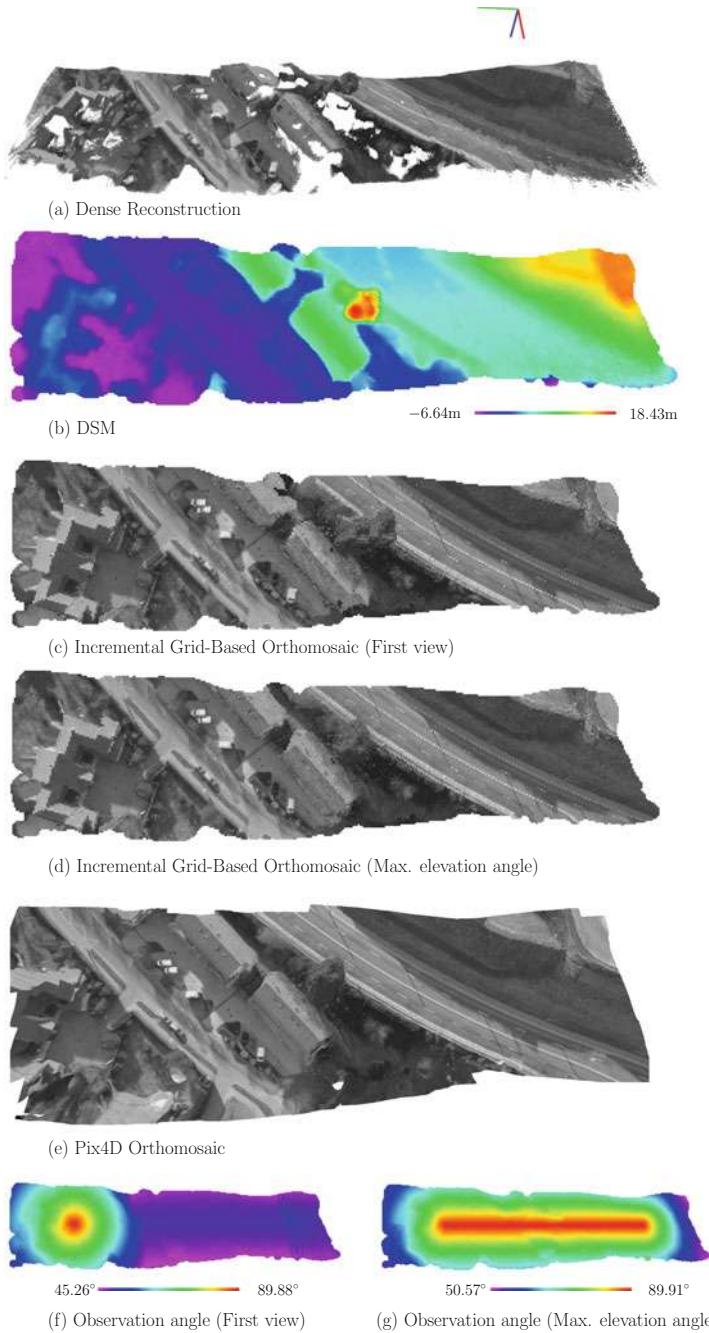


Fig. 4 Simulation results for dataset *cadastre* [23]

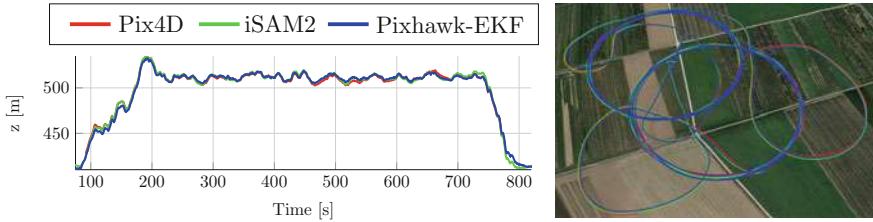
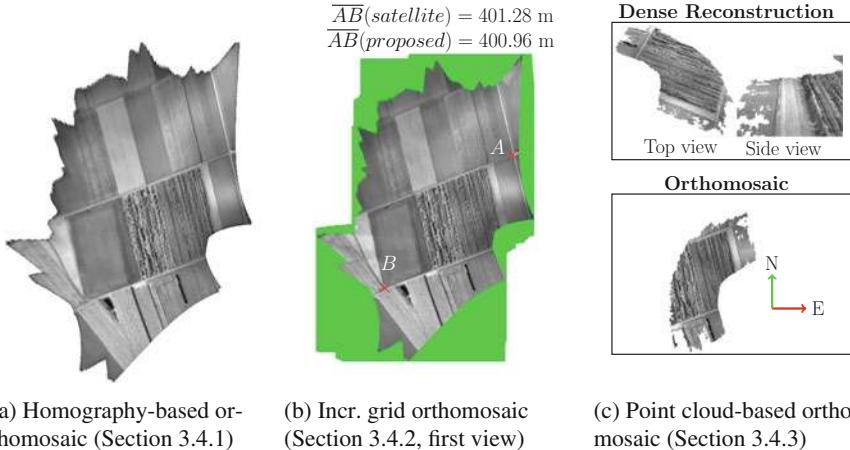


Fig. 5 Comparison of *Pix4D*, *Pixhawk-EKF* [22] and *iSAM2*-based estimation



(a) Homography-based orthomosaic (Section 3.4.1)

(b) Incr. grid orthomosaic (Section 3.4.2, first view)

(c) Point cloud-based orthomosaic (Section 3.4.3)

Fig. 6 Mapping results based on the *iSAM2* state estimates using a fixed-wing UAV

reconstruction and point cloud-based orthomosaic is even slightly faster than the homography-based approach. The caveat of the former is that, due to image distortions at the outer regions, a smaller field of view will be covered by the virtual stereo pair (cf. Fig. 6c). Both the homography- and point cloud-based approach outperform the methods presented in [2] by an order of magnitude. Note that the variants proposed in [2] do not generate a DSM and thus visual artifacts are introduced into the orthomosaic when the planar assumption is violated. The proposed incremental grid-based approach speeds up the computation by a factor of 10 compared to the batch variant. Both, the batch and incremental grid approach are several magnitudes faster than the triangle mesh implementation [14]. However, the implementation in [14] also performs color matching and identifies obscured pixels during the ray casting process adding up to a runtime of 62 min per image.

Table 1 Runtime results for dense reconstruction and orthomosaic generation

	Time/image	Total time	# images	Resol.	CPU	Type	Impl.
Homography (Sec. 3.4.1)	17.4 ms	4.33 s	249	-	2.8 GHz	Forw.	C++
Dense rec. (Sec. 3.2)	16.7 ms	0.384 s	23/249	-	2.8 GHz	Forw.	C++
Point cloud (Sec. 3.4.3)	0.2 ms	0.51 s	249	10 m	2.8 GHz	Forw.	C++
Point cloud (Sec. 3.4.3)	0.79 ms	1.97 s	249	1 m	2.8 GHz	Forw.	C++
Point cloud (Sec. 3.4.3)	31 ms	7.72 s	249	0.1 m	2.8 GHz	Forw.	C++
“Position” [2]	0.47 s	17.31 s	37	n/a	2.66 GHz	Forw.	Matlab
“Pose” [2]	0.5 s	18.33 s	37	n/a	2.66 GHz	Forw.	Matlab
“Image” [2]	12.41 s	459.2 s	37	n/a	2.66 GHz	Forw.	Matlab
“Hybrid” [2]	3.68 s	136.28 s	37	n/a	2.66 GHz	Forw.	Matlab
Grid (batch) (Sec. 3.4.2)	0.43 s	107.41 s	249	10 m	2.8 GHz	Backw.	C++
Grid (batch) (Sec. 3.4.2)	1.73 s	430.27 s	249	1 m	2.8 GHz	Backw.	C++
Grid (incr.) (Sec. 3.4.2)	171 ms	42.6 s	249	1 m	2.8 GHz	Backw.	C++
Triangle Mesh [14]	62 min	620 min	10	0.15 m	2.8 GHz	Backw.	C#, Matlab

Implemented

Proposed

7 Conclusion

In this publication, we demonstrated that incremental end-to-end dense reconstruction and orthomosaic generation for UAVs is feasible in real-time allowing, for instance, advanced autonomous missions of UAV fleets by relying on orthomosaic-based localization only. We highlight the characteristics of our implemented orthomosaic generation approaches in particular with respect to runtime and the influence of the flight altitude to terrain height ratio: The advantage of homography-based orthomosaic generation is the seamless blending, the fast computation and the optimal integration of all pixels but is only suited for planar scenery or, alternatively, high flight altitudes. The benefit of the point cloud-based orthomosaic is the lowest computation time among the evaluated methods, the seamless blending and the direct way of considering the surface elevation. However, depending on the dense reconstruction algorithm the area of coverage is smaller and sparse regions can only be overcome by interpolating nearby point intensities potentially leading to incorrect orthomosaics. Our proposed backward incremental grid-based orthomosaic is suited for arbitrary terrain, renders a true orthomosaic by considering the surface model and optimal viewing angle and still achieves real-time performance.

Acknowledgements The research leading to these results has received funding from ArmaSuisse under project n°050-45 and the European Commission’s Seventh Framework Programme (FP7/2007–2013) under grant agreement n°600958 (SHERPA). The authors thank Andreas Jäger and Sammy Omari for the implementation of the planar rectification algorithm, Lucas Pinto Teixeira for the synthetic image rendering pipeline, and Thomas J. Stastny for comments that greatly improved the publication.

References

1. Yol, A., et al.: Vision-based absolute localization for unmanned aerial vehicles. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3429–3434, Sept 2014
2. Yahyanejad, S., Rinner, B.: A fast and mobile system for registration of low-altitude visual and thermal aerial images using multiple small-scale UAVs. *ISPRS J. Photogramm. Remote Sens.* **104**, 189–202 (2015)
3. Fankhauser, P., et al.: A universal grid map library: implementation and use case for rough terrain navigation. In: Robot Operating System (ROS), vol. 1, ch. 5. Springer (2016)
4. Brown, M., Lowe, D.G.: Automatic panoramic image stitching using invariant features. *Int. J. Comput. Vis.* **74**, 59–73 (2007)
5. Steedly, D., Pal, C., Szeliski, R.: Efficiently registering video into panoramic mosaics. In: ICCV, pp. 1300–1307. IEEE Computer Society (2005)
6. Agarwala, A., et al.: Photographing long scenes with multi-viewpoint panoramas. *ACM Trans. Graph.* **25**, 853–861 (2006)
7. Laganière, R.: Composing a birds eye view mosaic. *Vis. Interface* 382–386 (2000)
8. Yahyanejad, S.: Orthorectified mosaicking of images from small-scale unmanned aerial vehicles. Ph.D. thesis, Alpen-Adria Universität Klagenfurt (2013)
9. Hemerly, E.M.: Automatic georeferencing of images acquired by UAV's. *Int. J. Autom. Comput.* **11**(4), 347–352 (2014)
10. Olawale, B.O., et al.: A Four-Step Ortho-Rectification Procedure for Geo- Referencing Video Streams from a Low-Cost UAV, vol. 9, no. 8, pp. 1445–1452 (2015)
11. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th international joint conference on Artificial Intelligence, pp. 674–679 (1981)
12. Azzari, P., et al.: An Evaluation Methodology for Image Mosaicing Algorithms, pp. 89–100. Springer, Berlin (2008)
13. Furrer, F., et al.: RotorS—A Modular Gazebo MAV Simulator Framework, pp. 595–625. Springer International Publishing, Cham (2016)
14. Nielsen, M.: True orthophoto generation, Master's thesis, Technical University of Denmark (2004)
15. Biadgie, Y., Sohn, K.A.: Feature detector using adaptive accelerated segment test. In: 2014 International Conference on Information Science Applications (ICISA), pp. 1–4, May 2014
16. Kaess, M., et al.: iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In: ICRA, pp. 3281–3288. IEEE (2011)
17. Leutenegger, S., et al.: Keyframe-Based Visual-Inertial SLAM using nonlinear optimization. In: Proceedings of Robotics: Science and Systems. Berlin, Germany, June 2013
18. Forster, C., et al.: IMU preintegration on manifold for efficient Visual-Inertial Maximum-a-Posteriori estimation. In: Proceedings of Robotics: Science and Systems. Rome, Italy, July 2015
19. Fusiello, A., et al.: A compact algorithm for rectification of stereo pairs. *Mach. Vis. Appl.* **12**(1), 16–22 (2000)
20. Bresenham, J.E.: Algorithm for computer control of a digital plotter. *IBM Syst. J.* **4**, 25–30 (1965)
21. Nikolic, J., et al.: A synchronized Visual-Inertial sensor system with FPGA Pre-Processing for accurate Real-Time SLAM. In: ICRA, pp. 431–437 (2014)
22. Leutenegger, S., et al.: Robust state estimation for small unmanned airplanes. In: 2014 IEEE Conference on Control Applications (CCA), pp. 1003–1010, Oct 2014
23. Pix4D dataset Cadastre.: <https://support.pix4d.com/hc/en-us/articles/202561399-Example-Datasets-Available-for-Download-Cadastre->

Aerial and Ground-Based Collaborative Mapping: An Experimental Study

Ji Zhang and Sanjiv Singh

Abstract We here present studies to enable aerial and ground-based collaborative mapping in GPS-denied environments. The work utilizes a system that incorporates a laser scanner, a camera, and a low-grade IMU in a miniature package which can be carried by a light-weight aerial vehicle. We also discuss a processing pipeline that involves multi-layer optimization to solve for 6-DOF ego-motion and build maps in real-time. If a map is available, the system can localize on the map and merge maps from separate runs for collaborative mapping. Experiments are conducted in urban and vegetated areas. Further, the work enables autonomous flights in cluttered environments through building and trees and at high speeds (up to 15 m/s).

1 Introduction

The paper is aimed at solving a mapping problem. In particular, we seek for collaboration between mapping from the ground and air due to each own characteristics. Ground-based mapping is not prone to limitations of space or time. Typically, a mapping device carried by a ground vehicle is suitable for mapping in large scale and can move at a high speed. On the other hand, a tight area can be mapped in a hand-held deployment. However, ground-based mapping is limited by the sensor's altitude, difficult to realize a top-down looking configuration. As illustrated in Fig. 1, the ground-based experiment produces a detailed map of the surroundings of a building, while the roof has to be mapped from the air. If a small aerial vehicle is used, aerial mapping is limited by time due to the short lifespan of batteries. Space also needs to be open enough for aerial vehicles to operate safely. In this paper, we carry out experimental studies to pursue the advantages of both.

The collaborative mapping is based on our previous work [1–3] which develops a data processing pipeline for real-time ego-motion estimation and mapping. The method utilizes a laser scanner, a camera, and a low-grade IMU, processes data

J. Zhang (✉) · S. Singh
Kaarta, Inc., Pittsburgh, USA
e-mail: ji@kaarta.com

S. Singh
Near Earth Autonomy, Inc., Pittsburgh, USA

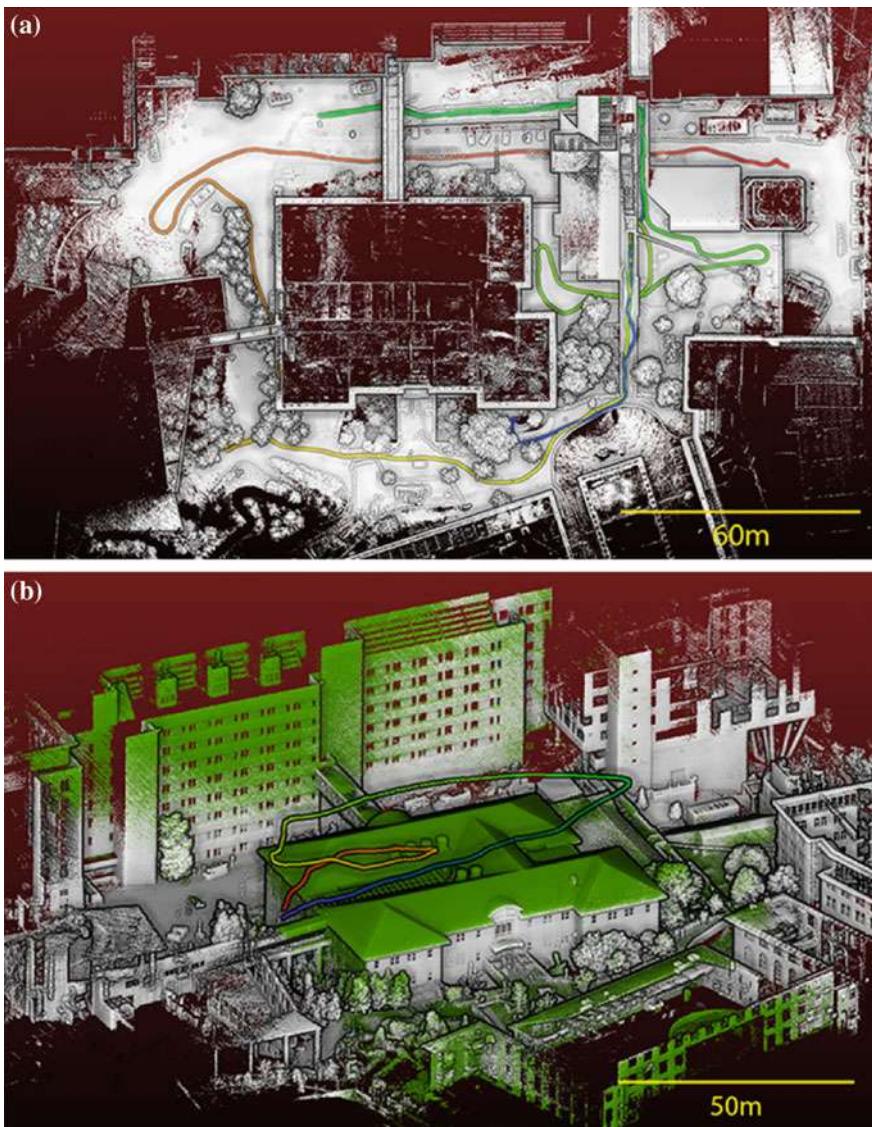


Fig. 1 Example of air-ground collaborative mapping. **a** shows the ground-based map and sensor trajectory (colored curve starting with blue and ending with red) produced by an operator holding a sensor pack and walking around a building at 1–2 m/s for 914 m of travel. The ground-based map covers details surrounding the building except the roof. Then in **(b)**, the same sensor pack is mounted to an aerial vehicle flying over the building at 2–3 m/s for 269 m. The green point cloud in **(b)** is the aerial map and the colored curve is the sensor trajectory in the air

through multi-layer optimization. The resulting motion estimates are at a high rate (200 Hz) with a low drift (typically <0.1% of the distance travel).

Benefit from the high-accuracy processing pipeline, the paper develops a method to merge the maps from the ground and air in real-time. This is by localization of one output w.r.t. the map from the other. In the existing literature, prior map based localization often involves particle filtering [4–9]. In addition, Nieuwenhuisen et al. use multi-resolution scan matching to localize an aerial vehicle on a 3D point cloud map [10]. Also employing scan matching, our method enables both high-speed navigation (up to 15 m/s) and large scale mapping (over 1 km).

While the proposed scheme fulfills collaborative mapping, it further reduces the complexity of aerial deployments. With a ground-based map, flight paths are defined and the aerial vehicle conducts mapping in autonomous missions. In experiments, the aerial vehicle is able to accomplish challenging flight tasks autonomously.

2 Method

2.1 Sensor Configuration

Figure 2 presents the sensor/computer pack utilized by the paper to enable collaborative mapping. Our processing software is not limited to a particular sensor configuration. However, introducing sensors in the front helps readers understand the technology. The sensor pack (see Fig. 2a) consists of a Velodyne Puck laser scanner generating 0.3 million points/s, a camera at 640×360 pixels resolution and 50 Hz frame rate, and a low-grade IMU at 200 Hz. An onboard i7 computer processes data from the sensors in real-time for ego-motion estimation and mapping. Figure 2c, d illustrate the sensor field of view. An overlap is shared by the laser and camera, with which, the processing software associates depth information from the laser to image features (more discussion in Sect. 2.2).

2.2 Odometry and Mapping

The odometry and mapping method is originally proposed in [3]. For completeness, we include an overview of the method. The software processes data from a range sensor such as a laser scanner, a camera, and an inertial sensor. Instead of combining data from all sensors in a large, full-blown problem, we parse the problem as multiple small problems, solve them sequentially in a coarse-to-fine manner. Figure 3 gives a block diagram of the software system. In such a system, modules in the front conduct light processing, ensuring high-frequency motion estimation robust to aggressive motion. Modules in the back take sufficient processing, run at low frequencies to warrant accuracy of the resulting motion estimates and maps.

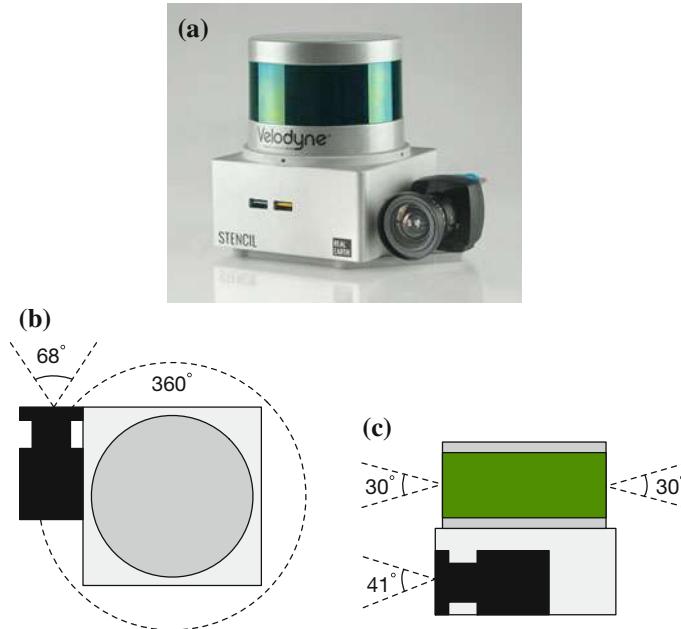
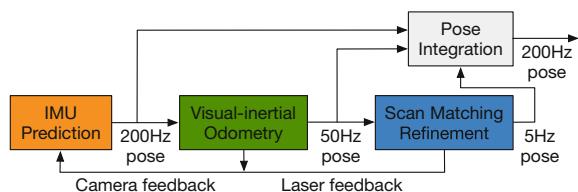


Fig. 2 **a** Sensor pack including a Velodyne Puck laser scanner, a camera, and a low-grade IMU. An onboard i7 computer processes data from the sensors to conduct real-time ego-motion estimation and mapping. **b** and **c** Horizontal and vertical field of view of the laser and camera

The software starts with IMU data processing (orange module in Fig. 3). This module runs at the IMU frequency to predict the motion based on IMU mechanization. The result is further processed by a visual-inertial coupled method (green module in Fig. 3). The method tracks distinctive image features through the image sequence and solves for the motion in an optimization problem. Here, laser range measurements are registered on a depthmap, with which, depth information is associated to the tracked image features. Since the sensor pack contains a single camera, depth from the laser helps solve scale ambiguity during motion estimation.

The estimated motion is used to register laser scans locally. In the third module (blue module in Fig. 3), these scans are matched to further refine the motion estimates. The matched scans are registered on a map while scans are matched to the map. To

Fig. 3 Block diagram of the laser-visual-inertial odometry and mapping software system



accelerate the processing, scan matching utilizes multiple CPU threads in parallel. The map is stored in voxels to accelerate point query during scan matching. Because the motion is estimated at different frequencies, a fourth module in the system (gray module in Fig. 3) takes these motion estimates for integration. The output holds both high accuracy and low latency beneficial for vehicle control.

The modularized system also ensures robustness w.r.t. sensor degradation, by selecting “healthy” modes of the sensors when forming the final solution. For example, when a camera is in a low-light or texture-less environment such as pointing to a clean and white wall, or a laser is in a symmetric or extruded environment such as a long and straight corridor, processing typically fails to generate valid motion estimates. Our system automatically determines a degraded subspace in the problem state space [11]. When degradation happens, the system only solves the problem partially in the well-conditioned subspace of each module. The result is that the “healthy” parts are combined to produce the final, valid motion estimates.

2.3 *Localization and Map Merging*

When a map is available, the method described in Sect. 2.2 can be extended to utilize the map for localization. This is using a scan matching method similar to the blue block in Fig. 3. The method extracts two types of geometric features – points on edges and planar surfaces, based on the curvature in local scans. Feature points are matched to the map. An edge point is matched to an edge line segment, and a planar point is matched to a local planar patch. On the map, the edge line segments and local planar patches are determined by examining the eigenvalues and eigenvectors associated with local point clusters. The map is stored in voxels to accelerate processing. The localization solves an optimization problem minimizing the overall distances between the feature points and their correspondences. Due to the fact that we use the high-accuracy odometry estimation to provide initial guess to the localization, the optimization usually converges in 2–3 iterations.

Compared to Sect. 2.2, the difference is that the localization does not process individual scans but stacks a number of scans for batch processing. Thanks to the high-accuracy odometry estimation, scans are registered precisely in a local coordinate frame where drift is negligible over a short period of time (a few seconds). A comparison is given in Fig. 4, where Fig. 4a is a single scan that is matched in Sect. 2.2 (scan matching executes at 5 Hz), and Fig. 4b shows stacked scans over 2 s, which are matched during localization (scan matching runs at 0.5 Hz). One can see the stacked scans contain significantly more structural details, contributing to the localization accuracy and robustness w.r.t. environmental changes. Additionally, low-frequency execution keeps the CPU usage to be minimal for onboard processing (localization consumes about 10% of a CPU thread).

Our localization is compared to a particle filter based implementation. The odometry estimation provides the motion model to the particle filter. It uses a number of 50 particles. At each update step, the particles are resampled based on low-variance

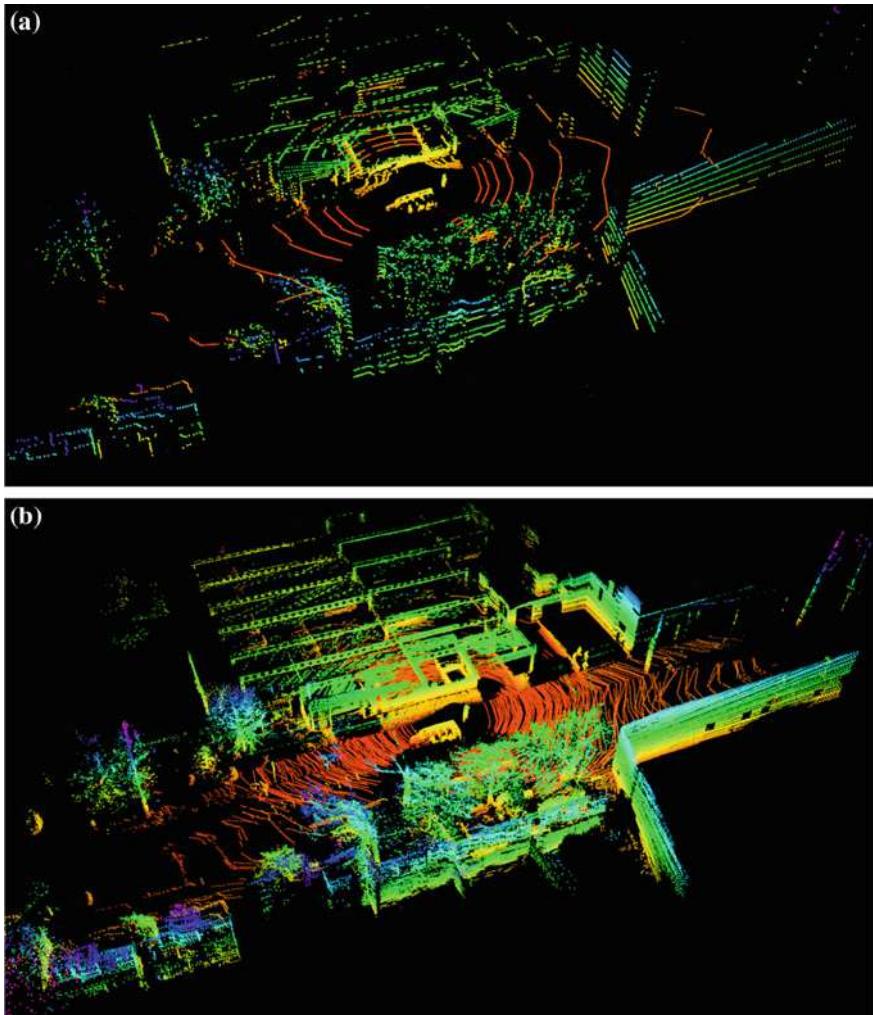


Fig. 4 Comparison of scans involved in odometry estimation and localization. In odometry estimation, each individual scan is processed in scan matching at 5 Hz. While in localization, a number of locally registered scans are stacked and batch processed at 0.5 Hz

resampling [12]. Comparison results are shown in Fig. 5 and Table 1. Here, errors are defined as the absolute distances from localized scans to the map. During the evaluation, we choose a number of planar surfaces and use the distances between points in localized scans to the corresponding planar patches on the map. Figure 5 shows the error distribution. When running the particle filter at the same frequency as our method (0.5 Hz), the resulting error is five times as large. While in Table 1, the CPU processing time is more than twice of ours. In another test, running the particle

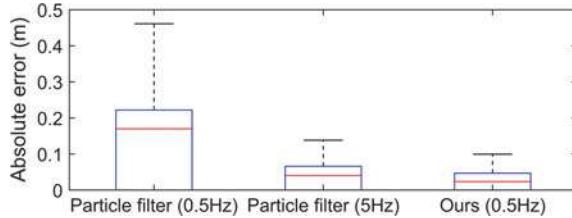


Fig. 5 Comparison of scan matching accuracy in localization. We use the absolute distances from localized scans to the corresponding local patches on the map as the metric. Points are selected from a number of planar surfaces in Fig. 4. The red lines illustrate medians, the blue boxes represent 75% of the distributions, and the black lines are the maximum errors

Table 1 Comparsion of CPU processing time in localization. When running the particle filter at 5 Hz, sensor data is processed at 25% of real-time speed due to high CPU demand

Method	Particle filter		Ours
Frequency (Hz)	0.5	5	0.5
Time per execution (ms)	493	478	214
Time per second (ms)	247	2390	107

filter at 5 Hz helps reduce the error to be slightly larger than our method. However, the corresponding CPU processing time increases to over 22x of ours. These results imply a particle filter based method does not take full advantage of the high-accuracy odometry estimation as compared to our implementation.

3 On Sensor Orientation

In previous work [3], we studied the system performance w.r.t. sensor degradation. We concluded that the system is robust to individual sensor failures, i.e. when the laser or camera is degraded, the corresponding module is bypassed while the rest of the system is staggered to generate the solution. In this section, we further carry out studies where the sensor pack is orientated differently. This is especially motivated by aerial mapping due to the fact that during “up and away” flights, the sensor pack has to be tilted downward in order to capture data from the ground.

The first set of study is conducted in Figs. 6 and 7. First, we carry the sensor pack horizontally in a garage building. Figure 6a shows the map built and sensor trajectory. Figure 6b is a single scan. In this scenario, the scan contains sufficient structural information. When bypassing the camera processing module (green module in Fig. 3), the system produces the same trajectory as the full pipeline. On the other hand, we run another test with the sensor pack tilted vertically down toward the ground. The results are shown in Fig. 7. In this scenario, structural information in a scan is much sparser (see Fig. 7b). The processing fails without usage of the camera and succeeds

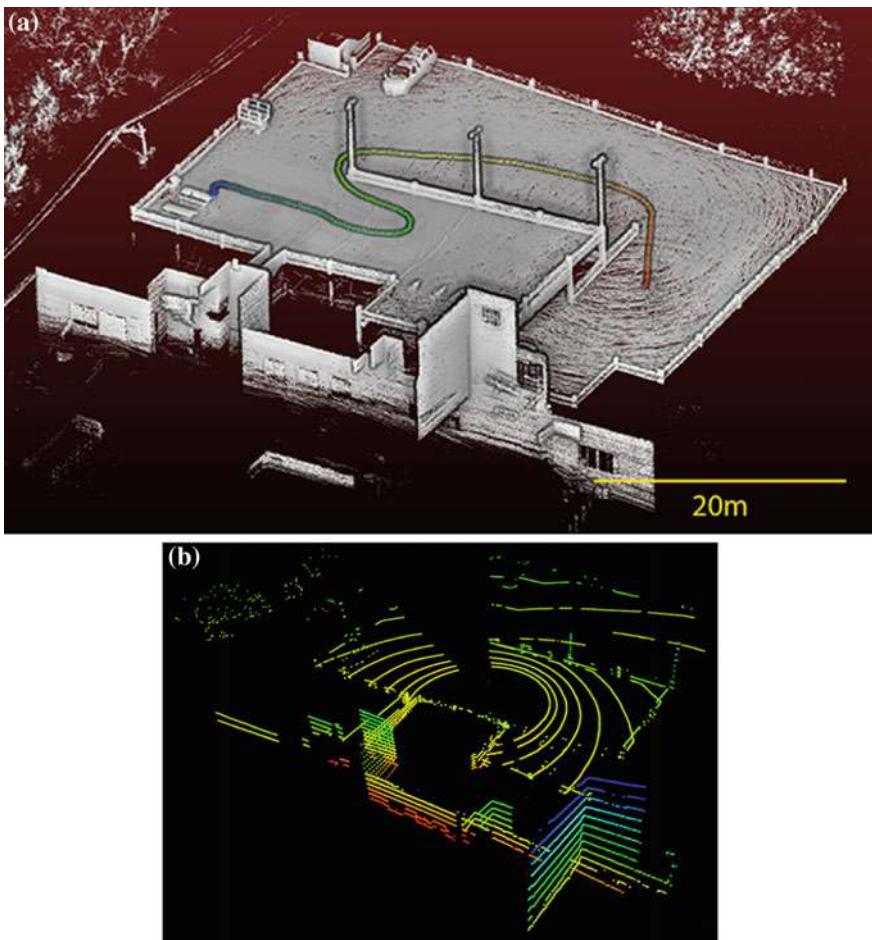


Fig. 6 Example of horizontally orientated sensor test. The processing succeeds with and without the camera, both yield the same map and sensor trajectory (colored curve starting with blue and ending with red) in (a). (b) shows a raw laser scan during the test which contains plenty of structural information for scan matching based methods to function

with the full pipeline. The results indicate the camera is critical for high-altitude flights where tilting of the sensor pack is required.

The second set of study compares the drift rate w.r.t. different sensor orientations. As shown in Fig. 8, the sensor pack is held by an operator walking through a circle at 1–2 m/s speed with an overall traveling distance of 410m. Figure 8a shows the map built and sensor trajectory with a horizontally orientated sensor configuration. The sensor is started and stopped at the same position. The test produces 0.18 m of drift through the path, resulting in 0.04% of relative position error in comparison to the distance traveled. Then, the operator repeats the path with two sensor packs

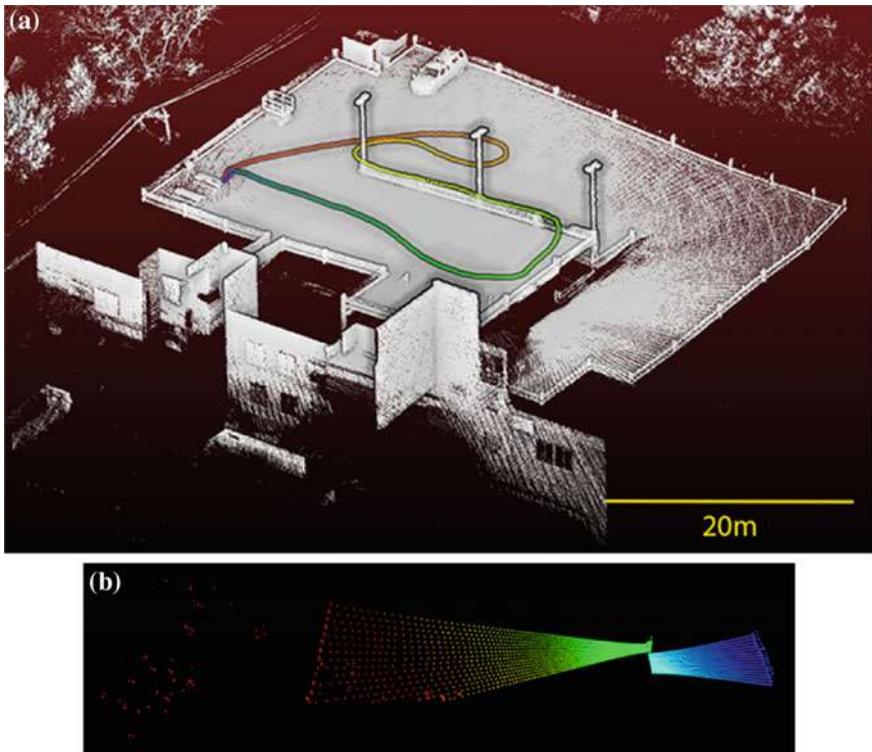


Fig. 7 Example of vertically orientated sensor test. The processing succeeds only when the camera is present, which yields the map and sensor trajectory in (a). Due to the lack of structural information in laser data (see a raw laser scan in (b)), odometry estimation requires assistance from the camera. Methods only relying on scan matching are not functional

held at 45° and 90° angles, respectively. The resulting sensor trajectories are shown in Fig. 8b. Clearly, tilting introduces more drift, where the relative position errors are 0.6% at 45° (blue dash curve) and 1.4% at 90° (red dash-dot curve). Finally, by localizing on the map in Fig. 8a using the method in Sect. 2.3, the drift is canceled and both configurations result in trajectories as the black solid curve.

4 Flight Experiments

4.1 Drone Hardware

The drone platform is a DJI S1000 aircraft as shown Fig. 9. The aircraft weights 6.8 kg (including batteries) and can carry a maximum of 4.2 kg payload. The sen-

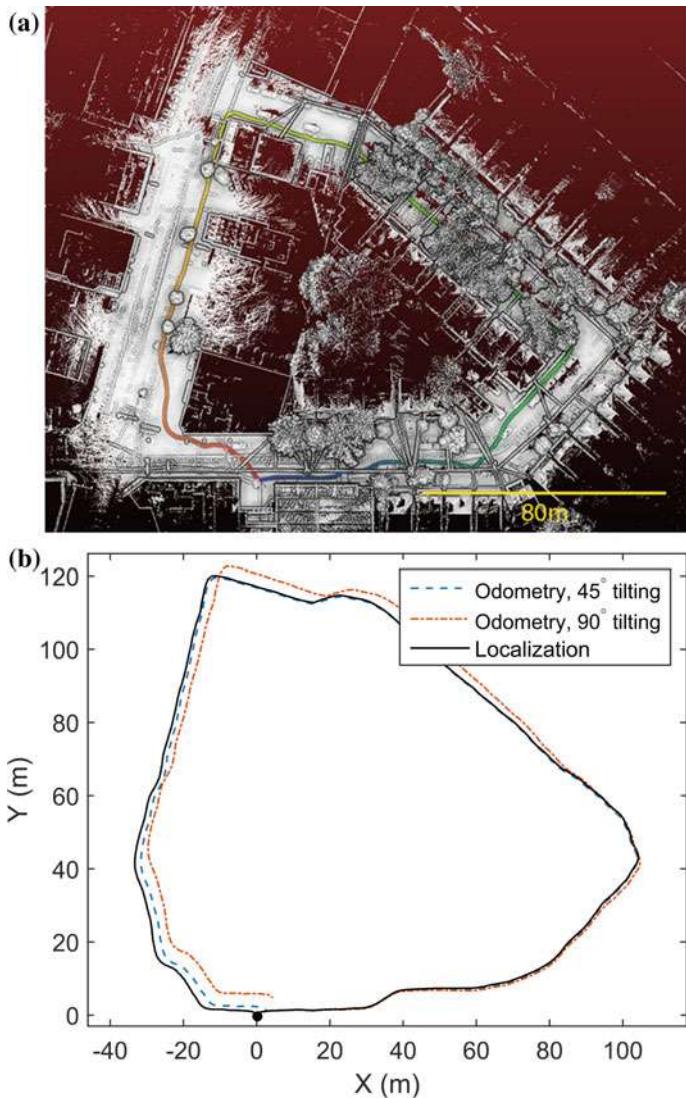
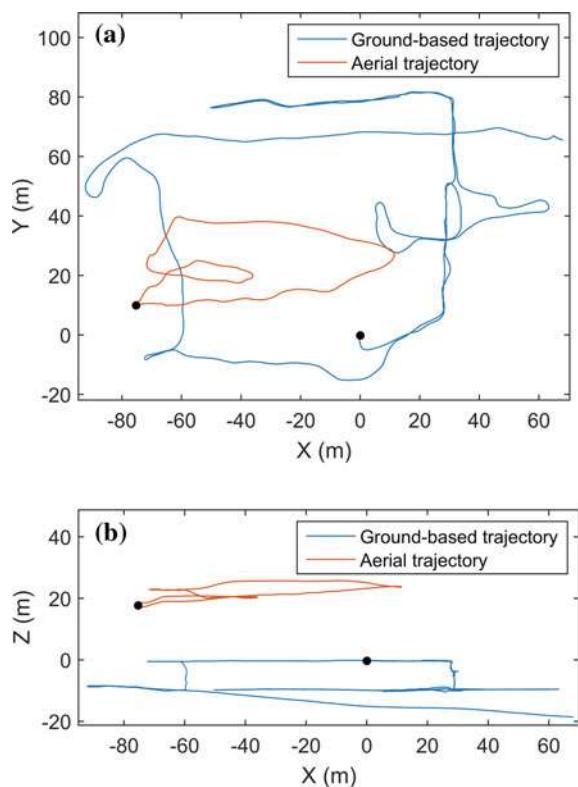


Fig. 8 Accuracy comparison between horizontally orientated and downward tilted sensor tests. **a** shows the map and sensor trajectory (colored curve starting with blue and ending with red) of a horizontally orientated setup. The sensor pack is started and stopped at the same position, and held by an operator who walks at 1–2 m/s through a circle for 410 m. The odometry estimation produces 0.18 m of drift resulting in 0.04% of relative position error w.r.t. the distance traveled. **b** shows results of a repeated test with the operator holding two sensor packs, one at 45° (blue dash-dot curve) and the other at 90° (red dash-dot curve). Tilting at 45° results in 2.3 m of drift and correspondingly 0.6% of relative position error, while tilting at 90° generates 5.8 m of drift and 1.4% of relative position error. Finally by localizing w.r.t. the map in **(a)**, both setups produce trajectories as the black solid curve. All trajectories start at the black dot



Fig. 9 DJI S1000 aircraft with sensor pack. The aircraft is built with a GPS receiver (on top of the aircraft). GPS data is not used in mapping or autonomous missions as in this paper

Fig. 10 Sensor trajectories corresponding to Fig. 1. The ground-based mapping is at 1–2 m/s with an overall distance of 914 m. The aerial mapping is at 2–3 m/s with 269 m of travel. The trajectories start at the black dots



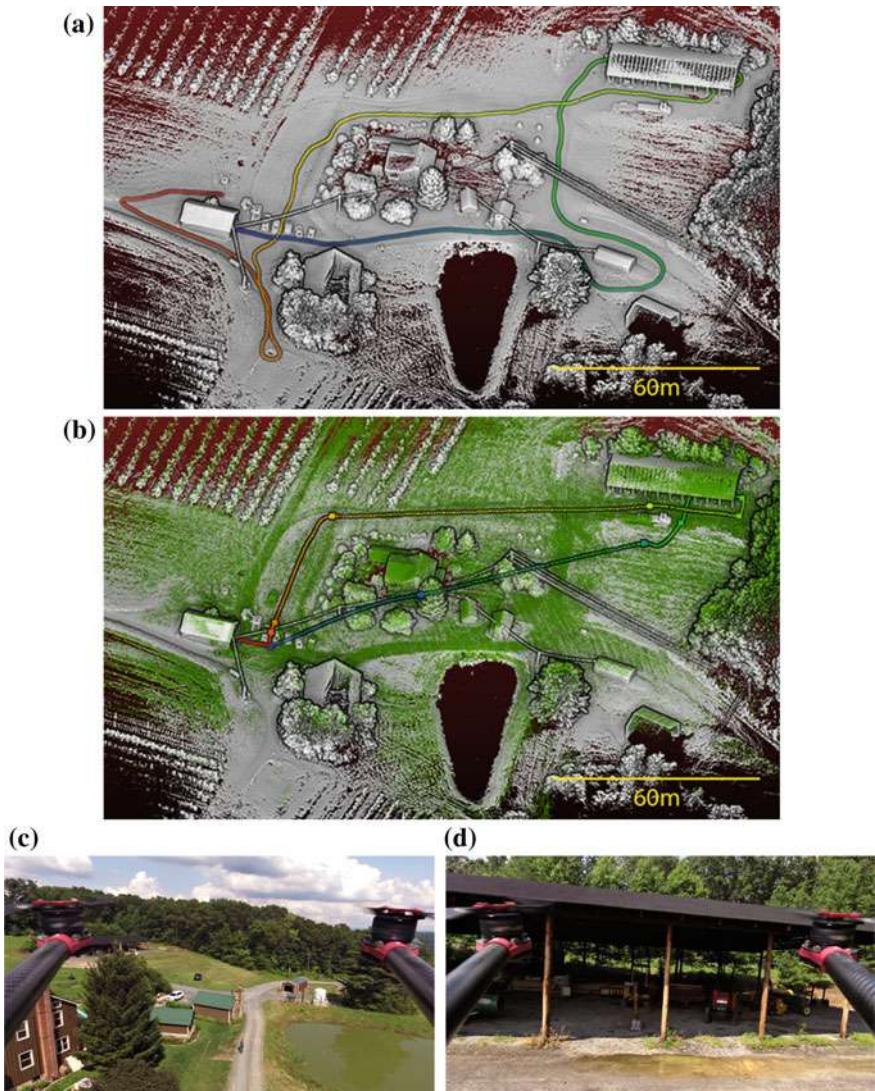


Fig. 11 Autonomous flight result through a shed. **a** shows the ground-based map and sensor trajectory (colored curve starting with blue and ending with red) built by an operator holding the sensor pack and walking at 1–2 m/s for 672m. With the ground-based map, way-points are defined and the drone executes an autonomous mission. It takes off inside a shed on the left side of the figure, navigates across the site at 4 m/s, passes through another shed on the right side at 2 m/s, and returns to the first shed over 390 m of travel. In **(b)**, the green point cloud is the aerial map, the colored curve is the sensor trajectory in the air, and the large points on the curve are the way-points. **c** and **d** are two images taken by an onboard camera when the drone flies between the sheds and is about to enter the shed on the right. **e** shows the estimated speed through the route

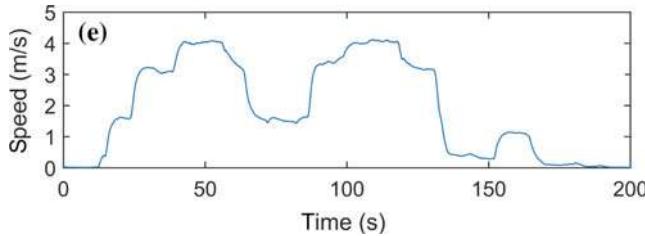


Fig. 11 (continued)

sor/computer pack is mounted to the bottom of the aircraft, weighting 1.7kg. The bottom right of the figure shows the remote controller. During autonomous missions, the remote controller is operated by a safety pilot to override the autonomy if necessary. Note that the aircraft is built with a GPS receiver (on top of the aircraft). GPS data is not used in mapping or autonomous missions through the paper.

4.2 Teleoperated Flight Results

In the first collaborative mapping experiment, an operator holds the sensor pack and walks around a building. Results are shown in Fig. 1. In Fig. 1a, the ground-based mapping covers surroundings of the building in detail, conducted at 1–2 m/s over 914 m of travel. As expected, the roof of the building is empty on the map. Second, the drone is teleoperated to fly over the building. In Fig. 1b, the flight is conducted at 2–3 m/s with a traveling distance of 269m. The processing uses localization w.r.t. the map in Fig. 1a. That way, the aerial map (green points) is merged with the ground-based map (white points). After the ground-based map is built, the take-off position of the drone is determined on the map. The sensor starting pose for the aerial mapping is known, and from which, the localization starts. Figure 10 presents the aerial and ground-based sensor trajectories, in top-down and side views.

4.3 Autonomous Flight Results

Further, we conduct autonomous flights to realize aerial mapping. In Fig. 11, we first build a ground-based map by hand-held mapping at 1–2 m/s for 672 m of travel around the flight area. The map and sensor trajectory are shown in Fig. 11a. Then based on the map, way-points are defined and the drone follows the way-points to conduct aerial mapping. As shown in Fig. 11b, the colored curve is the flight path, the large colored points on the curve are the way-points, and the green points form the aerial map. In this experiment, the drone takes off inside a shed on the left side of the figure, flies across the site and passes through another shed on the right side, then

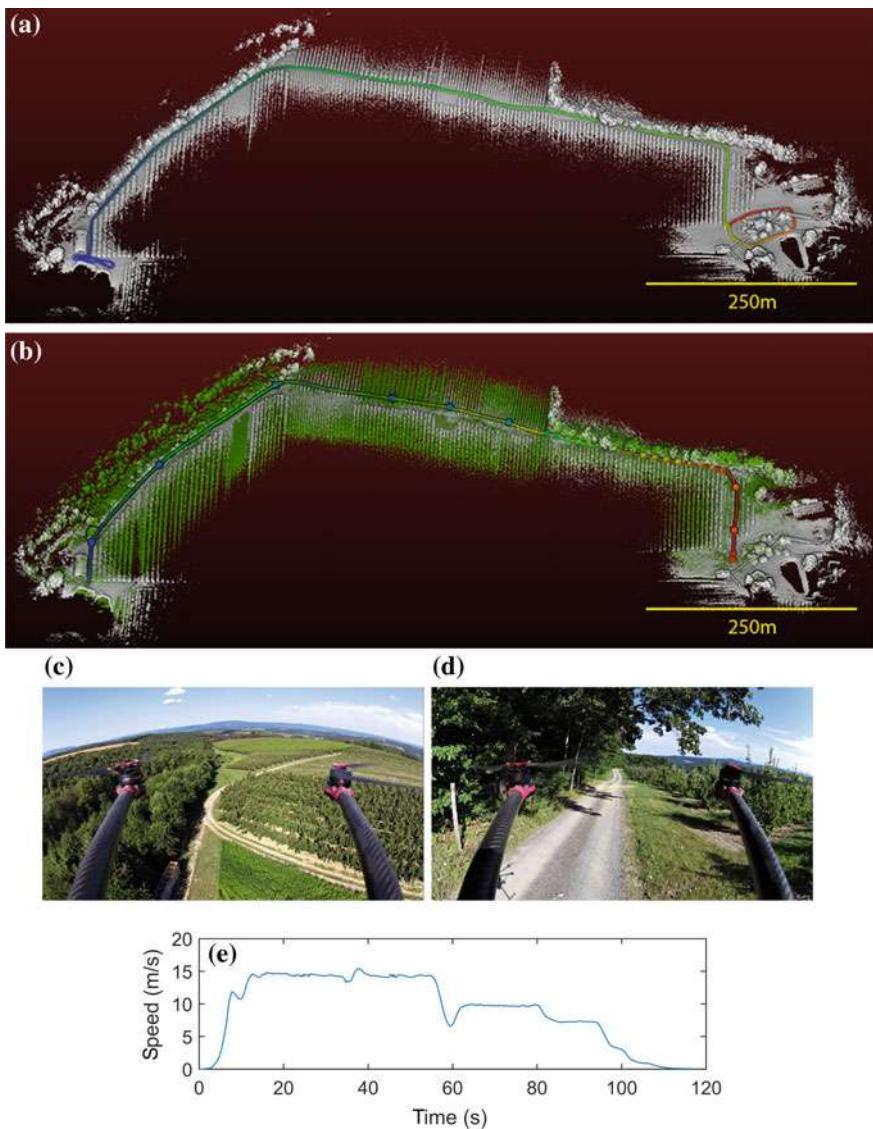


Fig. 12 Autonomous flight result over a long-run. **a** shows the ground-based map and sensor trajectory (colored curve starting with blue and ending with red) generated by the sensor pack mounted to an off-road vehicle, driven at 10 m/s for 1463m. **b** shows the result of the autonomous mission from an 1118 m flight. During the mission, the drone first flies at 20 m above the ground at 15 m/s and then descents to 2 m above the ground through a line of trees at 10 m/s. The green point cloud is the aerial map, the colored curve is the flight trajectory, and the large points on the curve are the way-points. **c** and **d** are from an onboard camera when the drone flies high above the trees and low underneath the trees. **e** shows the estimated speed through the route

returns to the first shed to land. The speed is 4 m/s crossing the site and 2 m/s passing through the shed. Figure 11c, d are two images taken by an onboard camera when the drone flies toward the shed on the right and is about to enter the shed. Figure 11e shows the estimated speed during the mission.

Finally, we conduct another experiment over a longer distance. As shown in Fig. 12, the ground-based mapping involves an off-road vehicle driven at 10 m/s from the left end to the right end, over 1463 m of travel. With the ground-based map and way-points, the autonomous flight crosses the site. Upon take-off, the drone ascends to 20 m high above the ground at 15 m/s. Then, it descends to 2 m above the ground to fly through a line of trees at 10 m/s. The flight path is 1118 m long as the colored curve in Fig. 12b. Two images are taken as the drone flies high above the trees (see Fig. 12c) and low underneath the trees (see Fig. 12d).

5 Conclusion

The paper presents experimental studies on collaborative mapping from the ground and air. Utilizing a miniature sensor pack consisted of a laser scanner, a camera, and a low-grade IMU, we demonstrate aerial and ground-based mapping while the maps are merged in real-time during the flights. This benefits from a data processing pipeline with multi-layer optimization—modules in the front execute at high frequencies to handle aggressive motion and produce high-rate motion estimates, while modules in the back run at low frequencies and take sufficient computation to warrant accuracy. We evaluate the system in both teleoperated and autonomous flights, through cluttered environments and at high speeds (up to 15 m/s).

Acknowledgements Special thanks are given to R. Chadha, D. Murphy, S. Spiker, K. Rugg, D. Duggins, K. Dowling, and M. Bergerman for their insightful inputs and invaluable help.

References

1. Zhang, J., Singh, S.: LOAM: Lidar odometry and mapping in real-time. In: Robotics: Science and Systems Conference (RSS), Berkeley, CA, July 2014
2. Zhang, J., Kaess, M., Singh, S.: Real-time depth enhanced monocular odometry. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL, Sept 2014
3. Zhang, J., Singh, S.: Enabling aggressive motion estimation at low-drift and accurate mapping in real-time. In: IEEE International Conference on Robotics and Automation (ICRA), Singapore, May 2017
4. Fallon, M., Johannsson, H., Leonard, J.: Efficient scene simulation for robust monte carlo localization using an rgb-d camera. In: IEEE International Conference on Robotics and Automation (ICRA), St. Paul, MN, May 2012
5. Ganganath, N., Leung, H.: Mobile robot localization using odometry and kinect sensor. In: IEEE International Conference on Emerging Signal Processing Applications (ESPA), Las Vegas, Nevada, Jan 2012

6. Tipaldi, G., Meyer-Delius, D., Burgard, W.: Lifelong localization in changing environments. *Int. J. Robot. Res.* **32**(14), 1662–1678 (2013)
7. Nurminen, H., Ristimäki, A., Ali-Löytty, S., Piché, R.: Particle filter and smoother for indoor localization. In: International Conference on Indoor Positioning and Indoor Navigation (IPIN), Montbliard, France, Oct 2013
8. Redondi, A., Chirico, M., Borsani, L., Cesana, M., Tagliasacchi, M.: An integrated system based on wireless sensor networks for patient monitoring, localization and tracking. *Ad Hoc Netw.* **11**(1), 39–53 (2013)
9. Bry, A., Richter, C., Bachrach, A., Roy, N.: Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments. *Int. J. Robot. Res.* **34**(7), 969–1002 (2015)
10. Nieuwenhuisen, M., Droeschel, D., Beul, M., Behnke, S.: Autonomous navigation for micro aerial vehicles in complex gnss-denied environments. *J. Intell. Robot. Syst.* (2015)
11. Zhang, J., Kaess, M., Singh, S.: On degeneracy of optimization-based state estimation problems. In: IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, May 2016
12. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. The MIT Press, MA, Cambridge (2005)

Part VI

Navigation and Planning

I Can See for Miles and Miles: An Extended Field Test of Visual Teach and Repeat 2.0

Michael Paton, Kirk MacTavish, Laszlo-Peter Berczi,
Sebastian Kai van Es and Timothy D. Barfoot

Abstract Autonomous path-following systems based on the Teach and Repeat paradigm allow robots to traverse extensive networks of manually driven paths using on-board sensors. These methods are well suited for applications that involve repeated traversals of constrained paths such as factory floors, orchards, and mines. In order for path-following systems to be viable for these applications they must be able to navigate large distances over long time periods, a challenging task for vision-based systems that are susceptible to appearance change. This paper details Visual Teach and Repeat 2.0, a vision-based path-following system capable of safe, long-term navigation over large-scale networks of connected paths in unstructured, outdoor environments. These tasks are achieved through the use of a suite of novel, multi-experience, vision-based navigation algorithms. We have validated our system experimentally through an eleven-day field test in an untended gravel pit in Sudbury, Canada, where we incrementally built and autonomously traversed a 5 Km network of paths. Over the span of the field test, the robot logged over 140 Km of autonomous driving with an autonomy rate of 99.6%, despite experiencing significant appearance change due to lighting and weather, including driving at night using headlights.

1 Introduction

Autonomous path-following algorithms based on the Teach and Repeat paradigm allow robots to repeat networks of connected paths previously driven by human operators using only on-board sensors.

The unique task of autonomously traversing a human-taught path gives a robot a strong prior on where it is safe to drive [2]. This allows for confident, autonomous navigation through rough, outdoor terrain that would otherwise be inaccessible or

M. Paton (✉) · K. MacTavish · L. -P. Berczi · S. K. van Es · T. D. Barfoot
University of Toronto Institute for Aerospace Studies,
Toronto, ON M3H 5T6, Canada
e-mail: mpaton@robotics.utias.utoronto.ca

T. D. Barfoot
e-mail: tim.barfoot@utoronto.ca

require complex, generic, and potentially risky terrain-assessment algorithms. Furthermore, these methods can be implemented to have bounded computation costs and minimal map sizes [7], making them well suited for long-range navigation. These benefits make autonomous path following appealing for industrial applications that consist of repeated traversals over constrained paths, such as factory floors, orchards, and mines. They are also well suited to applications that consist of autonomous exploration and retrotraverse such as search-and-rescue and hazardous-exploration robots. However, autonomous path-following systems suited for these applications need the ability to navigate large-scale environments over long time periods. Furthermore, they require constant metric localization to the manually driven path as input to a path-tracking controller to ensure minimal drift, and the ability to recognize and cope with obstacles blocking the path. These requirements pose a serious challenge for vision-based systems whose advantages of cost and commercial ubiquity come at the expense of robustness to appearance change. This paper presents Visual Teach and Repeat (VT&R) 2.0, a path-following system capable of long-term, navigation on large-scale networks of paths using only a stereo camera through the integration of a suite of recently published navigation algorithms [2, 14, 19, 21]. Furthermore, we present results from an extensive outdoor field test, illustrated in Fig. 1, that consisted of incrementally building and autonomously traversing a 5 km network of connected paths over the span of eleven days at an untended gravel pit in Sudbury, Canada. Over these eleven days, the robot traversed over 140 km with an autonomy rate of 99.6% of distance traveled while experiencing significant appearance change due to lighting, weather, and terrain modification.

The remainder of this paper is outlined as follows. Work related to VT&R 2.0 is summarized in Sect. 2. Details of the VT&R 2.0 system are presented in Sect. 3.



Fig. 1 A Grizzly RUV deployed with an autonomous path-following algorithm navigating a 5 km network of manually taught paths. Applications that rely on repeated traversals of constrained paths will greatly benefit from such algorithms: such as mining, agriculture, and patrol robots. In order to be useful for such applications, autonomous path-following algorithms will need to be able to cope with large-scale maps, and appearance change over long periods of time. Using a novel multi-experience localization and mapping method, the robot pictured above autonomously traversed over 140 km over two weeks, experiencing significant appearance changes in the environment

Section 4 provides information on the experimental set up of the field test with results presented in Sect. 5. Failure conditions of VT&R 2.0 and lessons learned from the field are presented in Sect. 6 with a conclusion in Sect. 7.

2 Related Work

VT&R 2.0 is an evolution of our previous system that provides short-term, vision-based path following on large-scale tree structures, VT&R 1.0 [6, 7]. While effective at long-range navigation, the system is highly susceptible to lighting change while operating outdoors, limiting successful operation to a window of only a few hours. This method was extended to multi-day operation through the use of color-constant images and multiple stereo cameras [18], but is still susceptible to longer-term appearance change due to weather and seasons. Apart from VT&R 1.0, short-term, vision-based path-following systems have been demonstrated using heading-only navigation [4, 10]. Despite the fusion of wheel odometry, the lack of a reliable translation estimate makes navigation in constrained environments unsafe. Path-following systems that rely on active sensors provide long-term, lighting-invariant navigation and have been demonstrated using appearance-based methods on lidar-generated intensity images [15] and point-cloud registration on dense 3D scans [11]. However, these systems struggle with motion distortion issues and rely on expensive and sometimes commercially unavailable sensors.

It is well known that constant-time localization and mapping can be achieved using globally inconsistent, topometric pose graphs [20], even with loop closures. Long-range navigation with VT&R 2.0 is possible through the use of this representation, allowing the system to build large-scale networks of paths with smooth path tracking across loop closures [21]. Long-term navigation with VT&R 2.0 is achieved through the use of the Multi-Experience Localization (MEL) algorithm [19]. This method is inspired by the Experience-Based Navigation (EBN) framework [5], which localizes against a number of past experiences, providing metric localization to the most similar experience. In contrast, MEL provides localization to the *single* manually taught path using experiences gathered *during* autonomous operation. Due to their reliance on multiple experiences, both EBN and MEL are inherently computationally intractable if left unbounded. A viable strategy to overcome this issue is to select a fixed-size subset of experiences for the localizer. [12] use past localization success to recommend experiences most likely to localize well in the future for the EBN system. In contrast, our system selects experiences most similar visually to the *live view* [14]. Another work closely related to MEL is the ‘Summary Maps’ method [16]. This method provides metric localization across seasonal appearance change through a multi-experience map that is pruned and curated offline. This method is successful, but requires downtime between traverses to perform mapping on an offline server—a constraint which restricts use for some applications.

Safe navigation with VT&R 2.0 is achieved through a place-dependent, multi-experience terrain-assessment algorithm [2]. Traditional terrain-assessment methods

are limited by the ability to label the terrain in a human-interpretable way, often resulting in conservative estimates [8]. More recent methods alleviate these issues through the use of previous experiences to learn traversability [3, 9], but still lead to conservative estimates in difficult environments. In contrast, our system learns a separate classifier at every place on the path [2], achieving better performance than a single general classifier applied to every place.

3 System Overview

This section provides an overview of the following components of VT&R 2.0: (i) map structure, (ii) network construction, (iii) route planning, (iv) autonomous path following, (v) multi-experience localization, (vi) appearance-based experience selection, and vii) place-dependent terrain assessment.

The Spatio-Temporal Pose Graph: Our system stores the map in a database indexed by a Spatio-Temporal Pose Graph (STPG). Depicted in Fig. 2, the graph structure contains vertices, temporal edges, and spatial edges. Vertices, each with a reference frame, \mathcal{F} , store raw sensor observations and triangulated 3D landmarks with associated covariances and descriptors.¹ The edges link vertices with uncertain, relative $SE(3)$ transformations. Temporal edges (blue lines) connect temporally adjacent vertices, while spatial edges (green lines) connect those that are spatially close (but may be temporally distant). Edges are considered *privileged* (solid lines) if the robot was being manually driven, or *autonomous* (dashed lines) if the robot was autonomously repeating a route. VT&R 2.0 uses the STPG to represent a multi-experience network of connected paths, where each *experience* is a collection of vertices linked by temporal edges. The subgraph containing *all* privileged experiences represents the collection of safe, drivable paths. Autonomous experiences linked to this privileged subgraph are used to aid the navigation algorithms, by providing a wealth of place-specific information.

Network Construction: VT&R 2.0 is operated through the user interface shown in Fig. 3, which provides an intuitive means to construct networks of paths, and command autonomous traversal to goals on the networks. A network is built by adding a teach goal (left panel) and manually driving the robot; this adds privileged experiences to the STPG. If the network exists prior to the teach, then a localization search centered around the robot's topological state estimate is performed. Upon successful localization, a privileged spatial edge is created, *branching* the new experience off the existing network. The robot will then add vertices and temporal edges to this new experience through our stereo VO pipeline [19] as it drives. Live experiences can be merged back into the network through loop closures initiated through the UI. The operator selects a region for the merge, and the system attempts to localize the

¹We use SURF features triangulated from greyscale and color-constant stereo measurements in our implementation, but the overall system is generic to any point-based, sparse visual feature.

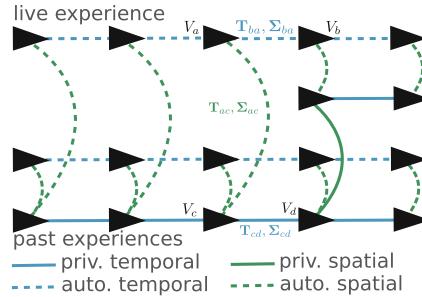
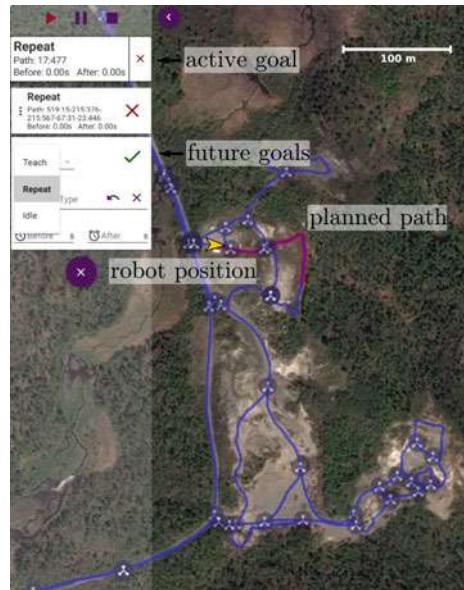


Fig. 2 Overview of the STPG data structure used to represent our multi-experience network of paths. Experiences are shown as rows of vertices (black triangles) connected metrically through blue temporal edges calculated via VO while the robot is either being manually driven (solid) or autonomously repeating (dashed). Experiences are related metrically through green, spatial edges, calculated through localization and can either be added autonomously while driving (dashed) or manually while adding a branch or loop closure (solid)

Fig. 3 Overview of the VT&R 2.0 user interface used to build networks of connected paths and command the rover to autonomously traverse the network



live view to the privileged vertices in this selection. Upon successful localization, the operator can confirm the result, adding a privileged spatial edge from the live vertex to the target. Otherwise, the user may continue driving to improve alignment, or cancel the merge.

Route Planning: Autonomous traversal begins with planning a route in the UI by adding a repeat goal (active in Fig. 3) to the queue, and selecting a sequence of waypoints for the robot. To plan the path, the system uses the safe, privileged subgraph of the network, including privileged experiences, and privileged spatial

edges from branching and merging. Given the robot's current topological position in the network and a set of waypoints, the planner finds the minimum-cost path that covers all selected waypoints in sequential order. Two edge costs we find useful are the path distance, and the temporal age between the live experience and the most recent traversal of the edge (combining absolute time and time-of-day). Since our system localizes against multiple autonomous experiences, planning over recently traversed edges is a heuristic to improve the likelihood of successful localization.

Path Following: Given a planned route through the privileged network, the system creates a new autonomous experience in the network, and attempts to localize the live view to a vertex in the privileged path. This process is identical to the first step of teaching, except the added spatial edge is flagged as autonomous. Once connected to the privileged experience, the system propagates the position estimate using stereo VO, which is sent to a model-predictive-control path tracker [17]. When a new keyframe is added as a vertex in the live run, it is localized to the closest vertex in the privileged path using MEL, detailed in the next subsection. Upon successful localization, an autonomous spatial edge is added between the two vertices.

Multi-Experience Localization: VT&R 2.0 provides metric localization with respect to the privileged path through the Multi-Experience Localization (MEL) algorithm [19]. Illustrated in Fig. 4, MEL estimates the uncertain transformation, $\{\hat{T}_{bd}, \hat{\Sigma}_{bd}\}$ (purple, dashed line), between the live vertex, V_b , and the estimated closest privileged vertex, V_d . This process takes a window of recommended experiences (green rectangles) as input, and transforms their landmarks to the coordinate frame

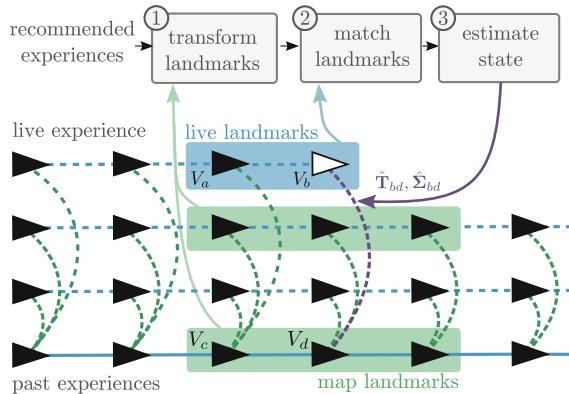


Fig. 4 An overview of the MEL algorithm. Given a selection of experiences to localize against (green rectangles), The algorithm solves for the transformation between the vertex in the live experience, V_b , and the vertex in the map experience, V_d , by transforming all landmarks in the localization window into the coordinate frame of V_d and performing a simple keyframe-to-keyframe bundle adjustment problem

of the target privileged vertex, V_d , using the temporal and spatial edges. The uncertainty of the transformed landmarks includes that from the source landmark and the transform itself. Landmarks originating from the live vertex, V_b , are then matched against these consolidated map landmarks. Matching consists of checking the similarity of unmatched, live landmarks to landmarks at each map vertex, starting at V_d , and expanding in a breadth-first search. Similarity is determined by the key-point Laplacian sign, feature descriptor distance, and a weak check on projected landmark position in image space using the prior. The process continues until one of the following exit conditions is met: (i) enough matches are found, (ii) the time limit has expired, or (iii) the map window is exhausted. The matches are first verified to remove outliers and initialize \mathbf{T}_{bd} , and a motion prior is built by compounding VO transforms and the most recent successful localization. The uncertain transform, $\{\mathbf{T}_{bd}, \Sigma_{bd}\}$, is iteratively refined in a robust, nonlinear, least-squares optimization using our Simultaneous Trajectory Estimation and Mapping (STEAM) engine [1].

Appearance-Based Experience Selection: Matching against all intermediate landmarks in the MEL algorithm becomes computationally intractable as the number of experiences grows. Only a subset of these are actually required to localize the live vertex—if we restrict our localization problem to the most relevant, we maintain real-time performance. To determine a relevant subset of experiences, we would like to find those with similar appearance to the *live view*. To this end, we have developed an algorithm [14], that selects a small subset of experiences based on a BoW appearance summary, illustrated in Fig. 5.

Each vertex in the STPG contains a BoW descriptor, quantized from the stable (tracked by VO) features in the keyframe against a local visual vocabulary. To improve robustness to viewpoint and signal-to-noise ratio, a grouped descriptor is

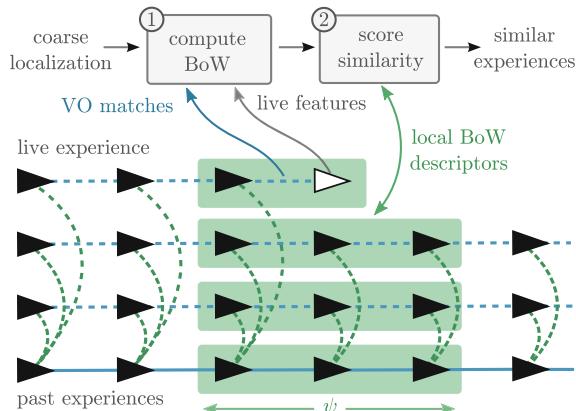


Fig. 5 An overview of the experience selection algorithm using BoW. 1. Given the VO feature matches for the live frame, a sliding-window BoW descriptor is constructed. 2. The BoW descriptor is compared against those for past experiences, centered on the coarse localization. The most similar experiences are selected for use in the remainder of the localization problem

formed by summing the descriptors from a local group of vertices, ψ , from the same experience [13]. At the start of each MEL problem, the local BoW descriptor (green rectangle) around the live vertex (white triangle) is compared to those of each past experience (lower green rectangles) using the cosine similarity. The N experiences with the highest similarity to the live experience are selected for MEL, where N is chosen to maintain real-time performance (5–10). Since this comparison is very fast, this method allows us to triage a large number (100s) of experiences very quickly.

Place-Dependent Terrain Assessment: We exploit the fact that during VT&R, the robot only needs to assess the traversability of terrain that it has already seen and driven. All of the drivable paths were at one point manually taught, and we assume that they were safe at that point in time. Traversability of the terrain ahead of the robot is determined by comparing it to known safe examples from previous experiences, and labeling any terrain that is sufficiently different as unsafe. This reduces the problem of terrain assessment to the simpler problem of change detection—allowing slow, gradual change over the lifetime of the path (such as growing grass), and stopping for large, sudden changes (such as a fallen tree). This place-dependent terrain-assessment algorithm first appeared in [2], and an overview of the pipeline is shown in Fig. 6. Patches are compared by taking the absolute difference of individual cell heights between the lookahead patches and each training patch individually. The patch cost is defined as the worst of these cell differences for each lookahead-training pair, and the traversability of the terrain is determined based on the lowest cost to previous experiences.

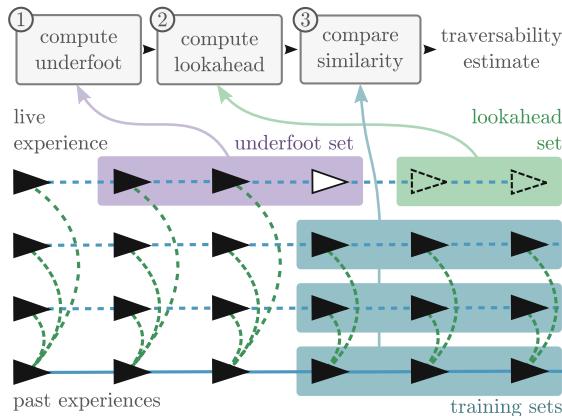


Fig. 6 An overview of the place-dependent terrain assessment. 1. Patches are computed at the robot location using recent data in the *underfoot set* (purple). 2. Patches are computed ahead of the robot at vertices in the *lookahead set* (green). 3. Lookahead patches are compared to previously computed underfoot patches from a spatially local *training set* (blue). The traversability ahead of the robot is based on its similarity to previous experiences

Fig. 7 Orthomosaic imagery of the 5 km network of paths at the Ethier Sand and Gravel in Sudbury, Ontario, Canada



4 Experimental Setup

Between the dates of 10/06/2016 and 16/06/2016, an extended field test of VT&R 2.0 was conducted at an unended gravel pit in Sudbury, Canada. Illustrated in Fig. 7, this field test was designed to stress test our system's ability to traverse a large-scale network of paths, in an unstructured environment, for an extended period of time, over significant appearance change. This location was selected for its variety of challenging environments, including rich vegetation and shifting sand with little visual texture.

The hardware configuration for the this field test consisted of a Clearpath Robotics Grizzly RUV, shown in Figs. 7 and 7, equipped with a Point Grey Research (PGR) Bumblebee XB3 stereo camera, and a pair of 9-watt LED headlights for nighttime operation. All of our VT&R 2.0 code ran on a Lenovo P50 laptop with a Intel® Core™ i7-6820HQ CPU equipped with a Quadro M2000 GPU.

Daily field test activities are outlined in Table 1. The majority of the network was taught on the first three days of testing during overcast conditions. During the first five days, the network was traversed from day to night, accumulating over 95 km of driving. On each of the remaining six days, the robot autonomously traversed between 5 and 10km a day. For each day, the autonomy rate remained above 99%. It is interesting to note that the majority of manual interventions occurred on the first two sunny days. Details on the causes of manual interventions are left for Sect. 6.

Table 1 Overview of the 2016 Sudbury Field Test

Date	Start (hh:mm)	End (hh:mm)	Weather	Teach Dist. (km)	Auto. Dist. (km)	Intervention Dist. (m)	Auto. Rate
2016/06/06	11:15	21:19	Rainy	1.56	13.45	12.56	99.91
2016/06/07	05:42	20:42	Overcast	1.88	17.61	8.74	99.95
2016/06/08	07:21	21:10	Rainy	0.83	23.95	72.66	99.70
2016/06/09	06:25	21:23	Sunny	0.03	19.10	148.69	99.22
2016/06/10	07:13	21:17	Sunny	0.33	20.76	171.62	99.17
2016/06/11	07:40	20:46	Sunny	0.22	09.64	41.49	99.57
2016/06/12	09:59	22:35	Sunny	0.20	08.95	20.04	99.78
2016/06/13	10:38	22:45	Sunny	0.00	07.87	27.65	99.65
2016/06/14	07:33	22:50	Sunny	0.00	07.63	53.42	99.30
2016/06/15	12:16	17:57	Sunny	0.00	07.37	2.27	99.97
2016/06/16	09:16	14:57	Sunny	0.00	04.15	2.85	99.93
Total	–	–	–	5.0	140.5	561.99	99.60

5 Results

This section presents the results of our field test. We begin with a detailed look at localization performance for a 240 m section of the network whose appearance is shown in Fig. 8 and path is highlighted in the top half of Fig. 1 as a blue line. The path begins in a meadow with tall grass and rapidly inclines up to a ridge containing thick vegetation bordered by a tree line, finishing with a steep descent into a sandy gravel pit. During the field test, this stretch of the network was autonomously traversed 109 times with only two manual interventions required. We chose to highlight the performance results of this section of the network in particular because it showcases every variety of appearance change seen during this field test and the rich vegetation and tree line make the environment challenging for vision-based navigation.

Figure 9 shows the relationship between the similarity score used by the experience selector to recommended experiences and feature inlier matches for all autonomous traverses of the example path. The plot shows that experiences with higher scores (>0.6) in general have higher feature match counts to each other, validating the selector’s ability to select experiences that will provide more matches.

Figure 10 shows localization results for all 109 autonomous traverses of the path. The left- and right-hand plots show the Cumulative Distribution Function (CDF) of the cross-track uncertainty and inlier feature matches, respectively. We define cross-track uncertainty as the one-standard-deviation uncertainty of our lateral translation estimate relative to the privileged path. The plot can be read as “for y % of the traverse, the localizer reported less than x m of cross-track uncertainty”. For the majority of traverses, the cross-track uncertainty stayed below 10 cm. The exceptions are three traverses where the uncertainty rose to 5, 10, and 15 cm at the 80% mark. This same trend can be seen in the right-hand feature inlier CDF, which shows that for all but



Fig. 8 The changing appearance of the 240 m example path. The top-left image was taken during network construction (no tire tracks), and all subsequent images were successfully localized to it using MEL. This section of the network is challenging for vision-based navigation due to strong shadows cast by a tree line and tall vegetation that moves in the wind

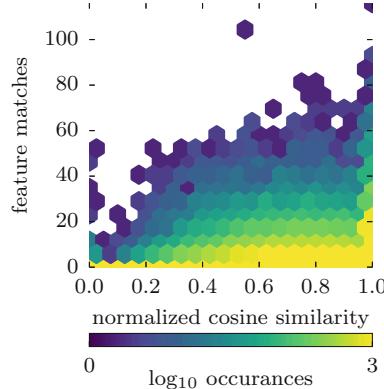


Fig. 9 Experience selector results: Normalized cosine similarity versus feature inlier matches for the vegetation-rich area

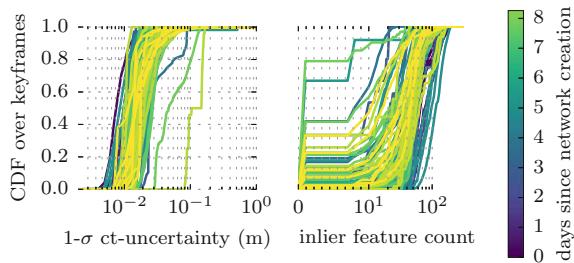


Fig. 10 Localization results for a 240 m section of the network, highlighted in Figs. 1 and 8. *left:* CDF of the $1 - \sigma$ localization uncertainty for all autonomous traverses on this path. *right:* CDF of the inlier feature matches for all autonomous traverses on this path. *note: log scale on x-axis*

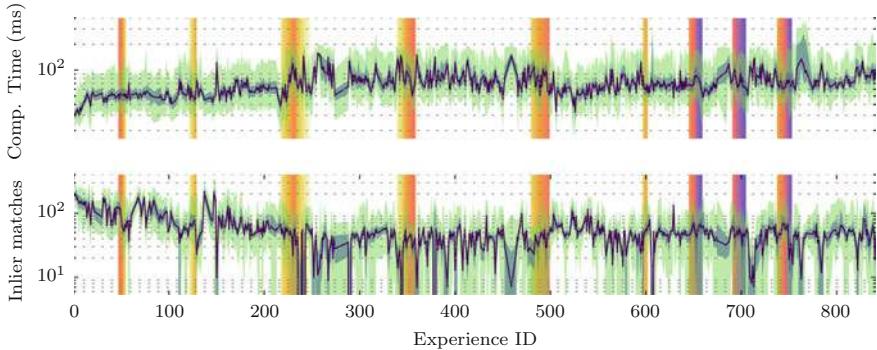


Fig. 11 Localization performance over the entire 11 day field trial with the daylight cycle colored. For each plot the dark line shows the median, the dark shaded area shows the interquartile region, and the light shaded area shows the mini/max extents. *top*: localization computation time of MEL including experience selection. *bottom*: inlier localization feature matches. *note*: log scale on y axis

three runs the localizer had more than 20–30 matches at the 80% mark. This high uncertainty and low match count can be correlated to traverses during sun glare and high winds, causing poor localization performance. More details on these localization failure cases is discussed in Sect. 6.

Figure 11 shows localization computation times and inlier feature matches for every autonomous traverse in the field test. For both plots, the dark-purple line shows the median values of the traverses, the dark-green, shaded area shows the upper (Q2) and lower (Q1) quartile, and the light-green, shaded area shows the min/max inlier values. The background of the plots are colored with respect to daylight conditions. Note the three instances of night driving on the far right. Timing results (top plot) show that the median localization computation time for most traverses is below or near the 100 ms mark. This value is safely within the tolerance for online driving for the VT&R 2.0 system, whose parallelization allows for VO at the frame rate of the sensor and localization at the rate of keyframe creation which is between 2 and 4 Hz.

Feature matches (bottom plot) show that the median and Q1 match count typically stayed between 100 and 30 inlier matches for the majority of traverses. However, there are instances of median values dropping to single digits with Q1 values of zero. These cases can be attributed to navigation in environments challenging for vision-based navigation during difficult weather conditions. These include high winds in vegetation-rich areas, sunshine and terrain modification in open desert areas, strong shadows on tree-lined roads, and glare when the elevation of the sun is low. A detailed analysis of these corner cases can be found in Sect. 6.

During the field tests, the VT&R 2.0 system ensured safe driving through the place-dependent terrain-assessment algorithm. A qualitative example of the algorithm is illustrated in Fig. 12, in which a human blocked the path on the vegetation-rich ridge area illustrated in Figs. 1 and 8. The robot was able to identify a change in the environment caused by the human obstacle and decided to stop. The

Fig. 12 Example of the place-dependent terrain-assessment algorithm correctly classifying a human in the path as unsafe



Table 2 Accuracy results for the place-dependent terrain-assessment algorithm.

	Obstacles		No obstacles		
	True Positives	False Negatives	True negatives	False positives	FPR
Count	29	0	20120	56	
Percentage (%)	0.14	0	99.58	0.28	0.28

quantitative performance of the algorithm is measured using the number of true positives (TP) (obstacles correctly identified), false positives (FP) (obstacle detected when no obstacle present), true negatives (TN) (safe terrain correctly identified), false negatives (FN) (missed obstacles), and false-positive rate (FPR), defined as $\frac{FP}{FP+TN}$. which can give more insight into the performance of the system since the nature of the terrain-assessment problem results in many more examples of safe terrain than examples of obstacles.

The terrain-assessment results are shown in Table 2. Notably, there are no false negatives, meaning that all obstacles were identified on the path (100% recall). In the absence of obstacles, only 56 false positives occurred in over 20000 estimates which translates to a false-positive rate of 0.28%. This is extremely low, and is a better indicator of algorithm performance than the typically reported precision (34%) because of the large imbalance of obstacle to obstacle-free examples. The results show that the algorithm is able to safely navigate in challenging environments with very few false positives per distance traveled.

6 Challenges/Lessons Learned

Difficult Conditions: For the majority of the 140 km of autonomous driving, the median inlier match count was at or above a safe value of 50, as shown in Fig. 11.

However, there were a select few traverses with median values near 10 matches, with a Q1 value of zero. This poor localization performance can be attributed to traversing in conditions that are difficult for vision-based navigation. The conditions

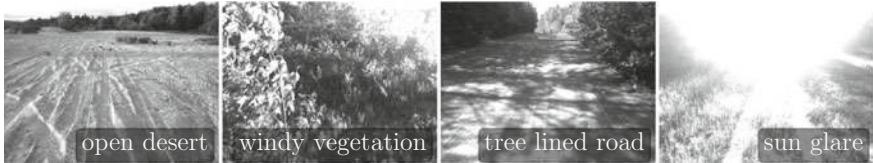


Fig. 13 Example images of areas that remain difficult for vision-based navigation. from left to right: *i* open desert areas with heavy vehicle traffic, *ii* lush vegetation during high winds, *iii* tree-lined corridors with strong shadows, and *iv* sun glare

that were encountered during the field test are highlighted in Fig. 13. Open, desert areas contain few features that persist over time with vehicles, wind, and weather changing the shape of the sand on a daily basis, which causes any stable features to be limited to the horizon. Vegetation-rich areas are typically not a problem for the VT&R 2.0 system, except when high winds are present, which causes the vegetation to rapidly sway back and forth, causing issues for outlier rejection for both localization and VO. Tree-lined corridors cast strong shadows on the road on sunny days. In these conditions, the majority of inlier matches originate from these ephemeral features. With multi-experience systems, this can lead to incorrect state estimates if the majority of inlier matches arise from features that have all moved with the elevation of the sun. The final difficult condition encountered during the field trial is glare from when the sun is low on the horizon, causing images to be oversaturated. It was in these conditions on the sunniest days of the field test where the majority of manual interventions occurred.

Manual Interventions: Figure 14 displays the manual interventions encountered due to localization failures. Trivial interventions were manifestations of a software bug that occurred when the system experienced a VO error, which would stop the robot until the operator drove it forward approximately 0.3 m to trigger a new vertex in the graph. As the nature of this error is somewhat random, the distribution of trivial interventions is nearly uniform across the network. This bug was a minor issue related to the logic of switching between states after a VO error and was resolved after the conclusion of this field test.

Minor interventions were the result of the robot being slightly out of its tracks, which required a small course correction to continue autonomous operation. This was a result of poor localization for short distances, which corresponded to the difficult conditions previously mentioned. Major interventions (yellow lines) occurred when the robot was significantly off course and could not recover. These are a result of extended localization failures without the ability to recover. During the field test the robot experienced interventions in the following areas: i) the tree-lined road when it was sunny and windy (top left), ii) the vegetation-rich area during high winds (mid right), and iii) the open desert area in sunny conditions after heavy vehicle traffic.

A more detailed look at localization performance in this desert area is displayed in Fig. 15. This 50 m path was the most difficult to localize against, running through flat, sandy ground with daily vehicle traffic. After the fifth day of testing, a new path

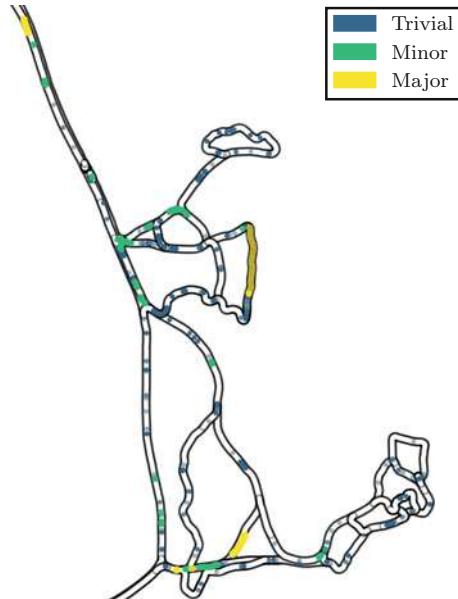


Fig. 14 Manual interventions during the 140 km field trial due to localization issues

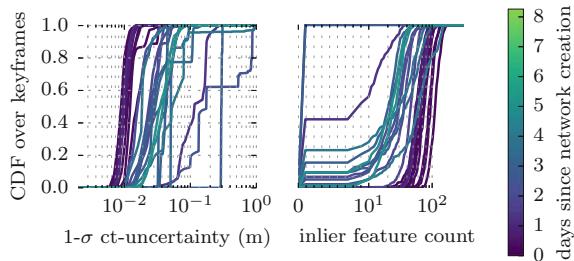


Fig. 15 Localization results a 50 m, open desert section of the network, highlighted in Fig. 13. *left:* CDF of the $1 - \sigma$ localization uncertainty for all autonomous traverses on this path. *right:* CDF of the inlier feature matches for all autonomous traverses on this path. *note: log scale on x-axis*

was rerouted around the trouble area. It can be seen in the figure that the uncertainty is much higher than the vegetation-rich ridge area highlighted in Sect. 5, with $1 - \sigma$ cross-track uncertainty between 5 and 10 cm for the majority of the traverses. for three traverses, the maximum uncertainty was as high as 1 m. This corresponds with the feature inlier count, where 40% of the traverse had less than 10 inlier feature matches, for traverses older than 2 days since map creation.

7 Conclusion/Future-Work

This paper presents the long-term, long-range autonomous path-following system, Visual Teach and Repeat (VT&R) 2.0. We present results from an extended field test demonstrating our system’s ability to safely traverse large-scale networks of paths across appearance change as different as night-vs-day in challenging, unstructured environments. Over an eleven day period our system traversed a 5 km network of paths accumulating over 140 km of autonomous driving with an autonomy rate of 99.6%. However, there remain challenges for vision-based, autonomous path-following systems related to difficult outdoor conditions that must be addressed. Chief amongst them are environments and conditions with ephemeral ground features such as open deserts, tree-lined roads, and high winds.

Future work on the VT&R 2.0 system will be focused on quantifying the scale of our uncertainty, which is calculated at every stage of the localization process, but has not undergone a rigorous evaluation with respect to ground truth to ensure consistency. Once this process is undertaken, the uncertainty can be used to inform the robot to abandon an autonomous traversal *before* it is too far off the path. Because the VT&R 2.0 system is always adding a new experience into the map, in the event of localization failure, it should be possible to use the current experience to backtrack to a safe area and replan or reattempt the traversal.

Acknowledgements This work was supported financially and in-kind by Clearpath Robotics and the Natural Sciences and Engineering Research Council (NSERC) through the NSERC Canadian Field Robotics Network (NCFRN). The authors would like to also extend their deepest thanks to Ethier Sand and Gravel for allowing us to conduct our field test at their site.

References

1. Anderson S., Barfoot, T.D.: Full steam ahead: Exactly sparse gaussian process regression for batch continuous-time trajectory estimation on $\text{se}(3)$. In: IROS (2015)
2. Berczi, L.-P., Barfoot, T.D.: It’s like déjà vu all over again: Learning place-dependent terrain assessment for visual teach and repeat. In: IROS (2016)
3. Berczi, L.-P., Posner, I., Barfoot, T.D.: Learning to assess terrain from human demonstration using an introspective gaussian-process classifier. In: ICRA (2015)
4. Chen, Z., Birchfield, S.T.: Qualitative vision-based path following. IEEE Trans. Robot. **25**(3), 749–754 (2009)
5. Churchill, W.S., Newman, P.: Experience-based navigation for long-term localisation. IJRR **32**(14), 1645–1661 (2013)
6. Clement, L., Kelly, J., Barfoot, T.D.: Robust monocular visual teach and repeat aided by local ground planarity and color-constant imagery. JFR **34**(1), 74–97 (2017)
7. Furgale, P., Barfoot, T.D.: Visual teach and repeat for long-range rover autonomy. JFR **27**(5), 534–560 (2010)
8. Goldberg, S.B., Maimone, M.W., Matthies, L.: Stereo vision and rover navigation software for planetary exploration. In IEEE Aerospace Conference Proceedings (2002)
9. Jackel, L.D., Krotkov, E., Perschbacher, M., Pippine, J., Sullivan, C.: The DARPA LAGR program: Goals, challenges, methodology, and phase I results. JFR **23**(11–12), 945–973 (2006)

10. Krajnk, T., Faigl, J., Vonsek, V., Konar, K., Kulich, M., Peuil, L.: Simple yet stable bearing-only navigation. *JFR* **27**(5), 511–533 (2010)
11. Krüsi, P., Bücheler, B., Pomerleau, F., Schwesinger, U., Siegwart, R., Furgale P.: Lighting-Invariant Adaptive Route Following Using ICP. *JFR* (2014)
12. Linegar, C., Churchill, W., Newman, P.: Work Smart. Recalling relevant experiences for vast-scale but time-constrained localisation. In: ICRA, Not Hard (2015)
13. MacTavish, K., Barfoot, T.D.: Towards hierarchical place recognition for long-term autonomy. In: ICRA Workshop (2014)
14. MacTavish, K., Paton, M., Barfoot, T.D.: Visual triage: a bag-of-words experience selector for long-term visual route following. In: ICRA (2017)
15. McManus, C., Furgale, P., Stenning, B., Barfoot, T.D.: Visual teach and repeat using appearance-based lidar. In: ICRA (2012)
16. Mhlfellner, P., Brki, M., Bosse, M., Derendarz, W., Philippsen, R., Furgale, P.: Summary maps for lifelong visual localization. *JFR* (2015)
17. Ostafew, C., Schoellig, A.P., Barfoot, T.D., Collier, J.: Learning-based nonlinear model predictive control to improve vision-based mobile robot path tracking. *JFR* **33**(1), 133–152 (2016)
18. Paton, M., MacTavish, K., Ostafew, C., Pomerleau, F., Barfoot, T.D.: Expanding the limits of vision-based localization for long-term route following autonomy. *JFR* **34**(1), 98–122 (2017)
19. Paton, M., MacTavish, K., Warren, M., Barfoot, T.D.: Bridging the appearance gap: Multi-experience localization for long-term visual teach & repeat. In: IROS (2016)
20. Sibley, G., Mei, C., Reid, I., Newman, P.: Adaptive relative bundle adjustment. In: RSS (2009)
21. van Es, K., Barfoot, T.D.: Being in two places at once: Smooth visual path following on globally inconsistent pose graphs. In: CRV (2015)

Dynamically Feasible Motion Planning for Micro Air Vehicles Using an Egocylinder

**Anthony T. Fragoso, Cevahir Cigla, Roland Brockers
and Larry H. Matthies**

Abstract Onboard obstacle avoidance is a challenging, yet indispensable component of micro air vehicle (MAV) autonomy. Prior approaches for deliberative motion planning over vehicle dynamics typically rely on 3-D voxel-based world models, which require complex access schemes or extensive memory to manage resolution and maintain an acceptable motion-planning horizon. In this paper, we present a novel, lightweight motion planning method, for micro air vehicles with full configuration flat dynamics, based on perception with stereo vision and a 2.5-D egocylinder obstacle representation. We equip the egocylinder with temporal fusion to enhance obstacle detection and provide a rich, 360° representation of the environment well beyond the visible field-of-regard of a stereo camera pair. The natural pixel parameterization of the egocylinder is used to quickly identify dynamically feasible maneuvers onto radial paths, expressed directly in egocylinder coordinates, that enable finely detailed planning at extreme ranges within milliseconds. We have implemented our obstacle avoidance pipeline with an Asctec Pelican quadcopter, and demonstrate the efficiency of our approach experimentally with a set of challenging field scenarios.

A. T. Fragoso (✉)
California Institute of Technology, Pasadena, CA, USA
e-mail: afragoso@caltech.edu

C. Cigla · R. Brockers · L. H. Matthies
NASA Jet Propulsion Laboratory, California Institute of Technology,
Pasadena, CA, USA
e-mail: cevahircigla@gmail.com

R. Brockers
e-mail: brockers@jpl.nasa.gov

L. H. Matthies
e-mail: lhm@jpl.nasa.gov

1 Introduction

Micro air vehicles possess size, weight, and power (SWaP) constraints on computing resources that require efficient obstacle detection, representation, and avoidance algorithms in order to successfully operate in unknown environments. Accordingly, vision-based techniques using one or more cameras, particularly binocular stereo, have emerged as a popular and successful method for obstacle detection due to their light weight, compactness, and ability to generate dense depth maps of a scene. These advantages have been extended into the internal onboard representation of obstacle environments through the use of egospace data structures [7], such as the disparity images and egocylinders, that are based on the geometry of camera projection, possess a favorable resolution pattern tailored to visual detection, and offer efficient collision-checking and temporal fusion of depth data.

Egospace obstacle representations also simplify reactive obstacle avoidance because their underlying geometry reduces any radially-aligned path to a single pixel, which in turn can be evaluated and selected against depth data using a light-weight pixel scan. Egospace also admits full configuration flat dynamics, of which the quadcopter and commonly-used car and airplane models are examples, and allow all control inputs and vehicle states to be determined uniquely in terms of the trajectories as they appear, in egospace coordinates, in relation to obstacles.

In this paper, we extend reactive egospace trajectory selection for micro air vehicles with negligible dynamics [1] to incorporate full configuration flat dynamics. We present experimental verification on a quadcopter platform in *a priori* unknown environments, using stereo obstacle detection, temporal filtering of depth data, and an egocylindrical obstacle and motion planning representation.

2 Related Work

Traditional approaches for MAV motion planning rely on flatness-based trajectory generation (as in [14]) over a three-dimensional voxel model of the environment. [12] populates a uniform resolution occupancy representation of an unstructured environment, which is used to identify a minimum-jerk trajectory using layered short-range and long-range planners based on convex segmentation followed by a graph search. For a known indoor environment, [17] achieves high-speed flight within a volumetric world model by seeding a polynomial motion planner with the output of a preliminary, dynamics-free RRT* search. Although almost any standard motion-planning method can be used within a voxel representation, uniform grids scale very poorly in size in large outdoor environments and complex access is required to distribute resolution in a more favorable way [9].

Egospace representations, on the other hand, can encode collision-free radial paths using a single pixel and offer an efficient trajectory search space for dynamics-free, reactive motion planning in unstructured environments [1]. Matthies et al.[13]

uses a disparity image representation to efficiently collision-check the segments of a closed loop RRT (CL-RRT) motion planner after a projection, but perform trajectory selection and generation in world coordinates using forward integration of a non-linear plant model at severe computational expense. Although reactive methods in egospace [1] are extremely lightweight, they scale poorly to high speeds because full trajectories are neither known nor collision-checked in advance—even if an instantaneous collision-free action is identified, there is no way to practically verify if it will remain dynamically feasible and trackable in an unknown environment. Deliberative motion planning performed entirely within egospace representations, rather than simply projected into egospace for collision-checking, is introduced and characterized in [7] for robots with configuration flat dynamics—completeness properties are established for the egospace equivalents of motion planners in world coordinates, and motion primitives expressed directly in egospace coordinates extend the advantages of world trajectory libraries to also encompass fast collision-checking and trajectory selection.

Depth data can be obtained from active depth sensors or passive stereo matching. Stereo matching approaches work well both indoors and outdoors and have an adjustable depth range that provides computational flexibility for the design of the perception system. Depth estimation is typically performed on each frame independently, which introduces errors and holes due to environmental factors as well as stereo matching failures. As a result, some degree of temporal fusion is required to propagate reliable data over time and generate a complete and reliable scene representation for later motion planning steps—unfused egospace approaches, such as [1], require additional side-looking cameras to achieve a large field-of-regard, and suffer from flicker and missed-obstacle incidents that result in collisions. Fusion can be performed during the matching step [5, 10, 16], in image space [2, 3, 18] or on 3D voxel representations [4, 8].

In [16] and [10], temporal fusion incorporates temporal data in cost functions during estimation as an extension to spatial aggregation. SLAM techniques [5] also constrain temporal consistency in the estimation framework, where online updates are applied in key frames. In [5], a simple Gaussian model represents depth measurements and limits the search range according to the standard deviation of prior hypotheses. Multi-view filtering is another approach that improves depth maps by removing outliers and compensating holes. In [18], the frames within a specified time window (a fixed number of the most recent frames, for example) are mapped to the most recent frame according to the related depth maps. These hypotheses are merged probabilistically via probability density estimation. Cigla [3] and Cigla et al. [2] use Gaussian Mixture Models (GMM) to represent depth observations in a compact manner, with an online update and propagation approach that decreases computational complexity and memory requirements. These methods tend to produce more reliable depth maps than can be obtained using filter-based fusion approaches.

Three-dimensional models [4, 8] are also commonly used to merge multiple depth map observations and provide temporal fusion. In this regime, depth data is mapped to voxels [4] and accumulated in a surface representation [8] that produces highly accurate maps at the expense of memory usage and computational efficiency.

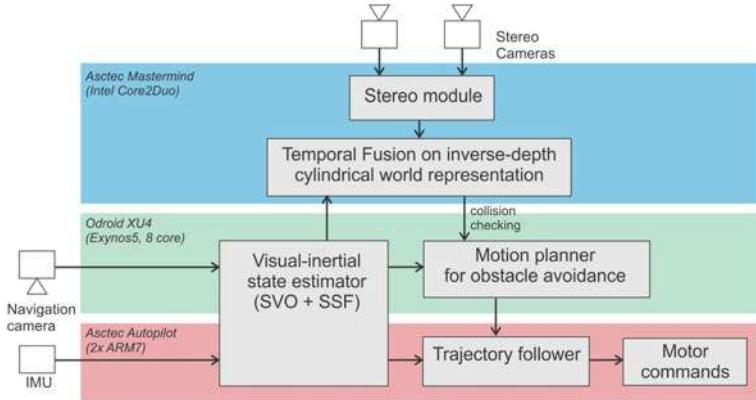


Fig. 1 System architecture for the implementation on an Asctec Pelican quadrotor that is equipped with an Asctec Mastermind and an Odroid XU4 computing board

3 System Overview

The perception, representation, and motion planning modules are implemented within a pipeline architecture onboard an Asctec Pelican quadcopter, with vision and perception tasks performed on an Asctec Mastermind embedded computer and motion planning and state estimation performed on an Odroid XU4 (Fig. 1). After acquiring stereo imagery of a scene using a set of forward-looking cameras, depth data is calculated, temporally filtered, and used to populate an egocylinder structure in which obstacles are artificially expanded in order to abstract the vehicle to a point mass for planning. The expanded egocylinder is then passed to a two-stage motion planner, which first performs a pixel scan to identify a candidate flight path, and then attempts to identify a dynamically feasible, collision-free maneuver onto the candidate path using minimum-jerk paths directly in image space. Once a motion plan is successfully found, a low-level flatness-based controller calculates feedforward inputs and tracks the reference trajectory until a new motion plan is required. Vehicle pose is provided to both the temporal filtering and motion planning modules using visual odometry from a downward-looking camera (SVO [6]) fused with IMU data using the SSF framework [19].

3.1 Perception and Representation

The egocylinder representation of the scene [1] is used to represent the environment for collision-checking, which avoids much of the memory and computation expense used by voxel maps to represent free space. A full 360° field-of-regard is achieved by integrating depth maps forward as the vehicle moves within the environment.

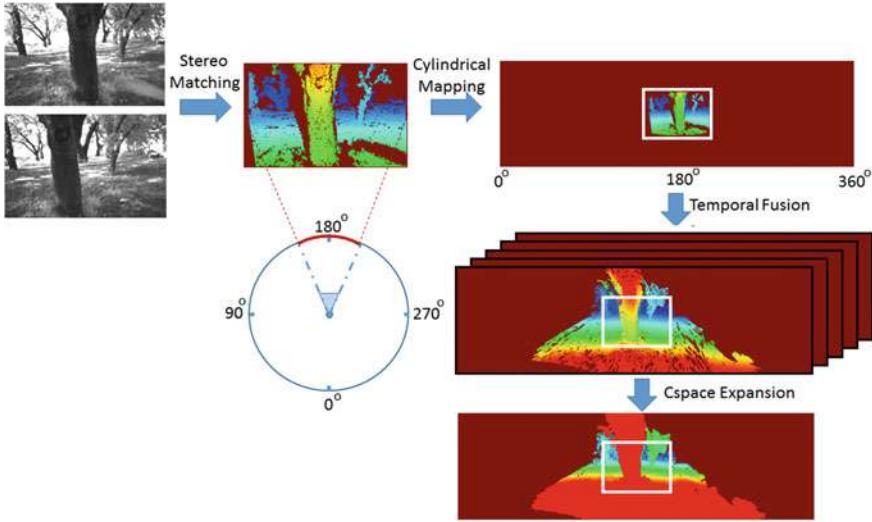


Fig. 2 The visual perception pipeline uses stereo matching to acquire depth data, which is then mapped onto an egocylinder for temporal fusion and C-space expansion directly in egocylinder coordinates. Areas of the egocylinder visible to the stereo cameras are enclosed within a white rectangle for illustration

Visual perception for collision avoidance is based on the approach shown in Fig. 2. The most recent stereo vision-based depth map estimate is first mapped onto an egocylinder representation fixed to the frame of the left camera—the optical center of the left camera maps to the center of the structure, with the forward direction aligned with its viewing axis. The internal representation of the egocylinder is a disparity image which wraps around the cylinder structure, which is called the egocylinder image.

Obstacles are then expanded into configuration space (C-space) [13] by a characteristic radius of the vehicle, directly on the coordinates of the fused egocylinder map, in order to abstract the vehicle to a point mass and simplify path planning. As motion planning proceeds and the vehicle continues to move through the environment, new depth data is observed in the central area of the egocylinder corresponding to the field-of-view of the stereo cameras.

We use GMM-based fusion [2, 3] to efficiently merge the current depth map with past observations directly in image space. In this approach, each pixel is modeled by a mixture of Gaussian distributions, in pixel coordinates (u, v) and disparity d , and propagated to the recent depth map using vehicle pose estimates. The final depth map is obtained by constraining both the standard deviation of the disparity models and an observation count to remove flicker and rely on frequently observed depth values. Cigla et al. [2] demonstrates that GMM-based fusion improves depth quality by removing large depth errors such as false obstacles, which are typically due to errors in stereo matching, and by assigning reliable depth values for empty

pixels in the recent depth map. Temporal fusion in this region of the egocylinder mitigates sensing failures, such as incorrect depth estimates or empty pixels, by constraining the temporally accumulated data. Regions of the egocylinder not visible to the stereo sensors do not receive new depth data, and are instead updated by transferring temporally accumulated models subject to a forgetting factor at each update. As the vehicle moves around and encounters obstacles at a variety of viewing angles, a more complete representation of the scene emerges as temporal data is fused into the structure.

3.2 Motion Planning

The first stage of the motion planning module replicates the approach of [1] in which collision-free, yet dynamically infeasible radial paths are represented by their pixel locations and identified using a scan over the expanded egocylinder structure. During this scan, the distance to obstacles at each pixel is compared to a predetermined planning horizon distance h , which is chosen to maintain a safe time-to-contact towards obstacles at the desired speed and can take values up to the maximum reliable sensor range depending on the expected complexity of the environment. The collision-free pixel that is closest to that of the destination is selected as a candidate path (Fig. 3), and if no safe candidate path is found the process is repeated with a lower desired speed and shorter planning horizon.

In the second stage, a trajectory generation module attempts to identify a feasible maneuver that can smoothly merge the vehicle onto the infeasible candidate path at the desired final velocity V , which greatly narrows the set of admissible trajectories that must be calculated and collision-checked. The straight-line segment that follows is known to be collision-free from the first stage and need not be checked again, and

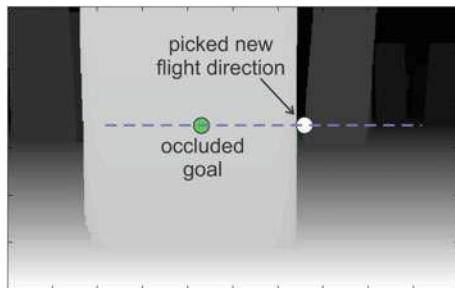


Fig. 3 The first stage of the planner identifies collision-free flight paths by checking the pixels of the egocylinder against a planning horizon chosen to guarantee a minimum time-to-contact. A goal is projected into the egocylinder (green, occluded) and the nearest collision-free pixel (white) is selected as a candidate path and passed to the trajectory generator. Here, brighter pixels are closer to the camera

the required acceleration, velocity, and position are automatically specified at the beginning and end of the turning maneuver (for position, up to an undetermined range):

$$\begin{aligned}\ddot{\mathbf{x}}(0) &= \mathbf{a}_0, & \dot{\mathbf{x}}(0) &= \mathbf{v}_0 & \mathbf{x}(0) &= \mathbf{x}_0, \\ \ddot{\mathbf{x}}(t_f) &= 0, & \dot{\mathbf{x}}(t_f) &= V\hat{\lambda} & \mathbf{x}(t_f) &= (\lambda h)\hat{\lambda}.\end{aligned}$$

Here, the final flight direction $\hat{\lambda}$ is chosen in the first planning stage, t_f is the total maneuver time, and $\lambda \in (0, 1]$ is a free parameter that modulates the distance to the end of the maneuver.

The set of possible trajectories is further narrowed by insisting on a minimum jerk maneuver, which can be shown to consist of fifth degree polynomials $(x(t), y(t), z(t)) = \sum_{i=0}^5 \mathbf{a}_i t^i$ using a straightforward application of the Pontryagin minimum principle. Once the endpoints are prescribed, the maneuver has only two remaining free parameters, t_f and λ , that can either be modulated independently or constrained to each other to produce a one-parameter family of admissible trajectories—for example, the total time can be chosen to maintain an average vector velocity associated with the candidate path (fixed speed, fixed direction) such that $t_f = (\lambda h)/V$. Minimum-jerk polynomials can be shown by direct substitution to satisfy the differentially flat quadcopter plant model of [11],

$$\begin{aligned}\ddot{x} &= (\sin \phi \cos \psi - \cos \phi \sin \theta \sin \psi) \frac{u_{\text{collective}}}{m}, \\ \ddot{y} &= (-\sin \phi \sin \psi - \cos \phi \sin \theta \cos \psi) \frac{u_{\text{collective}}}{m}, \\ \ddot{z} &= -g + (\cos \phi \cos \theta) \frac{u_{\text{collective}}}{m}, \\ \dot{\phi} &= u_{\text{roll}}, \\ \dot{\theta} &= u_{\text{pitch}}, \\ \dot{\psi} &= u_{\text{yaw}},\end{aligned}$$

where (ϕ, θ, ψ) are roll-pitch-yaw Euler angles and $u_{\text{collective}}$ is the collective motor thrust. The trajectories $(x(t), y(t), z(t))$ correspond to unique control inputs, subject to actuator saturation and a yaw angle trajectory that can be chosen independently of the translation of the vehicle, that are determined algebraically using the associated flatness relations.

The coefficients of the minimum-jerk polynomials are determined symbolically, in advance, as a solution of a small linear system of equations in terms of the initial and final vehicle states and the search parameter λ . This solution step dramatically reduces the amount of computation that needs to be performed onboard and allows for a large number of potential trajectories to be considered during a planning cycle (see Sect. 4 for runtime statistics). Evaluation for control saturation and collisions is enhanced by a simultaneous dual representation of the trajectories (Fig. 4) in world coordinates and egocylinder coordinates, which both have a simple closed form and are determined by the single search parameter—the world coordinate trajectories

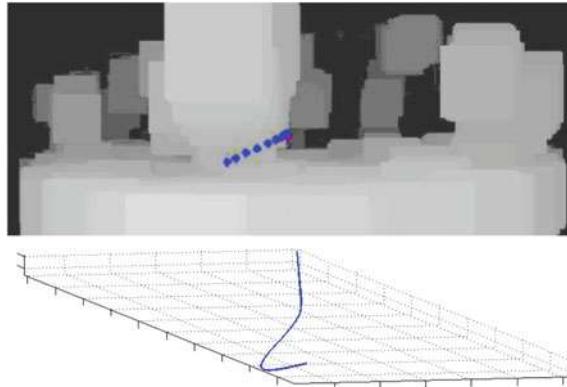


Fig. 4 Closed-form trajectories are simultaneously maintained in egocylinder coordinates for collision-checking and world coordinates for control. Here, a trajectory (top, blue dots proceeding towards the right) has been selected to avoid a tree (brighter pixels are closer) using a pixel scan and collision-check in egocylinder coordinates. The trajectory is tracked in world coordinates (blue line, bottom) by the flatness-based controller

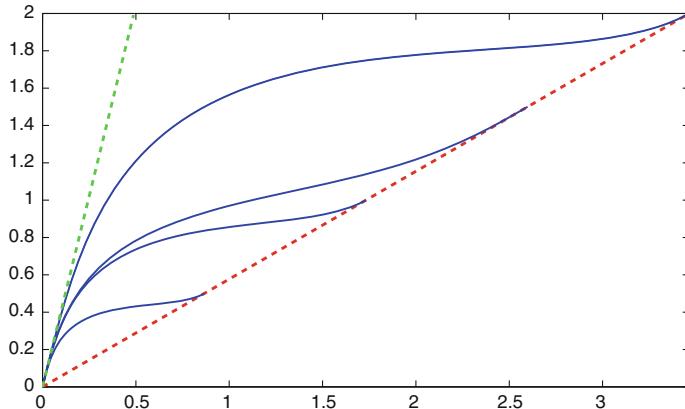


Fig. 5 The trajectory generator, shown in 2-D for clarity, attempts to merge the vehicle smoothly from the green heading (left, dashed) to the red heading (right, dashed) using minimum-jerk polynomials (blue) that are parametrized by the distance covered before merging. By decreasing the length of the maneuver, the trajectory generator can provide a more aggressive turn if a collision is detected

can be easily checked for saturation and converted into control inputs, while the egocylinder coordinates can be immediately collision-checked using a pixel-by-pixel comparison against the egocylinder data structure.

The trajectory search itself begins by attempting to merge the vehicle onto the candidate radial path at a distance corresponding to the planning horizon (Fig. 5). The proposed maneuver is collision-checked, evaluated against the control constraints, and returned if successful—otherwise, the distance to the end of the maneuver is

decreased along with the total time until either a valid trajectory is identified or the controls saturate, in which case either the total time is increased independently or a different candidate radial path is attempted. Once the trajectory is available, a low level controller tracks the reference using a standard two degree-of-freedom feed-forward design [15], with nested position/velocity and attitude loops corresponding to the slow and fast dynamics of the vehicle model.

Our receding-horizon approach does differ from an RRT-type formulation, like that of [17], in that it lacks a completeness guarantee and can become stuck in dead ends. For obstacle avoidance in unstructured environments where frequent replanning is required, however, RRT-type methods can bottleneck the pipeline [13] and limit flight speeds. Receding-horizon formulations such as ours, however, do share some degree of deliberative action with complete approaches and offer forward collision-checking unavailable to reactive approaches such as [1].

4 Results

The visual perception pipeline is implemented on the Asctec Mastermind embedded computer equipped with a 1.86 GHz Intel Core2Duo processor. The forward-looking stereo cameras are installed with a baseline of 25 cm, and frame-wise stereo disparity maps (376×240) are calculated by block matching with a search range of 100 pixels. The resolution of the egocylinder image is 660×200 . The full pipeline maintains a 10 Hz update rate using both cores of the processor, with computation times for each step in the visual perception pipeline given in Table 1.

Typical results for temporal fusion on the egocylinder are illustrated in Fig. 6. The left stereo image, unfused disparity map, and the corresponding egocylinder images are shown sequentially as the vehicle approaches an obstacle. Temporal fusion yields a more complete scene representation compared to the unfused disparity maps—known regions that would be otherwise discarded at each frame are instead retained in the egocylinder. The propagation process is particularly important for obstacles at close range, which eventually become invisible to stereo matching due to a limited search range and would otherwise present a missed-detection hazard (Fig. 7).

Table 1 Onboard computation times for each step in the visual perception pipeline

Step	Time (ms)
Stereo matching	100
Cylindrical mapping	14.4
Temporal fusion	52.6
C-space expansion	4.5

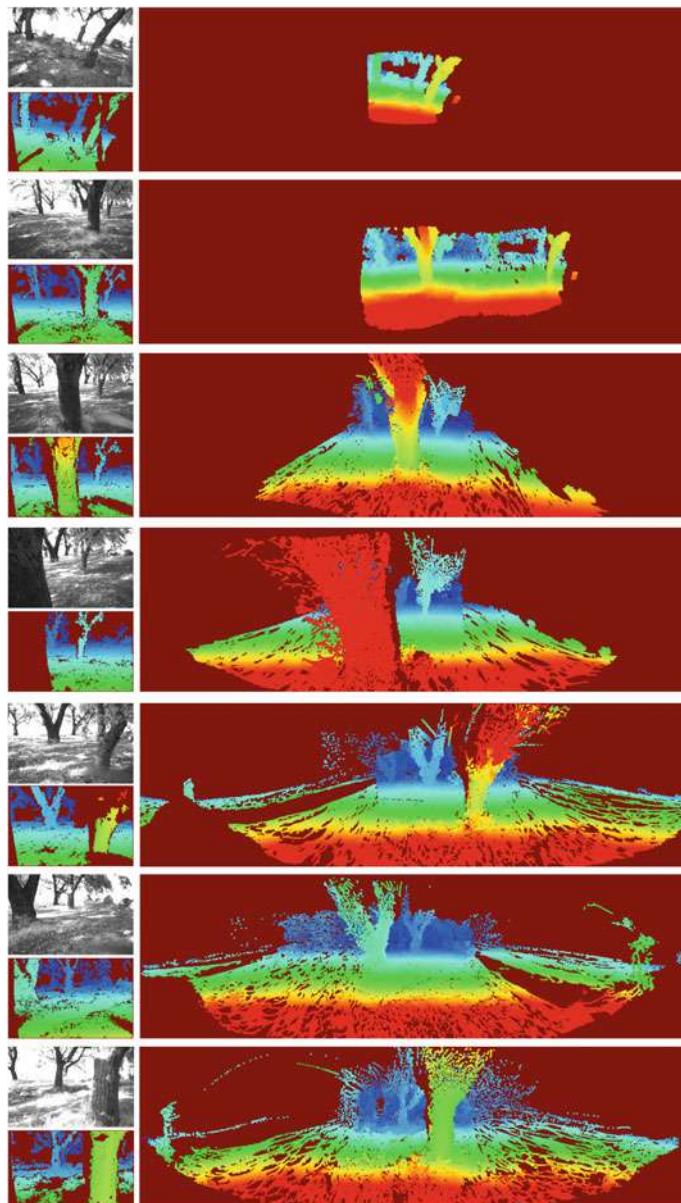


Fig. 6 The perception pipeline was tested in a forested environment. In the first column are raw images (grayscale) from the left stereo camera and the unfused disparity map, and in the second are full egocylinder maps with temporal fusion. Here, warmer colors are closer, cooler colors are farther, and the maroon background represents no data. In the fourth row, temporal fusion prevents a collision at close range (first column) by propagating obstacle data forward from an earlier detection (second column)

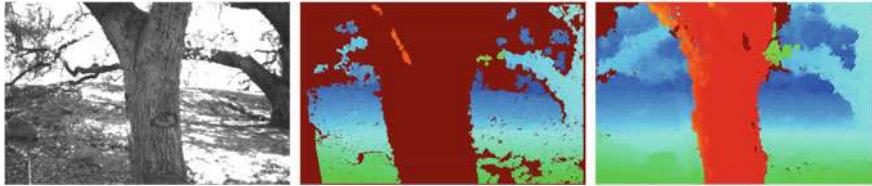


Fig. 7 While the maximum disparity calculated by the stereo algorithm is too low to resolve the nearby obstacle, temporal fusion propagates the previously observed obstacle, which is then used in the motion planner. Left: left rectified image, Middle: stereo disparity map, Right: temporally fused disparity map

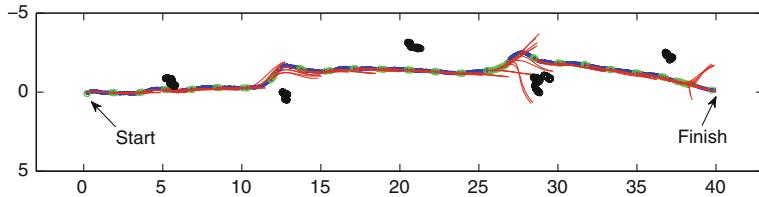


Fig. 8 While navigating through a forested environment (tree trunks at 2 m height shown as black point clouds), the vehicle took evasive maneuvers around three trees and successfully reached a predetermined destination 40 m away. The path (blue, vehicle travels from left to right) consists of segments of dynamically feasible trajectory segments (red arcs) calculated using the two-step procedure of Sect. 3 and followed in a receding-horizon fashion. Axis scale is in meters

Motion planning experiments were conducted at a number of challenging sites, including a 40 m course through a forested area, a small dead-end alcove in a built-up area, and an open, obstacle-free wash with distant buildings.

While maintaining a speed of 1 m/s, the vehicle encountered and successfully avoided three trees in the forest using the minimum-jerk trajectories of Sect. 3 executed in a receding-horizon fashion (Fig. 8). This environment required frequent replanning as new obstacles became visible, which was performed efficiently with average runtimes on the Odroid XU4 listed in Table 2.

The egocylinder also proved highly useful in preventing the vehicle from becoming stuck in confined areas when paired with temporal propagation of depth data. After exploring a small alcove and discovering a dead end (Fig. 9), the motion planner was able to extricate the vehicle towards an open area to its left that was invisible to the stereo pair, but already stored within the egocylinder structure and available to the motion planner.

Table 2 Onboard computation times for each step in the motion planning pipeline

Step	Time (ms)
Egocylinder scan	4.0 ^a
Trajectory generation	0.02
Collision-checking	0.2

^aFor a complete replan (motion planner can reuse old targets)

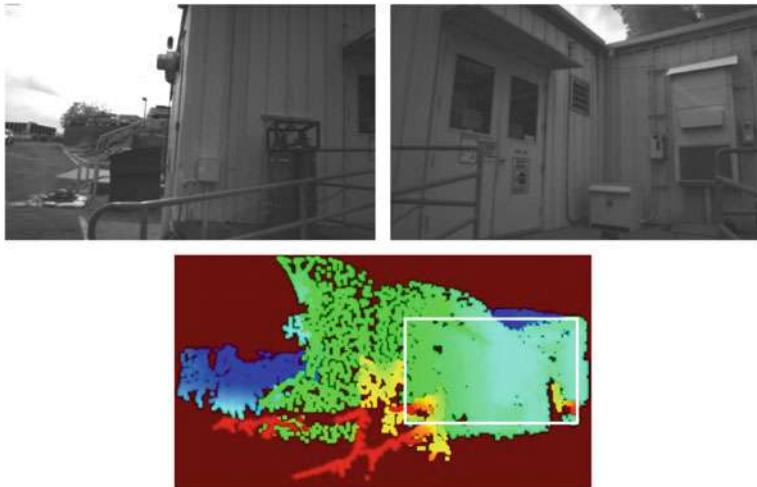


Fig. 9 When combined with temporal fusion, the egocylinder facilitates exploration of confined areas by propagating knowledge of open areas when they become invisible to the stereo camera. Here, the vehicle has reached a dead end within an alcove (top, raw images shown), but continues to represent an exit to its left within the egocylinder (blue area, bottom left) beyond the visible field-of-regard of unfused stereo (white rectangle). An escape route is also available by climbing over the building, but was excluded by a mission-level altitude ceiling

By identifying a restricted trajectory search space and exploiting the natural resolution pattern of the egocylinder, the motion planning module was also able to efficiently generate detailed, dynamically feasible trajectories at ranges near the limits of reliable stereo detection. When presented with a destination waypoint inside a parking garage over 40 m away, within 5 ms the motion planner identified a 5 m \times 2 m opening into the garage as the best path towards the target and calculated a collision-free and dynamically feasible path into the building (Fig. 10).

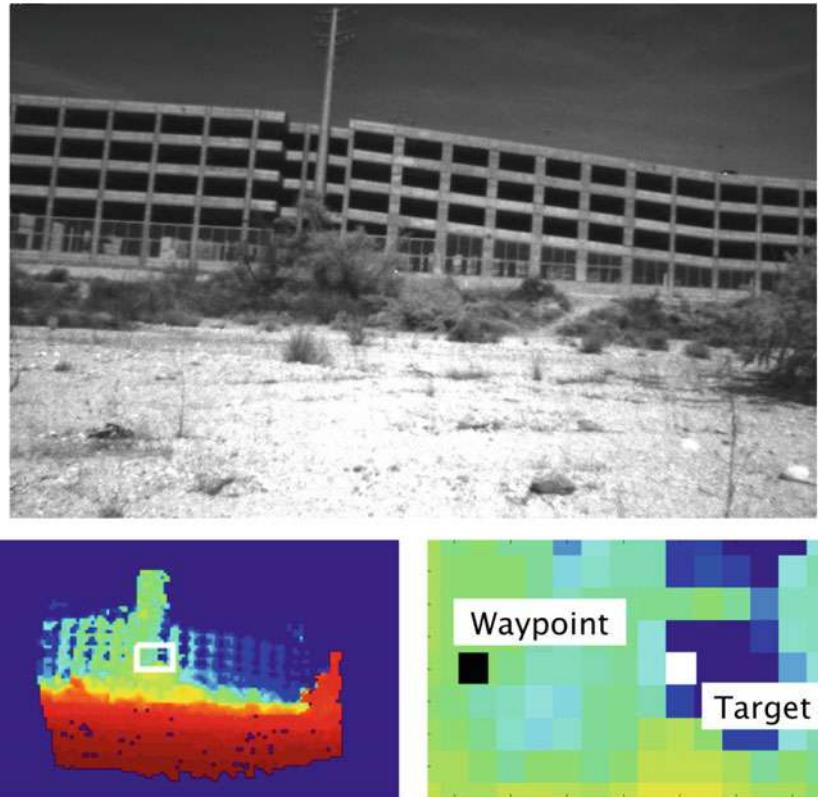


Fig. 10 Egospace representations are particularly well-suited for producing finely detailed trajectories at extreme ranges. Here, the motion planner has been assigned a waypoint behind a pillar inside a parking garage 45 m away (raw image, top). After generating and expanding an egocylinder representation (original, bottom left, zoomed for clarity, right), the motion planner projects the waypoint (black square, bottom right) into the image and identifies the closest collision-free target (white square, bottom right) as a candidate flight path for the trajectory generator

5 Conclusion

We have demonstrated an obstacle avoidance pipeline over the full dynamics of a micro air vehicle using a temporally fused egocylinder representation. In addition to being simple to access and compact to store, the egocylinder representation admits an extremely lightweight trajectory selection procedure, based on a parametrization of flight paths by pixels and differential flatness, that can quickly generate dynamically feasible turn maneuvers towards targets at the range limit of stereo sensing. This advantage is made more reliable and flexible with temporal fusion of obstacle data within the egocylinder geometry, which improves the accuracy of sensed data and

accumulates a 360° representation with a single set of stereo cameras that is useful for navigation in confined and cluttered environments.

Our perception framework also addresses a number of weaknesses of stereo vision, primarily due to the limited and fixed baseline of MAV stereo applications, and significantly increases its power as an obstacle detection tool. The resolution pattern of the egocylinder, which decreases favorably with distance without additional overhead, allows stereo data to inform motion planning and remain useful even at long ranges where it is less accurate. Detection errors at close ranges, which eventually become inevitable for raw stereo due to a limited field of view and disparity search, are prevented by propagation and storage of sensed data with real-time temporal fusion.

Acknowledgements This work was funded by the Army Research Laboratory under the Micro Autonomous Systems & Technology Collaborative Technology Alliance program (MAST-CTA). JPL contributions were carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

1. Brockers, R., et al.: Vision-based obstacle avoidance for micro air vehicles using an egocylindrical depth map. In: *ISER* (2016)
2. Cigla, C., Brockers, R., Matthies, L.: Image-based visual perception and representation for collision avoidance. In: *IEEE International Conference on Computer Vision and Pattern Recognition, Embedded Vision Workshop* (2017)
3. Cigla, C., et al.: Gaussian mixture models for temporal depth fusion. In: *IEEE Winter Conference on Applications of Computer, Vision* (2017)
4. Dryanovski, W., et al.: Multi-volume occupancy grids: an efficient probabilistic 3d mapping model for micro aerial vehicles. In: *IROS* (2010)
5. Engel, J., et al.: Semi-dense visual odometry for a monocular camera. In: *2013 IEEE International Conference on Computer Vision*, pp. 1449–1456. IEEE (2013)
6. Forster, C., Pizzoli, M., Svo, S.D.: Fast semi-direct monocular visual odometry. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2014)
7. Fragoso, A., Matthies, L., Murray, R.: A fast motion planning representation for configuration flat robots with applications to micro air vehicles. In: *ACC* (2017)
8. Hane, C., et al.: Stereo depth map fusion for robot navigation. In: *IROS* (2011)
9. Hornung, A., et al.: Octomap: an efficient probabilistic 3D mapping framework based on octrees. *Auton. Robot.* **34**(3) (2013)
10. Hosni, A., et al.: Temporally consistent disparity and optical flow via efficient spatio-temporal filtering. In *Pacific Rim Symposium on Image and Video Technology* (2011)
11. How, J., et al.: Real-time indoor autonomous vehicle test environment. *IEEE Control Syst. Mag.* **28**(2) (2008)
12. Liu, S., et al.: High speed navigation for quadrotors with limited onboard sensing. In: *ICRA* (2016)
13. Matthies, L., et al.: Stereo vision-based obstacle avoidance for micro air vehicles using disparity space. In *International Conference on Robotics and Automation (ICRA)* (2014)
14. Mellinger, D., et al.: Minimum snap trajectory generation and control for quadrotors. In: *ICRA* (2011)
15. Powers, C., et al.: Quadrotor kinematics and dynamics. In: *Handbook of Unmanned Aerial Vehicles*. Springer, Netherlands (2015)

16. Richardt, C., et al.: Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In: *European Conference on Computer Vision* (2010)
17. Richter, C., et al.: Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments. In: *ISRR* (2013)
18. Unger, C., et al.: Probabilistic disparity fusion for real-time motion stereo. *Mach. Vis. Appl.* **25** (2011)
19. Weiss, S., et al.: Monocular vision for long-term micro aerial vehicle state estimation: a compendium. *J. Field Roboti.* **30**(5) (2013)

Informed Asymptotically Near-Optimal Planning for Field Robots with Dynamics

Zakary Littlefield and Kostas E. Bekris

Abstract Recent progress in sampling-based planning has provided performance guarantees in terms of optimizing trajectory cost even in the presence of significant dynamics. The STABLE_SPARSE_RRT (SST) algorithm has these desirable path quality properties and achieves computational efficiency by maintaining a sparse set of state-space samples. The current paper focuses on field robotics, where workspace information can be used to effectively guide the search process of a planner. In particular, the computational performance of SST is improved by utilizing appropriate heuristics. The workspace information guides the exploration process of the planner and focuses it on the useful subset of the state space. The resulting Informed-SST is evaluated in scenarios involving either ground vehicles or quadrotors. This includes testing for a physically-simulated vehicle over uneven terrain, which is a computationally expensive planning problem.

1 Introduction

This work focuses on the performance and properties of motion planning algorithms in the context of field robots that exhibit interesting and challenging dynamics. Examples include unmanned aerial vehicles, which are becoming increasingly available and can be used for aerial reconnaissance [8], or wheeled rovers for exploration applications [4, 13], or aquatic robots for observing and monitoring large bodies of water [12, 26].

The objective is anytime performance, which means that the planner should be able to quickly generate trajectories that drive the robot to the goal. Then, given additional computational budget, the solution trajectory can be refined or replaced by a better one. Many factors can hinder a motion planning algorithm's performance, such as complex environments and challenging dynamics. This paper builds on top

Z. Littlefield · K. E. Bekris (✉)

Rutgers, The State University of New Jersey, New Brunswick, USA
e-mail: kostas.bekris@cs.rutgers.edu

Z. Littlefield

e-mail: zakary.littlefield@rutgers.edu

of a framework that can provide formal guarantees for systems with dynamics and improves its practical performance toward stronger anytime properties. This new planner is able to solve less complex challenges quickly, but is also able to provide solutions in more complex cases. In essence, the algorithm tries to balance the classic “exploration vs exploitation” trade-off during planning while still maintaining desirable formal guarantees.

Related Work: A straightforward path-finding framework relies on performing a discretization of the environment into a grid. Then, an A* search can be performed on the grid, thereby providing waypoints for the robot to follow. There are also methods that aim to smooth the paths via interpolation or visibility checks, and have been used for several successful navigation tasks [10, 25].

In order to more accurately take a robot’s dynamics into account, a discrete structure of dynamically feasible trajectories can be built. By choosing these trajectories carefully, the end states of these trajectories can coincide, creating a lattice structure that covers the state space [9, 22, 29, 32]. Then path finding can be done with search-based methods like A*. As long as the branching factor in this lattice is finite and the lattice is connected, the search on this lattice will terminate with the optimal sequence of motions.

This type of approach faces issues when the effects of controls are unknown *a priori*, i.e. when dynamics have significant effect. If many different control inputs cannot arrive at the same state, the search space becomes intractably large. This is also a problem when considering a continuous control space where any of the admissible controls may be selected from any state. When the number of actions is infinite due to a continuous action space, completeness and optimality are sacrificed for practical performance [5, 7, 14, 23].

Once a feasible path is found, it can then be used as input to a trajectory optimization procedure [30, 33]. Such methods work well for optimizing trajectories locally, and can be applied as post processing on any motion planning algorithm. It is important that the input trajectory is as close to optimal as possible to aid convergence.

A family of algorithms that aims to handle continuous action spaces and high-dimensional state spaces are sampling-based planners [17, 19]. These algorithms trade traditional completeness for probabilistic completeness and aim for a comprehensive exploration of the underlying state space. Recently, research efforts have focused on the conditions under which these methods converge to an optimal path [16] including for systems with dynamics [20, 21] and under uncertainty [24]. The latter work on the SST approach is the basis for this paper. The SST planner is able to provide asymptotic near-optimality properties for systems with dynamics, as well as for physically-simulated robots. Trajectories are generated in an anytime fashion, where an initial suboptimal solution is found and improved given more computation. It becomes more difficult to improve solutions over time, however, when computational effort is wasted on parts of the state space that will never provide better solutions. Heuristic and anytime search-based methods handle this problem well and focus computation only on viable parts of the search space.

There are prior methods that tried to bring insights from search-based methods into sampling-based planning [2, 3, 11, 31], while others consider what sampling-based insights can do for search-based methods [27, 28]. The integration of these two methodologies is generally more efficient than using one alone. Using heuristic information focuses computation on useful areas in the state space that will lead to good quality trajectories. This can be seen as contradictory to the “exploration” property that sampling-based planners traditionally focus on but guiding this exploration to the useful parts of the state space provides practical benefits.

Contribution: This paper focuses on an effective integration of heuristic search principles with sampling-based kinodynamic planning. There are multiple changes to the underlying SST method that after experimentation with simulated models of field robots with dynamics have been shown to be effective: (a) the node selection process of the algorithm is deterministic—more similar to that of A*—instead of uniform sampling in state space, i.e., RRT-like, (b) a workspace-based heuristic is used to guide the exploration and node pruning routines of SST, (c) the method can formally utilize desirable prespecified maneuvers, while still allowing for the consideration of random controls that allow for probabilistic completeness, (d) multiple controls are considered at each iteration of the algorithm and the best maneuver according to the heuristic information is utilized, (e) furthermore, a branch-and-bound process focuses the search method once an initial solution has been found. These adaptations result in improved success rate for finding solutions within given time limits and improved path quality over SST, the basic RRT, or a Randomized A* algorithm given the same amount of computation.

2 Problem Setup and Background

This paper considers systems with dynamics that follow differential equations of the form:

$$\dot{x}(t) = f(x(t), u(t)), \quad x(t) \in \mathbb{X}, \quad u(t) \in \mathbb{U}, \quad (1)$$

where $x(t)$ is an element of the state space \mathbb{X} , and $u(t)$ is in the control space \mathbb{U} . Some systems have their dynamics in closed-form, but when using a physics engine, the dynamics are numerically computed.

A trajectory π is a function $\pi(t) \in \Pi : [0, T_\pi] \rightarrow \mathbb{X}$, where T_π is the time duration of this trajectory. A feasible trajectory is one where all states $x \in \pi(t)$ are collision-free, i.e. belong in the free space $\mathbb{X}_f \subset \mathbb{X}$. With a slight abuse of notation, π is implied to always include its time parameter implicitly, and a trajectory generated between state x and x' can also be denoted as $\overline{x \rightarrow x'}$.

Each state $x \in \mathbb{X}$ is assigned a state cost $g : \mathbb{X} \rightarrow \mathbb{R}^+$, which encodes the difficulty of moving through that part of the state space. One common state cost function involves the use of a discrete cost map, where different areas of the state space are

more costly than others. This can encode difficult to traverse areas, making them last resorts for a motion planner. Then, the cost of a trajectory π is denoted by

$$c(\pi) = \int_0^{T_\pi} g(\pi(t))dt.$$

Common choices for a cost function include time to execute the trajectory, amount of energy expended, or simply distance traveled. In general, as long as the cost function is strictly monotonic and additive over a trajectory, SST can provide path quality guarantees.

The overall goal for a motion planner is to find trajectories, which satisfy Eq. 1, start from an initial state x_o and bring the system to a goal region \mathbb{X}_G . The secondary objective is to provide anytime performance, i.e., find an initial solution quickly and then improve the quality of this solution over time given additional computation, getting as close to the optimal solution cost as possible. The optimal cost $c^*(\pi)$ is

$$c^*(\pi) = \{\min c(\pi) : \pi \in \Pi, \pi(0) = x_o, \pi(T_\pi) \in \mathbb{X}_G, \pi(t) \in \mathbb{X}_f, \forall t \in [0, T_\pi]\}$$

If a trajectory between two states is not available, and a lower bound on the cost between those two states is needed, a heuristic can be defined:

$$h : \mathbb{X} \rightarrow \mathbb{R}, s.t. h(x) \leq c^*(\pi), \pi \in \Pi, \pi(0) = x, \pi(T_\pi) \in \mathbb{X}_G \quad (2)$$

A trivial heuristic function $h(\cdot) = 0$ satisfies this relationship, but provides no useful information about the expected cost between two states. If possible, a heuristic should be as close to the cost of the optimal trajectory between the two states. Another consideration for a heuristic function is that it is in the same unit as the cost function. If energy or time is the cost function, the heuristic function needs to account for this and create a lower bound on these costs. A properly constructed h function can help guide the motion planner toward useful areas to search.

3 Algorithm

SST framework: The algorithm that this work builds upon is outlined in Algorithm 1 [21]. It includes a series of additions to the RRT framework so as to achieve asymptotic near-optimality guarantees in terms of solution trajectory quality for systems with dynamics, while also requiring low computational resources in terms of space and time. There are three basic additions to RRT that SST employs. They correspond to node selection, random propagation, and pruning:

- **Node Selection:** In RRT, nodes are selected via a nearest neighbor call around a randomly sampled state. This type of selection biases the selection of nodes toward those on the frontier of the tree. SST employs a `Best_Near` procedure that selects a node within a radius δ_{BN} of a random sample that has the lowest path

Algorithm 1: SST($\mathbb{X}, \mathbb{U}, x_o, T_{prop}, N, \delta_{BN}, \delta_w$)

```

1  $\mathbb{V}_{active} \leftarrow \{x_o\}$ ,  $\mathbb{V}_{inactive} \leftarrow \emptyset$ ;
2  $G = \{V \leftarrow (\mathbb{V}_{active} \cup \mathbb{V}_{inactive}), \mathbb{E} \leftarrow \emptyset\}$ ;
3  $w_0 \leftarrow x_o$ ,  $w_0.rep = x_o$ ,  $W \leftarrow \{w_0\}$ ;
4 for  $N$  iterations do
5    $x_{selected} \leftarrow \text{Best\_Near}(\mathbb{X}, \mathbb{V}_{active}, \delta_{BN})$ ;
6    $x_{new} \leftarrow \text{Random\_Prop}(x_{selected}, \mathbb{U}, T_{prop})$ ;
7   if CollisionFree( $x_{selected} \rightarrow x_{new}$ ) then
8     if Is_Locally_Best( $x_{new}$ ,  $W$ ,  $\delta_w$ ) then
9        $\mathbb{V}_{active} \leftarrow \mathbb{V}_{active} \cup \{x_{new}\}$ ;
10       $\mathbb{E} \leftarrow \mathbb{E} \cup \{x_{selected} \rightarrow x_{new}\}$ ;
11      Prune( $x_{new}$ ,  $\mathbb{V}_{active}$ ,  $\mathbb{V}_{inactive}$ ,  $\mathbb{E}$ );
12 return  $G$ ;

```

cost from x_o , the root of the tree. This selection procedure biases toward nodes that are likely to have good path quality.

- **Random Propagation:** Implementations of RRT perform edge extensions in two ways: going toward the randomly sampled state from the selection process, or random propagation. It was recently shown that in some cases, the propagation toward the random sample is incomplete [18]. In addition to this, randomly sampled controls play an integral part for the asymptotic near-optimality proof for SST.
- **Pruning:** Especially when searching high-dimensional spaces, the number of nodes stored in an RRT can become quite large. By maintaining a witness set that “claims” a radial region around states, the number of nodes can be minimized. This also makes algorithm iterations much faster than RRT. An illustration of these witness nodes with respect to the tree of states is shown in Fig. 1.

These three modifications make SST asymptotically near-optimal, details of the proof of which can be found in the original paper [21].

Informed Extension: Algorithm 2 outlines the changes that make SST into an informed algorithm, iSST. Introduction of the heuristic function, h , allows for more effective selection, more informed edge extension, and more accurate pruning capabilities relative to SST. Each of these changes are explained in the coming sections in more detail. In the abstract, iSST borrows insights from heuristic search methods for graphs and trees, and applies them to a continuous motion planning problem.

As explained in Sect. 2, the construction of the heuristic function, h , should be a lower bound estimate of the optimal cost to the goal. For many applications, the distance between the two is used to estimate this cost. This function, however, could lead an informed search method toward environmental obstacles, subsequently hitting a local minimum. This makes trajectories difficult to find and negates much of the benefit of a heuristic. To account for this, a simpler motion planning problem (using a roadmap method [16]) is solved that accounts for the geometric aspects of the dynamic motion planning problem. A geometric path tree built from this roadmap

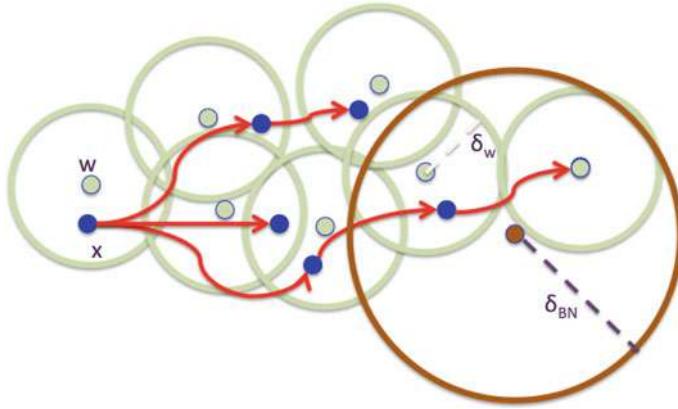


Fig. 1 Illustration of the main operations in the SST framework. The witness set W is a set of unconnected nodes, which indicate areas that have been visited. During the node selection process, a radial region around a uniformly sampled point is examined for low cost nodes

Algorithm 2: iSST($\mathbb{X}, \mathbb{U}, x_o, T_{prop}, N, \delta_w, h$)

```

1  $\mathbb{V}_{active} \leftarrow \{x_o\}$ ,  $\mathbb{V}_{inactive} \leftarrow \emptyset$ ;
2  $G = \{V \leftarrow (\mathbb{V}_{active} \cup \mathbb{V}_{inactive}), \mathbb{E} \leftarrow \emptyset\}$ ;
3  $w_0 \leftarrow x_o$ ,  $w_0.rep = x_o$ ,  $W \leftarrow \{w_0\}$ ;
4  $\mathbb{O} \leftarrow \{x_o\}$ ,  $\mathbb{O}' \leftarrow \emptyset$ ;
5 for  $N$  iterations do
6    $x_{selected} \leftarrow \text{SearchSelection}(\mathbb{O}, \mathbb{O}')$ ;
7    $x_{curr} \leftarrow x_{selected}$ ;
8   while  $x_{curr} \neq \text{NULL}$  do
9      $x_{new} \leftarrow \text{Blossom}(x_{curr}, h)$ ;
10     $x_{curr}.priority \leftarrow x_{curr}.priority + 1$ ;
11    if  $x_{curr} \neq x_{selected}$  then
12       $\mathbb{O} \leftarrow \mathbb{O} \cup \{x_{curr}\}$ ;
13    if  $\text{CollisionFree}(x_{curr} \rightarrow x_{new}) \wedge \text{BranchAndBound}(x_{new})$  then
14      if  $\text{Is\_Node\_Locally\_the\_Best\_SST}(x_{new}, W, \delta_w)$  then
15         $\mathbb{V}_{active} \leftarrow \mathbb{V}_{active} \cup \{x_{new}\}$ ;
16         $\mathbb{E} \leftarrow \mathbb{E} \cup \{x_{curr} \rightarrow x_{new}\}$ ;
17         $\text{Prune\_Dominated\_Nodes\_SST}(x_{new}, \mathbb{V}_{active}, \mathbb{V}_{inactive}, \mathbb{E})$ ;
18         $x_{curr} \leftarrow x_{new}$ ;
19      continue;
20     $x_{curr} \leftarrow \text{NULL}$ ;
21    $\mathbb{O}' \leftarrow \mathbb{O}' \cup \{x_{selected}\}$ ;

```

is then used to estimate the true optimal cost to a goal region, \mathbb{X}_G . In this way, geometrically misleading areas are avoided in favor of areas of the state space that are more likely to lead to solution trajectories.

Algorithm 3: SearchSelection(\mathbb{O}, \mathbb{O}')

```

1  $x_{selected} \leftarrow \text{NULL};$ 
2 repeat
3   if  $\mathbb{O} == \emptyset$  then
4      $\mathbb{O} \leftarrow \mathbb{O}';$ 
5      $\mathbb{O}' \leftarrow \emptyset;$ 
6    $x \leftarrow \min_{c+h}(\mathbb{O});$ 
7    $\mathbb{O} \leftarrow \mathbb{O} \setminus \{x\};$ 
8    $n \sim U([0, 1]);$ 
9   if  $\text{quality}(x) < n$  then
10     $\mathbb{O}' \leftarrow \mathbb{O}' \cup \{x\};$ 
11     $x.priority = 1;$ 
12  else
13    return  $x;$ 

```

Informed Selection: A major departure from the SST framework is that node selection is not done via random sampling in the state space. Instead, a queue system is employed that ensures that every node has a chance to be selected. Initially, the open set, \mathbb{O} , contains only the start vertex, x_o , and the auxiliary set, \mathbb{O}' , is empty. Every time a node is selected and tries to add a new child node, it is removed from \mathbb{O} and gets inserted into \mathbb{O}' , with one exception that is explained in the blossom propagation section.

Because this is a continuous state and control space problem, one expansion of a node is not sufficient to exhaust all possible control inputs for that node. For this reason, when \mathbb{O} is empty, the auxiliary set, \mathbb{O}' , is reassigned as \mathbb{O} and is subsequently cleared. In this way, all nodes that are in the tree will be continuously re-expanded, unless pruned for path quality.

For each iteration of iSST, the SearchSelection is called to return a node to expand (the algorithm can be found in Algorithm 3). This is replacing the Best_Near procedure from SST. Among all nodes in the open set, \mathbb{O} , the node

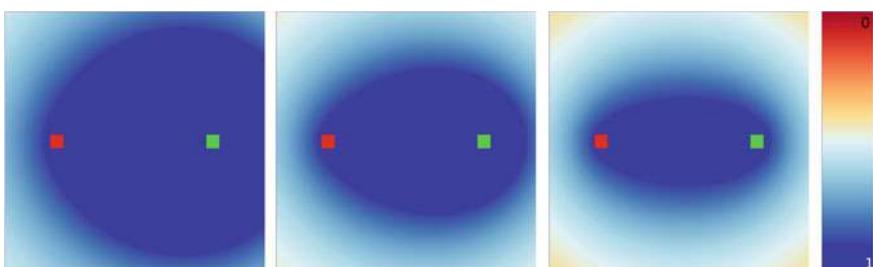


Fig. 2 Quality measures of nodes in a 2D workspace when the best known trajectory cost to the goal is $3 \times$ the optimal cost (left), $1.5 \times$ the optimal cost (middle), and $1.05 \times$ the optimal cost (right)

with the lowest $c + h$ value is returned, where c is the path cost from the root node, x_o , to that node and h is the heuristic value to the goal. Similar evaluations can be found in A^* formulations. After a node is found in the open set, a quality measure for that node is computed:

$$\text{quality}(x) = \left(\frac{1}{\frac{x.c}{x_{goal}.c} + \frac{h(x)}{h(x_o)}} \right)^{x.\text{priority}}. \quad (3)$$

This expression aims to bias selection toward nodes that are most likely to improve upon the current solution found, which has cost $x_{goal}.c$. A simple illustration of how this function biases toward nodes that are more likely to provide better solution trajectories can be found in Fig. 2. When a solution has not been found, the expression reduces to $h(x_o)/h(x)$ which says that any node that has heuristic value less than the start node's will be selected with probability one.

Algorithm 4: Blossom(x, h)

```

1  $X_{\text{new}} \leftarrow \emptyset;$ 
2 repeat
3    $x_{\text{new}} \leftarrow \text{MonteCarlo - Prop}(x, u \sim U(\mathbb{U}), t \sim U([0, T]));$ 
4   if CollisionFree( $x_{\text{new}}$ )
5    $\wedge \text{Is\_Node\_Locally\_the\_Best\_SST}(x_{\text{new}}, W, \delta_w)$ 
6    $\wedge \text{BranchAndBound}(x_{\text{new}})$  then
7      $X_{\text{new}} \leftarrow X_{\text{new}} \cup \{x_{\text{new}}\};$ 
8 until  $M$  tries;
9 return  $\min_h(X_{\text{new}});$ 

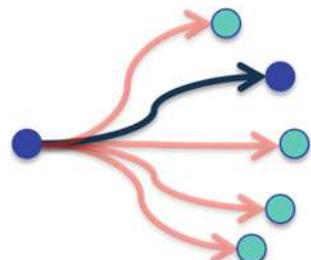
```

Blossom Propagation: Another deviation from the SST framework is the use of a “blossom” variant of extension [15] (see Fig. 3). Instead of only trying out one control input every iteration, a series of propagations are performed. Among all of these propagations that are simulated, any that collide with the environment (Algorithm 4, Line 4), would be pruned via the witness sample set (Line 5), or would never produce a better solution trajectory to the goal (Line 6) are all removed from

Fig. 3 A basic illustration of the Blossom procedure.

After trying many different candidate edges, only one is added to the search tree.

Edges can be disregarded due to collisions, suboptimal paths, or subpar heuristic values



consideration. Of the remaining nodes, the node with the minimal heuristic value will be the next node added to the search tree. This set of propagations from the same start state aims to mimic the expansion step in A^* searches. But given that the control space is continuous, all possible control inputs cannot be tried immediately. Instead, a number of tries are attempted, M . This number is unique to each node, and is reduced upon each expansion. When M becomes 1, the expansion procedure reverts to MonteCarlo-Prop from SST.

Another modification that can be done in the Algorithm 4 is introducing pre-built maneuver sets. For certain robot platforms, an expert user may be able to provide a set of maneuvers known to have a variety of end states. For car-like systems, this may be maneuvers that human drivers would perform. The MonteCarlo-Prop call in Algorithm 4 can be replaced with a lookup into this maneuver library, as long as random propagations are restored when all maneuvers have been attempted. Using maneuvers can help speed up iSST's performance by using known good controls, similar to lattice-based methods.

Informed Pruning: In SST, nodes that share a witness node, i.e. are within a δ_w radius of a witness node w , are evaluated against each other. The node with the smaller path cost from x_o would be kept, and the other node would be pruned. In iSST, a similar procedure is done, but the evaluation criterion is now $x.c + h(x)$. With this additional information about expected path cost, the algorithm can be more confident that pruned nodes will be more unlikely to improve the current best solution trajectory.

The commonly used technique of branch-and-bound can also be used in iSST. Whenever a solution is found, any node that will never provide a better solution can be pruned, i.e. $x.c + h(x) < x_{goal}.c$. This further reduces the number of nodes that have to be expanded in future iterations, making it more likely that better solutions can be found.

Effects of Changes on Properties: The proof of SST's probabilistic completeness property relies on showing that every node in the tree will be selected infinitely often and eventually all control inputs will be tried. With the Best_Near selection process, this involved showing that every node had a volume of the state space where if a random sample is thrown, that node will be selected for propagation. In the case of iSST, each node is selected infinitely often by construction. The open set selection guarantees that each node has a positive probability of being selected. Regarding the propagation properties, as long as iSST reverts to a single random control in the Blossom procedure, we can maintain probabilistic completeness. The number of blossomed propagations is reduced over time to a single propagation.

The proof of SST's probabilistic completeness and asymptotic near-optimality rely on the definition of a δ -robust trajectory. A δ -robust trajectory is a trajectory, π , that is always δ distance away from the obstacle space, \mathbb{X}_{obs} , and has more than δ dynamic clearance. Then, it can be shown that if $\delta_{BN} + 2\delta_w < \delta$, the properties of probabilistic completeness and asymptotic near-optimality apply to SST. For iSST, since the Best_Near procedure has been replaced with SearchSelection, δ_{BN} is not a needed parameter to consider in the analysis. It can then be shown that

$2\delta_w < \delta$ is the condition of the properties outlined above. This bound may not be tight, but follows directly from the analysis in previous work [21]. Having an adaptive δ_w in different regions of the state space is the subject of future work, which would also help alleviate this condition in the analysis. In practice, most reasonable values of δ_w work well.

For a given value of δ_w , asymptotic near-optimality can be proven in SST, and subsequently for iSST. The analysis in prior work [21] provides a method for achieving asymptotic optimality by reducing the δ_{BN} and δ_w radii over time. The same method can be applied in the case of iSST. It is, however, not recommended to implement this modification, since it is only needed for an asymptotic property. For practical run times, the iSST algorithm presented here provides good performance.

4 Experimental Evaluation

Experimental Setup: In order to test the performance of iSST, a few robotic system models are considered for evaluation. These robotic system abstractions are meant to approximate real-world analogs. There are three different robotic systems implemented for the evaluation of the new motion planning algorithm.

Second-order car: A four-wheeled vehicle with dynamics needs to reach a goal region traversing through a parking lot environment filled with vehicles (see Fig. 4). The state space is 5D (x, y, θ, v, ω), the control space is 2D ($\dot{v}, \dot{\omega}$) and the dynamics are: $\dot{x} = v \cos(\theta) \cos(\omega)$, $\dot{y} = v \sin(\theta) \cos(\omega)$, $\dot{\theta} = v \sin(\omega)$.

The optimization criterion for trajectories is time to execute the trajectory, thus a minimum time path is the objective. The heuristic function for the car takes the shortest path from the preprocessed data structure, and then computes the minimum

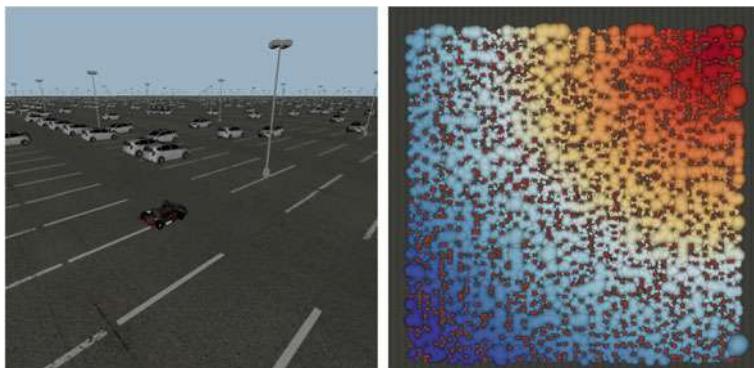
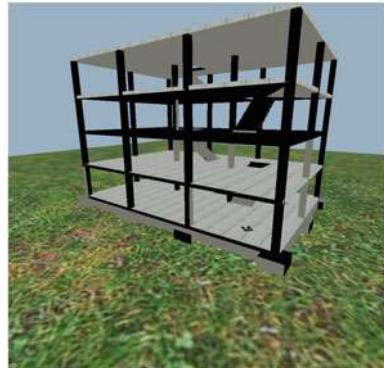


Fig. 4 The test case environment for the second-order car (left). The task to traverse a parking lot with lightpoles and cars as obstacles. The heuristic function gradient is shown at the left with the goal in the top right of the aerial view

Fig. 5 The test case environment for the quadrotor. The task is to traverse the multi-level building to reach the roof. Walls are removed to show the floor plan



time to traverse that distance if the car were to follow that path. This time is not feasible due to the dynamics of the car, but provides a reasonable lower bound. There is no state cost used here.

Quadrotor: A quadrotor moving in three dimensional workspace of a building (see Fig. 5). The dynamics are derived from [1]. The state space is 12 dimensional and the control space is four dimensional, corresponding to the thrust from each rotor. The cost function is the workspace distance traveled by the quadrotor, while the heuristic is the same.

Physically-simulated Rover: A physically-simulated rover system tasked with traversing an uneven environment. The system is simulated with the Bullet physics engine [6]. The system is the most computationally expensive among the three test cases, and will be given extra time in the experiments to show algorithm performance. The time for simulation is only going to be possible for planetary exploration tasks that can have a large lead time before navigation. Future work aims to alleviate this computational cost to get closer to the car and quadrotor runtimes.

The heuristic function for this test case utilizes a state cost function in the workspace. The terrain generates a heightmap in the two-dimensional workspace of the car, which can then be transformed into a state cost function. This height cost function then weights trajectories according to their elevation, promoting paths that do not change elevation significantly.

Results: To evaluate the motion planners, several statistics are gathered from each individual planning instance. Solution cost over time is reported to show how well algorithms can improve their solution trajectory quality. When solutions with better costs are found quickly, this builds confidence in the algorithm's capability to find near-optimal solutions in practice.

To evaluate the informed motion planner, iSST, several comparison methods are utilized. For sampling-based comparisons, the canonical RRT is evaluated as a baseline [19]. Then, the original SST algorithm is shown as the state-of-the-art asymptotically near-optimal algorithm for finding motion plans for dynamical systems [21]. Finally, in order to evaluate the speed of finding solution trajectories,

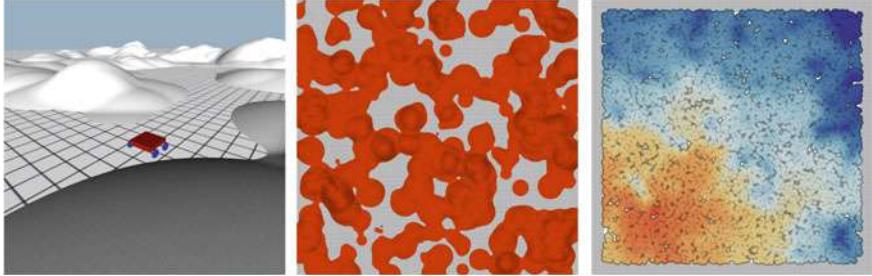


Fig. 6 The test case environment for the physically-simulated rover (left). The task is traversing to the bottom left of the map (see middle overhead view). Since the heuristic takes the terrain height into account, the heuristic does not follow a regular wavefront behavior (see right figure)

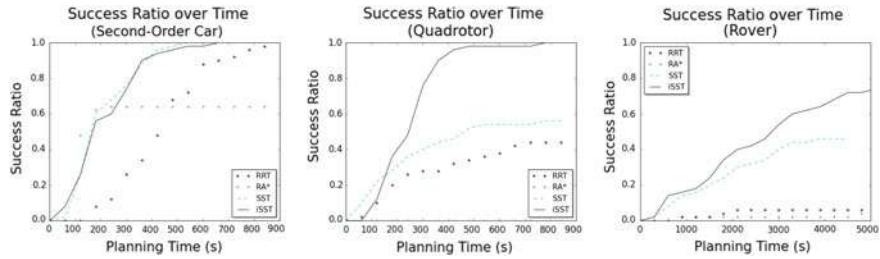


Fig. 7 Ratio of found solutions to planner attempts over time for the second order car (left), the quadrotor (middle), and physically-simulated rover (right). Every planner was given 50 opportunities to return a result within their computational budget (15 min for the car and quadrotor, and an iteration limit for the rover). Ratios closer to one are better

Randomized A^* is implemented [7] and uses the same heuristic information that iSST uses. In addition, Randomized A^* also makes use of the same maneuver sets from iSST.

The success rate over time for finding solution trajectories is shown in Fig. 7. In the case of the second-order car, SST and iSST get similar number of solutions over time, since random controls naturally explore the workspace well in the case of SST. Randomized A^* is unable to return a solution reliably due to its incomplete nature.

In the case of the quadrotor, iSST and SST diverge more. It is more difficult in this case to find solutions without a guiding heuristic, which makes iSST find solutions quicker. Just using the heuristic is not sufficient however, as shown by the Randomized A^* not finding a single solution in this problem. By allowing retries on nodes, the success rate is improved in iSST.

When moving to the more computationally expensive rover, the amount of time necessary to find solutions is much larger. But relative to the comparison methods, iSST is able to provide more solutions. Randomized A^* takes significantly more time to execute the same number of iterations as the other algorithms, so the graph in Fig. 7 (right) is truncated for readability.

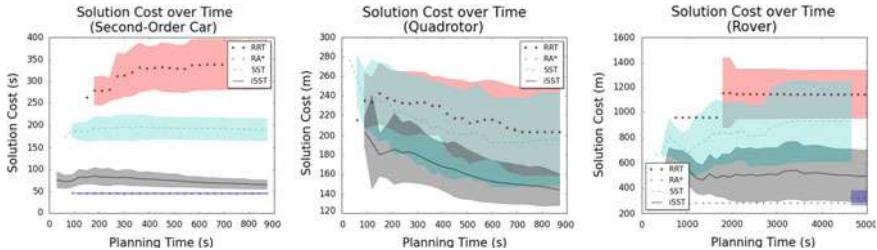


Fig. 8 Average solution cost over time for the second order car (left), the quadrotor (middle), and physically-simulated rover (right). Every planner was given 50 opportunities to return a result within their computational budget (15 min for the car and quadrotor, and an iteration limit for the rover) and all solutions at each time instance are averaged for an overall evaluation. Lower solution costs are better

The solution cost results for each test case can be found in Fig. 8. Interestingly, when Randomized A* finds solutions, they are usually better than those returned by all other algorithms compared here. The drawback, however, is that Randomized A* does not guarantee returning solutions, and in some cases like the quadrotor, likely will not return a solution. iSST, on the other hand, returns low cost solutions and finds those solutions relatively quickly. iSST also decreases the best solution cost over time at a much faster rate than SST.

5 Discussion and Conclusion

By leveraging insights from many heuristic search methods, iSST is able to provide solution trajectories for systems with dynamics for a small computational cost. Additionally, these trajectories can be improved over time, leveraging the anytime nature of the SST framework. This is an encouraging result for achieving the goal of effective and practical motion planning, but there are many outstanding issues to address in future steps.

In regards to the SST framework, there are two parameters that drastically affect the performance of the method, the selection radius, δ_{BN} , and the pruning radius, δ_w . iSST removes the requirement for choosing δ_{BN} , but the pruning radius still requires careful selection. One way to potentially address this is to adapt the pruning regions based on clearance from obstacle regions. This information can be generated during the construction of the heuristic estimate, and may alleviate some conditions of the analysis necessary for asymptotic optimality.

Another way to improve the performance of iSST is to provide more diverse maneuver sets. This work used maneuver sets that are generated by an expert user. While this is sufficient for well-defined robots like cars and quadrotors, more complex systems may need a computational approach to generating this set. Leveraging machine learning techniques for finding maneuvers will enable this informed plan-

ning framework to work for novel robot designs that previously were difficult to study. These maneuver sets can also provide estimates of the expected update to the state of the robot, making heuristic computation at the end of trajectories possible without a full simulation. This would make the physically-simulated rover experiment shown in this work more computationally efficient, bringing the computation down to similar levels for the other experimental setups.

References

1. Ai-Omari, M.A.R., Jaradat, M.A., Jarrah, M.: Integrated simulation platform for indoor quadrotor applications. In: 2013 9th International Symposium on Mechatronics and its Applications (ISMA) (2013)
2. Bekris, K., Kavraki, L.: Informed and probabilistically complete search for motion planning under differential constraints. In: First International Symposium on Search Techniques in Artificial Intelligence and Robotics (STAIR), Chicago, IL (2008)
3. Choudhury, S., Gammell, J.D., Barfoot, T.D., Srinivasa, S.S., Scherer, S.: Regionally accelerated batch informed trees (rabit*): A framework to integrate local information into optimal path planning. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 4207–4214. IEEE (2016)
4. Clement, L., Kelly, J., Barfoot, T.D.: Monocular visual teach and repeat aided by local ground planarity. In: Field and Service Robotics, pp. 547–561. Springer (2016)
5. Cohen, B.J., Chitta, S., Likhachev, M.: Search-based planning for manipulation with motion primitives. In: 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 2902–2908. IEEE (2010)
6. Coumans, E.: Bullet Physics Engine. <http://bulletphysics.org> (2012)
7. Diankov, R., Kuffner, J.: Randomized statistical path planning. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007, pp. 1–6. IEEE (2007)
8. Fang, Z., Yang, S., Jain, S., Dubey, G., Roth, S., Maeta, S., Nuske, S., Zhang, Y., Scherer, S.: Robust autonomous flight in constrained and visually degraded shipboard environments. *J. Field Robot.* **34**(1), 25–52 (2017)
9. Ferguson, D., Howard, T.M., Likhachev, M.: Motion planning in urban environments: Part II. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008, pp. 1070–1076. IEEE (2008)
10. Ferguson, D., Stentz, A.: Using interpolation to improve path planning: the field d* algorithm. *J. Field Robot.* **23**(2), 79–101 (2006)
11. Gammell, J.D., Srinivasa, S.S., Barfoot, T.D.: Informed rrt*: optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. [arXiv:1404.2334](https://arxiv.org/abs/1404.2334) (2014)
12. Hollinger, G.A., Sukhatme, G.S.: Sampling-based robotic information gathering algorithms. *Int. J. Robot. Res.* **33**(9), 1271–1287 (2014)
13. Inotsume, H., Creager, C., Wettergreen, D., Whittaker, W.R.L.: Finding routes for efficient and successful slope ascent for exploration rovers. In: The International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS) 2016 (2016)
14. Islam, F., Narayanan, V., Likhachev, M.: Dynamic multi-heuristic A. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 2376–2382. IEEE (2015)
15. Kalisiak, M., van de Panne, M.: RRT-blossom: RRT with a local flood-fill behavior. In: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006, pp. 1237–1242. IEEE (2006)
16. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* **30**(7), 846–894 (2011)

17. 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)
18. Kunz, T., Stilman, M.: Kinodynamic RRTs with fixed time step and best-input extension are not probabilistically complete. In: *Algorithmic Foundations of Robotics XI*, pp. 233–244. Springer (2015)
19. LaValle, S., Kuffner, J.: Randomized kinodynamic planning. *IJRR* **20**(5), 378–400 (2001)
20. Li, Y., Littlefield, Z., Bekris, K.E.: Sparse methods for efficient asymptotically optimal kinodynamic planning. In: *Workshop on Algorithmic Foundations of Robotics (WAFR)* (2014)
21. Li, Y., Littlefield, Z., Bekris, K.E.: Asymptotically optimal sampling-based kinodynamic planning. *Int. J. Robot. Res.* **35**(5), 528–564 (2016)
22. Likhachev, M., Ferguson, D.: Planning long dynamically-feasible maneuvers for autonomous vehicles. *Int. J. Robot. Res. (IJRR)* **28**, 933–945 (2009)
23. Likhachev, M., Stentz, A.: R* search. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. Citeseer (2008)
24. Littlefield, Z., Klimenko, D., Kurniawati, H., Bekris, K.E.: The importance of a suitable distance function in belief-space planning. In: *International Symposium on Robotic Research (ISRR)* (2015)
25. Nash, A., Daniel, K., Koenig, S., Felner, A.: Theta*: any-angle path planning on grids. In: *AAAI*, pp. 1177–1183 (2007)
26. Pereira, A.A., Binney, J., Hollinger, G.A., Sukhatme, G.S.: Risk-aware path planning for autonomous underwater vehicles using predictive ocean models. *J. Field Robot.* **30**(5), 741–762 (2013)
27. Persson, S.M., Sharf, I.: Sampling-based A* algorithm for robot path-planning. *Int. J. Robot. Res.* **33**(13), 1683–1708 (2014)
28. Pivtoraiko, M., Kelly, A.: Kinodynamic motion planning with state lattice motion primitives. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2172–2179. IEEE (2011)
29. Pivtoraiko, M., Knepper, R.A., Kelly, A.: Differentially constrained mobile robot motion planning in state lattices. *J. Field Robot.* **26**(3), 308–333 (2009)
30. Posa, M., Kuindersma, S., Tedrake, R.: Optimization and stabilization of trajectories for constrained dynamical systems. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1366–1373. IEEE (2016)
31. Wells, A., Plaku, E.: Adaptive sampling-based motion planning for mobile robots with differential constraints. In: *Conference Towards Autonomous Robotic Systems*, pp. 283–295. Springer (2015)
32. Ziegler, J., Stiller, C.: Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009*, pp. 1879–1884. IEEE (2009)
33. Zucker, M., Ratliff, N., Dragan, A.D., Pivtoraiko, M., Klingensmith, M., Dellin, C.M., Bagnell, J.A., Srinivasa, S.S.: CHOMP: covariant Hamiltonian optimization for motion planning. *Int. J. Robot. Res.* **32**(9–10), 1164–1193 (2013)

Strategic Autonomy for Reducing Risk of Sun-Synchronous Lunar Polar Exploration

Nathan Otten, David Wettergreen and William Whittaker

Abstract Sun-synchronous lunar polar exploration can extend solar-powered robotic missions by an order of magnitude by following routes of continuous sunlight. However, enforcing an additional constraint for continuous Earth communication while driving puts such missions at risk. This is due to the uncertainty of *singularities*: static points that provide weeks of continuous sunlight where communication blackouts can be endured. The uncertainty of their existence and exact location stems from the limited accuracy of lunar models and makes dwelling at singularities a high-risk proposition. This paper proposes a new mission concept called *strategic autonomy*, which instead permits rovers to follow preplanned, short, slow, autonomous drives without communication to gain distance from shadow and increase confidence in sustained solar power. In this way, strategic autonomy could greatly reduce overall risk for sun-synchronous lunar polar missions.

1 Introduction

Solar-powered lunar rovers (Fig. 1) could achieve months of exploration by following sun-synchronous polar routes to maintain continuous sunlight for power and heating while avoiding prolonged exposure to extreme cold. These routes favor peak elevations at high latitudes and could be accomplished with slow driving speeds as low as 0.1 cm/s. Such routes are intriguing because of their extended durations and their proximity to areas of scientific and commercial interest. Planning reliable sun-synchronous routes prior to launch requires detailed maps of future lighting conditions, which can be estimated based on prior 3D models of the Moon. The resolution

N. Otten (✉) · D. Wettergreen · W. Whittaker
Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh,
PA 15213, USA
e-mail: otten@cmu.edu

D. Wettergreen
e-mail: dsw@cmu.edu

W. Whittaker
e-mail: red@cmu.edu

Fig. 1 Prototype solar-powered lunar polar rover “Polaris.” Image credit: Astrobotic Technology, Inc



and certainty of predicted insolation maps is limited by the quality and accuracy of the underlying elevation models. The inherent uncertainty of lunar topography data leads to uncertainty that rovers following preplanned sun-synchronous routes will remain in continuous sunlight. The magnitude of this uncertainty is strongly influenced by the strictness of a mission’s requirement for rover-to-Earth communication.

One ostensibly conservative strategy permits rovers to drive only when in direct communication with ground controllers to facilitate teleoperation and/or highly supervised autonomy. Sans satellite relay, rovers would be required to maintain an unobstructed view of Earth when driving. This strict requirement severely constrains rover movement and, in many cases, would force rovers to dwell motionless near the day–night terminator for weeks at a time to achieve multiple lunar days of operation. Locations of enduring sunlight must be reached prior to losing Earth communication, which must later be reestablished before driving can resume. Such locations are so small and rare that they effectively behave as single points or *singularities*, through which all valid routes must pass. Due to the uncertainty of predicted insolation maps, the existence of such points cannot be guaranteed. A mission dependent on dwelling at a presumed singularity that turns out not to exist will almost certainly terminate prematurely. So, while solar-powered missions with a strict requirement for sustained communication when driving require singularities to achieve multiple lunar days, the uncertainty of such phenomena makes them unreliable as a means to that end. Rather than reducing overall mission risk, this “conservative” strategy ultimately increases odds of failure.

This paper proposes an alternative strategy for achieving extended lunar polar rover missions. In the absence of Earth communication, *strategic autonomy* would permit short, slow, autonomous drives following preplanned paths in areas adjacent to singularities to guarantee continuous sunlight. Under this new mission concept, the majority of a rover’s distance traveled could still be teleoperated and highly supervised. The key difference is that in situations where a communication blackout is inevitable, the rover would be liberated to drive autonomously to increase its estimated distance from shadow and thus improve its expectation of solar power. While the limited use of unsupervised autonomy does incur some additional risk, it is outweighed by increased confidence in uninterrupted sunlight. The adoption of strategic autonomy for sun-synchronous lunar polar exploration could substantially reduce overall risk and enable greater mission durations and returns.

The remainder of this paper is organized into five sections followed by a conclusion. Section 2 provides relevant background information regarding the lunar polar

environment, sun-synchronous exploration, and the models used for route planning. Section 3 describes a common concept of operations for lunar roving that enforces a strict communication constraint and presents a baseline route. Section 4 introduces a 2D representation of dynamic 3D constraints useful for analyzing and comparing routes. Section 5 formulates the concept of strategic autonomy and presents a corresponding alternative route. Section 6 highlights the benefits of strategic autonomy.

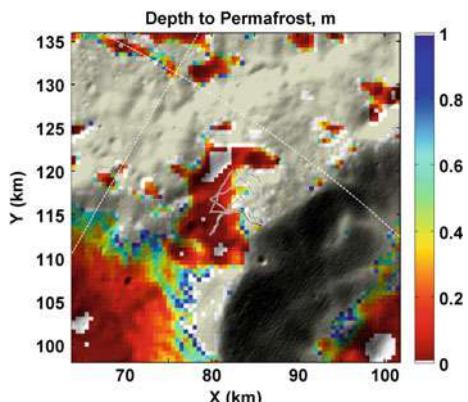
2 Background

2.1 The Lunar Polar Environment

Water The abundance of water on the Moon is well-documented, and the next steps of exploration are surface missions to visit and characterize local concentrations. Lunar water is a key resource because it can be converted into breathable air, drinkable water, and combustible propellant, all three of which are vital to sustaining exploration beyond Earth. Orbiting satellites have collected overwhelming evidence of vast quantities of frozen water ice concentrated at the lunar poles [3, 11], and NASA is considering a solar-powered robotic surface mission to verify and quantify this water and other frozen volatiles [2]. The estimated subsurface depth of frozen water permafrost [10] at a candidate landing site near the lunar south pole is shown in Fig. 2. In addition to accessible water, Nobile exhibits a combination of topography and latitude ideal for demonstrating the advantages of strategic autonomy.

Sunlight Because the synodic period of the Moon averages approximately 708 hours, a single lunar day is equivalent to about 29.5 Earth days [4]. At non-polar latitudes, this yields alternating day and night periods of sunlight and darkness averaging

Fig. 2 Estimated depth of permafrost below the lunar surface for a roughly 40-by-40-km area of the Nobile Crater rim near 86° S [10]. Gray terrain indicates an estimated permafrost depth of greater than 1 meter. A 74-day sun-synchronous route, which crosses the steep gradient between deep and shallow permafrost several times, is located near the center of the area shown. Image credit: Richard Elphic



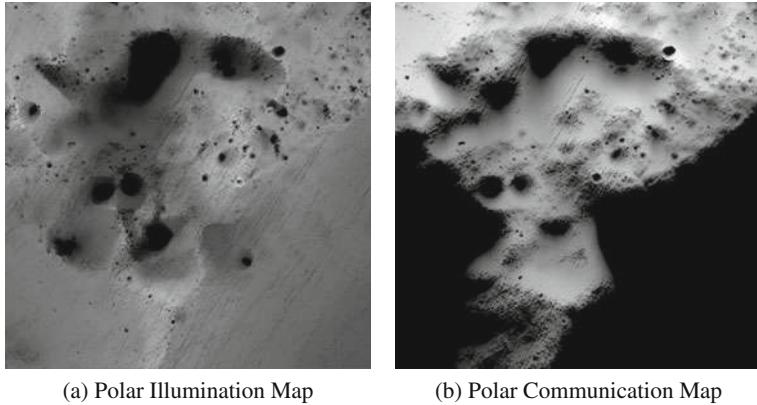


Fig. 3 These images represent the composite illumination **(a)** and communication **(b)** coverage over a 4-month period of a 20-by-20-km area on the rim of Nobile Crater near the lunar south pole. Brighter values indicate greater cumulative exposure (to the Sun or Earth, respectively) as a percentage of total time: *white* indicates 100%, while PSRs appear as *black*. Earth is upward

14.75 Earth days each. Limited to 2 weeks of solar power, slow-moving¹ rovers that lack a sustainable internal heat source (e.g., nuclear isotopes) are doomed by the unavoidable cryogenic temperatures characteristic of lunar night [9, 10, 15].

Near the poles, however, seasonal effects and local topography dominate, resulting in more dynamic and complex lighting conditions. The Moon's 1.5° axial tilt relative to the ecliptic induces seasons that are subtle relative to Earth's yet pronounced enough to leave the poles almost completely dark during winter and predominately lit during the summer. The Sun circles the summer pole once per lunar day and never deviates far above (or below) the horizon; thus, sunlight grazes the lunar polar surface. The low-angle light renders shadows that stretch many kilometers and sweep across the rough terrain, locally blocking out sunlight for days at a time. Although no polar peak is lit 100% continuously [6], slow-moving rovers can maintain uninterrupted solar power for 3–6 months by navigating strategic sun-synchronous routes. This enables mission durations an order of magnitude greater than what is possible at lower latitudes or without sun-synchronous planning.

A 4-month composite of polar illumination based on 3D modeling is shown in Fig. 3a. This is the same area as that shown in Fig. 2. The permanently shadowed regions (PSRs) generally correspond to high concentrations of ice, while the small points of maximum illumination at local peaks are characteristic of singularities.

Communication After power, the most critical resource for lunar rovers is communication with Earth. Without frequent communication, rovers cannot adequately receive commands nor transmit telemetry and scientific data. A satellite relay could provide (near) constant communication with polar rovers, but such infrastructure is

¹Here, less than 10 cm/s is considered slow. (Circumnavigating the equator requires ~ 4.3 m/s.).

costly to deploy. Polar rovers can instead rely on direct line-of-sight to Earth-based antennae. Due to tidal locking, only one hemisphere of the Moon ever faces Earth; however, at the poles, the Moon's axial wobble causes the Earth to rise and set periodically as viewed from the surface. This induces intermittent communication coverage with periodic blackouts common at local depressions and on non-Earth-facing slopes. These conditions are not conducive to highly supervised autonomy or teleoperation. A composite of polar communication coverage (via line-of-site to Earth) is shown in Fig. 3b. The steepest communication gradient coincides with the peak of maximum illumination, another characteristic of singularities.

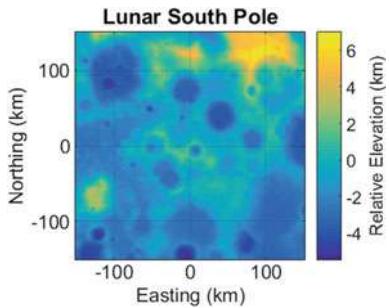
Topography Another important aspect of the lunar polar environment is local topography, which dictates not only the sunlight and communication available to rovers but also the terrain that must be negotiated. While rover-scale terrain features cannot yet be resolved using the best lunar data, topographic models do yield maps of gross ground slope, much of which is too steep for typical wheeled rovers to traverse.

2.2 *Sun-Synchronous Mission Planning*

Sun-synchronous circumnavigation routes for maximizing solar power and extending missions were first envisioned without the benefit of high-fidelity lunar terrain models [18]. The concept was later partially demonstrated in field experiments each spanning a single day on Earth [17]. The concept of mission-directed path planning was developed to enable global planning and navigation capabilities for planetary rovers over large scales and long durations amenable to sun-synchronous exploration [14]. The TEMPEST planner embodied a sun-synchronous navigation strategy tailored to solar-powered polar exploration and was demonstrated onboard the “Hyperion” rover on Devon Island north of the Arctic Circle. It was not extended to multiple diurnal periods, interplanetary communication, nor the use of lunar data.

Prior work by the authors presents examples of sun-synchronous lunar polar routes generated using data from Lunar Reconnaissance Orbiter (LRO) [5, 8]. South polar routes at Malapert Massif and Shackleton Crater maintain uninterrupted sunlight and traversable slopes for 2 months with maximum driving speeds of 1 cm/s and 1 m/s, respectively, but do not account for communication [8]. A route near Nobile Crater (similar to the routes presented in this paper) achieves 74 days with a strict requirement for direct line-of-sight to Earth when driving [5]. Although this route dwells twice at a ‘singularity,’ prior research did not identify or define this concept nor did it address the related risk.

Fig. 4 This 10-m LOLA DEM covers the lunar south pole out to 85° S. The Nobile Crater rim is among the peak elevations



2.3 Lunar Models

Prior work and this research use predictions of future lighting generated using lunar digital elevation models (DEMs), lunar and solar ephemeris data, and ray-tracing software. The DEMs are a data product of LRO's Lunar Orbiter Laser Altimeter (LOLA) instrument.² The highest resolution LOLA DEM covering the Nobile Crater region is 10 m per pixel [12]. This 2.5D elevation model, shown in Fig. 4, was converted into a 3D mesh model in Cartesian coordinates, and a simulated Sun³ was positioned according to the SPICE toolkit [1]. Ray-tracing software renders an orthographic map of lighting conditions for each time in a predefined sequence. The result is a 3D binary array that defines each (x, y, t) state as either lit or shadowed, in this case with spatial resolution of 10 m and a temporal resolution of approximately 2 hours. The same process was used to estimate communication coverage by substituting the Earth's position for the Sun's.⁴ Principal ground slope was computed by applying divided difference to the DEM.

The resolution and accuracy of LOLA data is orders of magnitude better than what was available previous to LRO [19]; however, uncertainty remains an issue. The accuracy of individual LOLA ground points is approximately 10 m radially and 50–100 m spatially [13], and gridded LOLA DEMs contain a myriad of visible artifacts. In low-angle lighting, these errors are magnified, resulting in substantial uncertainty about the exact location of sunlight–shadow boundaries. This uncertainty makes dwelling near the terminator a risky proposition.

²LRO data products are accessible via NASA's Planetary Data System Geosciences Node [7, 16].

³The Sun can be approximated as a directional or area light source. The latter yields a range of solar flux values, which can be thresholded to produce a binary output for planning purposes.

⁴Since radio waves behave differently than visible light, this yields a slightly optimistic estimate.

3 Supervised Teleoperation

This section describes a common concept of operations that uses a mix of highly supervised autonomy and teleoperation. An example route is presented as a baseline for later comparison with a route demonstrating the concept of strategic autonomy.

The default concept of operations is loosely modeled after those used by NASA's Mars rovers, albeit at an accelerated cadence. The Mars Science Laboratory "Curiosity" and the Mars Exploration Rover "Opportunity" are each actively managed by a team of rover operators that upload a sequence of commands no more than once per day based on previously returned telemetry and science data. Upon receiving the command sequence, the rover executes the actions (if possible) with the aid of a limited set of semi-autonomous navigation capabilities, transmits new data back to Earth, and awaits further instruction. This cyclical process generally repeats every 24 hours. The pace is dictated by the 20-min average communication delay, limited solar flux, and the length of a sol. Real-time teleoperation is prohibited by high latency due to the large distance between Earth and Mars, and rover movement is highly supervised, never going beyond the previous day's horizon.

A similar approach could be used for lunar exploration but at a far more rapid cadence enabled by the Moon's proximity to Earth, greater solar flux, and lesser gravity. Command cycles could iterate every few hours or minutes instead of once per day. This pace approaches that of pure teleoperation, which is possible at slow driving speeds and with low latency of a few seconds. In this strategy, no unsupervised rover autonomy would be required nor permitted. The rover would operate under constant supervision with humans in the loop ready to intervene if necessary.

Constraints The supervised teleoperation ("teleop") concept of operations imposes the following constraints on sunlight, communication, and terrain slope.

Sunlight The rover must remain in direct sunlight at all times and cannot enter or be overcome by shadow.

This constraint is constant and unconditional. While a rover's onboard battery, depending on its size, could enable it to operate, drive, and possibly even heat itself for a limited time in the absence of sunlight, it is simpler and safer to impose a strict constraint to always remain in sunlight.⁵ This constraint guarantees consistent power and heating and leverages the unique character of polar lighting to extend operational life. For the work presented here, being in shadow is defined as being at any state (location and time) for which the ground is shadowed. Extensions that account for the height of the solar array above the ground are possible; however, the definition used here provides a reasonable and conservative approximation.

Communication The rover must maintain an unobstructed view of Earth at all times during which it is driving.

⁵To complete certain science objectives, a rover may be required to enter a PSR or other unlit area for a brief period of time; however, this extension is outside the scope of the work presented here.

This constraint is conditional, as it is only active when the rover is moving. The proximity of Earth and the Moon enables a level of control not possible for Mars. The opportunity for near real-time communication invites a seemingly prudent approach that not only takes advantage of constant supervision but mandates it.

Slope The rover must remain on terrain of principal slope not exceeding 20°.

This value was chosen such that it would not dominate route planning in the following examples. In practice, this constraint is dependent on the mobility platform and could be higher or (most likely) lower.

Planning Method The gridded lunar model representations described in Sect. 2.3 are conducive to grid-based planning methods. The distinctions of sunlight, communication, and slope yield a planning problem with heterogeneous constraints. Enforcing a constraint on slope is the simplest, since it is defined by a static 2D map of binary go/no-go conditions. Enforcing a constraint on sunlight is similarly simple, with the only difference being that the map is dynamic, represented by a 3D binary array composed of stacked 2D maps for each time step. The communication condition must be enforced at planning time, since it is conditional on whether or not two consecutive rover states share the same spatial position. For two graph nodes of differing positions to be legally connected by an edge, both must have communication coverage (and sunlight). Sunlit nodes of identical position are connected along the time dimension, regardless of communication coverage. This conditional constraint is straightforward to implement in the state transition or ‘get_child_node’ function of any standard heuristic graph search algorithm.

To generate the following example route, a starting point was selected based on favorable landing site criteria, and several major waypoints were manually selected such that the rover would visit numerous sites of scientific interest near PSRs. A minimum distance path passing through all waypoints was generated using A* graph search on a 3D adaptation of an eight-connected grid (see [8] for details). This was done first at a resolution of 80 m and then the coarse route was refined to a resolution of 20 m using the 80-m route sequence as waypoints for a second iteration of A*. This hierarchical approach was used to reduce computation time.

Supervised Teleoperation Route The baseline teleop route is illustrated by a series of snapshots in Fig. 5. Each frame represents only a single instant in time; however, each path represents the full route history up to that point. Hence, *green* path overlaid on top of *black* or *blue* ground does not indicate that the rover passed through shadow, only that a visited location later became shadowed. The route covers approximately 63.5 km in 74 Earth days without exceeding 10 cm/s. It returns to its starting point after each of three drives to wait out the communication blackout. Dwelling at this singularity is the only means of achieving three lunar days of solar-powered exploration under the given constraints.

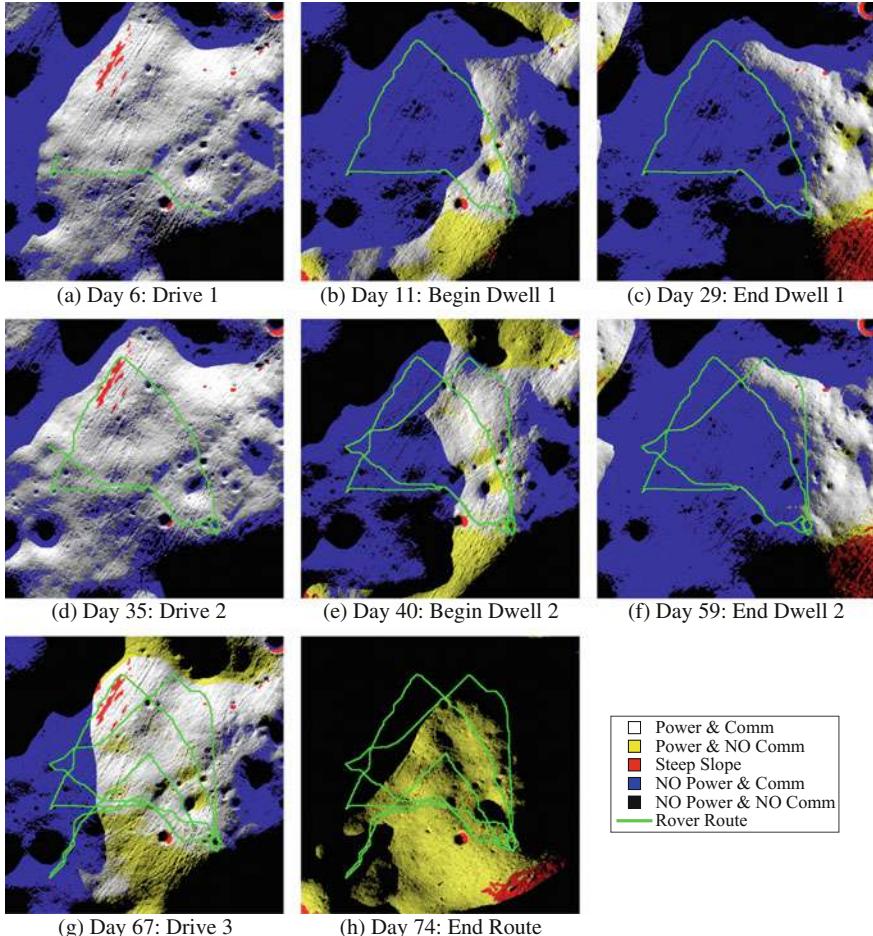
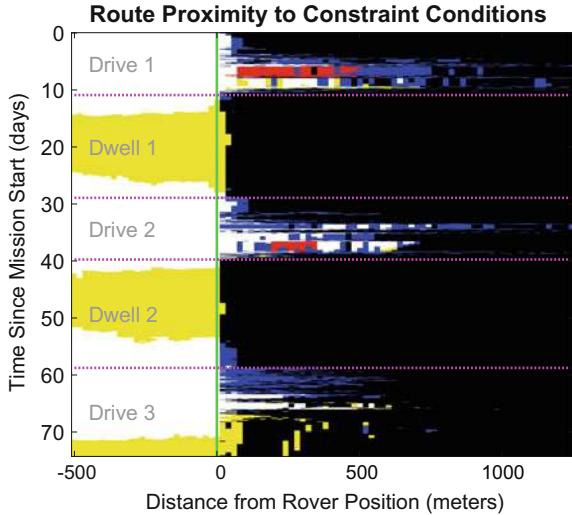


Fig. 5 *Supervised Teleoperation:* This sequence of snapshots illustrates a 3-lunar-day route near Nobile Crater. The route starts in the bottom-right quadrant and proceeds in a clockwise fashion with a long dwell period at a singularity near the starting point between each of the three drives. The rover is stationary from Day 11 (b) to Day 29 (c) and from Day 40 (e) to Day 59 (f) due to the communication constraint. It theoretically maintains uninterrupted solar power near the edge of the terminator while dwelling, but this is uncertain. The terrain is color-coded by constraint conditions

4 Route Analysis

Visualizing 3D Route Constraints in 2D This section introduces a novel method for visualizing three-dimensional spatiotemporal routes in two dimensions. The route's spatial proximity to an ordered set of dynamic constraint conditions is viewed as a function of time. This 2D projection is useful for visualizing, analyzing, and

Fig. 6 This graph shows the proximity of the teleop route to the best- and worst-case constraint conditions as a function of time. The vertical green line at 0 represents the relative position of the rover's planned path. On the right side of the green line is the most dominant constraint at every distance from the rover's position. On the left side is the least dominant constraint. The horizontal magenta dotted lines mark when the rover arrives at and departs from the singularity. See Fig. 5 for color-coding



comparing routes, particularly where 3D formats (e.g., videos) are prohibited. Furthermore, this technique reveals the risk associated with dwelling at singularities.

This 2D graphical representation was inspired by weather radar history graphs and can be explained using that analogy. A video playback of weather radar can be condensed from three dimensions to two by plotting the distance from a given location (e.g., a city center) to the nearest storm front in any direction for all times. An even richer representation can be constructed by plotting the most severe weather condition that occurs a specific distance away (in any direction) for all distances with a given range and for all instants over a span of time. Essentially, this type of graph displays the minimum distance to the worst case conditions as a function of time. A similar graph can be constructed for a moving vehicle instead of a static point, where distances are relative to the vehicle's instantaneous position. Rover constraints (e.g., sunlight, communication, slope, and combinations thereof) can be substituted for weather conditions as long as they are ordered by severity.

Applying this concept to the teleop route yields the graph shown in Fig. 6. The y-axis represents time and advances downwards like a strip chart. The x-axis marks distance from the rover's instantaneous position, which is indicated by a vertical green line. Distances are relative and can be in any direction. Each row of the plot represents a snapshot of the dominant constraints at each distance from the rover's position at that instant in time. To the right of the green line is the worst-case condition at every distance, analogous to the distance outside the storm's edge. To the left of the green line is the best-case condition at every distance (distance inside the storm's edge). The precedence of the color-coded conditions match that of Fig. 5 and are in order of increasing severity: white (power and comm), yellow (power and no comm), red (steep slope), blue (no power and comm), black (no power and no comm).

Computational Method The graph is computed one row at a time. Distance is divided into discrete uniform bins, each defined by two radii. For example, at a distance of $d = 100$ m with a resolution of 20 m, $r_1 = 90$ m and $r_2 = 110$ m. All unique constraint conditions lying within the ring formed by the two radii are isolated. The most severe condition within that subset is plotted at distance d , and the least severe condition is plotted distance $-d$. This is repeated for every distance to complete the first row and for every row to complete the graph.

Interpretation The proximity graph efficiently conveys useful information about the route plan and reveals potential concerns. By examining Fig. 6, it is clear that the rover spends two periods of nearly 20 days each within 10–30 m of total blackout from sunlight. This happens when the rover is parked at a singularity as the terminator rotates about the peak. During this time, the rover is not permitted to move due to communication denial. Because the DEM contains errors of up to 50–100 m, the location of this single point of light may vary significantly, or it may not even exist. Both cases would strand the rover in darkness and cold for weeks, likely ending the mission before communication is restored.

5 Strategic Autonomy

The concept of strategic autonomy is motivated by concerns of dwelling at an uncertain singularity as illustrated by the constraint proximity graph. This new operational concept seeks to reduce overall mission risk by distancing the rover from singularities (akin to the way robotic manipulators avoid kinematic singularities). This comes at the cost of some accepted added risk related to unsupervised autonomy during communication outages. This paper asserts that this tradeoff is beneficial.

Constraints Strategic autonomy is defined by the following constraints.

Sunlight The rover must remain in direct sunlight at all times (same as teleop).

Communication The rover must maintain an unobstructed view of Earth when driving except when doing so would cause the rover to dwell at a singularity for a period of time exceeding 24 hours. In this case, the rover is permitted to drive autonomously without communication at 20% of the nominal maximum drive speed.

This is the key distinguishing feature of the strategic autonomy concept of operations. This modified constraint prevents the rover from dwelling motionless at a singularity for long periods of time. Instead, the planner is free to maximize the rover's distance from shadow and thus confidence of sunlight.

Slope Principal ground slopes must not exceed 20° (same as teleop).

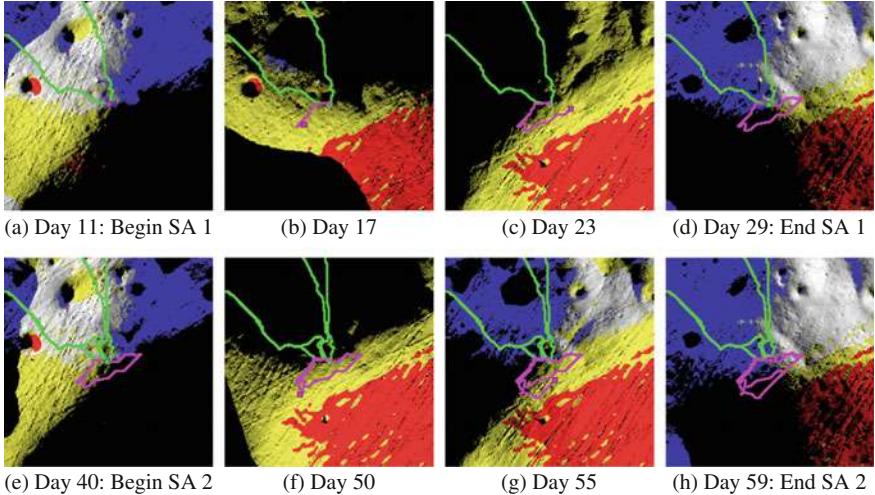
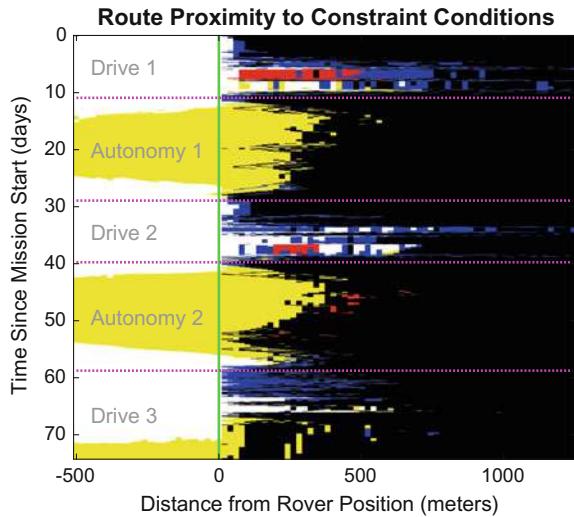


Fig. 7 Strategic Autonomy (SA): This sequence of snapshots (zoomed in relative to Fig. 5 and centered at the route's starting point/singularity) illustrates the two autonomous route segments that differ from the baseline teleop route. The top row (a–d) replaces first teleop dwell period, and the bottom row (e–h) replaces the second teleop dwell period. The autonomous drive segments are drawn in magenta, whereas unaltered route segments are green. See Fig. 5 for terrain color-coding

Planning Method The planning method begins with the baseline teleop route and modifies discrete segments according to the altered communication constraint. Singularities are identified manually with the aid of the 2D constraint proximity graph. At the instant the rover reaches an identified singularity, the altered constraints are activated. A small set of new waypoints are selected (manually, for this example) to maximize distance from darkness and steep slopes such that the new route segment splices into the original route at the instant the original dwell would have ended. Minimum distance paths passing through all new waypoints are computed using the same A* method as before with reduced maximum drive speed (from 10 to 2 cm/s) while suspending the requirement for communication.

Strategic Autonomy Route Using this method, two instances of dwelling at singularities were identified and replaced. The altered route segments, which exhibit strategic autonomy, are shown in Fig. 7. All other route segments remain unaltered and are identical to those in Fig. 5. Like the baseline, the strategic autonomy route spans 74 Earth days but covers a total distance of 71.8 km, a 11.5% increase compared to the teleop route. The constraint proximity graph is shown in Fig. 8.

Fig. 8 This visualization shows the proximity of the strategic autonomy route to the best- and worst-case constraint conditions as a function of time. The vertical green line at 0 represents the position of the rover as it traverses the planned path. This plot should be interpreted in the same manner as Fig. 6. Likewise, constraints are color-coded as in Fig. 5. The large yellow areas to the right of the green line indicate a wide margin of predicted sunlight in all directions around the rover



6 Discussion

Notable differences between the supervised teleoperation and strategic autonomy routes are shown in Table 1 and Fig. 9. Most notable is the increased mean distance to shadow. Whereas the teleop route dwells approximately 10 and 30 m from the nearest shadow (the day–night terminator) the corresponding strategic autonomy route segments achieve a mean separation of approximately 180–190 m. On a 10-m DEM, this is the difference between 1–3 pixels and nearly 20. This difference becomes even more significant when considering that the LOLA topography models have an estimated spatial accuracy of 50–100 m. If this translates to an error of up to 100 m, the supervised teleoperation route is problematic, whereas the strategic autonomy route might still retain a safety margin of almost 100 m. Compared to dwelling at uncertain singularities, this added buffer substantially increases the likelihood of maintaining solar power and continued exploration.

Table 1 Route segment comparison

Route segment	Total time (days)	Dwell time (%)	Distance traveled (km)	Max speed (cm/s)	Mean speed (cm/s)	Min dist. to shadow (m)	Max dist. to shadow (m)	Mean dist. to shadow (m)
Dwell 1	18	100	0	0	0	10	50	29
Dwell 2	19	100	0	0	0	10	30	12
Autonomy 1	18	36	4.1	1.5	0.26	10	390	191
Autonomy 2	19	38	4.2	1.6	0.26	10	370	178

Dwell supervised teleoperation dwell segment; *Autonomy* strategic autonomy drive segment

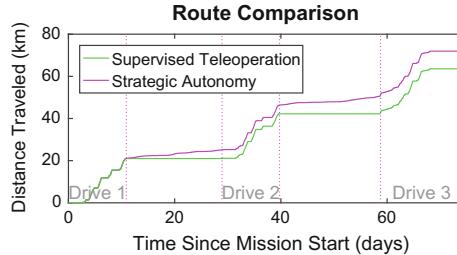


Fig. 9 The supervised teleoperation route dwells statically at singularities while the strategic autonomy route continues driving to avoid them. Vertical lines mark the start and end of the two dwell/autonomy segments corresponding to Table 1. The three segments labeled “Drive” are identical between the two routes

Strategic autonomy is not without drawbacks. Its reliance on unsupervised autonomous driving, even at very slow driving speeds, incurs some added risk. The total magnitude of this risk depends on numerous complex factors related to the rover’s perceptual, computational, and mechanical design and can therefore not be estimated easily or without rigorous testing of the actual systems. However, the high level of risk inherent to dwelling at an uncertain, unverified singularity the size of which is less than the estimated error of the models used to predict such a point almost certainly exceeds that of short, slow, autonomous drives preplanned to favor regions of almost certain sunlight.

7 Conclusion

Solar-powered rovers are a logical next step for exploring the poles of the Moon, and multi-month missions are possible using a new concept of operations called strategic autonomy. Strategic autonomy maintains continuous exposure to sunlight by permitting short, slow, autonomous drives during periodic communication blackouts—a natural consequence of sun-synchronous routes. These drives follow preplanned paths and distance rovers from the day–night terminator where sustained insolation is uncertain. Mission concepts that prohibit autonomy and enforce a strict requirement on constant communication for teleoperation inevitably force rovers to dwell stationary for weeks at small peaks of predicted sunlight; however, the existence and exact locations of such singularities cannot be confidently predicted due to limitations in lunar topographic data. This paper asserts that the risk associated with dwelling at uncertain singularities exceeds that of driving autonomously with guaranteed solar power. Strategic autonomy ultimately reduces overall mission risk while enabling extended lunar polar exploration. Future work will quantify this claim.

Acknowledgements The authors thank Dr. Tony Colaprete and Dr. Richard Elphic for their advice on the development of this work and for providing information on relevant lunar sites. This research was supported by NASA Innovative Advanced Concepts (NIAC) Grant # NNX13AR25G.

References

1. Acton, C.H.: Ancillary data services of NASA's navigation and Ancillary Information Facility. *Planet. Space Sci.* **44**(1), 65–70 (1996)
2. Andrews, D.R., Colaprete, A., Quinn, J., Chavers, D., Picard, M.: Introducing the Resource Prospector (RP) Mission. In: AIAA SPACE 2014 Conference and Exposure Reston, Virginia (2014)
3. Colaprete, A., Schultz, P., Heldmann, J., Wooden, D., et al.: Detection of water in the LCROSS ejecta plume. *Science* **330**(6003), 463–468 (2010)
4. Heiken, G.H., Vaniman, D.T., French, B.M.: *Lunar Sourcebook—A user's Guide to the Moon*. Cambridge University Press, New York, New York, USA (1991)
5. Heldmann, J., Colaprete, A., Elphic, R.C., Bussey, B., McGovern, A., Beyer, R., Lees, D., Deans, M.C., Otten, N., Jones, H., Wettergreen, D.: Rover Traverse Planning to Support a Lunar Polar Volatiles Mission. In: LEAG. NASA Ames Research Center (2015)
6. Mazarico, E., Neumann, G., Smith, D., Zuber, M., Torrence, M.: Illumination conditions of the lunar polar regions using LOLA topography. *Icarus* **211**(2), 1066–1081 (2011)
7. NASA: Welcome to the Planetary Data System. <https://pds.nasa.gov/>
8. Otten, N.D., Jones, H.L., Wettergreen, D.S., Whittaker, W.L.: Planning routes of continuous illumination and traversable slope using connected component analysis. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 3953–3958. IEEE (2015)
9. Paige, D.A., Foote, M.C., Greenhagen, B.T., Schofield, J.T., et al.: The lunar reconnaissance orbiter diviner lunar radiometer experiment. *Space Sci. Rev.* **150**(1–4), 125–160 (2010)
10. Paige, D.A., Siegler, M.A., Zhang, J.A., Hayne, P.O., et al.: Diviner lunar radiometer observations of cold traps in the moon's south polar region. *Science* **330**, 479–482 (2010)
11. Pieters, C.M., Goswami, J.N., et al.: Character and Spatial Distribution of OH/H₂O on the Surface of the Moon Seen by M3 on Chandrayaan-1. *Science* **326**(5952), 568–572 (2009)
12. Smith, D.E., Zuber, M.T., Jackson, G.B., et al.: The lunar orbiter laser altimeter investigation on the lunar reconnaissance orbiter mission. *Space Sci. Rev.* **150**(1–4), 209–241 (2010)
13. Smith, D.E., Zuber, M.T., Neumann, G.A., Lemoine, F.G., et al.: Initial observations from the Lunar Orbiter Laser Altimeter (LOLA). *Geophys. Res. Lett.* **37**(18) (2010)
14. Tompkins, P., Stentz, A., Wettergreen, D.: Mission-level path planning and re-planning for rover exploration. *Robot. Auton. Syst.* **54**, 174–183 (2006)
15. Vasavada, A.R., Bandfield, J.L., Greenhagen, B.T., Hayne, P.O., et al.: Lunar equatorial surface temperatures and regolith properties from the diviner lunar radiometer experiment. *J. Geophys. Res.-Planet.* **117**(4) (2012)
16. Washington University in St. Louis: PDS Geosciences Node Data and Services: LRO LOLA. <http://pds-geosciences.wustl.edu/missions/lro/lola.htm>
17. Wettergreen, D., Tompkins, P., Urmson, C., Wagner, M., Whittaker, W.: Sun-synchronous robotic exploration: technical description and field experimentation. *Int. J. Robot. Res.* **24**(1), 3–30 (2005)
18. Whittaker, W.L., Kantor, G., Shamah, B., Wettergreen, D.S.: Sun-synchronous planetary exploration. In: AIAA Space (2000)
19. Wieczorek, M.: The gravity and topography of the terrestrial planets. *Treatise Geophys.* (2007)

Towards Visual Teach and Repeat for GPS-Denied Flight of a Fixed-Wing UAV

M. Warren, M. Paton, K. MacTavish, A. P. Schoellig and T. D. Barfoot

Abstract Most consumer and industrial Unmanned Aerial Vehicles (UAVs) rely on combining Global Navigation Satellite Systems (GNSS) with barometric and inertial sensors for outdoor operation. As a consequence, these vehicles are prone to a variety of potential navigation failures such as jamming and environmental interference. This usually limits their legal activities to locations of low population density within line-of-sight of a human pilot to reduce risk of injury and damage. Autonomous route-following methods such as Visual Teach and Repeat (VT&R) have enabled long-range navigational autonomy for ground robots without the need for reliance on external infrastructure or an accurate global position estimate. In this paper, we demonstrate the localisation component of VT&R outdoors on a fixed-wing UAV as a method of backup navigation in case of primary sensor failure. We modify the localisation engine of VT&R to work with a single downward facing camera on a UAV to enable safe navigation under the guidance of vision alone. We evaluate the method using visual data from the UAV flying a 1200 m trajectory (at altitude of 80 m) several times during a multi-day period, covering a total distance of 10.8 km using the algorithm. We examine the localisation performance for both small (single flight) and large (inter-day) temporal differences from teach to repeat.

This work was supported by the NSERC Canadian Field Robotics Network (NCFRN) and a Mathematics of Information Technology and Complex Systems (MITACS) Accelerate Fellowship in partnership with PrecisionHawk Ltd.

M. Warren · M. Paton · K. MacTavish · A. P. Schoellig · T. D. Barfoot (✉)
University Of Toronto Institute For Aerospace Studies (UTIAS), 4925 Dufferin St,
Toronto M3H 5T6, Canada
e-mail: tim.barfoot@utoronto.ca

M. Warren
e-mail: michaelwarren@robotics.utias.utoronto.ca

M. Paton
e-mail: mpaton@robotics.utias.utoronto.ca

K. MacTavish
e-mail: kirk.mactavish@robotics.utias.utoronto.ca

A. P. Schoellig
e-mail: angela.schoellig@robotics.utias.utoronto.ca

Through these experiments, we demonstrate the ability to successfully localise the aircraft on a self-taught route using vision alone without the need for additional sensing or infrastructure.

1 Introduction

With increasing use in civilian airspace, UAVs need to be able to navigate reliably and safely using a variety of redundant sensing modalities. Typically, low-cost, commercial UAVs (Fig. 1) used for mapping and surveillance tasks rely on a combination of GNSS such as the Global Positioning System (GPS) in combination with barometric, airspeed and inertial sensing to navigate outdoors. However, these sensors are prone to both malicious and environmental interference (e.g., jamming, poor sky view, obstruction and mechanical stress). This means that airspace regulators often tightly restrict their operation to line-of-sight and low-population-density locations to minimise risk.

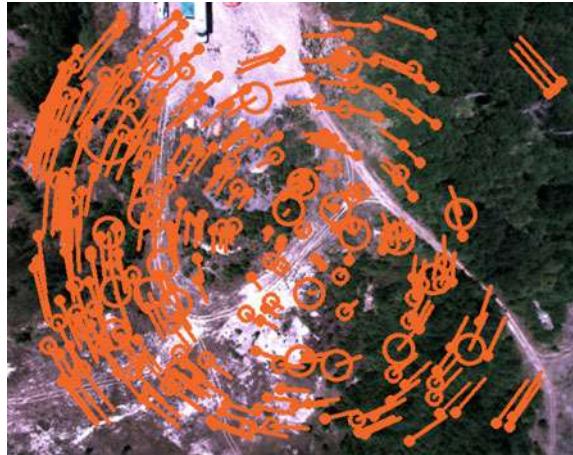
Autonomous route-following methods such as VT&R [12] have enabled long-range navigational autonomy for ground robots without relying on external infrastructure or an accurate global position estimate. By first building a visual map while under control of a human operator (the ‘teach’ phase), VT&R then allows the vehicle to autonomously re-follow the taught path (the ‘repeat’ phase) by matching sensor observations back to the original map in a local co-ordinate frame and providing path-tracking errors to a suitable vehicle controller [22]. We seek to adapt VT&R for use on a fixed-wing UAV as a demonstration of a low-cost navigation method in case of GPS, communications, or other navigational failure.

We see the applicability of VT&R on aircraft in two different cases: (1) a method of emergency return during an exploratory or traditional mapping flight, by following the ‘visual breadcrumbs’ home, and (2) acting as a complement or complete replacement of primary navigational systems when performing flights over repeat trajectories (e.g., inter-warehouse delivery or linear infrastructure inspection) in cases where GPS may be unreliable (e.g., due to poor sky view or jamming). This type of



Fig. 1 The PrecisionHawk Lancaster fixed-wing UAV used in experiments, seen here during take-off. Note the payload bay in the centre of the fuselage housing the downward-looking camera used for experiments. (Image: François Pomerleau)

Fig. 2 Post-MLESAC matches (orange lines) and features (orange circles, size denotes octave) during localisation for a repeat flight. Large orientation differences between teach and repeat phases account for the large pixel offsets seen in matching, while a high number of inliers is representative of the short temporal difference between teach and repeat (~ 20 min) in this case



safety net could open the door to operation beyond line-of-sight in more urbanised environments, and in less than ideal physical conditions.

To date, VT&R has been primarily demonstrated on ground vehicles following restricted routes. In this paper, we demonstrate a core aspect of VT&R adapted to a fixed-wing UAV: accurately localising during a repeat flight over a pre-taught route using a downward-facing, onboard camera in a large-scale, outdoor experiment (Fig. 2). This demonstration of VT&R on a UAV has some critical differences to a ground-based robot: (1) reliance on stereo for accurate scale is not feasible due to the ratio of practical baseline to altitude; (2) perspective of the scene can be radically different due to changes in altitude, orientation and position; meaning map observations can often be fleeting, (3) trajectories are no longer restricted to specific routes as there are few traversability concerns like that of ground robots when flying at sufficient altitude.

This paper presents the first demonstration of the VT&R localisation engine, or any visual route-following method, in this scenario. We present performance statistics related to localisation robustness and algorithm speed and discuss the implications and challenges of adapting VT&R to this scenario. To sufficiently limit the scope of this paper, we leave a number of tasks to future work; including closing the control loop on navigating the aircraft along the autonomously taught route, the planning of an efficient return route, and identifying when traditional navigation has failed in order to switch over to the emergency return mode.

The rest of this paper is outlined as follows: Sect. 2 describes related work, Sect. 3 outlines the monocular VT&R framework and application-specific modifications, Sect. 4 describes the vehicle, datasets, and experiments used to test the VT&R localisation engine, Sect. 5 demonstrates the results of experiments, while Sect. 6 discusses the outcomes and challenges of this work. The paper is concluded in Sect. 7.

2 Related Work

Today, most small-scale UAVs utilise GPS and inertial measurements in a filtered framework for 6 Degree-of-Freedom (DOF) state estimation [20]. However, many civil aviation authorities have made clear, through statements and regulations [8, 15], that for UAVs to perform routine operations over urban and other sensitive environments, reliance solely on GPS and radio-based communication for accurate navigation is not sufficient. New technology is attempting to bridge or mitigate this gap with improved air-to-air communications, localisation from existing infrastructure, and the ability to land safely in the event of an emergency.

Non-GPS-based navigation on aerial vehicles has seen increased interest in recent years due to these regulatory and operational issues, with many demonstrations in GPS-denied environments [9, 10, 29] using LiDAR [5] and stereo [16], visual-inertial systems [1, 26], and with vision alone [11, 18]. In most cases of outdoor, vision-only navigation, the literature has mostly been restricted to visual odometry or relatively small maps with few online examples [6, 17], mostly due to the mass and compute limitations on board the aircraft, or sometimes offloading processing to a more powerful ground-based computer with a high-rate data link.

Visual route-following on pre-built maps has been studied for some time [14]. As a modern technique, VT&R has been extensively tested in ground-based applications with stereo cameras [12], with LiDAR [19], and with multiple experiences for long-term autonomous navigation [24], and has made use of colour-constant imagery to improve resistance to lighting change [23]. It has also been tested with monocular cameras by taking advantage of the ground-plane assumption [7] and has seen preliminary testing in the air on board a Micro Aerial Vehicle (MAV) [25], demonstrating its wide applicability. Our work differs from [7] in that it does not make strict assumptions about the ground plane nor require continuous knowledge of the camera altitude, as in [25].

3 Methodology

In this paper, we intend to demonstrate robust localisation on imagery gathered from a fixed-wing UAV suited to the task of autonomous route following, without requiring input from additional sensors. We use the same software system as [24], adapted to suit a monocular front end for the single camera on board the aircraft. Similar to [24], the algorithm consists of separate teach and repeat phases. During the teach phase, the aircraft flies under control of an on-board autopilot during a primary data gathering task, analogous to the human operator used in ground-vehicle demonstrations, inserting the visual observations from this privileged experience into a relative map of pose and scene structure. During the repeat phase, without reliance on GPS or other sensors, the vehicle should autonomously re-follow the route by visually localising to the map of the privileged path. The vehicle repeats a path

by sending high-frequency localisation updates to a path-tracking controller [21]. While such a system has been demonstrated online on ground vehicles (by using a human operator and stereo vision to follow the privileged path) [24], in this paper we demonstrate the localisation engine of VT&R using datasets and leave closing the control loop to future work. The remainder of this section provides details on the mapping process (Sect. 3.2), Visual Odometry (VO) (Sect. 3.3) and the localisation process (Sects. 3.3–3.3.3).

3.1 Map Building

The map used in our system, which we refer to as a Spatio-Temporal Pose Graph (STPG), is depicted in Fig. 3. This data structure is an undirected graph, $G = \{V, E_s, E_t\}$, where V is a set of vertices, E_t is a set of *temporal* edges, and E_s is a set of *spatial* edges. Vertices, each with an associated reference frame, \mathcal{F} , store raw sensor observations and triangulated 3D landmarks with associated covariances and descriptors. Landmarks and associated descriptors are stored in the first vertex at which the feature corresponding to the landmark is observed. An edge in the graph links vertices metrically with a relative 6 DOF $SE(3)$ transformation with uncertainty. Temporal edges (blue lines) link vertices that are temporally adjacent, while spatial edges (green lines) link vertices that are temporally distant yet spatially close, i.e., from the repeat to the teach pass. Temporal edges are furthermore denoted as *privileged* if they were collected while the aircraft was self-teaching a route under autopilot control or *repeated* if the aircraft was following a privileged route; this distinction is illustrated in Fig. 3 as solid and dashed lines, respectively. We define an *experience* as a contiguous collection of vertices linked by temporal edges. Mapping

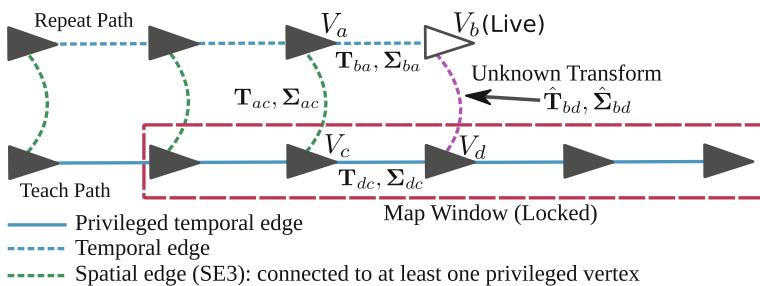


Fig. 3 Overview of the localisation problem and the spatio-temporal pose graph (STPG) data structure used as our map. We wish to estimate the unknown transform and uncertainty, $\{\hat{\mathbf{T}}_{bd}, \hat{\Sigma}_{bd}\}$ (dashed, purple line), between the live vertex, V_b , and the target vertex, V_d , in the privileged path (solid blue line). This is achieved by matching all landmarks in V_b to landmarks observed in the map window (dashed, red rectangle), transformed into the coordinate frame of V_d . This setup allows for outlier rejection and a simple optimisation of $\{\hat{\mathbf{T}}_{bd}, \hat{\Sigma}_{bd}\}$ against a map of locked landmarks with uncertainty



(a) High-rate: frame-to-vertex VO. Landmark estimates (black stars) at the latest vertex are locked, and the motion estimate is solved using new matches from the live frame (purple triangle) to the last vertex (black triangle) while using a smoothing trajectory prior (black dot).

(b) Low-rate: sliding-window vertex bundle adjustment. Transforms and landmarks connected to unlocked vertices (white triangles) are optimised and those connected to locked vertices (black triangles) are locked.

Fig. 4 VO pipeline showing the parallel high-rate, approximate **a** and low-rate, accurate **b** estimators similar to [17]

consists of adding either a privileged or autonomous experience to a new or existing STPG while computing data products and temporal edge transformations through a monocular VO pipeline, which is illustrated in Fig. 4. For each incoming frame (the *live* frame), sparse visual features are extracted and their descriptors computed. Features are represented by a single measurement, $\{y, Y\}$, where y is the 2×1 keypoint position of the feature and Y is the 2×2 covariance on the measurement. We use oriented Speeded-Up Robust Features (SURF) [4] to detect and describe keypoints and calculate Y based on the octave and Hessian of the response. When two measurements are matched through their descriptors, the 3D landmark is triangulated via the inverse camera model and the relative transform between the cameras to obtain a 3D landmark including uncertainty, $\{p, \Phi\}$, where p is the 4×1 positional mean in homogeneous coordinates and Φ is the uncertainty represented by a 3×3 covariance.

3.2 Visual Odometry

To initialise the VO, features and descriptors are extracted from the first image, and then matched against those from subsequent frames. Both an Essential matrix and 2D Homography matrix are computed using MLESAC [28] to extract a relative transformation for each new frame, subject to a Geometric Robust Information Criterion (GRIC) test [27] to select the best estimate. Once the inlier count for each frame-to-frame transformation drops below a threshold (as an analogue for translational motion), landmarks are triangulated (subject to a re-projection and plane-distance test to eliminate gross outliers) and the pair of frames placed as the first two vertices in the graph, with the computed transformation inserted as the edge. To initialise the scale appropriately, a ground plane is fitted to the triangulated landmarks from which the height from the scene is extracted, then the true height from the ground is retrieved from a GPS position at a similar time-point to find the scaling parameter. This is then applied to the transformation and landmarks. In practice, any approximate scaling data can be used, such as from a calibrated barometer, laser altimeter,

or other suitable sensor. Perfect scaling is not crucial to the function of VT&R, drift of global estimates is easily handled.

For subsequent frames, extracted features from the live view are matched via their appearance to locked landmarks in the latest graph vertex (a.k.a., keyframe) and motion computed (Fig. 4a) by solving the Perspective-Three-Point (PnP) problem [13], again using MLESAC. New landmarks are triangulated from new matches that are not associated with an existing landmark, subject to the same plane-distance and re-projection tests to remove outliers. A trajectory (velocity and position) estimate is produced at frame rate from the optimisation and can be queried to predict future motion. This prediction is used to project landmarks into the new frame (reducing image search space for matching) and compensates for latency between the localisation system and the path-tracking controller. If the translational or rotational motion is large, or the number of matched features between the live view and the last graph vertex drops too low, the live frame is inserted as a new vertex in the graph; otherwise, it is discarded. Upon insertion of a new vertex, a temporal edge linking to the previous vertex is added. If the aircraft is in GPS-based teach mode, this edge is flagged as privileged. Following vertex insertion, bundle adjustment is performed on a sliding window of the latest vertices in the graph (Fig. 4b) using our Simultaneous Trajectory Estimation And Mapping (STEAM) [2] engine; smoothing factors are added to the relative transforms to ensure stability in the estimated trajectory during areas of poor feature tracks. After optimisation, the updated poses, landmarks, and their uncertainties are re-inserted into the graph.

3.3 Localisation

When repeating a path, the overall objective of the algorithm is to estimate the posterior transform and uncertainty, $\{\hat{\mathbf{T}}_{bd}, \hat{\Sigma}_{bd}\}$, between the most recent vertex in the live run, V_b , and the estimated closest vertex in the privileged path, V_d . This is achieved by minimising the measurement error of landmarks in the map window (red, dashed rectangle in Fig. 3) observed by V_b . Throughout the algorithm, we make use of the prior term, $\{\check{\mathbf{T}}_{bd}, \check{\Sigma}_{bd}\}$, obtained by compounding the uncertain transforms [3],

$$\{\mathbf{T}_{ba}, \Sigma_{ba}\}, \{\mathbf{T}_{ac}, \Sigma_{ac}\}, \{\mathbf{T}_{cd}, \Sigma_{cd}\}, \quad (1)$$

which are computed through previous VO and previous localisation estimates. The localisation pipeline consists of the following main steps: (a) Landmark Transformation, (b) Localisation Matching, and (c) State Estimation.

3.3.1 Landmark Transformation

The first step of localisation is to transform all landmark means and uncertainties in the active map window from their respective coordinate frames in each vertex to \mathcal{F}_d , the coordinate frame of V_d and the one in which localisation is to be computed. We use the same process as Paton et al. [24] to transform landmarks expressed in a nearby vertex map frame, \mathcal{F}_m , with mean and covariance, $\{\mathbf{p}_m, \Phi_m\}$, to \mathcal{F}_d , giving $\{\mathbf{p}_d, \Phi_d\}$, ensuring uncertainty is appropriately transformed along with the landmark co-ordinates. This process is carried out on all landmarks in the map window to produce a set of landmarks with 3D position and uncertainty, all expressed in the privileged frame, \mathcal{F}_d . The locations and uncertainties of all landmarks are transformed, even if they are not matched, as these help refine the matching process, making it faster and more robust.

3.3.2 Localisation Matching

The goal of localisation matching is to associate every *feature* observed by V_b to a landmark in the map window, even if the feature is not associated with a landmark in V_b . The process begins with labeling all features in the live vertex as unmatched. Vertices in the map window are sequentially examined starting from V_d in an outward search pattern. We choose to center the search around the privileged target vertex as a heuristic for prioritising landmarks that have the lowest uncertainty in the target privileged frame. For every new vertex visited, the transformed map landmarks associated with this vertex are projected into the camera frame of vertex V_b using the prior term, $\{\check{\mathbf{T}}_{bd}, \check{\Sigma}_{bd}\}$. Each feature associated with this vertex is then checked for matching feasibility to the unmatched live features by comparing keypoint position and descriptor appearance. This process continues until one of three criteria are met: (i) a sufficient number of matches are found, (ii) the amount of time has surpassed an allowed limit, or (iii) the map window of vertices is exhausted. As the process of comparing visual features is costly, this process is the most computationally expensive step of localisation, but it is performed in parallel to the main VO pipeline for each new vertex on the Graphics Processing Unit (GPU), meaning online operation is possible. Upon completion of localisation matching, the problem is set up so that there are candidate features in V_b associated with landmarks in V_d . This information is sent through a MLESAC PnP estimator to initialise the relative transform between V_b and V_d (as this may be significantly different from the prior) and remove outliers.

3.3.3 State Estimation

We now seek the optimal posterior,

$$\{\hat{\mathbf{T}}_{bd}, \hat{\boldsymbol{\Sigma}}_{bd}\}, \quad (2)$$

given the prior term, $\{\check{\mathbf{T}}_{bd}, \check{\boldsymbol{\Sigma}}_{bd}\}$, as well as associated data between V_b and map landmarks in the coordinate frame of V_d . This can be achieved by minimising the negative-log-likelihood cost function:

$$J(\mathbf{T}_{bd}) = \frac{1}{2} \sum_{j=1}^M \mathbf{e}_j^T \mathbf{R}_j^{-1} \mathbf{e}_j + \frac{1}{2} \mathbf{e}^T \mathbf{R}^{-1} \mathbf{e}, \quad (3)$$

with the first term in J summing the squared reprojection error of map landmarks and the second term encoding the transform prior. Given a map landmark, j , with mean and uncertainty, $\{\mathbf{p}_{d,j}, \boldsymbol{\Phi}_{d,j}\}$, expressed in the co-ordinate frame of V_d and a monocular measurement of j , \mathbf{y}_j , with uncertainty, \mathbf{Y}_j , expressed in the camera frame of V_b , the reprojection error is defined by

$$\mathbf{e}_j = \mathbf{y}_j - \mathbf{g}(\mathbf{T}_{bd} \mathbf{p}_{d,j}), \quad (4)$$

$$\mathbf{R}_j = \mathbf{Y}_j + \mathbf{G}_j \mathbf{T}_{bd} \mathbf{D} \boldsymbol{\Phi}_{d,j} \mathbf{D}^T \mathbf{T}_{bd}^T \mathbf{G}_j^T, \quad (5)$$

where $\mathbf{g}(\cdot)$ is the monocular measurement model and \mathbf{G}_j is its Jacobian (evaluated at $\mathbf{p}_{b,j} = \mathbf{T}_{bd} \mathbf{p}_{d,j}$), with

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}. \quad (6)$$

This weights each error by uncertainty in the measurement and the map. The second term of Eq. (3) constrains the optimisation problem by the prior with

$$\mathbf{e} = \ln(\check{\mathbf{T}}_{bd} \mathbf{T}_{bd}^{-1})^\vee, \quad \mathbf{R} = \check{\boldsymbol{\Sigma}}_{bd}, \quad (7)$$

where \vee is the inverse operator of \wedge [3]. To obtain an optimal posterior estimate, $\hat{\mathbf{T}}_{bd}$ is iteratively refined in a nonlinear least-squares optimisation using our STEAM engine [2]. In the absence of any matches between the live image and map, the prior estimate (based on VO) is returned.

4 Experimental Setup

To evaluate the performance of the modified VT&R localisation in this new application, a series of experiments were conducted offline using a monocular dataset collected using a PrecisionHawk Lancaster Rev IV (Fig. 1). This aircraft is the target system for the developed algorithms, with a take-off weight of ~ 2.5 kg and wingspan of 1.5 m. The typical flight time is 30–45 min. This UAV was fitted with a custom payload consisting of a single Point Grey Chameleon machine-vision camera, configured to face straight down from the payload bay. This camera uses a $1/2''$ global shutter CMOS sensor, with an approximately $90^\circ \times 80^\circ$ Field of View (FOV). Bayer-encoded imagery is captured at ~ 22 Hz and 1280×1024 pixel resolution (converted to grayscale and down-sampled to 640×512 for this experiment). Imagery is recorded using an on-board, 1.6 Ghz Intel Atom PicoITX computer along with GPS data at 5 Hz from an on-board Ublox LEA-6N receiver. The data was gathered at a disused open-pit gravel mine in Sudbury, in central Canada, during early summer. This site consists of dirt roads, both undisturbed and naturally reforested boreal forest and exposed regolith from prior mining operations. The dataset consists of multiple flights using the custom payload, covering a square box pattern (Fig. 5) with segments approximately 400 m in length. This pattern is flown in sequence, multiple times per flight. Each flight is approximately 15–25 min in duration, with 1–7 repeats per flight, and these are split into individual ‘experiences’ that cover a full loop of the flown square pattern. A selection of these are used in the experiments for this paper, shown in Table 1, chosen to cover a variety of test scenarios (other flights were for different test configurations).

We test the performance of VT&R localisation on the airborne data by experimenting with these selected sets of flights, focusing on increasing time differences between the initial teach pass and repeat to evaluate the performance of the system under appearance change and compare to our well established ground-based system. For each experiment, performance is evaluated by examining both relative uncertainty of the UAV and inlier matches during localisation in the repeat phase. The

Fig. 5 The configuration of the flight path. Start and end of route at bottom left corner. Note forested areas in top-left and right of image



Table 1 Overview of the selected experiences in the Sudbury dataset

ID	Flight	Start time condition	Conditions
e0	2	14/6/2016 12:53	sunny, calm
e4	2	14/6/2016 13:05	sunny, calm
e5	4	14/6/2016 14:57	sunny, calm
e11	4	14/6/2016 15:14	sunny, calm
e12	8	14/6/2016 17:54	sunny, calm
e18	8	14/6/2016 18:07	sunny, calm
e19	10	15/6/2016 12:07	sunny, windy
e24	10	15/6/2016 12:19	sunny, windy
e25	15	16/6/2016 12:02	sunny, windy

Table 2 Overview of the configurations used for localisation experiments

ID	Live experience	Privileged experience	ΔT teach to repeat hh:mm (24 hr hh:mm)
g0	e4	e0	0:09
g1	e24	e19	0:12
g2	e18	e12	0:13
g3	e11	e5	0:17
g4	e25	e19	23:51 (-0:09)
g5	e19	e0	23:17 (-0:43)
g6	e25	e0	47:04 (-0:54)
g7	e5	e0	2:07
g8	e12	e0	5:04

makeup and included experiences in each experiment are listed in Table 2. These experiments can be grouped into three general categories: (1) same-flight repeats (g0–g3), (2) temporally close repeats (g4–g6), and (3) temporally distant repeats (g7–g8).

Experiments g0–g3 include teach and repeat from the same flight (the first pass of the pattern to the last). In all these experiments, the time difference between teach and repeat is less than 17 min. Experiments g4–g6 include teach and repeat from flights that are temporally close, but different days. Experiment g4 includes a repeat approximately 24 h after the teach, but with only a 9-min time-of-day difference. Experiments g5 and g6 are one and two days after the teach, but 43 and 54 min temporally distant from the teach. Finally, experiments g7–g8 are conducted on the same day, but approximately 2 and 5 h after the teach. We use both grayscale and colour-constant imagery [23] in all experiments to ensure the best performance in VO and localisation.

By examining the performance of VT&R localisation in this way, we can establish the temporal limitations on safe and accurate repeats for emergency returns,

and make comparisons to the performance of VT&R localisation in the more traditional ground-vehicle environment. We use the localisation uncertainty as the primary metric for judging localisation success, and define it as the one-standard-deviation uncertainty of our 3D translation estimate relative to the privileged path. This tells us how uncertain we are of the distance of the vehicle to the privileged path. This is plotted as a Cumulative Distribution Function (CDF), where better performance is indicated by lower uncertainties over a greater percentage of the path. It is important to note that while uncertainty is calculated at every stage of the algorithm from key-point detection to landmark transformation, we have not yet performed a rigorous evaluation of our uncertainty estimates with respect to ground truth to ensure consistency. Therefore we treat this metric as a way to compare relative performance between experiments and do not necessarily trust the exact scale of our uncertainty estimates.

We also include the inlier count for each localisation on the repeat path, and use a count of 15 inliers as the minimum number to constitute a successful localisation. Fewer than 15 inliers generally indicates either a poor or degenerate estimate. We plot this as inlier count vs. time since repeat start, grouped into three figures corresponding to the experiment type described above, to highlight the reliability of localisation over the course of each flight. Each experience is from 120 to 170 s long, and we normalise the inlier count vs. time results to 170 s to improve the consistency of comparison when discussing sections that cover the same area.

5 Results

Results are presented in Figs. 6 and 7. In Fig. 6, it can be seen that short time differences (< 20 min) mean the probability of successful localisation is high (Table 3). This corresponds well with our intended application of emergency return, where the repeat would typically be conducted in the same flight as the teach. Since the aircraft's typical flight time is 30–45 min, these results indicate that return within a single flight is feasible and reliable. With increasing temporal difference, however, the average uncertainty grows rapidly. Performance rapidly drops after 45 min difference (with a total time difference of 24 h). At 2 h (g7) and 5 h (g8) difference from teach to repeat, uncertainty is high and localisation performance is significantly degraded (Table 3).

These results are corroborated by plots of localisation inliers over time (Fig. 7). For experiments g0–g3 (Fig. 7a), localisation performance is strong, with the number of inliers at each localisation step approximated by the number (~ 500) of migrated points. Fig. 7b, c highlight the reduced performance of later repeats. The algorithm uses 1GB RAM and runs at $\sim 25\text{Hz}$ on an NVIDIA Tegra TX2, which exceeds the framerate used in the dataset of $\sim 22\text{Hz}$.

Fig. 6 The CDF of translational uncertainty for each experiment, in increasing temporal time difference. g0–g3 (solid lines) are within the same flight, g4–g6 (dashed lines) are different days but within 9–54 min of the teach and g7–g8 (dash-dot lines) are large temporal differences (2 and 5 h respectively)

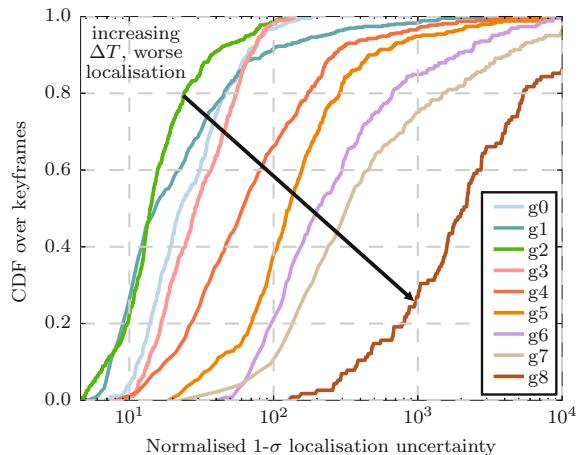


Table 3 Localisation performance for each experiment. Success is an inlier count >15

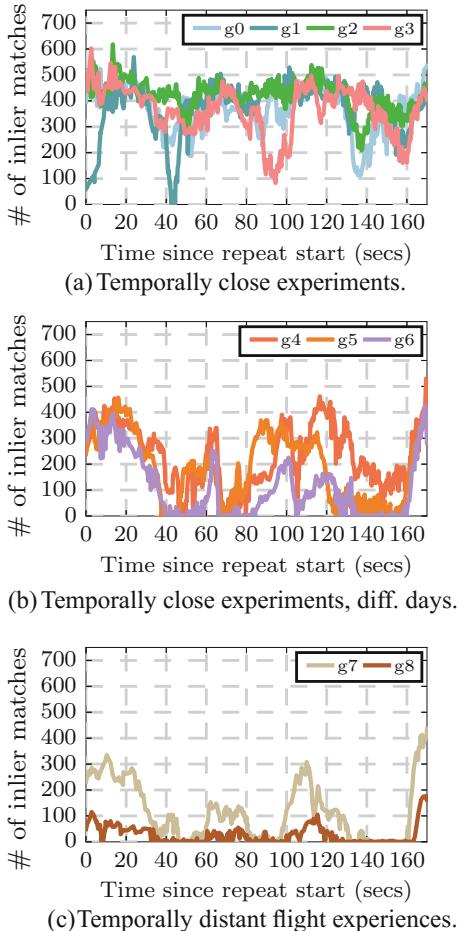
ID	Localisation keyframes	Successful localisations	%
g0	880	880	100.0
g1	938	932	99.4
g2	824	824	100.0
g3	832	832	100.0
g4	999	978	97.9
g5	797	701	88.0
g6	775	530	68.4
g7	664	443	66.7
g8	526	230	43.7

6 Discussion, Challenges and Future Work

These initial results show the concept of a vision-based UAV emergency navigation system is feasible using VT&R as a basis. This first prototype has generated valuable lessons and highlighted some significant challenges for future research, as highlighted below.

First, the localization performance (Fig. 6) when repeating over a path deteriorates an order of magnitude faster than experienced on ground vehicles [23]. Results from Paton et al. showed that with colour-constant imagery, strong performance in localisation during repeat was possible 7 h after the initial teach. In the airborne case, 2 h saw significant loss of accuracy and reliability. This is due to a number of factors: (1) A reduced perspective constraint due to unrestricted paths. When localising, minimising the perspective change from the live to the map view enhances reliability

Fig. 7 Inlier counts for each keyframe during the repeat phase for the three grouped sets of experiments: **a** g0–g3, **b** g4–g6, **c** g7–g8



in matching descriptor-based features. For ground vehicles, the restricted paths and locally 2D surface makes this an easier task. In the air, an un-closed control loop, unrestricted paths and 6-DoF motion mean perspective can significantly change from teach to repeat. (2) Significant shadow dependent appearance change. In the airborne perspective, particularly over forests (which have high depth variation), shadows move rapidly and the ‘opposition effect’ (bright halos seen around an object’s shadow when illuminated directly from behind) means that features are generally tracked for shorter distances.

This latter potential cause is highlighted by certain segments of the flight path. During the first and last segments of flight (seconds 0–30 and 170–180), the imagery consists of grass and exposed regolith, which tends to show better invariance to appearance change than areas that cover forest (seconds 30–60, and 140–170). The appearance change of these areas for different experiments are highlighted in Fig. 8.

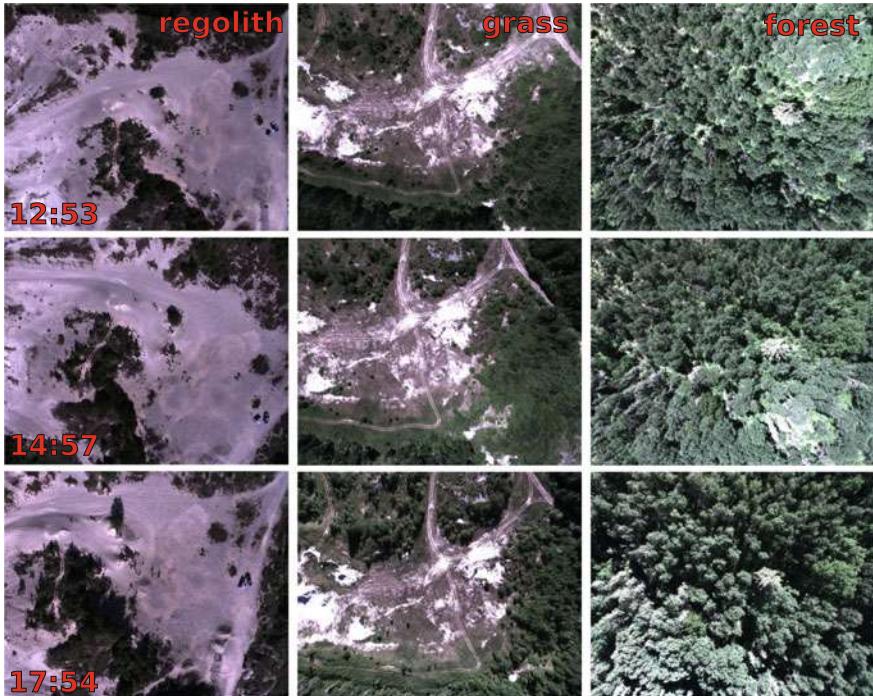


Fig. 8 Sample images from differing regions of the dataset during experiment g0 (top row), experiment g8 (centre row) and experiment g9 (bottom row). Exposed regolith (left column, approx. 70 s since start of repeat) and grass (centre column, 20 s since start of repeat) are significantly more robust to appearance change than forest (right column, 40 s since start of repeat). Note change in shadow and movement of halo due to the opposition effect in the right-hand column (identify the tall tree in centre right of image to assist in recognising scene)

Contributing to the significant appearance seen in forests are the rapidly moving shadows generated in small regions between tall trees, and less robust descriptors generated from typically smaller octaves due to significant fine detail.

Apart from these application specific challenges, the general use of feature descriptors presents the same limitations as that for ground vehicles: rapid appearance change due to cloud shadowing or featureless environments will reduce localisation performance. We have addressed these challenges through the development of colour-constant imagery [23] and Multi-Experience Localisation (MEL) [24] (see below). However, the airborne case is less strictly reliant on continuous localisation as the return trajectory does not need to be strictly the same. We expect long periods of dead-reckoning where the aforementioned factors cause localisation failure, and during the first stage of an emergency return (the turn-around). Our tests show a VO translational error of approximately 1% over the trajectories tested, in line with current state-of-the-art. We have tested with other feature types such as Oriented FAST and Rotated BRIEF (ORB), but have seen similar localisation performance.

The second major lesson is reflected in the shortage of results that leverage multiple experiences, as demonstrated for ground vehicles in [24], which is a current major focus for our lab. While not strictly required for emergency return, a multi-experience framework would remain useful in applications that require repeat trajectories (such as deliveries) in GPS-denied or GPS-intermittent areas. Given rapid appearance change in the airborne case, the need for accurately timed bridging experiences is critical, and effectively requires continuous flights with 10–15 min delay in order to generate enough inlier matches to successfully estimate pose relative to the original path. This is logically difficult to implement feasibly, so such investigations are left to future work.

Since the primary application is emergency return, the algorithm will be further developed in conjunction with a path-tracking controller suited to guidance of a fixed-wing aircraft. This will build on previous work for ground vehicles [22]. To improve localisation performance during large temporal differences, we are exploring techniques to learn place-specific binary descriptors that are more invariant to appearance change and localisation on data captured from differing sensors.

7 Conclusions

This paper presented the application of a monocular VT&R localisation engine on an outdoor, fixed-wing UAV. A key contribution is the demonstration of localisation using only a single camera in a configuration as-yet untested outdoors. Through an analysis of localisation performance and estimated uncertainty, we have shown that our algorithm is able to provide metric localisation to a privileged experience during a single flight within the capabilities of the PrecisionHawk Lancaster, such that it can be used for emergency return. Performance was also evaluated with increasing temporal difference, showing the current limitations of the algorithm given significant and rapid appearance change.

Acknowledgements Thanks to PrecisionHawk, MITACS, and the NCFRN for project funding, Ethier Sand and Gravel for property access to gather data, and Haowei Zhang for logistical support and data processing.

References

1. Achtelik, M.W., Achtelik, M.C., Weiss, S.M., Siegwart, R.: Onboard IMU and monocular vision based control for MAVs in Unknown in- and outdoor environments. In: International Conference on Robotics and Automation (ICRA), pp. 3056–3063. IEEE, May 2011
2. Anderson, S., Barfoot, T.D.: Full STEAM ahead: Exactly sparse gaussian process regression for batch continuous-time trajectory estimation on $SE(3)$. In: Intelligent Robots and Systems (IROS), pp. 157–164, Sept 2015
3. Barfoot, Timothy D., Furgale, Paul T.: Associating uncertainty with three-dimensional poses for use in estimation problems. *IEEE Trans. Robot.* **30**(3), 679–693 (2014)

4. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: speeded up robust features. In: European Conference on Computer Vision (2006)
5. Bry, A., Bachrach, A., Roy, N.: State estimation for aggressive flight in GPS-denied environments using onboard sensing. In: International Conference on Robotics and Automation (ICRA). IEEE (2012)
6. Caballero, F., Merino, L., Ferruz, J., Ollero, A.: Vision-based odometry and SLAM for medium and high altitude flying UAVs. *J. Intell. Robot. Syst.* **54**(1–3), 137–161 (2008)
7. Clement, L.E., Kelly, J., Barfoot, T.D.: Monocular visual teach and repeat aided by local ground planarity. In: Wettergreen, D., Barfoot, T.D. (eds.) Field and Service Robotics, chapter VI, pp. 547–561. Springer International Publishing, Toronto (2015)
8. Dillingham, G.L.: Unmanned aircraft systems: continued coordination, operational data, and performance standards needed to guide research and development. Technical report, U.S. Government Accountability Office (2013)
9. Fang, Zheng, Yang, Shichao, Jain, Sezal, Dubey, Geetesh, Maeta, Silvio: Robust autonomous flight in constrained and visually degraded environments. In: Wettergreen, D., Barfoot, T.D. (eds.) Field and Service Robotics. chapter IV, pp. 411–425. Springer International Publishing, Toronto (2015)
10. Forster, C., Faessler, M., Fontana, F., Werlberger, M., Scaramuzza, D.: Continuous on-board monocular-vision based elevation mapping applied to autonomous landing of micro aerial vehicles. In: International Conference on Robotics and Automation (ICRA), Seattle. IEEE (2015)
11. Forster, C., Pizzoli, M., Scaramuzza, D.: SVO: fast semi-direct monocular visual odometry. In: International Conference on Robotics and Automation (ICRA). IEEE (2014)
12. Furgale, P., Barfoot, T. D.: Visual teach and repeat for long-range rover autonomy. *J. Field Robot.* **27**(5), 534–560 (2010)
13. Gao, XS., Hou X.R, Tang, J., Cheng, H.F.: Complete solution classification for the perspective-three-point problem. *IEEE Trans Pattern Anal. Mach. Intell.* **25**(8), 930–943 (2003)
14. Goedemé, T., Nuttin, M., Tuytelaars, T., Van Gool, L.: Omnidirectional vision based topological navigation. *Int. J. Comput. Vis.* **74**(3), 219–236 (2007)
15. Transportation Infrastructure: Vulnerability assessment of the transportation infrastructure relying on the global positioning system. Technical Report, Center, John A. Volpe National Transportation Systems (2001)
16. Kelly, J., Sukhatme, G.S.: An Experimental Study of Aerial Stereo Visual Odometry. In: Symposium on Intelligent Autonomous Vehicles, pp. 1–6 (2007)
17. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 1–10. IEEE, Nov 2007
18. Majdik, L., Albers-schoenberg, Y., Scaramuzza, D.: MAV urban localization from google street view data. In: Intelligent Robots and Systems (IROS), pp. 3979–3986. IEEE (2013)
19. McManus, C., Furgale, P., Barfoot, T.D.: Towards appearance-based methods for LiDAR sensors. In: International Conference on Robotics and Automation (ICRA), pp. 1930–1935. IEEE (2011)
20. Oettershagen, P., Stastny, T., Mantel, T., Melzer, A., Gohl, P., Agamennoni, G., Alexis, K., Siegwart, R.: Long-endurance sensing and mapping using a hand-launchable solar-powered UAV. In: Wettergreen, D., Barfoot, T.D. (eds.) Field and Service Robotics. chapter IV, pp. 441–454. Springer International Publishing, Toronto (2015)
21. Ostafew, C.J., Schoellig, A.P., Barfoot, T.D.: VIusal Teach And Repeat, Repeat, Repeat: Iterative Learning Control To Improve Mobile Robot Path Tracking In Challenging Outdoor Environments. In: Intelligent Robots and Systems (IROS), pp. 176–181. IEEE (2013)
22. Ostafew, C.J., Schoellig, A.P., Barfoot, T.D.: Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments. In: International Conference on Robotics and Automation (ICRA), pp. 4029–4036. IEEE (2014)
23. Paton, M., Mactavish, K., Ostafew, C.J., Barfoot, T.D.: It's not easy seeing green: lighting-resistant stereo visual teach & repeat using color-constant images. In: International Conference on Robotics and Automation (ICRA). IEEE (2015)

24. Paton, M., Mactavish, K., Warren, M., Barfoot, T.D.: Bridging the appearance gap: multi-experience localization for long-term visual teach and repeat. In: Intelligent Robots and Systems (IROS) (2016)
25. Pfrunder, A., Schoellig, A.P., Barfoot, T.D.: A proof-of-concept demonstration of visual teach and repeat on a quadrocopter using an altitude sensor and a monocular camera. In: Conference on Computer and Robot Vision (CRV), pp. 238–245 (2014)
26. Shen, S., Michael, N., Kumar, V.: Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft MAVs. In: International Conference on Robotics and Automation (ICRA), pp. 5303–5310, Seattle, IEEE (2015)
27. Torr, P.: An assessment of information criteria for motion model selection. In: Computer Vision and Pattern Recognition, San Juan, pp. 47–52. IEEE (1997)
28. Torr, P.: MLESAC: a new robust estimator with application to estimating image geometry. *Comput. Vis. Image Underst.* **78**(1), 138–156 (2000)
29. Weiss, S.M., Scaramuzza, D., Siegwart, R.: Monocular SLAM Based Navigation for Autonomous Micro Helicopters in GPS Denied Environments. *J. Field Robot.* **28**(6), 854–874 (2011)

Local Path Optimizer for an Autonomous Truck in a Harbor Scenario

Jennifer David, Rafael Valencia, Roland Philippson and Karl Iagnemma

Abstract Recently, functional gradient algorithms like CHOMP have been very successful in producing locally optimal motion plans for articulated robots. In this paper, we have adapted CHOMP to work with a non-holonomic vehicle such as an autonomous truck with a single trailer and a differential drive robot. An extended CHOMP with rolling constraints have been implemented on both of these setup which yielded feasible curvatures. This paper details the experimental integration of the extended CHOMP motion planner with the sensor fusion and control system of an autonomous Volvo FH-16 truck. It also explains the experiments conducted on the differential-drive robot. Initial experimental investigations and results conducted in a real-world environment show that CHOMP can produce smooth and collision-free trajectories for mobile robots and vehicles as well. In conclusion, this paper discusses the feasibility of employing CHOMP to mobile robots.

1 Introduction

The motivation of this work comes from the Cargo-ANTS EU project¹ which aimed to create smart Automated Guided Vehicle (AGVs) and Autonomous Trucks (ATs) that can cooperate in shared workspace for efficient and safe freight transportation

J. David (✉) · R. Philippson
Intelligent Systems Lab, Halmstad University, Halmstad, Sweden
e-mail: jendav@hh.se

R. Philippson
e-mail: roland.philippson@gmx.net

R. Valencia
Robotics Institute, Carnegie Mellon University, Pittsburgh, USA
e-mail: rvalenci@andrew.cmu.edu

K. Iagnemma
Robotics Mobility Group, Massachusetts Institute of Technology, Cambridge, USA
e-mail: kdi@mit.edu

¹www.cargo-ants.eu.

in main ports and freight terminals [13]. The whole context of the project invokes research questions on multi-robot/vehicle path planning of AGVs and ATs in a constrained outdoor environment like the container terminal. Though there are different approaches available for solving multi-robot/vehicle planing like centralized and decentralized methods, a dedicated navigational framework called SPADES was proposed in [5].

SPADES (Simultaneous Planning and Assignment in Dynamic Environments) framework (Fig. 1) is based on the multi-robot task allocation methodology similar to [8]. It consists of a global task assignment and a global path planner that runs on the central station of the harbour and a local interconnected path adaptor running on individual vehicles. The global task assignment module assigns tasks to each of the vehicles based on the classical Hungarian algorithm [6] which minimizes the overall path length of all the vehicles. Thus, under a static environment, an optimal task (e.g. A vehicle picks N container, B vehicle picks O container, etc.) is given to each of the vehicles from the set of its all possible tasks (A vehicle can pick N or O and B vehicle can pick N or O , etc.) The minimization criteria is the path length of each of the vehicle, which is obtained from the global path planner module. This module finds all the possible paths between the vehicle-container-goal combinations. The cost of each combination is a simple navigation function that makes the overall assignment and planning faster. The resulting trajectories are later continuously refined by a local path adaptor to avoid collisions with dynamic obstacles or to avoid conflicts with other paths. When encountered with dynamic obstacles, the optimal paths assigned to all the vehicles can loose their optimality due to the path refinement by the local path adaptor. However, it always produces near-optimal solutions. This is because the conflicts are resolved locally by taking into account the trajectories of other vehicles.

In this paper, we detail on the lower navigational framework of SPADES framework by detailing on the adaptation of the local path adaptor for a non-holonomic vehicle, its implementation details on the Volvo FH-16 truck and the real-world experiments conducted on it. We also compare the results with a turtlebot2 differential drive robot to analyze the extent of this approach. The next section details on the recent works related to local obstacle avoidance techniques used for multi-robot path planning. Section 3 details the local path adaptor and the approach used for adapting it to a mobile robot. The implementation details are explained in Sect. 5 followed by the vehicle software architecture. The results and discussions are explained in the final Sects. 6 and 7 with concluding remarks at Sect. 8.

2 Related Work

There is a large body of literature in motion planning for multiple robots in general and intelligent vehicles in particular [7]. For local path adaptation, many of the simple obstacle avoidance techniques [12] could be used. One of the well-known reactive techniques like Reciprocal Velocity Obstacle (RVO) approach [15] is suitable for multi-vehicle scenario. This approach extends the older Velocity Obstacle concept

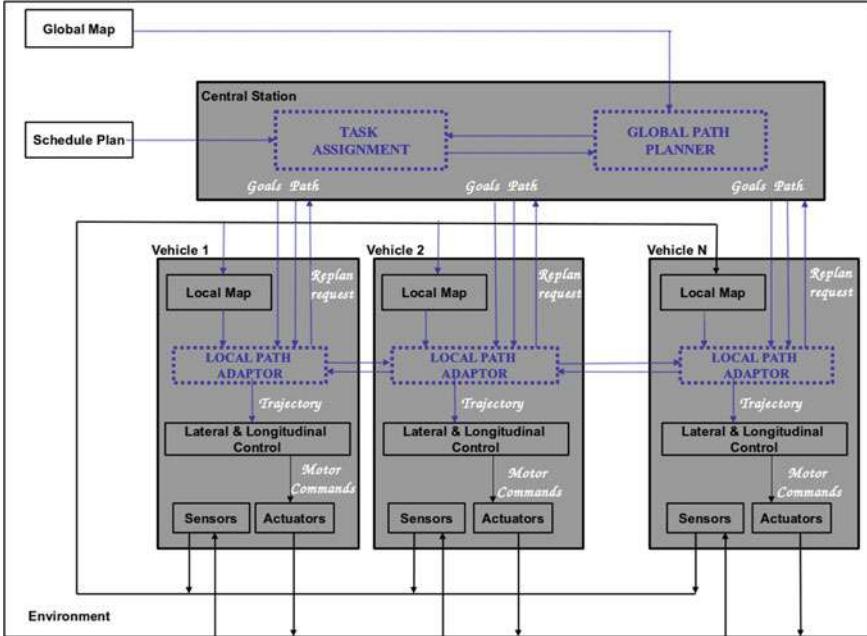


Fig. 1 SPADES architecture - Simultaneous Planning and Assignment in Dynamic EnvironmentS

[10] to appropriately address situations where interacting ATs can be assumed or designed to share the responsibility for avoiding collisions with each other. There are also recent generalizations to RVO which are able to handle significant kinematic and kino-dynamic constraints [1]. It should be noted that this type of reactive avoidance technique is more beneficial for on-board execution on each vehicle, as it increases the safety in unforeseen situations. However, this could break down at resource bottlenecks, for instance if there is a single gate through which all traffic must flow. This also lowers the throughput of the system due to non-optimal solutions.

The use of gradient based path optimization approaches have been very successful for articulated robots. They are fast and produce locally optimal solutions. For the SPADES framework, there is a need for a very fast reactive approach to be coupled with the task assignment. This is because in case of a very tight local path conflict situation, there is a need for a re-planning from the global path planner from the central station. Hence, in this paper, we seek to adopt the CHOMP algorithm by [9] as the local path adaptation algorithm for SPADES. It is used for optimizing the path generated by the global path planner, in this case, the E* algorithm [11] for a group of autonomous vehicles depending on the local obstacles. In this paper, the CHOMP algorithm has been extended to mobile robot/vehicle by incorporating simple constraints on the optimizer. In Sect. 3, a brief explanation on the concept of CHOMP algorithm and its extension is elaborated.

3 Chomp Preliminaries

CHOMP stands for Co-variant Hamiltonian Optimization for Motion Planning by [9]. It is a trajectory optimization technique used for motion planning in high-dimensional spaces. It produces near-optimal solutions without violating dynamical constraints. CHOMP iteratively improves the quality of an initial trajectory by optimizing an objective functional that trades off between desired qualities such as smoothness and obstacles avoidance. An additional merit of CHOMP is inherent invariance to the choice of trajectory parametrization used. Given a trajectory $\xi : [0, 1] \rightarrow \mathcal{C}$, i.e. a function mapping time to robot configurations, CHOMP optimizes a functional $U : \xi \rightarrow R$, which maps each trajectory ξ in the space of trajectories ξ to a real number. In the original formulation of CHOMP, an objective given as follows was optimized:

$$U[\xi] = \mathcal{F}_{obs}[\xi] + \lambda \mathcal{F}_{smooth}[\xi], \quad (1)$$

where the term \mathcal{F}_{smooth} penalizes the trajectories based on dynamical criteria such as velocities and accelerations to encourage smooth trajectories. At the same time, the term \mathcal{F}_{obs} penalizes proximity to objects in the environment to encourage trajectories that clear obstacles. In order to integrate CHOMP within SPADES architecture for a fleet of AGVs, it has to be adapted to include the non-holonomic constraints of the robot/vehicle. In this section, we elaborate the method to incorporate these constraints in CHOMP to be used with such robot/vehicles.

The main goal is to incorporate curvature constraints into a trajectory optimizer such as CHOMP. The problem, however, is that there may be no feasible curvature-constrained trajectory in the basin of attraction of the initial guess trajectory, particularly in our system where the initial guess currently involves significant simplifications. We see two options: either ensure that the initial guess trajectory is always good, or employ an optimizer that can switch basins of attraction. This implied system-wide trade-offs are beyond the scope of this paper and part of ongoing research.

In order to solve this, different methods were tried out without changing the nature of CHOMP, which are elaborated in [2]. In this paper, a constrained CHOMP approach as mentioned by [9] was derived, thus, generating feasible trajectories for non-holonomic vehicles. For the Cargo-ANTs project, we assume that the globally planned path which is used to initialize CHOMP lies close enough to a trajectory that respects curvature constraints and that a soft curvature-maximization objective function suffices to keep the optimizer from violating curvature constraints. In practice, this approach works well in the tested settings with less-cluttered scenarios.

4 Non-holonomic Constraints

CHOMP algorithm makes use of Lagrange multipliers to include additional constraints in its original form by changing the update rule. Similarly, we add the kinematic constraints to CHOMP, here, by explicitly including them in the optimization step via the method of Lagrange multipliers. In order to avoid this process to hamper real-time performance, we opted to instead include only rolling and forward motion constraints by this mean. To this end, we formulate them as equality constraints as we show next.² By doing so, it increases the smoothness to a point where the curvature constraints are respected.

Thus, we first formulate the rolling constraint and the forward motion constraints separately here. The rolling constraint expresses the fact that the midpoint in the rear axle has to move in a direction normal to the rear wheels axle, that is,

$$x' \sin(\theta) - y' \cos(\theta) = 0, \quad (2)$$

where x' and y' are the global linear velocities of the vehicle along the x and y axis and θ is the vehicle's orientation.

Thus, for a trajectory of $n + 1$ discrete vehicle's configurations $q_i = (x_i, y_i, \theta_i)^\top$, our rolling constraint functional can be expressed as follows

$$\mathcal{H}_{roll}(\xi) = \sum_{t=0}^{n-1} \frac{x_{t+1} - x_t}{\Delta t} \sin(\theta_t) - \frac{y_{t+1} - y_t}{\Delta t} \cos(\theta_t). \quad (3)$$

Next, to enforce forward motions, we can constraint the forward velocity in the local vehicle's frame to always be positive, that is,

$$\|x'_R\| - x'_R = 0, \quad (4)$$

where x'_R is the velocity along the x -axis of the local vehicle's frame R . Thus, for a trajectory of $n + 1$ discrete vehicle's configurations, our forward motion constraint functional can be expressed as follows,

$$\begin{aligned} \mathcal{H}_{fwd}(\xi) = & \frac{1}{\Delta t} \sum_{t=0}^{n-1} (x_{t+1} - x_t) \cos(\theta_t) + (y_{t+1} - y_t) \sin(\theta_t) \\ & - \|(x_{t+1} - x_t) \cos(\theta_t) + (y_{t+1} - y_t) \sin(\theta_t)\|. \end{aligned} \quad (5)$$

Finally, our restrictions are summarized into the following constraint functional

$$\mathcal{H}(\xi) = \mathcal{H}_{roll}(\xi) + \mathcal{H}_{fwd}(\xi) = 0. \quad (6)$$

²A C++ implementation of CHOMP using the rolling and forward motion constraints is available at <https://github.com/rafaelvalencia/path-adaptor>.

Now this equality constraints is added to CHOMP by expressing it as a constraint functional $\mathcal{H}(\xi) = 0$. It requires to iteratively update the trajectory as follows,

$$\begin{aligned}\xi_{i+1} = & \xi_i - \frac{1}{\eta_i} A^{-1} \nabla U[\xi_i] \\ & + \frac{1}{\eta_i} A^{-1} C^\top (CA^{-1}C^\top)^{-1} CA^{-1} \nabla U[\xi_i] \\ & - A^{-1} C^\top (CA^{-1}C^\top)^{-1} b,\end{aligned}\quad (7)$$

with $A = KK^\top$, where K is a finite differencing matrix (see Eq. 13 in [9]), $C = \frac{\partial}{\partial \xi} \mathcal{H}(\xi_i)$ is the Jacobian of the constraint functional evaluated at ξ_i and $b = \mathcal{H}(\xi_i)$. Thus, trajectory updates that follow our motion restrictions are formulated using the constraint functional \mathcal{H} and its Jacobian C and b here.

5 Real-Time Integration

The constrained CHOMP was integrated with the sensor fusion and control system of the Autonomous Truck. It should also be noted that, the overall SPADES framework has a single global path planner that computes optimal paths along with the global site planner in a central station. However, in this paper, we are studying the performance of the local path adaptor (the constrained CHOMP) for a single vehicle only and not the SPADES framework. Hence, a global path planner like the E* was run on the vehicle to generate paths for the constrained CHOMP. Also, the interference objective of CHOMP that considers the trajectories of other vehicles to adapt its own path has also been excluded. In this section, we detail the integration of the constrained CHOMP without the interference objective but with the local map, global path planner and the control system of the vehicle.

5.1 Autonomous Truck and Sensors

A Volvo FH-16 truck with tractor 4 × 2 was used for the project Cargo-ANTs as shown in Fig. 6. The sensors that are fitted with the truck are explained in the Fig. 2.³ Sensors include a GPS, front camera, two IBEO lux laser scanners with 7 layers, a front RADAR and four Delphi short range RADARs at either side of the truck and trailer. There is a navigation PC which fuses the high level sensor information to the low-level processed perception information to keep track of the vehicle positioning and tracking functions together with the map storage and distribution functionality.

³Courtesy: Volvo Trucks AB, Goteborg.

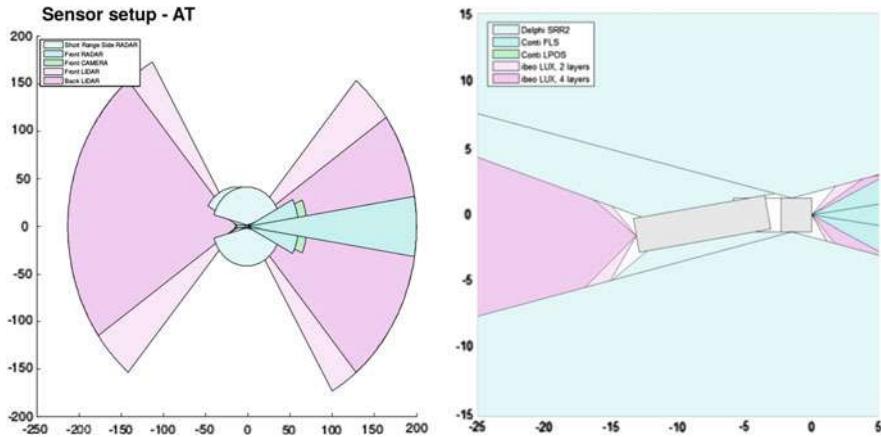


Fig. 2 Sensors location in the AT

It has Intel Core i3 1.6 Ghz processor running Ubuntu 14.04 and ROS Indigo version. A global grid map is generated offline by running the truck along the confined area and the truck is localized in this map [3]. The navigation PC is used to run the global mapping package and store this global map as shown in Fig. 3.

During autonomous driving of the truck, this navigation PC runs the global path planner on this stored global map and localizes its position. The E* algorithm generates smooth interpolated way points from the start to the goal point. This path consists of n number of (x,y) discretized way-points which are broken into smaller chunks of path of 20 m length ahead of the vehicle. The navigation PC also runs a dynamic

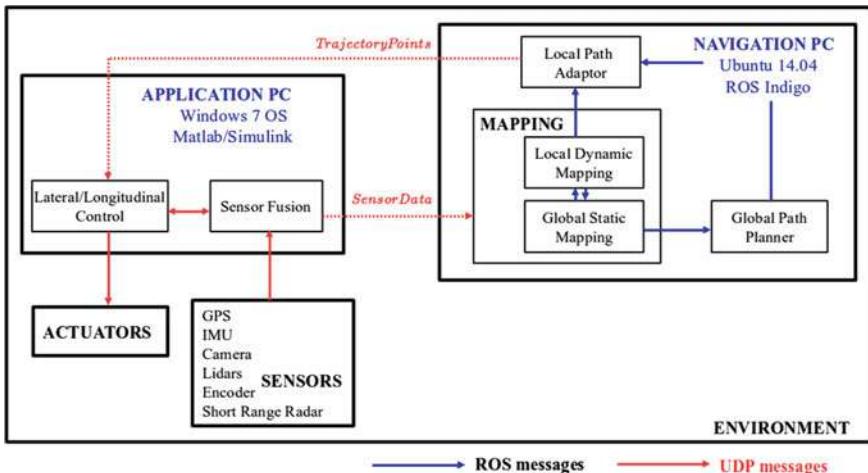


Fig. 3 Software components of an autonomous truck

local map simultaneously which generates a poly line map representing the dynamic obstacles. The local path adaptor, in our case, the constrained CHOMP takes the pruned path with 20 m length (start point taken as the current position and the goal point is 20 m ahead of the truck) and the local map as input for every 20 s. Thus, the constrained CHOMP produces new trajectory as it moves because the start and goal points shift at every 1 m. It then generates the desired trajectory points (x, y, th) , the desired velocities and acceleration of the path as (x', y', th') and (x'', y'', th'') respectively.

5.2 Integration with Controller

The Cargo-ANTs block diagram is explained in Fig. 3 that details the control flow between the navigation PC and the truck controls which is run on an application PC. In the Fig. 3, the vehicle control and sensor fusion components of the truck or applications runs on the application PC. The navigation applications or the ROS interfaces runs on the navigation PC. It should be noted that the navigation PC runs Ubuntu 14.04 and the vehicle low-level control is run under an application PC with Windows 7 OS and Matlab. This type of flow was adapted for ease and independent software development by various partners involved in the project. The flow of control between the two PCs are done via UDP control protocol. The sensor data from the truck is packed as ROS messages by the UDP to be used by the navigation PC and the trajectory points by the constrained CHOMP are then converted back to UDP messages to be used by the vehicle low-level controller.

A path follower controller based on [14] method is used to execute the desired trajectory. It takes the given trajectory way-points, which is obtained as UDP messages and traces to follow it, thus, controlling the lateral motion of the truck. The desired velocities and acceleration of the output trajectory is used for controlling the longitudinal motion of the truck. The local path adaptor takes care of any dynamic obstacles on the local map and avoids it. However, when a complicated scenario arises, the local path adaptor requests for a new path from the global path planner. There is also an emergency safety module for the autonomous truck which over takes the truck when the distance between the obstacle and the truck is less than 0.5 m.

5.3 Integration in a Turtlebot

The constrained CHOMP was implemented with the ROS Navigation stack by developing a plugin for the local planner. The existing local planners like Trajectory Rollout and Dynamic Window are replaced by the constrained CHOMP. Thus, it makes use of the *base controller* in the *move base* ROS package for controlling the turtlebot [4].

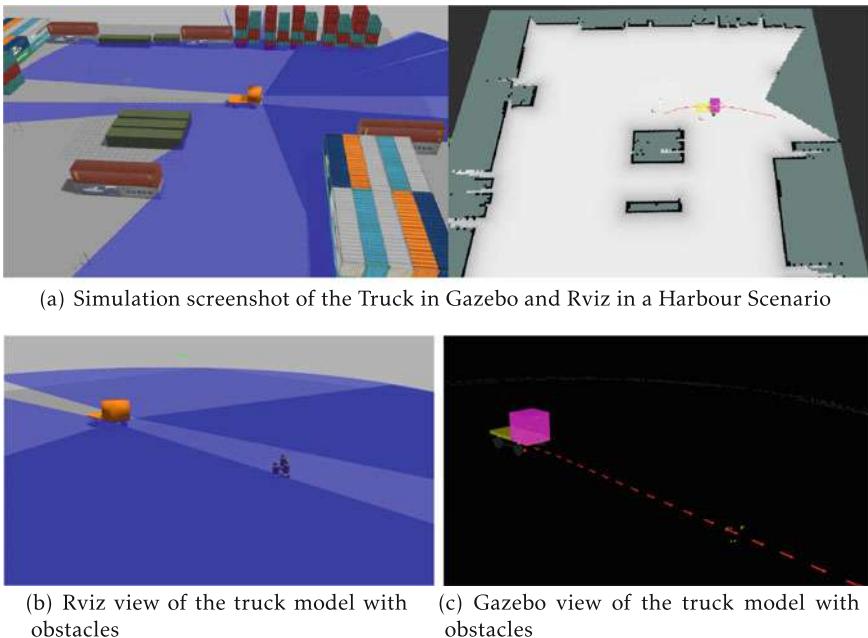


Fig. 4 Gazebo simulations on the truck

6 Experiments

6.1 *Simulation - Autonomous Truck*

The constrained CHOMP was also tested using a simulated Truck with ROS under Gazebo simulation environment. Models of the harbour entities were created as xacro models using dae sketches from Google Sketchup.⁴ The truck model was designed as per the specifications of the real Volvo FH-16 truck model that was used for the project in terms of sensor placements and dimensions of the truck chassis.

Figure 4a is one of the screen-shots that explains the truck model following a path in a simulated harbour scenario in Gazebo and RVIZ. Figure 4b and c explains the constrained CHOMP trajectory for an another simple obstacle avoidance scenario. The initial path is given to be a straight line of length 50 m from the truck's location which crosses an unexpected obstacle. This initial guess is iterated by CHOMP to produce feasible, smooth and constrained trajectory for the truck. The CHOMP

⁴Xacro models of Harbour Environment are available at <https://github.com/jenniferdavid/cargo-ants-ros/tree/master/cargo-ants-models>.

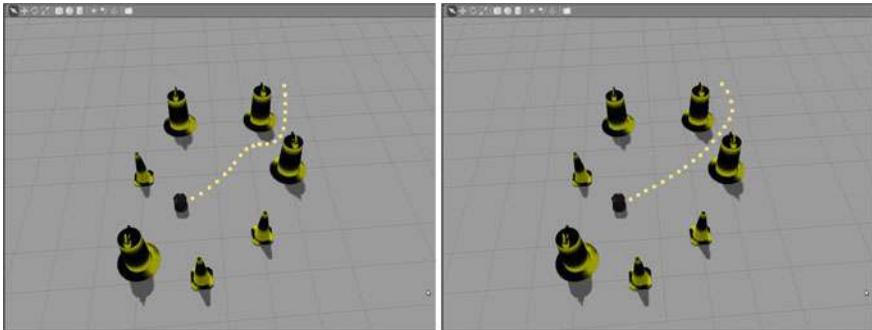


Fig. 5 Gazebo view of the constrained trajectory for an arbitrary goal for the turtlebot

parameters taken for both the scenarios are: $\lambda = 10.00$, time step of 1 ms, $\eta = 100$, number of poses in $\xi = 20$ for 100 iteration limits. The obstacle, (in this case, three barrels) are inflated to 0.04 m of their original sizes.

6.2 Simulations - Turtlebot

The constrained-CHOMP was tested for a differential-drive robot (Turtlebot) under a ROS-Gazebo environment with viz visualizer. A screenshot of the simulation is shown in Fig. 5. It also compares the trajectory generated by CHOMP with constrained CHOMP that gives a smooth and curvature constrained trajectory.

6.3 Real World Experiments on an Autonomous Truck

Real world experiments were conducted on the autonomous truck separately as well as with the truck hooked with a single trailer at Storaholm test track, Goteborg.⁵ In the test track, multiple wooden pallets of size $1 \times 0.75 \times 1.2$ m were added as obstacles in an area of about 1.5 sq. km. Two separate scenarios were demonstrated that explains the feasibility of constrained CHOMP on non-holonomic vehicles with and without obstacles. As shown in Fig. 6, constrained CHOMP was able to avoid the front obstacle with the trailer by generating a smooth and collision free trajectory.

⁵The video of the demo is with Volvo Trucks AB, Goteborg.



Fig. 6 Real-World Scenario - the AT avoiding an obstacle in the front along with the trailer

7 Discussions

7.1 Performance Comparison

A simple GUI of the CHOMP and constrained CHOMP was created as shown in Fig. 7. In this simple setting, we consider an initial planar trajectory connecting a starting robot configuration at $(-5.0 \text{ m}, -5.0 \text{ m}, \pi/4 \text{ rad})$ to a goal configuration in $(7.0 \text{ m}, 7.0 \text{ m}, \pi/2 \text{ rad})$, with 20 intermediate robot poses stacked initially into the starting pose. In the middle of a straight line, connecting the starting and ending configurations, we place two obstacles, enclosed by the blue and purple circles with a radius of 2 m. In practice, the radius of these circles account for the size of the obstacle plus some inflation, i.e. in Fig. 7 the robot is not in contact to the actual obstacles.

For these tests, we compare the unconstrained and constrained CHOMP using the same setting. For both CHOMP implementations, we used $\Delta t = 1$, $\eta = 100$ and $\lambda = 1$. In Fig. 7c we show the evolution of the smoothness for the trajectories found at each iteration with the unconstrained CHOMP and by adding rolling constraints. From Fig. 7b we notice that adding the rolling constraints allows us to get smoother trajectories than the unconstrained case. The unconstrained trajectory shows a sharper turn when approaching the obstacle (see Fig. 7a), which is especially not desirable for a heavy-duty vehicle. Moreover, the computation time is not significantly increased in the constrained case. Note also that we did not constraint the final pose in this implementation, however, a similar procedure [9] can be performed to include it.

Another set of experiments were conducted on the simulation truck and turtlebot. Table 1 illustrates the experiments conducted on 10 different scenarios varying from highly cluttered to scarcely cluttered on a simulated truck, simulated truck with trailer and a turtlebot. It could be seen that the trajectories produced by CHOMP was not feasible for the simulated truck to navigate at all compared to the turtlebot because of its minimum turning radius. However, the constrained CHOMP was able to safely

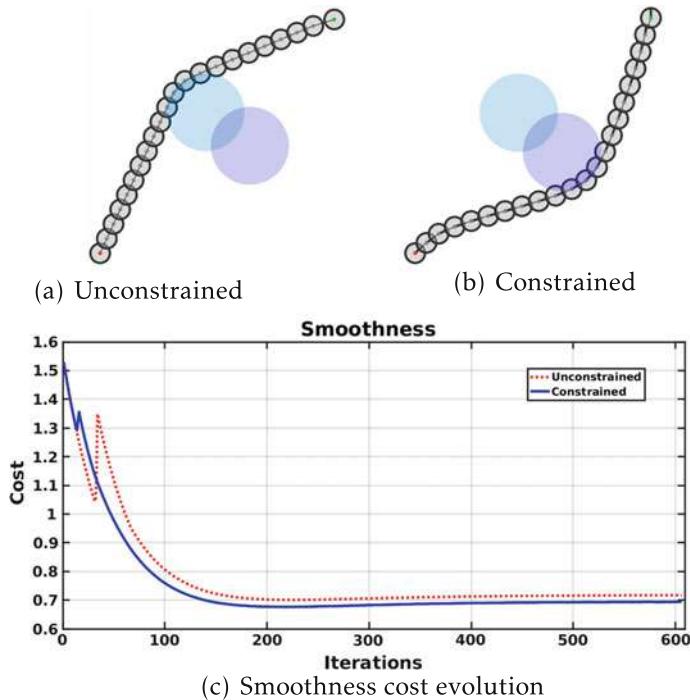


Fig. 7 Obstacle avoidance and the evolution of the smoothness cost at each iteration for the unconstrained CHOMP (red dotted line) and constrained CHOMP (blue solid line)

Table 1 Simulation results on the simulated truck and turtlebot for 10 different scenarios

	CHOMP			Constrained CHOMP		
	Turtlebot	Sim AT	Sim AT & trailer	Turtlebot	Sim AT	Sim AT & trailer
Number of successful maneuvers	8/10	3/10	1/10	10/10	9/10	7/10
Avg planning time (in secs)	0.8	1.1	1.2	1.08	0.9	0.97
Avg time for iterations (in secs)	41	34	25	32	29	38

navigate in almost all the cases as shown. It is also seen that the computation and updation of this extra Lagrangian term in Eq. 7 does not affect the regular planning and updation time of the CHOMP.

7.2 Insights

From these results, it can be observed that the constrained CHOMP did perform well with differential-drive robots more than with non-holonomic vehicles. This is because it does not capture all the kinematic constraints and physical dimensions for the truck or a truck with trailer. This way of adding constraints seems straightforward and easy. But adding all constraints would make Eq. 6 and its Jacobian more intricate. Nevertheless, for avoiding obstacles of the size we consider in our tests, this approach gives a fast and feasible solution. Using this approach for more complicate scenarios, i.e. with more clutter or larger obstacles, a replanning scheme should be followed, i.e. we would need to compute a new path with our global path planner when CHOMP cannot provide a feasible solution. Another issue we notice is that it also gets stuck in local minima, however, some solutions to this problem include random restarts [9] as well as proper initial trajectory input. Lastly, an alternative to adding motion restrictions in this way was to include inequality constraints into CHOMP instead, in order to restrict the motion of the vehicle inside a kinematic feasible region.

8 Conclusions

The SPADES framework was developed for a fleet automation in container handling with multiple layers of abstraction. Each layer consists of carefully chosen simplifications to achieve good overall system performance while making sure that each layer is tractable. In this paper, the lower level of the navigational framework is tested in simulations as well as in a real-world scenario. For this, we have adapted the CHOMP algorithm for use on non-holonomic vehicles as well as with differential-drive robots. Approaches involving constraints on the curvature with a separate objective functional as well as integrating along with the smoothness objective have already been investigated in [2]. The incorporation of sliding and rolling constraints on the CHOMP trajectory to obtain a smooth curvature for non-holonomic vehicles have been studied here in this paper. The results have been tested on a real-world scenario as well as in simulations where the truck with trailer was able to avoid obstacles. It has also been found that the basic algorithm of CHOMP does not allow us to capture the complete kinematic constraints of the vehicle. However, this approach has been found to work satisfactorily in fairly uncluttered environments. Future work will include in adding inequality constraints to CHOMP and improved versions of CHOMP that can truly respect the curvature constraints of the vehicle in a larger scale.

Acknowledgements The authors would like to thank Volvo Trucks AB, Gothenburg for their contributions in this work. This work has been supported by the EU Project CargoANTs FP7-605598.

References

1. Alonso-Mora, J., Breitenmoser, A., Rufli, M. et al.: Optimal reciprocal collision avoidance for multiple non-holonomic robots. In: Distributed Autonomous Robotic Systems, pp. 203–216 (2013)
2. Bosshard, P., Philippsen, R.: Investigation of Trajectory Optimization for Multiple Car-Like Vehicles, Semester Thesis report, Halmstad University, Sweden (2015)
3. Corominas-Murtra, A., Vallvé, J., Solà, J. et al.: Observability analysis and optimal sensor placement in stereo radar odometry. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 3161–3166 (2016)
4. David, V., Michael, F.: ROS navigation stack (2014)
5. David, J., Valencia, R., Iagnemma, K.: Robotics science and system. In: Workshop on Task and Motion Planning, Michigan, USA (2016)
6. Kuhn, H.: The hungarian method for the assignment problem. In: 50 Years of Integer Programming 1958–2008, pp. 29–47. Springer, Berlin (2010)
7. LaValle, S.: Planning Algorithms. Cambridge university press (2006)
8. Turpin, M., Mohta, K., Michael, N., Kumar, V.: Goal assignment and trajectory planning for large teams of aerial robots. Autonom. Robots (2014)
9. Matt, Z., Nathan, R., Anca, D., Mihail, P., et al.: CHOMP: covariant hamiltonian optimization for motion planning. Int. J. Robot. Res. **32**(9–10), 1164–1193 (2009)
10. Paolo, F., Zvi, S.: Motion planning in dynamic environments using velocity obstacles. Int. J. Robot. Res. **17**(7), 760–772 (1998)
11. Philippsen, R., Siegwart, R.: Smooth and efficient obstacle avoidance for a tour guide robot. In: IEEE International Conference on Robotics and Automation (ICRA), LSA-CONF-2003-018 (2003)
12. Siciliano, B., Khatib, O.: Springer Handbook of Robotics. Springer (2016)
13. Steenken, D., Voß, S., Stahlbock, R.: Container terminal operation and operations research - a classification and literature review. OR Spectr. **26**(1), 3–49 (2004)
14. Thrun, S., Montemerlo, M., Dahlkamp, H., et al.: Stanley: the robot that won the DARPA grand challenge. J. Field Robot. **23**(9), 661–692 (2006)
15. van den Berg, J., Stephen, J., Ming, L., Dinesh, M.: Reciprocal n-body collision avoidance. In: Springer Tracts in Advanced Robotics, Robotics Research: The 14th International Symposium (ISRR), pp. 3–19 (2011)

Part VII

Systems and Tools

Field Experiments in Robotic Subsurface Science with Long Duration Autonomy

Srinivasan Vijayarangan, David Kohanbash, Greydon Foil, Kris Zacny,
Nathalie Cabrol and David Wettergreen

Abstract A next challenge in planetary exploration involves probing the subsurface to understand composition, to search for volatiles like water ice, or to seek evidence of life. The Mars rover missions have scraped the surface of Mars and cored rocks to make ground breaking discoveries. Many believe that the chance of finding evidence of life is expected to increase by going deeper. Deploying a system that probes the subsurface brings its own challenges and to that end, we designed, built and field tested an autonomous robot that can collect subsurface samples using a 1 m drill. The drill operation, sample transfer, and sample analysis are all automated. The robot also navigates kilometers autonomously while making decisions about scientific measurements. The system is designed to execute multi-day science plans, stopping and resuming operation as necessary. This paper describes the robot and science instruments and lessons from designing and operating such a system.

S. Vijayarangan (✉) · D. Kohanbash · G. Foil · D. Wettergreen
Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, USA
e-mail: srinivasan@cmu.edu

D. Kohanbash
e-mail: dkohanba@andrew.cmu.edu

G. Foil
e-mail: gfoil@meshrobotics.com

D. Wettergreen
e-mail: dsw@cmu.edu

K. Zacny
Honeybee Robotics, New York, USA
e-mail: zacny@honeybeerobotics.com

N. Cabrol
SETI Institute, Mountain View, USA
e-mail: nathalie.a.cabrol@nasa.gov



Fig. 1 Panorama taken by Zoë in the Atacama desert showing the heterogeneous nature of the environment

1 Introduction

The search for life in the far reaches of the solar-system compels the use of robots to explore faster, cheaper, and safer than humans. Still these robotic systems and their operations are complex, so it is necessary to practice and test robotic missions on Earth to gain insights into the technical challenges and best methods. To that end, we deployed a robotic system and operated it in the Mars-analog Atacama Desert in Chile where evidence suggests that the interior is the most arid and lifeless region on Earth (Fig. 1). Our field investigation uses a rover to make controlled transects in the desert with instruments to characterize subsurface habitats (Fig. 2). Figure 3 shows the regions explored in two field seasons to accomplish the following goals:

- **Subsurface sample analysis:** drill into the surface autonomously, collect samples and analyze them using the onboard science payload.
- **Autonomous science sequences:** select sampling targets that maximize the information gain while minimizing the navigational cost.
- **Multi-day autonomy:** operate on a plan autonomously for multiple days: executing commands during the day, detecting the end of day, shutting down the robot gracefully, waking up the next morning and resuming the plan after the robot is fully charged.

The paper is organized as follows. Section 2 categorizes the related work based on our goals. Section 3 gives an account of the robot and its science payload. Sections 4, 5 and 6 details progress with respect to project goals. Section 7 discusses the lessons learned during the field operations.

2 Related Work

2.1 Subsurface Sample Analysis

The Mars Exploration Rovers Spirit and Opportunity landed on Mars in January 2004 carried the Rock Abrasion Tool (RAT) [1] for grinding and brushing. It drilled 0.045 m wide by 0.005 m deep holes in Martian rocks. The Curiosity rover which landed in November 2011 carried a percussion drill capable of drilling 0.016 m wide and up to 0.05 m deep holes [2].

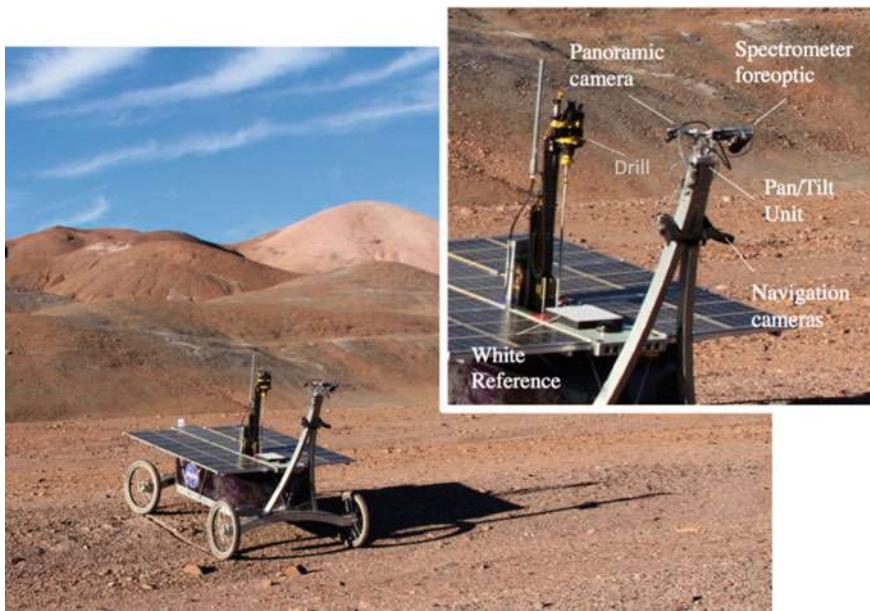


Fig. 2 Zoë



Fig. 3 Region traversed by Zoë in 2013 (right) and 2015 (left)

There are few simulation missions similar to our work. The Mars Astrobiology Research and Technology Experiment (MARTE) [3] conducted a drilling operation simulating a Mars mission in 2005. It collected core samples near the Rio Tinto river (southwest Spain). It drilled 6 m deep in a 30 day mission and collected 21 core samples. Scarab [4] simulated a lunar mission with a coring drill capable of drilling 1 m in Mauna Kea in Hawaii. It processed the core samples and analyzed the composition of captured soil. The Icebreaker mission simulated a Mars polar mission operating the Icebreaker drill [5] in the Arctic and the Antarctic Dry Valleys. The Icebreaker drill is a 1 m class drill with a triple redundant sample transfer mechanism which is capable of drilling 1 m in 1 h with approximately 100 W of power. However in all these missions, the drilling procedure and the sample handling and analysis procedures were tele-operated manually.

2.2 Autonomous Science Sequences

Earlier works of Thompson et al. [6] fused images from different views of the robot to detect rocks that could be sampled autonomously. This was followed by Thompson et al. [7] where the features of geologic interest were automatically detected using a probabilistic fusion technique. Smith et al. [8] proposed different modes of operation for science autonomy and reported qualitative results using Zoë [9].

2.3 Multi-day Autonomy

Wettergreen et al. used Hyperion [10] with goals of long duration autonomy. Given a command, the mission planner generated waypoints which the robot followed autonomously with the health monitor looking for faults. It introduced the basic software structure to operate autonomously for long durations. This was followed by the work on DepthX [11] which was capable of executing an elaborate plan. The missions involved diving into flooded sinkholes in Sistema Zacatón (Mexico), searching science worthy targets, collecting samples and surfacing, all without any telemetry. The plan also included an extensive list of contingency plans for safety. The missions lasted 4–6 h, thus extending the hands-off autonomous operation duration. Scarab [4] extended this capability by including a drilling operation along with the navigation goals. One of these systems have the capability to operate on a plan of possibly unlimited duration and diurnal hibernation.

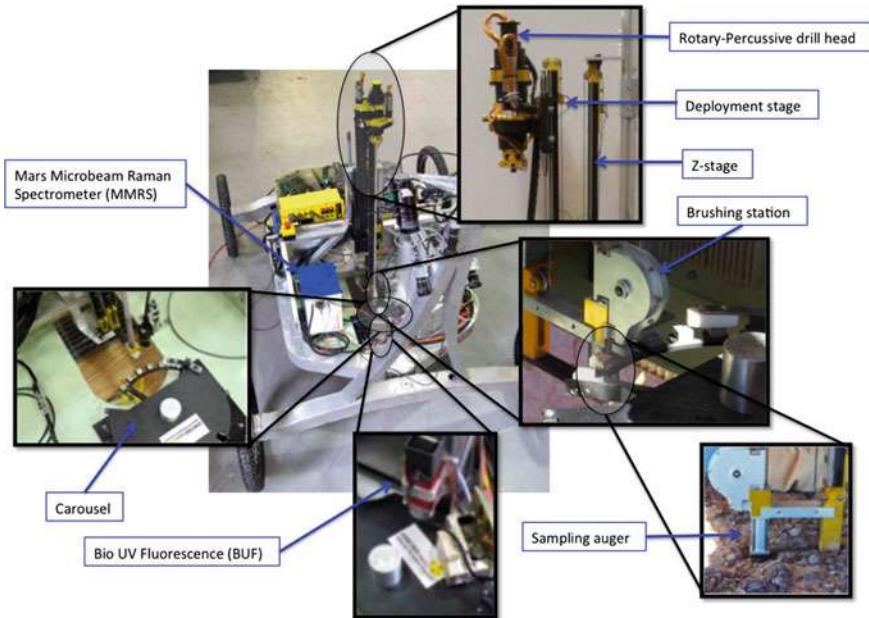


Fig. 4 Zoë with drill and science payload (MMRS and BUF imager)

3 System Overview

The system deployed for the Atacama field experiments is a rover with integrated 1 m drill, sample collection and handling to Raman spectrometer and fluorescence imager. Long range sensing with visible-near infrared spectrometer aid in sample selection.

3.1 Rover

We refurbished and reconfigured Zoë [9] to incorporate a drill and scientific instruments for field investigations in June 2013 and March 2015. Zoë is a solar powered robot with passive steering and passive suspension. The solar panels use triple junction GaAs cells 2.4 m^2 with 23% efficiency [12]. It has a pan-and-tilt unit that points a visible near-infrared spectrometer with 1° resolution and a high-resolution camera for taking close-up context images and panoramas.

Zoë has a stereo camera pair mounted on the mast used for navigation. However software is added to use those cameras for science autonomy tasks as well.

3.2 Drill

The drill is a two stage, rotary percussive mechanism capable of drilling up to 1 m deep. Its design accounts for flight constraints like weight and volume. It weighs 10 kg and consumes 300 W of power on average. The drill is designed and developed by *Honeybee Robotics Spacecraft Mechanisms Corporation* and consists of the following subsystems: (1) *Rotary-Percussive drill head*, (2) *Sampling auger*, (3) *Brushing station*, (4) *Z-stage*, (5) *Deployment stage* (6) *Carousel* (Fig. 4).

The drill head is designed with rotation and percussion decoupled. This allows use of the more energy intensive percussive system only when required (e.g., to penetrate harder formations). Both rotary and percussive motors are approximately 150 W each. To reduce sample handling complexity, the drill auger is designed to capture drill cuttings as opposed to cores. High sampling efficiency is possible through a dual design of the auger. The lower section of the auger has deep and low pitch flutes. This geometry creates natural cavities ideal for retaining granular materials (cuttings and soil). The upper section of the auger moves the cuttings out of the hole efficiently.

The drill mechanism lowers to contact with the ground surface with its first stage and stabilize the mechanism on the drill hole. The rotary-percussive second stage then drives the auger into the ground. The mechanism is co-designed with the rover mast for efficient integration and greater overall rigidity and stabilization. For sample collection, soil from the tip of the auger is captured in a collar which expels upon drill retraction down a chute into the sample cups. The carousel is a single degree of freedom system designed to move 20 cups underneath the drop off spout and the other science payload instruments. It has a cantilevered (fully passive) scraper that smooths out and compacts the top powder in each cup. Automatic coordination onboard the rover places specific cups beneath the chute which are then rotated into sealed storage.

The drill uses a *bite* sampling approach where samples are captured in \sim 10 cm intervals. That is, after drilling 10 cm, the auger with the sample is pulled out of the hole, and the sample is brushed off into one cubic centimeter cups by a passive brush within the brushing station. An advantage of the *bite* sampling approach is that stratigraphy is preserved and the provenance depth of the sample is known.

3.3 Mars Microbeam Raman Spectrometer

The Mars Microbeam Raman Spectrometer (MMRS) measures the Raman scattering of the collected samples using a focused laser beam. Raman scattering probes the fundamental vibrations of molecules that produces finger-print spectral patterns with sharp no-overlapping peaks. Laser Raman spectroscopy is a powerful technique for the detection and characterization, at fine-scale, of the major, minor and trace species in a mixture (rocks and soils).

The carousel delivers the samples collected at different depths (10, 30 and 80 cm) under the MMRS probe head. It then collects the Raman spectra for the sample in the sample cup by focusing a 532 nm laser beam. A stepper motor within the probe moves the optical bench linearly over the sample surface in a range of 10 mm, from which Raman spectra of 20 to 100 spots are collected without using autofocus.

The wavelength of the laser used in the MMRS is calibrated using a Ne lamp before, during and after the field experiments. In addition, we measure Raman spectra of substances like naphthalene and diamond which has strong and distinct Raman spectra and use as reference and measure multiple times in a day. This helps us to keep the laser wavelength calibrated and also evaluate the general performance such as sensitivity, noise levels and spectral resolution of the device.

3.4 Bio UV Fluorescence Imager

The Bio UV Fluorescence (BUF) imager shines UV and white light on the samples and measures fluorescence, if any. Fluorescence under UV light is a strong indicator of chlorophyll in the sample which provides evidence of life. 370 nm wavelength UV light LEDs are used in the BUF imager. The imager is a light-field camera manufactured by Lytro, Inc. One unique feature of the light-field camera is that it uses a lenslet array to simultaneously collect a number of images at slightly different perspective with no moving parts. These images comprise a *rayfield image* that can be re-focused after the fact. While this synthetic autofocus feature is not required to examine the flattened powdered samples presented to it in the carousel, this feature enables the BUF imager to examine unprepared rock samples in other venues without the need for a mechanical autofocus mechanism. The BUF imager is controlled by the MMRS which in turn is commanded from the rover computer. The BUF imager is mounted on the same carousel next to the MMRS.

3.5 Visible-Near Infrared Spectrometer and Panoramic Imager

The Visible-Near Infrared (VNIR) spectrometer on Zoë is an Advanced Spectral Devices (ASD) FieldSpec Pro with readings in the Visible-Near Infrared (350–2500 nm) range. It uses a one-degree foreoptic mounted on a pan/tilt unit alongside the panoramic imager on the mast. Visible/near-infrared spectroscopy involves studying the reflectance spectra of a material in the visible/ near-infrared wavelength range. The goal of the visible/near-infrared spectrometer is to support the rover and other instruments with mineralogical composition information. This can be used to help direct the rover when used in conjunction with orbital based spectral data and to determine locations of interest to the rover.

The Visible-Near Infrared Spectrometer (VNIR) was deployed successfully in the 2004 and 2005 LITA field investigations [13]. However software functionality is added to enable intelligent and complex workflows using the spectrometer. For instance, software functionality is added to calibrate the spectrometer to the camera so that its viewpoint within the image is known and the target of the spectrometer can be identified. Further, several algorithms for detecting features in the panoramic image have been developed and ported to the rover so that it can automatically identify salient features in the scene and then direct and record high resolution image and VNIR spectra.

4 Subsurface Sample Analysis

Drilling: Drilling in consolidated, fine grained soils is easy and the sample is retained in the auger successfully every time. Poorly consolidated, coarse-grained soil, is relatively easy to drill, but capturing and retaining of samples within the auger flutes is difficult. In most cases, the soil is pushed aside as the drill is lowered into the ground and in turn no soil is captured because of low friction angle and lack of cohesion. To address these issues we experimented with various combinations of the following: (1) We increased the diameter of the auger from 0.5 to 0.75 in. to enable greater sampling volume (2) We used shallower flutes to help with sample retention, (3) We optimized drilling software to shorten the sampling time, and (4) We drilled without percussion and retracted the drill without rotation. In all cases, the average drilling power is less than 15 W because the percussive system is not needed and hence not engaged most of the time. The weight-on-bit is also low, at 50 N or less. Table 1 lists the different locations we drilled successfully in March 2015 and their depths.

A drilling sequence which involved drilling to 10 cm, retracting and dumping to a cup on the carousel, followed by drilling to 30 and 80 cm, took a total of 130 min.

Table 1 Lists the sites where we successfully completed the drilling operation in March 2015 and their respective depths

Locale	Latitude	Longitude	Drill depths (cm)
12	24° 29'23.69"S	70° 08'52.05"W	10, 15
13	24° 29'15.65"S	70° 08'52.25"W	10, 19, 50
14	24° 29'06.11"S	70° 08'52.96"W	10, 20
15	24° 29'03.13"S	70° 08'53.43"W	10, 17
18	24° 29'17.26"S	70° 09'02.26"W	10, 15, 20, 50
19	24° 29'18.18"S	70° 09'04.85"W	20
20	24° 29'31.32"S	70° 09'59.09"W	20
24	24° 34'15.63"S	70° 09'31.02"W	10, 20

But the duration and sample collection efficiency depended largely on the drilled location and material drilled into.

Raman Spectras: Any changes, especially the optical alignment of MMRS affected by mechanical, optical or electronic fluctuations during the transverse of the rover, are apparent in the reference spectra. Despite Zoë's 50 Km transverse on a rough terrain and a wide diurnal temperature cycle, from -6 to 27°C , during the field campaign, the MMRS did not show noticeable performance change. We did notice the laser wavelength shift, due to insufficient temperature control in the laser unit within MMRS which was later corrected.

Multi-point Raman spectra were obtained for 31 samples. The spectral analysis showed the presence of three groups of minerals in the Atacama samples. They are: original igneous minerals (mainly feldspar and quartz); alteration products (e.g., TiO_2 and goethite); and hydrous or anhydrous salts (sulfates and carbonates) with variable origins.

Bio UV Fluorescence: The BUF imager recorded images of the samples illuminated under UV light. Abundances were negligible, so no fluorescence was detected but images were still useful when the samples were illuminated under white light as they served as a reference for the MMRS measurements.

Autonomy: The system accomplished end-to-end operation collecting samples to analyzing them, several times. This involved drilling to a specified depth, followed by retraction and dumping to a cup on the carousel, moving the carousel to position the cup under the MMRS, recording the Raman spectra, repositioning the cup under the BUF imager, collecting images with UV and white light illuminations and then navigating to a different location. We found that batch processing the samples with MMRS and BUF is much more efficient than processing them individually after they are collected.

5 Autonomous Science Sequences

Zoë's science autonomy system includes two basic capabilities that operates the robot on mesoscale and macroscale features respectively. *Smart targeting* identifies science features in rover navigation imagery and uses this information to point the VNIR-ASD spectrometer. *Adaptive path planning* navigates on scales of tens or hundreds of meters, using satellite images to select waypoints with distinctive or novel spectra. A more detailed account of these techniques can be obtained from Wettergreen et al. [14].

Smart Targeting [15, 16]: Zoë began each autonomous target selection process by acquiring a navigation camera image. Onboard image processing then analyzed the scene to find large contiguous regions (using connected components analysis) of a

desired terrain class (using random forest classification). Typically these classes were rough surface features like rock outcrop or bright sediment patches with distinctive spectral signatures. Upon finding a feasible target, the rover re-calibrated its VNIR-ASD spectrometer, pointed at the feature and collected a small 33 raster of spectra centered on the target of interest. For context, it also acquired a high-resolution color image of the scene.

Adaptive Path Planning: The science autonomy system also operates on larger scales of tens or hundreds of meters, where it analyzes satellite data to adjust its traverse path. We model the explored environment using a standard geographic or area mixing model where each measurement is a mixture of a small number of endmember materials. Endmembers' spectra combine in proportion to their physical extent on the surface. In practice there is always residual error separating the reconstruction from the measurement. This is partly attributable to measurement noise, but unless the library is comprehensive there may also be incompleteness errors (e.g. spectral features that are expressed in the observations but not present in the library). A library that reconstructs all spectra well can be said to have explained the scene, and provides insight into the mineral compositions in the remote sensing data. This intuition provides a figure of merit for an adaptive path planning system to select future measurement locations. Zoë's planner selects locations, the measurements at which provide the largest expected reduction in unmixing error. As a consequence, it aims to visit locations that are spectrally distinctive, collecting samples that fully explain the orbital image.

6 Multiday Autonomy

In order to achieve multiday autonomy goals we created a tool called *Rover Commander* for scientists to generate plans for the rover. To enable the rover to power up and power down the devices through commands from the autonomy system, we built the *Power Management and Distribution (PMAD)* system. Finally, to navigate to a desired location autonomously we used *Reliable Autonomous Surface Mobility (RASM)* [17] software.

Rover Commander: Rover commander is a web-based planning tool that is used to generate a list of commands that can be executed by Zoë. The plan can span multiple days. Figure 5 shows a snapshot of the tool. The tool uses the Google Maps API to source the underlying terrain information. The tool has presets for different operations like drill with different depths, quick or full panoramas, location to move to, drive to location using adaptive science, etc. A sample plan file generated by the Rover Commander is shown in Table 2.

Power Management and Distribution: The PMAD is a low powered embedded computer that is developed to provide a way to turn on/off devices through software. It has solid-state relays connecting to all the devices on the robot which can

be controlled through software. This also allowed us to save power on the robot by commanding only the essential devices to be powered up. For example, we powered down the drill and the scientific instruments when driving. Another main utility of the PMAD is to support the idea of surviving the night. When the power goes below a set threshold at sunset, the PMAD sends a hibernate signal to the high-level software which suspends the current plan, shuts down the devices and powers down the robot. In the morning, after the batteries are sufficiently charged from the solar panels, it wakes-up the robot automatically and the high-level software will then resume execution of the previous day's plan.

Reliable Autonomous Surface Mobility: RASM [17] is the onboard navigation software in Zoë that is capable of local hazard avoidance and path planning using a 3D terrain representation.

Autonomy: Zoë demonstrated multi-day autonomy partially. During the end of the day, Zoë suspended the current plan, turned off its devices and hibernated. It survived the night and successfully booted back up the next morning and resumed the previous day's plan from where it left off. But unfortunately for a variety of different reasons, each day of the field season, the motor controllers got into an error state and the robot was not able to accomplish its complete plan but it moved a few meters before encountering the error state thus demonstrating the capability but not accomplishing it.

Table 3 shows the percentage of time each of the specific operation was carried out compared to the uptime of the robot. Figure 6 shows the time of the day when these operations were performed. Of all the operations, driving took the most amount of time. This is because we drove 25.6 kms in 10 days. 60.62% of the total distance was driven autonomously in these exploration experiments.

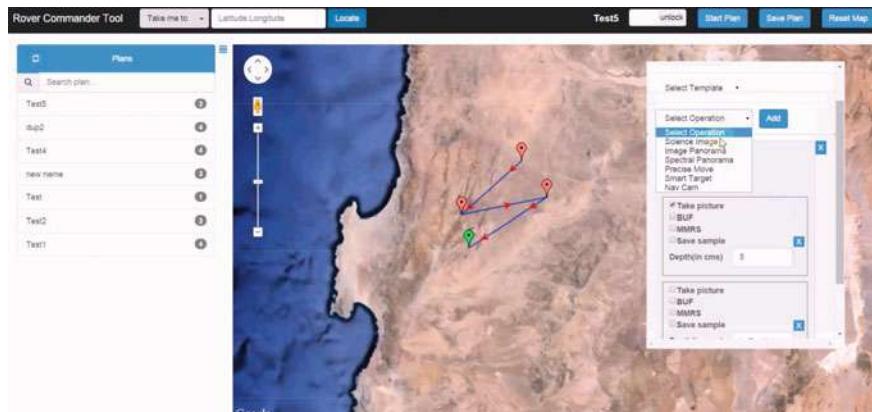


Fig. 5 Rover commander—a web based planning tool to generate science plans for Zoë

Table 2 Sample plan generated by Rover commander

Command	Description
Checkplan	Check plan for syntax error
Drill <i>locale12 500</i>	Drill command to drill 0.5 m and name the sample <i>locale12</i>
Sotf <i>-24.488383 -70.151347 0.2</i>	Use adaptive science (<i>science-on-the-fly</i>) to navigate to the specified lat/lon location. Will explore the region while ensuring the additional distance overhead to less than 20%
Panorama <i>-30 30 -20 20</i>	Generate a panorama with elevation angles (-30° , 30°) and azimuth angles (-20° , 20°)
Spanorama <i>-10 10 -10 10 10 1</i>	Generate a spectral panorama using the VNIR spectrometer and the high-resolution camera
Latlon <i>-24.488012 -70.150286</i>	Navigation command to drive to specified latitude and longitude
Sotf <i>-24.486816 -70.148548 0.4</i>	Use adaptive science (<i>science-on-the-fly</i>) to navigate to the specified lat/lon location. Will explore the region while ensuring the additional distance overhead to less than 40%
Drill <i>locale13 300</i>	Drill command to drill 0.3 m and name the sample <i>locale13</i>
Panorama <i>-40 40 -20 20</i>	Generate a panorama with elevation angles (-40° , 40°) and azimuth (-20° , 20°)
Spanorama <i>-10 10 -10 10 10 0</i>	Generate a spectral panorama using just the VNIR spectrometer
Latlon <i>-24.487680 -70.147848</i>	Navigation command to drive to specified location
mmrsbuf	Take MMRS and BUF measurements on all the unprocessed samples (<i>locale12</i> and <i>locale13</i> in this case)

Table 3 Percentage of total duration of each of the operations with respect to the uptime

Operation	Duration (% of uptime)
Science-on-the-fly (adaptive science)	3.20
MMRS and BUF	5.31
Panorama	6.35
VNIR spectra	9.36
Driving	18.84
Drilling	11.99

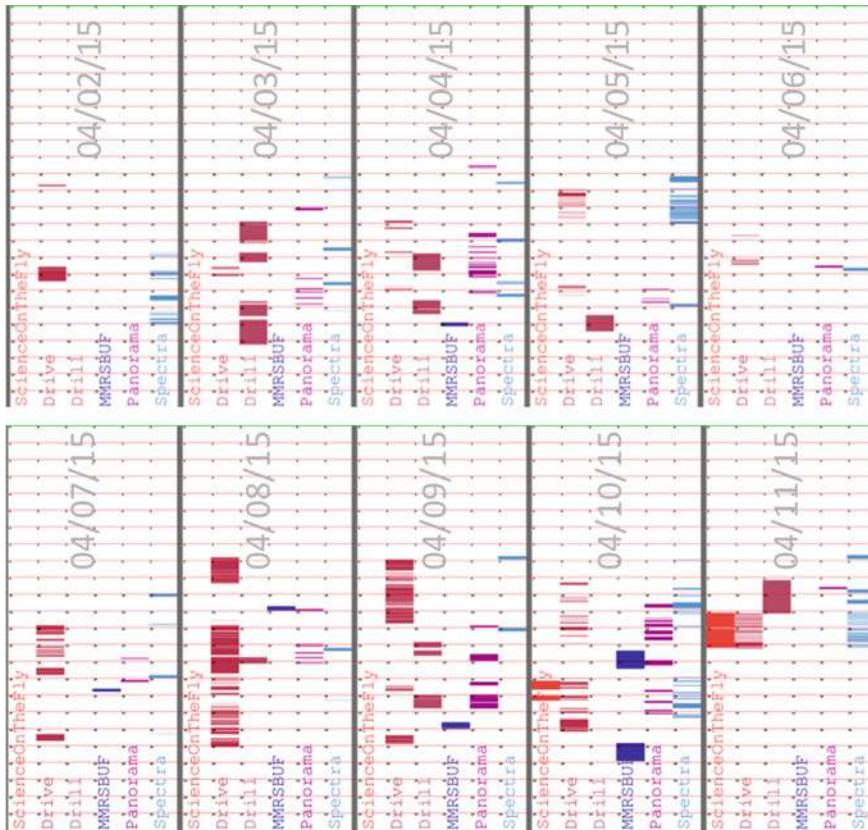


Fig. 6 Graph shows the time of the day each of these operations were performed. Each vertical column corresponds to a day during the field operations and the horizontal rows correspond to the hours in the day

7 Lessons Learned

- **Drilling in unconsolidated soil:** After digging and inspecting a pit, we noticed many layers of different soil combined with layers of rocks and void spaces. The possible explanation for not being able to generate sample in the cup may be the collapse of material into the void spaces, as evidence by the lack of tailings pile at some sites.
- **Time/power for MMRS:** The MMRS instrument consumed more power than that could be sourced by the solar panels. This is because the laser was designed to be operated on the cooler environment in Mars. Also each measurement took more than a few minutes which added up as the number of samples increased.
- **Fiddling with science plan:** Although the robot had the capability to continue executing a plan for several days, the plan was often modified multiple times.

This was mainly because the scientist found value in altering the plan when the robot arrived at a new locale. This was also common due to the fact that the Atacama desert received historic rains during our field investigations and the environment was changing drastically.

- **Registration of orbital data (salar experiment):** We used the data from ASTER satellite for the orbital data. However we found that the ASTER images are mis-registered. This prevented the science autonomy software from accurately guiding the rover in some cases.
- **Software integration earlier, rather than modularity:** We favored modularity and delayed the integration of the different software components. But we learned, once again, that integrating the software components early in the development process saves time.
- **Use a standard tool for logging:** Early in the software design we decided to use SQL based logger as opposed to a custom binary logging tool. It proved useful as it is hard to maintain a custom logging tool for several years. However since our data is logged into a SQL database we can use any SQL tool to access the data. This greatly aided data analysis.

References

1. Gorevan, S.P.: Rock abrasion tool: mars exploration rover mission. *J. Geophys. Res.* **108**(E12), 8068 (2003)
2. Jandura, L.: Mars science laboratory sample acquisition, sample processing and handling: subsystem design and test challenges drill. *Aerosp. Mech. Symp.* 233–248 (2010)
3. Stoker, C., Cannon, H., Dunagan, S., Lemke, L., Glass, B., Miller, D., Gomez-Elvira, J., Davis, K., Zavaleta, J., Winterholler, A., Roman, M., Rodriguez Manfredi, J., Bonaccorsi, R., Bell, M., Brown, A., Battler, M., Chen, B., Cooper, G., Davidson, M., Fernández-Remolar, D., Gonzales-Pastor, E., Heldmann, J., Martinez-Frias, J., Parro, V., Prieto-Ballesteros, O., Sutter, B., Schuerger, A., Schutt, J., Rull, F.: The 2005 MARTE robotic drilling experiment in río Tinto, Spain: objectives, approach, and results of a simulated mission to search for life in the Martian subsurface. *Astrobiology* **8**(5), 921–945 (2008)
4. Wettergreen, D., Jonak, D.: Field experiments in mobility and navigation with a lunar rover prototype. *F. Serv. Robot.* 1–10 (2010)
5. Zacny, K., Paulsen, G., McKay, C., Glass, B., Davé, A., Davila, A.F., Marinova, M., Mellerowicz, B., Heldmann, J., Stoker, C., Cabrol, N., Hedlund, M., Craft, J.: Reaching 1 m deep on Mars: the icebreaker drill (2013)
6. Thompson, D.R., Wettergreen, D.: Multiple-object detection in natural scenes with multiple-view expectation maximization clustering. In: 2005 IEEE/RSJ International Conference on, Intelligent Robots and Systems, 2005 (IROS 2005), pp. 562–567 (2005)
7. Thompson, D.R., Niekum, S., Smith, T., Wettergreen, D.S.: Automatic detection and classification of geological features of interest. In: IEEE Aerospace Conference Proceedings, Big Sky Montana (2005)
8. Smith, T., Niekum, S., Thompson, D., Wettergreen, D.: Concepts for science autonomy during robotic traverse and survey. *IEEE Aerosp. Conf. Proc.* **2005**, 1–9 (2005)
9. Wagner, M., Heys, S., Wettergreen, D., Teza, J., Apostolopoulos, D., Kantor, G., Whittaker, W.: Design and control of a passively steered, dual axle vehicle. *Eur. Sp. Agency, (Special Publ. ESA SP)* **2005**(603), 29–36 (2005)

10. Wettergreen, D., Cabrol, N., Teza, J., Tompkins, P., Urmonson, C., Verma, V., Wagner, M., Whitaker, W.: First experiments in the robotic investigation of life in the Atacama Desert of Chile. In: Proceedings—IEEE International Conference on Robotics Automation, vol. 2005, pp. 873–878 (2005)
11. Fairfield, N., Kantor, G., Jonak, D., Wettergreen, D.: DEPTHX Autonomy Software: Design and Field Results, July 2008
12. Calderon, F., Luders, A., Wettergreen, D., Teza, J., Guesalaga, A.: Analysis of high-efficiency solar cells in mobile robot applications. *J. Sol. Energy Eng.* **129**(3), 343–346 (2007)
13. Wettergreen, D., Cabrol, N., Baskaran, V., Calderón, F.: Second experiments in the robotic investigation. In: ISAIRAS 2005 Conference, German, Munich (2005)
14. Wettergreen, D., Foil, G., Furlong, M., Thompson, D.R.: Science autonomy for rover subsurface exploration of the Atacama desert. In: AI Magazine, pp. 47–61 (2014)
15. Bekker, D.L., Thompson, D.R., Abbey, W.J., Cabrol, N.A., Francis, R., Manatt, K.S., Wagstaff, K.L.: A field demonstration of an instrument performing automatic classification of geologic surfaces. *Astrobiology* **14**(6), 486–501 (2014)
16. Wagstaff, K.L., Thompson, D.R., Abbey, W., Allwood, A., Bekker, D.L., Cabrol, N.A., Fuchs, T., Ortega, K.: Smart, texture-sensitive instrument classification for in situ rock and layer analysis. *Geophys. Res. Lett.* **40**(1), 4188–4193 (2013)
17. Wettergreen, D., Wagner, M.D.: Developing a framework for reliable autonomous surface mobility. In: International Symposium on Artificial Intelligence Robotics Automation Space, vol. 1 (2012)

Design and Development of Explosion-Proof Tracked Vehicle for Inspection of Offshore Oil Plant

Keiji Nagatani, Daisuke Endo, Atsushi Watanabe and Eiji Koyanagi

Abstract French oil company TOTAL and ANR (L'Agence Nationale de la Recherche) organize the ARGOS (Autonomous Robot for Gas and Oil Sites) Challenge, which our research group had the opportunity to participate in. ARGOS is a research and development competition for mobile robots capable of autonomous inspection of instruments and teleoperated information gathering in oil plants, in place of human workers. One of the features of this challenge is that robots should be constructed with explosion-proof structures, because the target plants may have explosive atmospheres. To participate in the third competition of the ARGOS Challenge in March 2017, we developed AIR-K, an explosion-proof robot. The AIR-K is divided into three parts to make it explosion-proof. According to the features for robot functions and sensors, it uses a flameproof battery enclosure (Ex 'd'), a pressurized apparatus (Ex 'p') for its body, and intrinsic safety (Ex 'i') for sensors; the explosion-proof of the robot is achieved by a combination of these methods. In this paper, we introduce the design guidelines and implementations that allow our robot to be explosion-proof.

1 Introduction

In recent years in Japan, the practical use of field robots is expected for economic improvement, reduction of dangerous tasks, and social creation. Therefore, industries, government, and academia are continually researching and developing such robots. In particular, the aging of social infrastructure and plants constructed during periods of high economic growth is a significant problem. Therefore, the introduction of robot-based maintenance management methods is recommended to solve

K. Nagatani (✉) · D. Endo · A. Watanabe
Tohoku University, 6-6-10, Aramaki-Aoba, Sendai, Japan
e-mail: keiji@ieee.org

E. Koyanagi
Mobile Robot Research Co.LTD, 2-15-41, Dai, Kamakura, Kanagawa, Japan
e-mail: koyanagi@irobo.co.jp

the problem. With such robotic technologies, it is expected that the lifetime of the infrastructure will be extended and the total cost of maintenance will be reduced [1].

The need for robotic inspection of infrastructure and plants is not limited to Japan—it is a common problem for nations that possess similar facilities. A concrete example is the oil spill that occurred at the Deepwater Horizon in the Gulf of Mexico on April 20, 2010. During the accident, the concerned facility should have been investigated. However, it was impossible to enter the plant because it was too dangerous for human inspectors.

To facilitate robotic investigations in emergency situations, and inspections during normal operations, French oil company TOTAL and ANR (L'Agence Nationale de la Recherche) organized the ARGOS (Autonomous Robot for Gas and Oil Sites) Challenge, which was launched in December 2013. ARGOS was a research and development competition for mobile robots capable of autonomous inspection of instruments and teleoperated information collection in oil plants, in place of human workers. Five international teams, including ours, were selected for the challenge, based on a document review. One of the features of this challenge was that robots should be constructed with explosion-proof structures, because the target plants may have explosive atmospheres. Figure 1 shows a scene from the ARGOS Challenge final competition in March 2017.

Few small mobile robots have obtained explosion-proof certificates. It can be said that Sakura-II was the first small mobile robot with explosion-proof certification. It was developed by Mitsubishi Heavy Industries Ltd. with a New Energy and Industrial Technology Development Organization (NEDO) research grant for surveillance of tunnel disasters. The robot was certified according to the explosion protection (zone-1) of the Japanese standard (IEC spec conformable), which is almost equivalent to ATEX Cat.2.

With reference to the above robot, we conducted research and development of an explosion-proof robot, called AIR-K, in order to participate in the third competition of the ARGOS Challenge in March 2017. In this paper, we introduce the design guidelines and implementation for our explosion-proof robot.

Fig. 1 AIR-K prototype (non-explosion-proof version) in ARGOS Challenge



2 Design Guidelines of AIR-K

2.1 Requirements

The objectives of AIR-K are autonomous inspection of instruments during normal operations and teleoperated observations in emergency situations (e.g., gas leaks and fires). Therefore, the AIR-K requires numerous capabilities, including image processing, sound detection, temperature detection, gas detection, obstacle detection, localization, and stair traversal. To provide such capabilities, the following sensors and mechanisms were installed onto the AIR-K.

- Image processing: Six optic cameras were installed.
- Sound detection: Four ultrasound microphones were installed.
- Temperature detection: A thermal camera was installed.
- Gas detection: A gas sensor was installed.
- Localization, obstacle detection: Two 3D LIDARs were installed in the front and back.
- Stair traversal: Four subtracks were installed to negotiate stairs.

The configuration of the sensors on the robot is shown in Fig. 2.

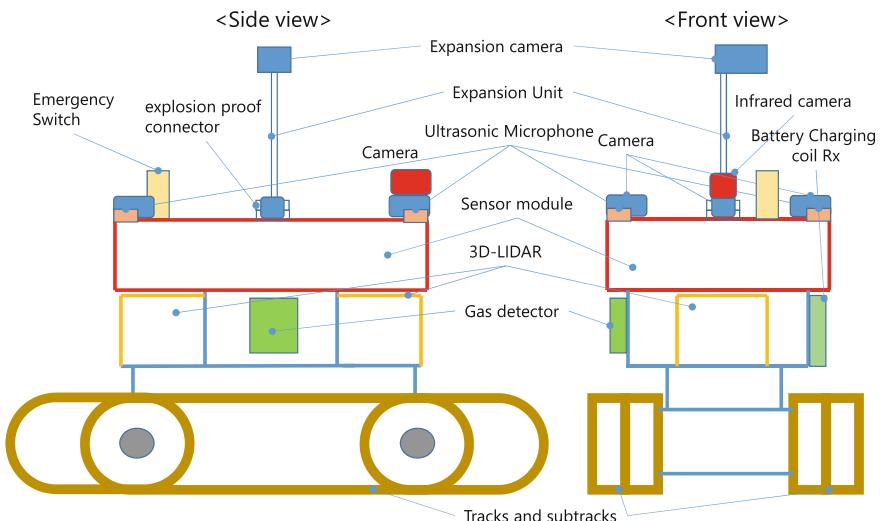


Fig. 2 Configuration of sensors and actuators on the AIR-K

2.2 Design Guidelines Providing Explosion-Proof Structure

To make the AIR-K explosion-proof, we performed design and development based on the following requirements:

1. Hazardous region: zone 1
2. Gas group: IIA
3. Temperature class: T3

where the explanation of the standards are in [2]. The hazardous region is zone 1, so the equipment protection level (EPL) should be Ga or Gb.

The AIR-K is divided into three parts to make it explosion-proof. Figure 3 shows a block diagram of the components of AIR-K. According to the features of robot functions/sensors, it uses a flameproof enclosure (Ex ‘d’), a pressurized apparatus (Ex ‘p’), and intrinsic safety (EX ‘i’); its explosion-proof capability is achieved by a combination of these methods.

The basic idea of the whole explosion proof of AIR-K is as follows:

1. A type ‘d’ flameproof enclosure protects the battery based on the philosophy of “ensuring its explosion proof capability even when the battery discharges all energy.” It is located inside the pressurized apparatus. The battery has the potential to ignite. However, when a flameproof enclosure covers it, and when the enclosure is located inside of the pressurized apparatus, it can meet the following

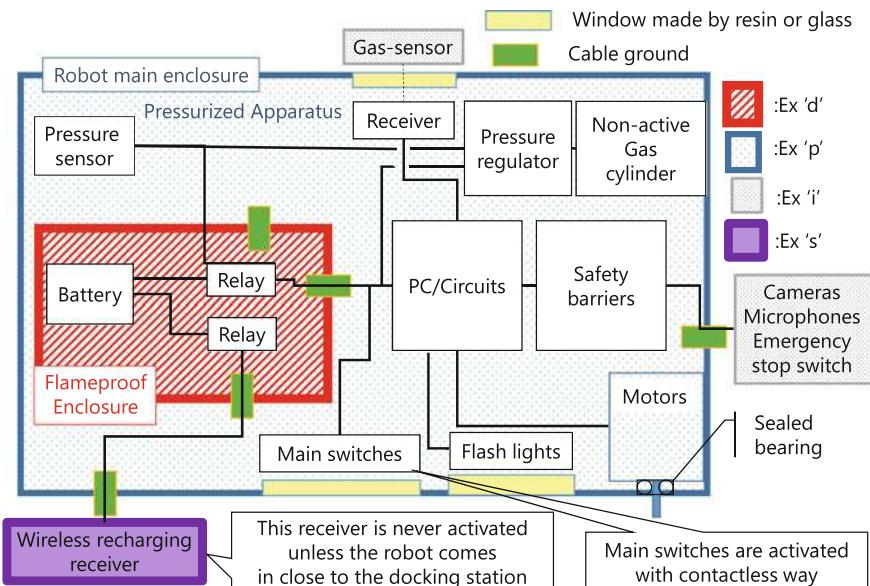


Fig. 3 Block diagram of components of AIR-K

requirement: “a device with ignitability is not built in the internal pressure container in the normal state.”

2. Some electric sensors, such as microphones and cameras, are difficult to make pressurized apparatus, individually. One possible method is that the internal pressure of the sensor is kept equal to the inside of the main body by using the through-holes. However, restrictions on design are large, and expandability is also impaired. Therefore, we configured intrinsically safe devices and located outside of the robot. The protection type is Ex ‘i’.
3. Other equipment, such as PC and integrated flashlight located inside the pressurized apparatus, do not have a flameproof enclosure. So, these are assumed to be ignition capable apparatus (ICA). Therefore, the protection type is Ex ‘p’.

We introduce additional design and implementation details for the flameproof battery enclosure in Sect. 3, the pressurized apparatus for the robot body in Sect. 4, and the intrinsically safe sensors in Sect. 5; other requirements for explosion-proof capability are discussed in Sect. 6.

3 Flameproof Enclosure for Batteries

A type ‘d’ flameproof enclosure protects the battery module. The whole module, including the flameproof container with the battery inside, is placed inside the pressurized apparatus (IEC60079-1 [3]). Thus, the module secures a protected circuit in cases where the internal pressure cannot be sustained. In addition, this circuit is connected to a pressure sensor that measures the pressure balance between the inside and outside of the explosion-proof structure. If the internal pressure is not at least +50 Pa higher than the external pressure, the relay inside the flameproof container will be switched off, so that the battery’s energy will not be discharged to the outside of the container. The protection type is ‘px,’ so this mechanism must be dual redundant. Because the battery must be rechargeable, the AIR-K’s battery is charged by a contactless electricity transmission coil. This configuration was chosen because it is assumed that the robot will be deployed on offshore platforms, where it may experience salty breezes and rain. If the battery is charged via a connected charger, there is a possibility of power leakage from a short circuit. Thus, we strongly believe that a contactless charging system is essential, even if the charging system is placed in the non-explosive atmosphere.

Figure 4 shows a battery module diagram for the AIR-K. The coil for receiving the power from the contactless charging system is placed on the outer side of the robot, which is connected to the charging circuit located inside the pressurized apparatus. The charging circuit is connected to the battery inside the flameproof container via the double protection circuit including the relay. The coil protection circuit actuates only when power is supplied from the charging station located in the non-explosive atmosphere, which makes it a non-electric component in hazardous zones. However,

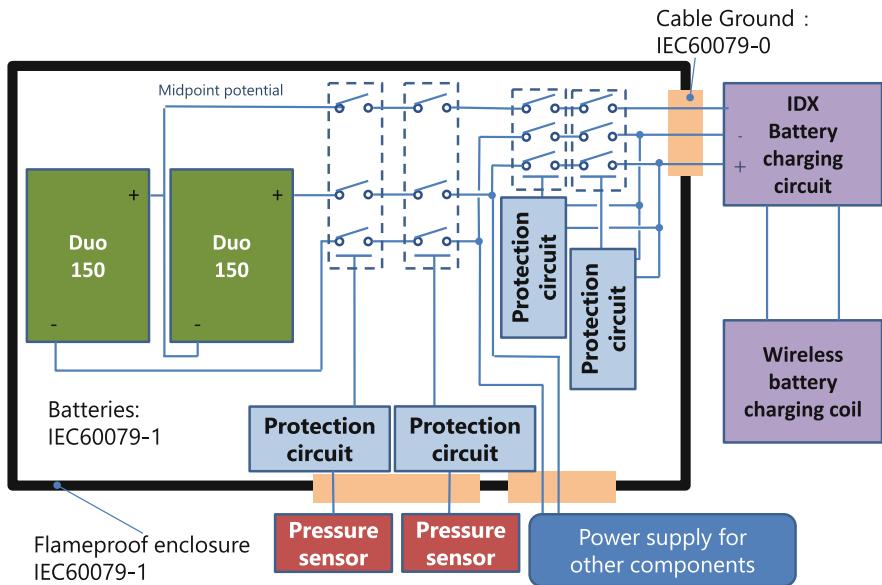


Fig. 4 Battery module diagram for AIR-K

all cables that connect the inside and outside of the flameproof container satisfy IEC60079-1 section 13 [3].

Standards for explosion-proof batteries state that cells must be connected in series only. Thus, lithium-ion batteries (L1A0N8C1: Maxell [4]) should be connected in series. However, owing to Japanese aviation regulations, we cannot transport such battery via an airplane. Therefore, as an alternative for the AIR-K, we chose a battery (DUO-150: IDX) that could be transported to the ARGOS challenge.

We conducted initial functional tests in which the robot was switched off when the internal pressure became lower than the external pressure, and confirmed that the functionality worked as we expected.

4 Pressurized Apparatus for Robot Body

The Ex ‘p’ pressurization method protects the robot body. The inner pressure should be 50 Pa higher than the outside pressure (IEC60079-2, section 7.10 “Value of over-pressure” [5]). Therefore, the AIR-K is equipped with an intake gas cylinder that contains an inert gas, and the inner pressure is adjusted by controlling a solenoid valve attached to the cylinder to supply the gas. Figure 5 shows a block diagram of the AIR-K’s internal pressure regulator. It uses a differential pressure sensor to control the inner pressure, and the threshold for opening/closing the valve is 1 kPa. Once

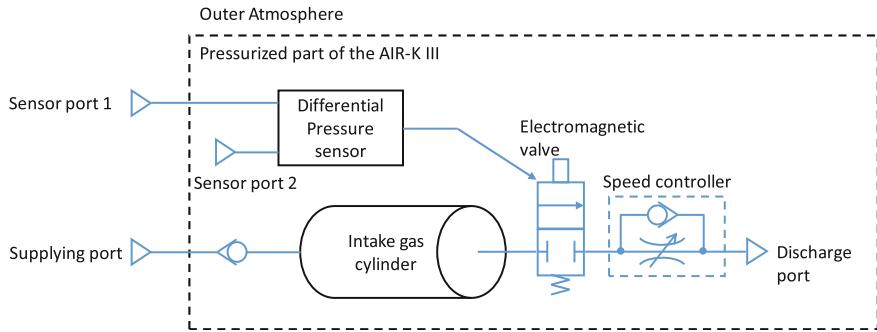


Fig. 5 Diagram of pressurized apparatus for AIR-K

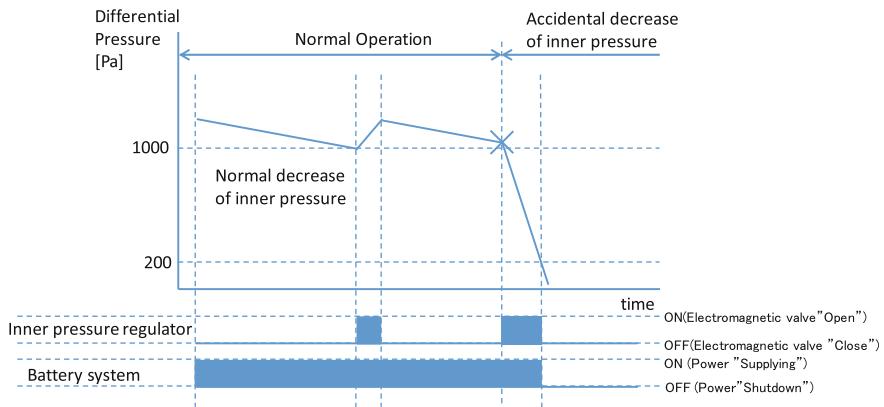


Fig. 6 Relationship between pressure sensor and solenoid valve

the robot opens the valve, the inert gas is supplied to the robot body. The threshold value can be changed, and it is a temporary value, at present.

Figure 6 shows the relationship between the differential pressure sensor and the status of the solenoid valve. When the inner pressure decreases, the robot opens the valve, and then the inner pressure increases. On the other hand, if an abnormal pressure drop occurs (the differential pressure sensor detects that the difference between the inner pressure and outside pressure is lower than 200 Pa), the system shuts down the battery module. Thus, the output voltage of the battery is never exposed to the outside of the flameproof enclosure.

As described above, the inside of the AIR-K is equipped with an intake gas cylinder to supply inert gas. Therefore, AIR-K is assumed to use a pressurized apparatus structure with a leakage compensation method. Figure 7-left shows a CAD model of the bottom part of the robot, and Fig. 7-right shows the actual location of the gas supply valve. According to Technology Institution of Industrial Safety (TISS) in

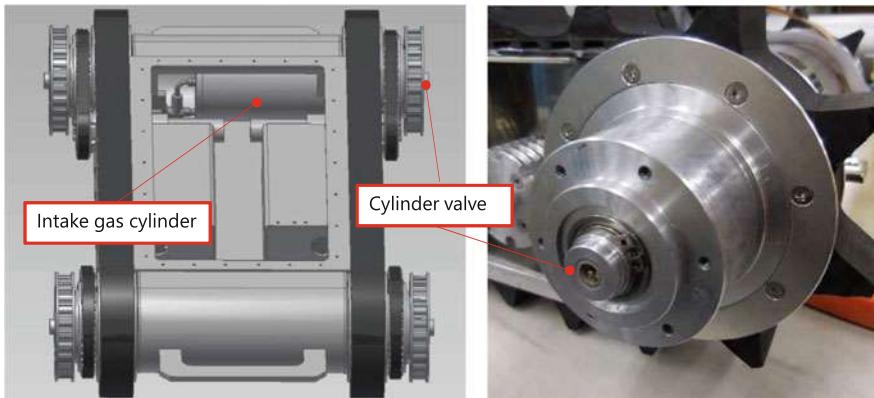


Fig. 7 Intake gas cylinder and cylinder valve

Japan, scavenging air is not required based on this method. Therefore, AIR-K does not have a function to scavenge air.

In the implementation of AIR-K's pressurized apparatus, the following values are ensured:

- Maximum: Outside pressure + 2 kPa
- Minimum: Outside pressure + 50 Pa
- Operating: Outside pressure + 1 kPa (Reference value)

The minimum pressure value of “outside pressure + 50 Pa” is the same as the value of type px in IEC60079-2, section 7.10, “Value of overpressure” [5].

To confirm the safety of the pressurized apparatus, the strength of the robot's body must be tested. The test includes a 1 kg-sharp-weight-drop from a height of more than 70 cm (IEC60079-26.4.2).

5 Intrinsically Safe Configuration for Sensors

The microphones, optical cameras, infrared camera, and gas sensor are difficult to make pressurized apparatus, individually, as described in Sect. 2.2. Therefore, these are configured as intrinsically safe devices (Fig. 8).

5.1 Intrinsic Safety Configuration for Optical and Thermal Cameras

Six optical cameras and one thermal camera are mounted on the robot to measure the instruments, to detect heat sources, and to send images for teleoperation. These

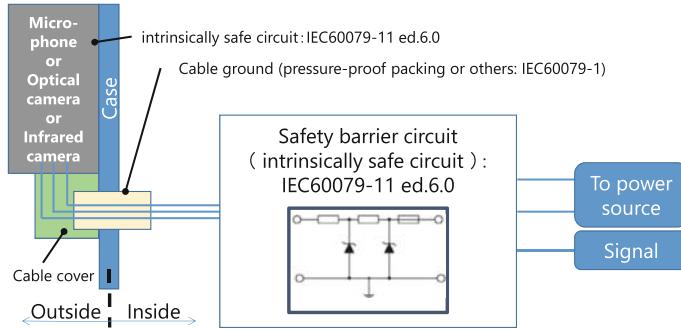


Fig. 8 Diagram of intrinsically safe configuration for cameras, etc

sensors have an intrinsic safety structure and are designed according to IEC60079-11 [6]. We install these modules outside the robot's body, and connect the signal and power lines to the controller located inside the pressurized apparatus via protected circuits. Therefore, they will not become an ignition source even when a short circuit occurs.

There are mainly two types of protected circuits: Zener barrier and isolated barrier. A Zener barrier requires an A-class earthing, so it is not applicable for mobile robots. Therefore, the robot adopts the isolated barrier so that there is no need for earthing. We configure the barrier circuit with the following components:

- Isolator: Insulates the USB communication line and power to prevent unintended ground loops.
- Current/Voltage regulation circuit: Prevents ignition even when there is a circuit failure or short circuit. Resistance and Zener diodes are used.

USB isolators (USB-029L2, HuMANDATA co., Ltd.) and DC-DC converters were selected, with a dielectric strength voltage of 2000 V. The factor of safety was 3.5, calculated from the French power source's peak voltage of 564 V. Figure 9 shows an isolated barrier circuit for USB devices.

The current limiting resistor in the barrier circuit is designed to satisfy the condition described in CENELEC (European Committee for Electrotechnical Standardization). Particularly, since the power lines send electricity, if the 5 V from the USB is supplied directly to the line, it will not satisfy the requirements. So, a voltage larger than 5 V is used to produce 5 V at the device, the electricity should be at a lower current. The value of the current limiting resistor was calculated by model-based design method.

I_{short} , the maximum current to avoid explosion when there is a short circuit, is calculated as

$$I_{short} = \frac{E}{R}, \quad (1)$$

where power voltage denotes E , and current limiting resistor R .

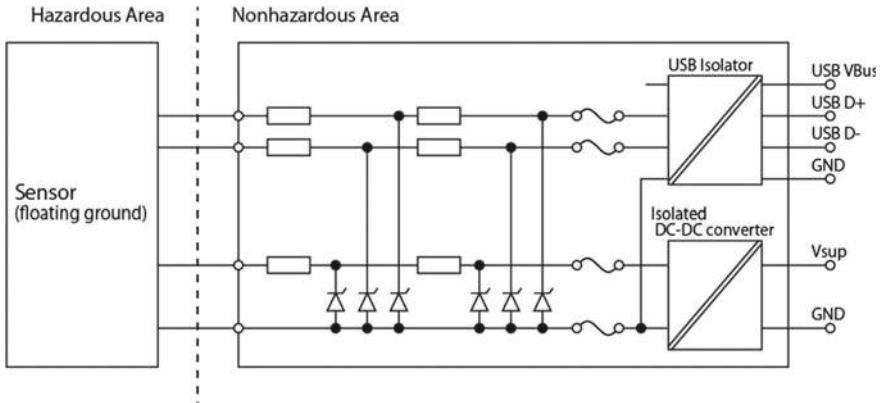


Fig. 9 Isolated barrier circuit for USB devices

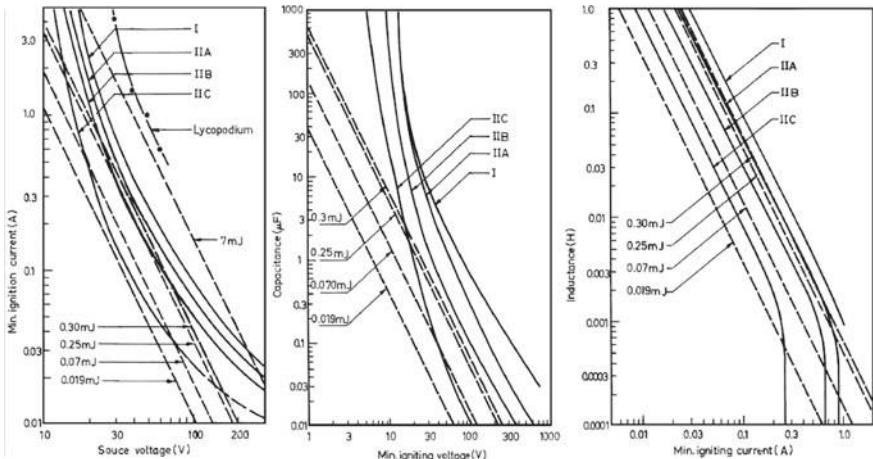


Fig. 10 Standards to determine if the system will ignite or not [7]

On the other hand, I , the current at maximum load for functioning devices, is calculated as

$$I = \frac{E - \sqrt{E^2 - 4 \cdot R \cdot P/\text{eff}}}{2 \cdot R}, \quad (2)$$

where electricity conversion efficiency (at the device) eff and the maximum power consumption (at the device) P . It is the equation to obtain the necessary current for the DC-DC converter to drive the devices under certain protection resistance and voltage. In case the resistance is too large, the power cannot be supplied, and I becomes an imaginary number. Therefore, I should be a real number to supply the power.

To avoid ignition when there is a short circuit, the circuit has to be under the Current-Inductance curve, Voltage-Electrostatic capacity curve, Current-Voltage curve, as shown in Fig. 10. The worst case scenario for ignition from induced currents is when a line is a cutoff, and the short circuit triggers another cutoff. So, I_{short} and inductance of circuit L must be below the Current-Inductance curve. The worst case scenario for ignition from the circuit capacity is when a line is a cutoff, the circuit is open, and the electrostatic capacity reaches the power voltage before triggering a cutoff. So, E and capacity of circuit C must be below the Voltage-Electrostatic capacity curve. The worst case scenario for ignition from the current/voltage of the barrier circuit has many conditions, so the design is made on the conservative side. Taking into account the worst case scenario for the current/voltage, E , I_{short} must be below Current-Voltage curve.

Based on a full search in 2-dimensional parameter space of current limiting resistor and supply voltage, these parameters are set as 66Ω and 23.8 V. Using these parameters, the factor of safety of 5 for the ignition energy can be guaranteed.

By setting the USB signal level to 3.3 V, and inserting a 36Ω current limiting resistor, the signal lines is guaranteed with a factor of the safety of 10. It is more conservative parameter.

5.2 *Intrinsically Safe Configuration for Ultrasonic Sensors*

Four ultrasonic microphone modules were mounted to detect gas leak sounds and to locate the source. This module has an intrinsic safety explosion-proof structure that is similar to those of the optical and thermal cameras, as per the standard stated in IEC60079-11 [6]. We installed these modules outside the robot's body, and connected the signal and power lines to the controller located inside the pressurized apparatus via protected circuits; this prevents them from becoming an ignition source, even when a short circuit occurs.

The microphone module consumes very little power; thus, it does not have a power line, and operates using the weak current from the signal line. Each signal line is connected to the microphone module in the isolated barrier circuit via a current/voltage limited circuit and an isolated circuit. The current/voltage limited circuit is identical to that of the USB signal lines of the optical and thermal camera modules, with a factor of safety of greater than 10 (even when a short circuit occurs).

5.3 *Gas Sensors*

A GX-2009 (RIKEN KEIKI Co., Ltd.) gas detector is mounted on the robot for detecting flammable gas. The structure of this sensor is intrinsically safe Ex ‘i’ (Exia CT4X), with a certificate from TIIS. It is also IP67-equivalent waterproof. This sensor performs infrared communication (IrDA) via a Pro Plus (ABS510700), a special

LEGASTIC IrCOMM adaptor receiver, to communicate with external interfaces. To enable the robot to detect flammable gas, we locate the detector outside of the pressurized apparatus, and locate the receiver inside the apparatus. The infrared source and receiver face each other, which enables communication through the transparent body.

We connect a power cable from the battery module to the sensor to extend the activity of the sensor. To guarantee its intrinsic safety, the maximum voltage is set to 5 V, and the current limiting resistor is set to $36\ \Omega$ to ensure that the current is within 30–50 mA (factor of safety: greater than 70).

6 Other Requirements for Explosion-Proof Operation

6.1 Dissipation of Static Electricity Buildup

The non-metal materials (plastic) used for the robot enclosure must satisfy at least one of the conditions below.

1. Surface resistivity based on the measurement method specified in IEC60079-0 section 26.13 is below $10^9\ \Omega/\text{sq}$.
2. Maximum surface area is below $10,000\ \text{mm}^2$ (Equivalent to EPL ‘Gb’).
3. Maximum layer thickness is below 2 mm (Equivalent to EPL ‘Gb’).

These conditions are intended to prevent static electricity, and are stated in IEC60079-0, section 7.4. (If the material is categorized as Gas Group II A, and Equipment Protection Level (EPL) ‘Gb.’)

On the other hand, plastic materials are used for the following sections inside the AIR-K robot: (1) Charging coil case, (2) Light transmitting plate, (3) 3D LIDAR case (includes translucent parts), (4) Wireless LAN enclosure, (5) Base plate for Gas sensor, (6) Base plate for receiver of Gas sensor, (7) Ultrasonic microphone case, (8) Thermal camera case, (9) Optical camera case, and (10) Main switch case. The surface areas of objects (2)–(10) are each below $10,000\ \text{mm}^2$. In the case of (1), the surface area of its acrylic case exceeds $10,000\ \text{mm}^2$, and the surface resistivity is $1,016\ \Omega$. Therefore, we should spray an anti-electrostatic agent on the surface to satisfy the standards.

6.2 Earthing Arrangement

The earthing arrangement is stated in IEC60079-0 (Connection facilities for earthing or bonding conductors), and equipment that needs grounding is described in section 15.1. On the other hand, equipment that does not require grounding is described in section 15.2, and the following statement is provided: “for which supple-

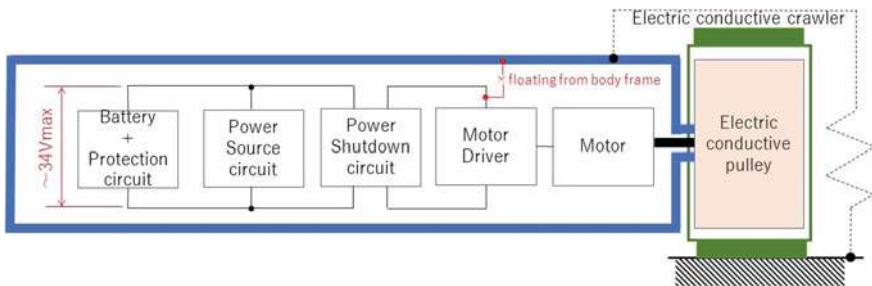


Fig. 11 Earthing structure of AIR-K

mentary earthing is not necessary, an internal or external earthing or bonding facility need not be provided.”

According to the following reasons, AIR-K can be classified as “Safety Extra Low Voltage” (SELV), and it does not require grounding.

1. There is no connection to the primary power supply in the danger zone.
2. The potential difference of the robot is a maximum of 34 V, and it does not exceed 42.4 V (which corresponds to a dangerous level of current for humans).
3. The surface of the robot has the same electric potential, and the potential difference between two arbitrary points (one point may be broken) is safe for human contact.

Figure 11 shows the earthing structure of AIR-K.

As described above, AIR-K does not require grounding. However, conductive belts should be used for tracks to satisfy high electrostatic breakdown strength, as described in IEC60079-0, section 7.4 (Electrostatic charges on external non-metallic materials). Therefore, for safety considerations, the structure has simply secured grounding.

7 Conclusions

In this study, we conducted research and development to create a mobile robot (AIR-K) as our participation entry for the ARGOS Challenge 3rd competition. The robot’s explosion-proof design was based on the IEC60079 series standard. As shown in this paper, the robot uses a flameproof enclosure (Ex ‘d’), a pressurized apparatus (Ex ‘p’), and intrinsic safety (EX ‘i’), and its explosion-proof is achieved by a combination of these methods. Figure 12 shows an image of the developed robot. Because of a problem with the locomotion, we did not use the robot in the 3rd competition but attended with a non-explosion-proof AIR-K prototype.

Because of space limitations, not all tests to confirm the robot’s explosion-proof capabilities were described in this paper. However, to obtain an explosion-proof certificate for the robot, remaining issues must be solved and additional tests must

Fig. 12 Explosion-proof robot AIR-K Ver.3



be cleared. As the main issues, the robot must consist of cells connected in series only, according to explosion-proof standards for batteries. We need to replace the serial battery configuration and conduct tests on their flameproof capabilities. Second, a thermal test inside the robot should be conducted during operations. Third, a strength test of the body should be conducted to confirm the safety of the pressurized apparatus.

In the future, we plan to develop explosion-proof mobile robots for autonomous inspection in oil plants, based on our experience from the above research and development.

Acknowledgements The ARGOS Challenge [8] supported this work.

References

1. Transport Ministry of Land, Infrastructure and Tourism (MLIT): Efforts to take a far-sighted vision of social infrastructures. <http://www.mlit.go.jp/common/001063080.pdf>
2. UL Standards: Explosive atmospheres Part 0: Equipment-general requirements. https://standardscatalog.ul.com/standards/en/standard_60079-0
3. UL Standards: Explosive atmospheres Part 1: Equipment protection by flameproof enclosures “d”. https://standardscatalog.ul.com/standards/en/standard_60079-1
4. Maxell: Lithium-ion rechargeable battery. http://biz.maxell.com/ja/rechargeable_batteries/pdf/LaminatedLi-ion_1508j.pdf
5. UL Standards: Explosive atmospheres Part 2: Equipment protection by pressurized enclosure “p”. https://standardscatalog.ul.com/standards/en/standard_60079-2
6. UL Standards: Explosive atmospheres Part 11: Equipment protection by intrinsic safety “i”. https://standardscatalog.ul.com/standards/en/standard_60079-11_6
7. Eckhoff, R.K.: Minimum ignition energy (mie)-basic ignition sensitivity parameter in design of intrinsically safe electrical apparatus for explosive dust clouds. J. Loss Prev. Process Ind. **15**(4), 305–310 (2002)
8. Kydd, K., Macrez, S., Pourcel, P., et al.: Autonomous robot for gas and oil sites. In: SPE Offshore Europe Conference and Exhibition. Society of Petroleum Engineers (2015)

Life Extension: An Autonomous Docking Station for Recharging Quadrupedal Robots

Hendrik Kolenbach and Marco Hutter

Abstract In this paper we describe the design of a fully autonomous docking station for the quadrupedal robot ANYmal. The autonomous recharging of mobile robots is a crucial feature when long-term autonomy is expected or human intervention is not possible. This is the case when a robot is used in environments that create a potential hazard to humans such as the inspection of oil rig platforms. If operated in such explosive environments, machines are usually required to be frequently purged with inert gas to avoid ignition through electric sparking (ATEX-P certification). Our docking station allows for recharging of ANYmal's battery as well as purging of its main body with gas. We present a robust docking strategy that negotiates positioning errors of the robot through guiding elements and flexible parts. The docking mechanism itself consists of an actuated plug which is inserted into a socket on the robot's belly for electrical and mechanical connection. The mechanism is designed for reliable, sealed and spark-free operation. The system has proven to be robust in a laboratory environment and under realistic conditions.

1 Introduction

The operational lifetime of autonomous mobile robots is limited to the energy supply of the system. While some robots use their own means to charge, for instance by using solar energy [1], digesting slugs [2] or carrying radioactive generators [3], most robots rely on energy provided by an external source. In environments where long-term autonomy of the mobile robot is expected, especially in industrial/home automation or exploration tasks, autonomous charging becomes a necessary asset.

Generally speaking, autonomous charging has been around since the early days of mobile robots. In the 1940s Grey Walter was using light to guide his tortoise robots into a hut, where they would make contact with a charging circuit [4]. Since then, the

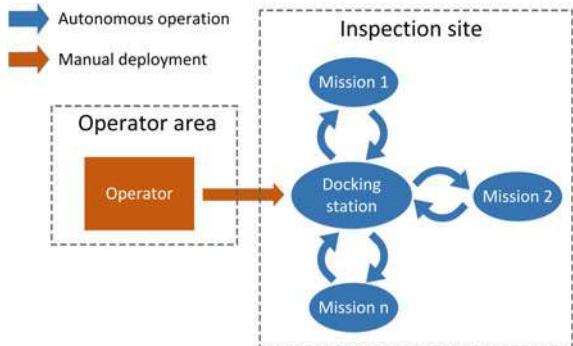
H. Kolenbach (✉) · M. Hutter

Robotic Systems Laboratory, ETHZ, Leonhardstrasse 21, 8092 Zurich, Switzerland
e-mail: hendrik.kolenbach@mavt.ethz.ch

M. Hutter

e-mail: mahutter@ethz.ch

Fig. 1 Scenario overview with docking station



general idea of using on-board sensors to identify a specifically designed docking station, move towards the station and physically dock using some sort of mechanism has not changed much [5]. Some researchers are beginning to tackle the problem more generally by trying to connect to common power outlets; however, to date this requires the robot to be fairly advanced [6]. Docking stations have recently become visible in everyday life due to their commercial application in home automation. A good example is presented by robotic vacuum cleaners or home surveillance robots [7]. These stations are designed for robots using wheel-based locomotion. Since the more recent arrival of robots using legged locomotion, the question of how to autonomously charge these systems has arisen.

So far, autonomous docking stations for legged robots have not yet been an active area of research. Most notable is the consumer market docking station of Sony's Aibo¹ robot introduced in the early 2000s. Here, the robot uses its on-board CCD camera to locate the docking station, walk over it and place its main body on the station, which matches the shape of the robot's belly. The charging contacts are exposed on the station and make physical contact with the robot once seated. Another docking station is the predecessor of this station developed in 2016 [8]. Here, the robot climbs over the station and seats itself on two 3d printed cones; the charging pins are exposed on the station. However, the system has several drawbacks such as poor success rates and its non-suitability for outdoor use.

Since legged robots provide an advantage over wheeled systems when it comes to negotiating with complex environments, it is only a matter of time until these systems are used extensively for demanding tasks such as inspection of disaster sites. This presents a scenario in which the system should be able to explore and operate for several hours without the need for human interaction (Fig. 1).

Such a task, an autonomous inspection of an oil-rig platform, was the goal of the ARGOS (Autonomous Robot for Gas & Oil Sites) challenge organized by Total.² The third and last round of the 3 year competition took place from March 9–17, 2017 at Lacq, France and presented a good opportunity to validate the docking station

¹<http://www.sony-aibo.com/>.

²<http://www.argo-challenge.com/>.

and our robot in a realistic scenario [9]. Our robot, ANYmal, is a quadrupedal robot specifically designed to fulfill inspection tasks in harsh environments [10]. The robot is equipped with two rotating Hokuyo laser scanners for localization and an actuated sensor head to perform inspection tasks. The recently updated power source consists of a 12 S 5P Panasonic NCR 16850-GA Lithium-Ion battery with a capacity 745 Wh, which provides the robot with roughly 2.5 h of autonomy under nominal load. Since the robot is operated in a potentially explosive atmosphere, it is necessary to purge and pressurize its body with inert gas [11].

In the following chapters, we describe the development of a new autonomous and robust docking station to charge the quadrupedal robot ANYmal with energy and purge the main body with inert gas. We introduce the system design, the docking strategy and experimental results.

2 System Design

The docking station consists of several subsystems of which the most important ones can be defined as follows: *Docking Mechanism*, *Sensing*, *Actuation and Control* and the *Pneumatic system*. All of these are integrated inside the station (Fig. 2), which has a size of roughly $1\text{ m} \times 0.14\text{ m} \times 0.27\text{ m}$ (Length \times Width \times Height) and a weight of 16.17 kg (without gas bottle).

A safe and secure docking is of high importance for reliable operation of the robot and for human interaction with the system. The hardware is intrinsically designed to not have any exposed contacts at high power unless a reliable connection is made.

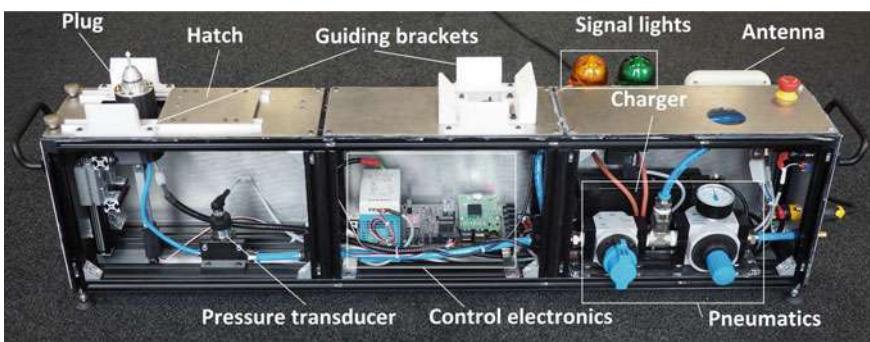


Fig. 2 The picture shows the hatch and plug assembly (docking configuration) on the left, the control electronics in the middle and the pneumatics and charger on the right. Handles, guiding brackets and signal lights are also visible

2.1 Docking Mechanism

Frame

The frame has to fulfill several functions. First, the dimension must be small enough for the robot to walk over it while providing enough space to house all components in a waterproof environment. To protect the charging plug and to keep the housing sealed, a linearly actuated hatch is mounted on top. An important feature of the frame is the guiding brackets. These brackets are used to compensate linear alignment errors of the robot of up to 20 mm. The corresponding interface on the robot is the belly plate, which is developed for shock absorption in the event of fall.

The brackets are made from polyoxometalate (POM) for low friction, and are adapted to match the design of the belly plate. To sense if the robot is correctly seated on the docking station, an inductive sensor is placed on top of the station between the guiding brackets. To visually notify the operator about the current status of the station, a yellow and green signal light is attached to the frame. A long-range wireless connection is made with a directional antenna. Handles are mounted for easy transport and adjustable feet are mounted below the station to ensure a stable stand on uneven terrain. To set the initial status of the station, three buttons are mounted. One button is used to power on and off the station, another to activate the fully autonomous mode and one button that enables the purging and pressurization procedure whenever the robot is docked. An emergency stop is installed that can cut power and stop all operation.

Plug

The plug (Fig. 3a) is housed inside the docking station and can be moved linearly through the opened hatch against the robot once it is in position. The main functions are to provide a stable connection to the socket and to allow energy and gas flow to the robot. The shape of the plug and socket was partially inspired by the mechanisms for spacecraft docking [12]. The round top and the conically shaped part allows for precise positioning of the plug with respect to the socket. To overcome the remaining tolerances, the plug is situated on top of a flexible rubber part which allows a certain freedom of motion. Four pogo pins are placed on the plug in a special, arbitrary pattern to avoid reverse polarity. Two pins are used for detecting contact and two to transmit the power. The contact detection pins are designed slightly shorter than the other two. This ensures that during docking, the shorter pogo pins are in contact after the power pins, as well as ensures they are the first ones that will lose contact when undocking. This is important to avoid sparking on the power pins.

Socket

The counterpart of the plug is attached to the belly of the robot. The socket assembly consists of a plastic part, two printed circuit boards (PCB), a lid, a check valve and a sealing (Fig. 3b). The plastic part, like the brackets, is made from polyoxometalate

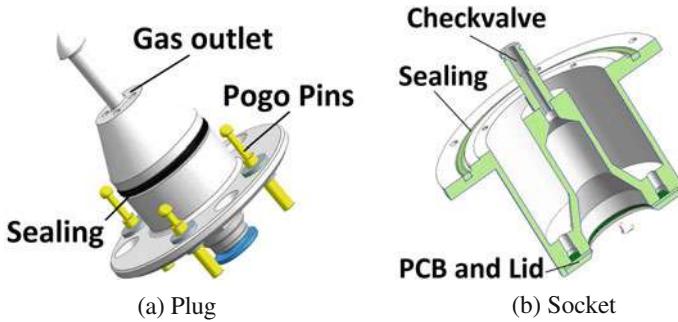


Fig. 3 The figure shows the plug (a) and the socket (b) with matching shape

(POM) for low friction. The inner shape matches the outer shape of the plug to guide it to a defined position. A circuit board with gold-plated contacts is located on the bottom of the socket part, partially covered by a lid. The bottom circuit board serves as the interface to the pogo contacts of the plug. A second circuit board (not visible in the figure) is located on top of the socket inside the robot. This board houses a passive set of electronics (relays and diodes), that are powered by the docking station. The electronics ensure a safe connection to the battery once contact is established and verified. The check valve ensures the gas flow is directed inwards. The socket is mounted from the outside to the main body of the robot and is sealed to avoid gas leakage.

2.2 Sensing, Actuation and Control

The docking station is controlled by a Raspberry Pi 3 situated on a custom circuit board, which interfaces the connected peripherals (Fig. 4). We use the Raspbian Jessie³ distribution as OS and installed ROS Indigo.⁴ We programmed a docking station node for ROS in the form of a state machine, which is started once the station is powered on and the program start button is pressed. The docking station node is incorporated in the overall system architecture, which includes ANYmal and the operator PC. Hence, easy communication and monitoring of the status of both ANYmal and the docking station is possible and available to the operator.

The controller board is the interface to the electromechanical components. The controller is able to move the linear actuators attached to hatch and plug, to indicate the status of the station via yellow and green signal lights, as well as read the various sensors (Induction, Pressure, Current, Contact) and the buttons mounted on the frame.

³<https://www.raspberrypi.org/>.

⁴Robotic Operation System—<http://wiki.ros.org/>.

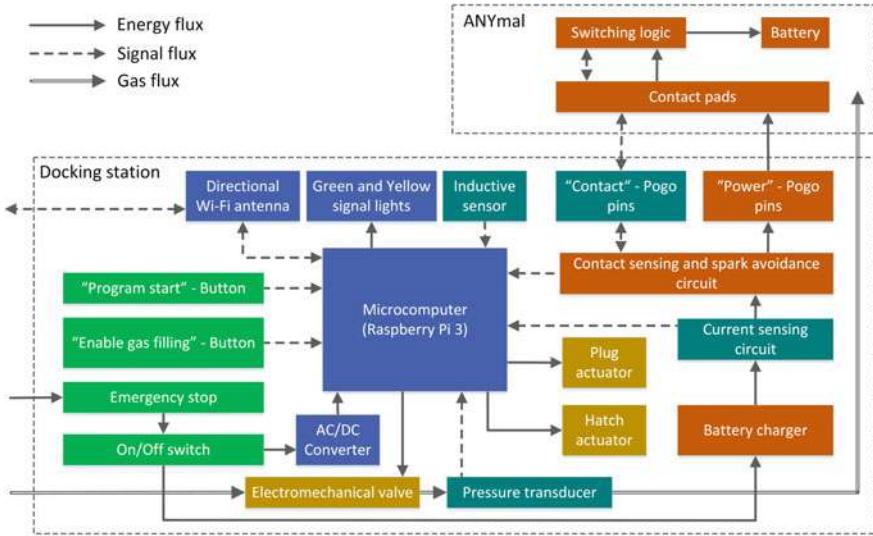


Fig. 4 Schematic overview of the electrical components and interfaces

We also developed an analog circuit that allows the sensing of passive electronics inside the robot.

This circuit inside the robot, which is mounted on the socket (Sect. 2.1), is decoupled from the rest of the system and can only be powered by the contact pins once contact is established. This allows one to safely open relays inside the robot and inside the docking station and minimize, together with the mechanical design of the pogo pins, the risk of sparks and voltage peaks. The current flow through the battery charger is measured and visible to the operator. Depending on the state of the gas filling enabling switch, the purging procedure can be started or neglected.

2.3 Pneumatic System

We can purge the robot with inert gas (nitrogen) provided by the docking station. Part of the pneumatic installation is installed inside the docking station, while additional parts are placed inside the robot (Fig. 5).

The docking station setup consists of a gas cylinder (C1) with a pressure gauge (P1); a setup of two pressure regulators to set operating pressure, minimize oscillations and the supply pressure effect (R1, R2); two respective pressure gauges (P2, P3); an electromechanical valve (Q1) with suppressor (R3); a manual switch connected in parallel (S1) and a pressure transducer (B1). The pneumatic system interfaces the robot through the plug mentioned in Sect. 2.1.

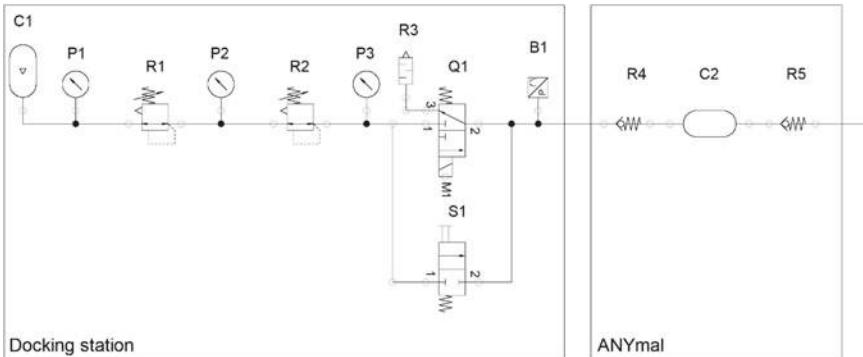


Fig. 5 Overview of the pneumatic system

The robot (C2) is equipped with two check valves (R4, R5). One check valve (R4) is connected to the socket with the flow-direction pointed inwards, while the other (R5) is connected to the front of the robot, directed outwards. Once a sealed connection between the plug and socket is made and confirmed, the docking station can start the purging procedure. The electromechanical valve (Q1) opens while the charging pressure is constantly monitored through the pressure transducer (B1). The two pressure regulators (R1, R2) are preset to maintain a constant output pressure. The check valve is integrated in the socket and opens once a pressure difference between working pressure and robot is reached. The nitrogen level inside the main body rises until the pressure difference between outside and inside is reached and the second check valve (R5) opens to release overpressure.

The point in time after the check valve opens is monitored by the docking station control. The valve stays open and closes within discrete time steps until a minimum of five times the robots volume equivalent of gas has purged through the body. After the successful purging, the electromechanical valve (Q1) closes, remaining pressure between the valve and the check valve (R4) is relieved through the suppressor (R3) and the robot's main body remains pressurized. As an alternative to the automatic docking procedure, the electromechanical valve can be bypassed by a manual on-off valve (S1). Additionally, the pressure inside the main body is constantly measured and monitored through the onboard electronics.

3 Docking Strategy

Deployment

On entering an inspection site, ANYmal can be transported seated on the docking station inside a wheeled box. The box can be moved by one person by pulling it by a metallic handle. The box is designed to overcome pavement edges of up to 8 cm.

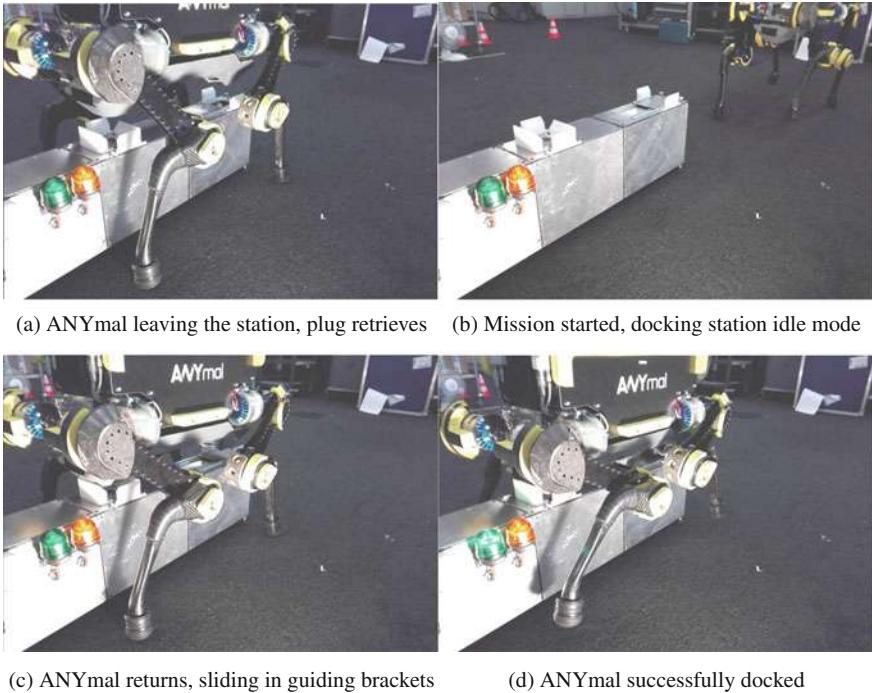


Fig. 6 The docking sequence

Once close to the area of interest, ANYmal and the docking station can be lifted from the box and transported by two people to the destination point by using two handles attached to the docking station.

Once deployed (Fig. 7), the robot starts localizing itself within the environment. Therefore, the Iterative Closest Point (ICP) algorithm [13] is used to match the local scan (perceived by the two rotating Hokuyos in the front and in the back) to the reference map. Starting from an initial guess of the robot's pose within the reference frame, the algorithm iteratively searches for the nearest neighbors between local and reference point clouds and minimizes the sum of all squared distances. The resulting absolute pose of the robot is used to correct the robot's odometry during the entire mission. In this case, the pose of the robot in seated configuration on the docking station is stored for future reference. After this, the robot lifts its body and walks off the station to start the mission (Fig. 6a). The docking station automatically retrieves the plug and closes the hatch once the robot is not sensed anymore. This state is called the idle state (Fig. 6b).

Approach and Docking

When the mission is finished or a low battery/pressure status is detected, the robot autonomously returns to the docking station. The target pose of the robot is the same as the starting pose during the deployment phase. The robot approaches the docking station and walks over it by using a special maneuver [14]. A two-step strategy is used in order to position the robot with high precision and account for potential localization errors. The first step is to position the robot above the guiding brackets, which can passively compensate misalignment errors of a few centimeters (Fig. 6c). The robot slides in a seating position and the successful seating of the robot is sensed by the inductive sensor of the station.

In the second step, the hatch on top of the station opens and the plug moves towards the socket. The plug has the ability to negotiate remaining, small misalignment errors of a few degrees/millimeters thanks to a rubber part that allows for certain flexibility. The successful docking is confirmed by two pogo pins on the plug, which sense the passive electronics of the socket (Fig. 6d). In case a confirmation was not received, the plug will retreat a few centimeters and tries to dock again. If docking was not successful following the third try, the station would return to the idle state. Since the operator will be aware of the situation, he could then try to walk-off the station and restart the docking maneuver.

Charging

Once the connection is made, the robot's batteries are automatically charged. Additionally, the robot can be pressurized with gas. This is important for ATEX-P certification, which the robot has to comply with when operating in areas with a potentially explosive atmosphere [11].

During this process, the main body of the robot is purged several times its volume with an inert gas (i.e. nitrogen) and has to remain pressurized for the time of operation. Both, current flow and the amount of gas flowing in the robot is measured and available to the operator. Signal lights indicate the status of the station. A flashing yellow light indicates the move of the actuators and a flashing green light indicates a successful docking and charging.

Undocking

In nominal operation, the robot stays connected through the plug on the docking station and batteries are constantly charged. In case the robot is lifted off or commanded to continue its mission, the station automatically returns to idle state. If necessary, the station can also be shut down remotely or manually while the robot is seated.

4 Experiments and Results

We used the docking station extensively during the last round of the ARGOS challenge. The scenarios in which the robot and the docking station were tested include the autonomous inspection of an oil rig platform under realistic environmental conditions. We deployed the robot with the docking station (Fig. 7), and several missions were started and finished on it. We carried out the charging with both gas and energy successfully.

Additionally, we performed end-to-end tests in a laboratory environment. For the first two experiments, the station was not powered and no docking was performed. We tested if the robot was able to walk off and on the station, and if the guiding brackets would allow a defined positioning of the robot. In the next four experiments, the docking station was activated and the robot walked off and immediately returned to the station without an intermediate mission simulation. In the following four experiments, the robot walked off, trotted in front of the station, and returned. The docking station was able to perform autonomous docking with confirmed contact in all eight cases (Table 1). In one case, the automatic retreat and re-dock mode as mentioned in Sect. 3 was triggered since the pogo pins did not confirm contact at first. Here, the plug retreats slightly and approaches the socket a second time while the robot remains seated. We observed this, albeit rarely, at the competition which is why we implemented the procedure. To guarantee a 100% docking at first try, further improvements could be made through optimization of the flexible part (allowing more flexibility).

Fig. 7 ANYmal in seated configuration on the docking station at the ARGOS challenge



Table 1 Results of docking station end-to-end testing

Exp. no	Task	Docking station	Result	Comment
1	Walking off and immediate return	Inactive	Success	Robot seated correctly
2	Walking off and immediate return	Inactive	Success	Robot seated correctly
3	Walking off and immediate return	Active	Success	
4	Walking off and immediate return	Active	Success	
5	Walking off and immediate return	Active	Success	
6	Walking off and immediate return	Active	Success	
7	Walking off, trotting, and return	Active	Success	
8	Walking off, trotting, and return	Active	Success	
9	Walking off, trotting, and return	Active	Success	
10	Walking off, trotting, and return	Active	Success ^a	

^aThe docking station docked successfully after short retreat and re-dock of the plug (Chap. 3)

Though never observed, one can imagine a case where the ICP algorithm fails to localize the robot with high certainty, e.g., in flat, open areas. In this case, the docking station could be surrounded by geometric items, i.e., poles. Alternatively, CV tags could be used to localize the station directly, or the guiding brackets could be enlarged. Overall, our experiments confirm the high robustness of the system, which we also observed during the ARGOS challenge.

5 Future Work

The docking station could be upgraded with additional features in the future. It can be used as a communication bridge between the operator and the robot to enlarge reach of the Wi-Fi and, consequently, the operational area. Additionally, sensors could be mounted on the station to monitor the environment in close proximity and inspect the robot for potential damage. In the event that the station is used in a scientific scenario, one can think of having additional instrumentation to facilitate the analysis of samples brought by the robot.

The current version is connected with a cable to an external power supply and with a hose to a gas bottle. The last building block for a completely independent and autonomous operation would be achieved by incorporation both a gas bottle and a large battery pack.

6 Conclusion

We introduced the first autonomous, outdoor-proof docking station for quadrupedal robots. Through testing and use in a realistic scenario we showed the capability of the system to safely charge the robot's batteries and purge and pressurize the main body.

The system is capable of increasing the robot's operational time in hazardous environments without human intervention. Furthermore, the docking station provides a solid base for future implementation of additional features.

References

1. Landis, G.A.: Exploring mars with solar-powered rovers. In: Conference Record of the Thirty-first IEEE Photovoltaic Specialists Conference 2005, pp. 858–861 (2005)
2. Kelly, I., Melhuish, C.: Slugbot: A robot predator. In: The 5th International Symposium on Artificial Life and Robotics for Human Welfare and Artificial Liferobotics, pp. 470–475 (2000)
3. Welch, R., Limonadi, D., Manning, R.: Systems engineering the curiosity rover: a retrospective. In: 2013 8th International Conference on System of Systems Engineering, pp. 70–75 (2013)
4. Walter, W.: The living brain (1953)
5. Silverman, M.C., Nies, D., Jung, B., Sukhatme, G.S.: Staying alive: a docking station for autonomous robot recharging. In: Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292), vol. 1, pp. 1050–1055, vol.1 (2002)
6. Nagatani, K., Yamada, T., Tanaka, Y.: Long-term activities for autonomous mobile robot. In: Field and Service Robotics, Results of the 5th International Conference, FSR 2005 (2005)
7. Song, G., Wang, H., Zhang, J., Meng, T.: Automatic docking system for recharging home surveillance robots. *IEEE Trans. Consumer Electron.* **57**(2), 428–435 (2011)
8. Ahlberg, M.: Design of an autonomous docking station. Bachelor thesis, ETHZ (2016)
9. Hutter, M., Diethelm, R., Bachmann, S., Fankhauser, P., Gehring, C., Tsounis, V., Lauber, A., Guenther, F., Bjelonic, M., Isler, L., Kolvenbach, H., Meyer, K., Hoepflinger, M.: Towards a generic solution for inspection of industrial sites. In: Field and Service Robots (FSR) (2017)

10. Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, C.D., Tsounis, V., Hwangbo, J., Bodie, K., Fankhauser, P., Bloesch, M., Diethelm, R., Bachmann, S., Melzer, A., Hoepflinger, M.: Anymal—a highly mobile and dynamic quadrupedal robot. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 38–44 (2016)
11. Explosive atmospheres—part 2: equipment protection by pressurized enclosure “p”. Standard, International Organization for Standardization, Geneva, CH (2014)
12. Portree, D.S.F.: Mir hardware heritage. Technical report, NASA Johnson Space Center (1995)
13. Pomerleau, F., Colas, F., Siegwart, R., Magnenat, S.: Comparing ICP variants on real-world data sets. *Auton. Robots* **34**(3), 133–148 (2013)
14. Fankhauser, P., Dario Bellicoso, C., Gehring, C., Dub, R., Gawel, A., Hutter, M.: Free gait: an architecture for the versatile control of legged robots. In: 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), pp. 1052–1058 (2016)

Autonomous Mission with a Mobile Manipulator—A Solution to the MBZIRC

**Jan Carius, Martin Wermelinger, Balasubramanian Rajasekaran,
Kai Holtmann and Marco Hutter**

Abstract This work presents the system and approach we employed to tackle the second challenge of the Mohamed Bin Zayed International Robotics Challenge (MBZIRC) (See <http://www.mbzirc.com/challenge>). The goal of this challenge is to find a tool panel on a field, pick an appropriate wrench from the panel, and operate a valve stem therewith. For this purpose we use a task-oriented field robot, based on Clearpath Husky with a customized series elastic arm, that can be deployed for versatile purposes. However, to be competitive in a robotic challenge, further specialization and improvements are necessary to achieve a certain task faster and more reliably. A high emphasis is put on designing a system that can operate fully autonomously and independently respond if a subtask was not executed successfully. Moreover, the operator can easily monitor the system through a graphical user interface and, if desired, interact with the robot. We present our algorithms to explore the field, detect the panel, and navigate to it. Furthermore, we use a support vector machine based object detection method to locate the valve stem and wrenches on the panel for visual servoing. Finally, we show the advantages of a force controllable manipulator to handle the valve stem with a tool. This system demonstrated its applicability when fulfilling the entire task fully autonomously during both trials of the Grand Challenge of the MBZIRC 2017.

1 Introduction

Robotic competitions and challenges give a strong motivation to foster innovation and research and to aim for a long-term goal. They support development and real-world

J. Carius · M. Wermelinger (✉) · B. Rajasekaran · K. Holtmann · M. Hutter
Robotic Systems Lab, ETH Zurich, Switzerland
e-mail: martin.wermelinger@mavt.ethz.ch

J. Carius
e-mail: jan.carius@mavt.ethz.ch

verification of single components, but also force to integrate those components into a complete system. This may be a reason why these contests gained more and more popularity over the past years and spread over various areas of robotics. Contests like *Cybathlon* [16], *DARPA Robotic Challenge (DRC)* [8], *ELROB* and *euRathlon* [17], *European Robotic Challenges (EuRoCs)* [18], *RoboCup* [12], and *RoCKIn* [13], to mention only a few, show the state of the art but also catalyze new developments. Furthermore they present a great opportunity to promote the progress in robotics to a broad audience. However, robotic challenges may also drive the technological innovation into an “reductionist design” [2], where simplistic but custom-tailored solutions are applied instead of generic approaches. Finally they expose clear weaknesses of current systems that require help from several humans to be set up and supervised.

This work presents the approach and system design that we used to master the second challenge of the MBZIRC 2017. The task of this challenge is to use an unmanned ground vehicle (UGV) to locate a wrench panel on a 60×60 m outdoor field, navigate to it, select and grip a suitable wrench hanging on the panel, and turn a valve stem 360° therewith. A special focus of this challenge lies on complete autonomous behavior, since any intervention by a human operator will directly result in significant score penalties. We decided to utilize a generic mobile manipulation platform that can be used for a variety of tasks. The modifications made on hardware and software are modular and complement the base system.

The remainder of this work will cover our contributions to accomplish the challenge task: design of a mobile robot for handling wrenches (Sect. 2); autonomy and mission state machine (Sect. 3); detection of the panel (Sect. 4); exploration, navigation, and positioning (Sect. 5); object detection and visual servoing (Sect. 6); valve manipulation (Sect. 7); and finally we conclude with the performance during the Grand Challenge.

2 Hardware

The system used for this robotics challenge consists of the four-wheeled skid-steer platform Husky (Clearpath Robotics), equipped with a custom six degree of freedom (DoF) manipulator arm (see Fig. 1 left). For state estimation, localization, and navigation in unknown environments the UGV is equipped with wheel encoders, a Microstrain 3DM-GX4-15 inertial measurement unit (IMU), a Velodyne HDL-32E light detection and ranging (LIDAR) sensor, and optionally a Garmin GPS-18x global positioning system (GPS) receiver. The arm is an advancement of the previously presented ANYpulator [1] with newly three DoF allocated at the wrist of the manipulator. All joints of the arm are actuated by the same high-performance series elastic actuator (SEA) units. These actuators, called ANYdrives [10], are composed of a brushless high-torque motor, harmonic drive gear, and a torsional spring. The usage of SEAs adds inherent compliance to the system, making it especially suitable for interaction tasks such as valve manipulation. The computational payload for

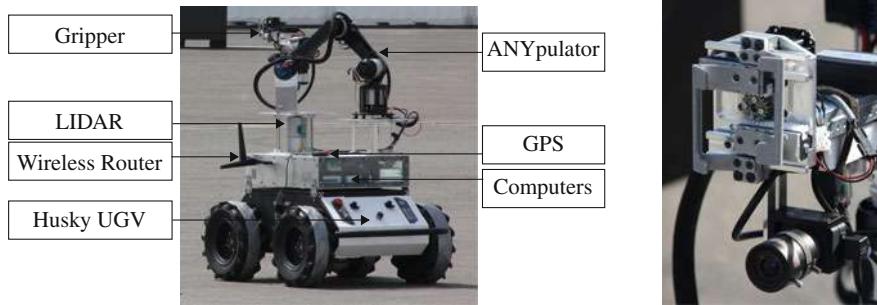


Fig. 1 **Left:** Integrated mobile manipulator platform with six DoF arm. **Right:** Clamp design for grasping wrenches with attached camera for visual servoing

navigation, arm control, and perception is distributed on three on-board computers, which connect to an operator computer through a wireless network.

Specifically designed for the task of the challenge is the end-effector tool for wrench grasping (see Fig. 1 right). The gripper's mechanical design is optimized to provide a lightweight but effective tool for holding the shafts of standard wrenches. A hollow aluminum structure attaches to the final joint of the robot arm and provides mounting points for a camera and a clamping mechanism. A single Dynamixel MX-64 Motor drives two sheet metal clamps through a rack and pinion mechanism. This ensures that the closing position is centered laterally with the end-effector reference frame. Additionally, the clamps are designed to center the head of a gripped wrench with the rotation axis of the last joint. This facilitates pure rotation motion of the tool, e.g. for turning a screw or the valve stem. Sensing of motor current and position allows to determine whether or not an object is successfully held by the gripper. For visual servoing, a calibrated monocular camera (PointGrey Chameleon 3) is attached to the aluminum shaft below the clamps, such that a grasped wrench does not obstruct the view of the camera.

3 Autonomy and Mission Control

A typical mission that we expect our robot to fulfill involves several sequential steps such as exploration, detection of the manipulation site, navigation, positioning, and manipulation itself. Tasks of this complexity call for a modular mission design since each sub-task may fail and needs to be repeated. Figure 2 displays our mission state flow diagram for the second task of the MBZIRC. Each state is responsible for a specific sub-task and is accompanied by a supervision module (not shown in the figure) that decides if the completion was successful or not. In case of failure specific recovery behaviors may be executed or the state machine simply transitions back to the previous task and starts another attempt. This architecture also allows the

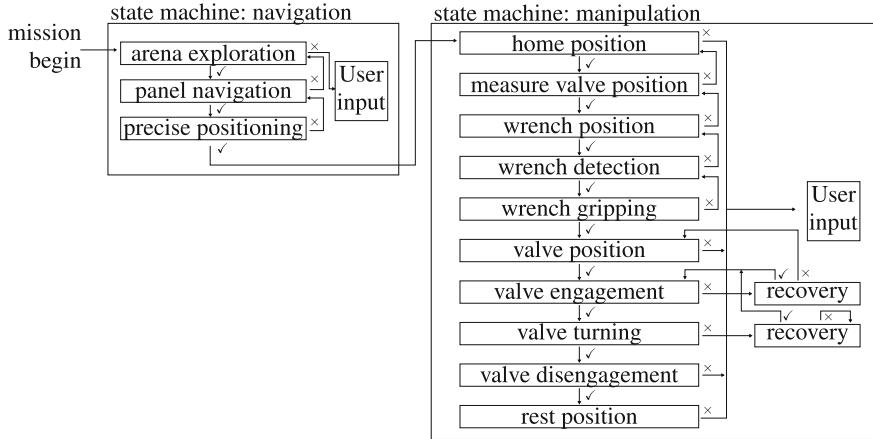
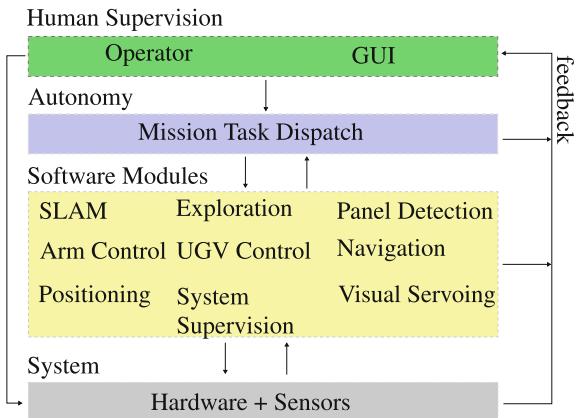


Fig. 2 State flow during the MBZIRC mission. Each state may succeed (✓) or fail (✗). On failure, the robot will execute the previous task again or request user input

Fig. 3 Hierarchical software architecture and abstraction. The operator can interact with the mission task dispatcher or teleoperate the system directly



interruption of a current task on predefined conditions, for example when the battery runs low or the operator requests the mission to stop (Fig. 3)

The operator can monitor the robot through an additional operator computer, which is connected to the robot's wireless network. A graphical user interface (GUI) displays information from all relevant software modules and diagnostic information such as battery charge state and temperatures. Additionally, a visualization of the constructed environment map and the image stream from the camera are displayed to the operator. At any time, the operator may interrupt the autonomous mission and provide manual commands through a joystick or through the GUI. This empowers an user to issue specific commands such as closing or opening the gripper, switching to a specific mission state, or controlling the manipulator or mobile platform.

All mission critical software is running on the robot itself and is independent from the quality of the WiFi network. Therefore, even under limited network connectivity, the robot maintains its autonomous behavior. For safety, the robot pauses if there is a connection time-out to the joystick.

4 Wrench Panel Detection

An important element for succeeding in the navigation part of the MBZIRC is the correct identification of the wrench panel and its subsequent pose estimation. Our method relies on 3D point clouds acquired from a laser scanner.

4.1 Point Cloud Processing

The panel detection software operates on any kind of point cloud (e.g., constructed map or raw scan) with the only requirement that the vertical direction of the cloud is known. For the MBZIRC, our setup directly uses raw scans, obtained by assembling laser data from one revolution of the scanner.

To identify the panel, we make use of the known geometric dimensions of the rectangular sides as given by the challenge description. Our algorithm can identify any planar side of the panel by executing the following operations:

1. The point cloud is cropped to a box with side length of 10 m around the robot. This reduces the computational requirements and prevents detection of false positives.
2. A normal vector is assigned to each point through principal component analysis of the covariance matrix from neighbors of the query point.
3. Region growing using conditional Euclidean clustering assigns each point to a cluster, thereby segmenting the point cloud. Two points share the same cluster if they are sufficiently close (Euclidean distance) and their normals are aligned.
4. In each cluster, random sample consensus (RANSAC) [6] plane extraction singles out individual planes. We reject non-vertical planes and those that lie outside of our search space (specified as a polygon corresponding to the arena geometry).
5. For each of the remaining planes, we check if its dimension matches any of the rectangular sides of the panel. Finally, we confirm that the point distribution across the plane is approximately uniform to reject any remaining artifacts.

4.2 Pose Estimation

If all of the above tests are successful, the identified plane, together with its orientation and position, is passed to an estimation module which keeps track of several

detections. The individual detections (*measurements*) of the panel pose are fused to potential matches (*candidates*). Each candidate i holds an internal state, containing the pose of the front plane of the panel with respect to a map-fixed reference frame and an improper probability $p_i \in [0, 1]$ indicating how likely it is the true match. New measurements will update existing candidates if there exists a sufficiently close one, otherwise a new candidate will be initialized with probability p_{init} .

- (a) **Initializing Candidates:** Upon construction of a new candidate, the pose information of the measurement is transformed to an equivalent pose of the panel's front side by applying a suitable translation and rotation. If the initial measurement is ambiguous, i.e., the algorithm detected the side of the panel, one cannot decisively infer if the detection corresponds to the right or left part. In this case, an arbitrary choice is made and the state of this candidate is set to ambiguous.
- (b) **Updating Candidates:** The first step in updating a candidate is again transforming the measurement to an equivalent pose of the front side of the panel. In ambiguous cases, the rotation is chosen such that it supports the current hypothesis. If a previously ambiguous candidate receives an update by a conclusive measurement (e.g., the front or back side), the ambiguity can be resolved. This potentially involves flipping front and back side if the wrong initial choice was made. The position is updated through a first order filter and the rotation through spherical linear interpolation and the probability is increased by p_{update} .

At each iteration, the candidate manager reorders its list in descending probability order, normalizes the probabilities such that the highest does not exceed 1.0, and removes candidates with probabilities below a given threshold. We accept the most likely candidate 0 if its probability $p_0 > p_{\text{threshold}}$ and it is significantly more likely than the next best candidate $p_0 - p_1 > p_{\text{diff}}$, if any. The parameters fulfill

$$0 < p_{\text{init}} + p_{\text{update}} < p_{\text{threshold}} < p_{\text{init}} + 2p_{\text{update}} < 1 , \quad (1)$$

hence at least three detections are necessary for acceptance of a candidate.

4.3 Evaluation

Our approach correctly identifies the pose of the wrench panel from a distance of 10 m, which is approximately the point at which the rectangular sides become visible in the point cloud visualization for a human observer.

The computational load of processing point cloud data allows update rates of 0.6 Hz on an Intel i7 processor, which places an upper bound on the speed at which the robot can drive past the panel without missing it. In practice, the maximum speed of our ground vehicle (1.0 m/s) does not cause any problems in this respect.

A major difficulty for the detection algorithm arises from occlusions in the laser scan. Our scanner is protected by a cover which is held by three vertical pillars,

casting conic shadows with opening angles of 7°. If the panel happens to fall inside this region, the occlusions render the detection difficult or at times impossible.

Apart from detections in the laser scan, the estimation module accepts panel measurements from several sources. While we did not employ this in the actual challenge, it would have been possible to take input from a user or measurements made by drones about the potential position of the panel. The rejection mechanism of single false positives proves essential in many cluttered environments.

5 Navigation

We applied a custom navigation routine to drive and position the robot in front of the wrench panel. This routine makes use of underlying iterative closest point (ICP) based localization and mapping [14, 15] and the ROS navigation stack¹ for path planning and following. The navigation is divided into two main phases, first locating the panel in an exploration phase and secondly positioning the robot relative to it.

5.1 Exploration

The panel has to be located before the robot can position itself precisely. Therefore, the exploration module creates a complete coverage path plan in a predefined rectilinear search space which is defined by the user via the input of GPS corner coordinates. This coverage path starts at the center of the search space and works its way towards the boundaries in counter-clockwise cycles around the middle point. The step size between those cycles thereby correlates directly with the detection range of the panel detection algorithm. An illustration of the coverage path plan is given in Fig. 4 (left). Furthermore, this module comes with a set of recovery behaviors to increase overall robustness. They guarantee that exploration continues when encountering scenarios such as unreachable waypoints due to obstacles. For increased efficiency of the exploration, heuristic priority waypoints can be specified. They are tracked prior to the retracing of the inherent coverage path.

The exploration phase is usually the first stage in a mission and terminates once the concurrently running panel detection algorithm (Sect. 4) succeeds in identifying the manipulation site. In a subsequent transition phase, the robot navigates to a fixed position relative to the wrench panel. Once this intermediate goal is reached, the transition completes and the positioning phase is triggered.

¹See <http://wiki.ros.org/navigation>.

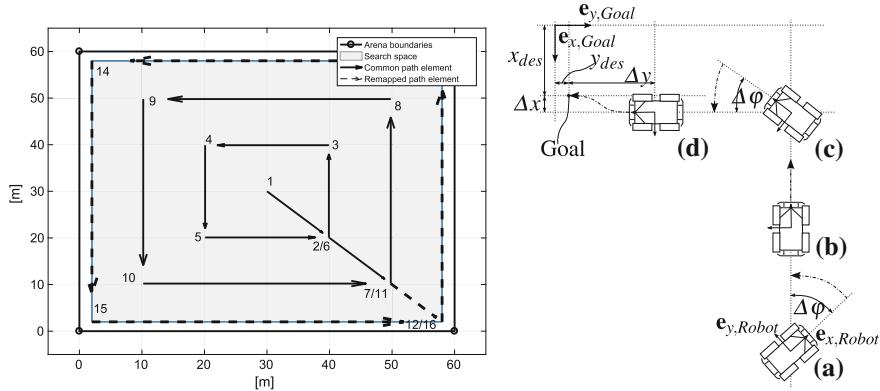


Fig. 4 Left: Waypoints for complete coverage of rectilinear search space. Right: The positioning procedure: **a** Facing, **b** Approaching, **c** Turning, and **d** Centering

5.2 Positioning in Front of the Wrench Panel

The purpose of this phase is to position the robot in a way that the panel is within the reach of the arm and subsequent manipulation tasks can be performed. To this end, a fixed sequence of maneuvers is executed. These maneuvers ensure that the desired goal pose of the robot in front of the wrench panel is reached reliably and accurately considering the skid-steer constraints of the mobile robot.

The positioning is divided into four stages: Facing, Approaching, Turning, and Centering. The schematic overview of these stages is illustrated in Fig. 4 (right). During the first three stages simple unidimensional velocity commands are applied, that is a constant rotation around the robot's z-axis in stage (a) and (c) and a linear velocity in stage (b). In the last stage both linear and angular velocities are controlled simultaneously. Thereby, potential residuals in lateral and longitudinal direction as well as rotational offsets from the previous stages are eliminated.

6 Object Detection and Visual Servoing

Manipulation of the valve using the appropriate wrench is the fundamental task of the mission. The key elements for succeeding in this task are correct detection of valve, precise estimation of depth, selection of the correct tool, and reliable visual servoing.

A support vector machine (SVM) [9] based object detection method using histogram of oriented gradients (HoG) features [4] is adopted in order to achieve robustness. HoG features are used as it captures the object's shape characteristic which is the most distinct property of the considered objects. A linear SVM model is trained using SVMLight [11] for each object i.e. valve, wrench, wrench box-end (referred

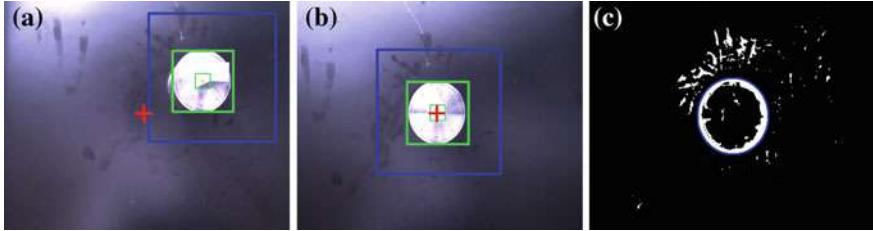


Fig. 5 Valve tracking (a) inside a region of interest (blue) to align the camera center (red) with the valve (green) (b). Detected valve contour (c) for depth estimation

to as wrench ring), and wrench open-end (referred to as wrench head). The training set contains images of the objects augmented for different illuminations to facilitate detection in different lighting conditions.

6.1 Valve Pose Estimation

We measure first the valve position and save it as reference point for the subsequent manipulation, as the wrench panel layout is specified.

- (a) **Valve Detection and Tracking:** The previously mentioned HoG based object detection method is used to detect the valve. The end-effector executes a predefined spiral motion pattern while searching for the valve. Once the valve is found with a certain confidence level, tracking starts using a Kalman Filter. Velocity commands are sent to the arm controller to reduce the alignment error between the center of the camera and the center of the valve as shown in Fig. 5a. After aligning the camera with the valve (Fig. 5b), the pose of the valve with respect to the robot's base can be computed using the current pose of the camera and the estimated distance between valve and camera.
- (b) **Depth Estimation:** The valve is segmented from the panel using adaptive thresholding [5] and its external contour is extracted using an algorithm based on connected components [19] as shown in Fig. 5c. This gives us the size of the valve in pixels (d_{image}). Knowing the focal length (f_{cam}) of the camera, we determine the distance between the valve and camera (Z_{depth}) using the known size of the valve in metric standards (D_{world}) via the projective equation

$$Z_{\text{depth}} = \frac{D_{\text{world}} f_{\text{cam}}}{d_{\text{image}}} . \quad (2)$$

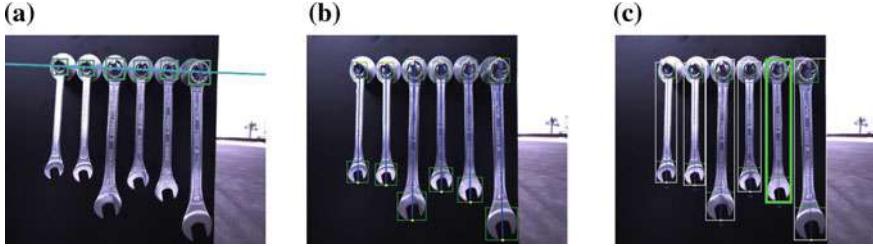


Fig. 6 **a** Pin detection to estimate view orientation. **b** Wrench length estimation through head and ring detection. **c** Selected wrench (green) after length voting

6.2 Wrench Selection

Corresponding to the given dimension of the valve stem, the appropriate wrench is selected according to its length with the following steps:

- (a) **Wrenches View Camera Pose Estimation:** As the panel layout is specified, we can infer the camera pose where all wrenches are visible from the camera lens parameters (perspective view angle) and the estimated valve position.
- (b) **Orientation Alignment and Depth Estimation using Wrench Rings:** The wrench rings are detected using the mentioned SVM detector and the center points of the rings are accumulated from consecutive image frames. Once enough points are recorded, a line is fit over those points as shown in Fig. 6a and its angle from the horizontal is calculated to correct the camera roll.
In order to be robust against positioning deviation of the UGV, we again estimate the depth from the wrench hanging pins. An approach similar to estimate the depth from the valve is used again. The distance (in pixels) between the consecutive pins in the image is equated to the distance (in metrics) between them from the panel layout using Eq. (2).
- (c) **Wrenches Length Estimation and Selection:** The head and ring of every wrench is detected using the trained SVM models. The distance (in pixels) between the mid-points of the detected ends of the wrench is calculated as shown in Fig. 6b and is converted to metric length.
In order to be robust against swinging of wrenches due to wind, a voting scheme is used. A wrench is selected in each image frame based on its estimated length. Each candidate is voted based on its frequency of occurrence as shown in Fig. 6c.

6.3 Wrench Grasping

Once the appropriate wrench is selected, it is approached by the gripper using visual servoing. Our method uses both position-based visual servoing (PBVS) and image-based visual servoing (IBVS) [3, 7] control techniques.

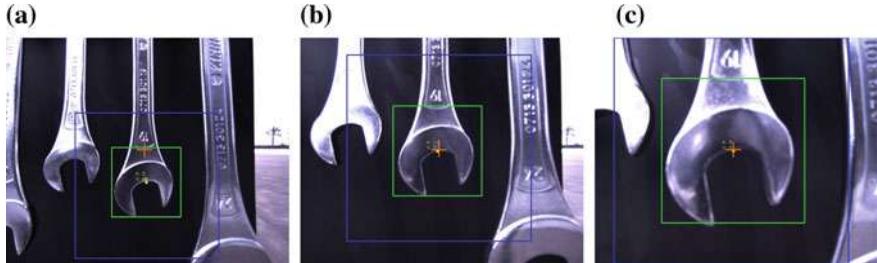


Fig. 7 Wrench head tracking for grasping: The wrench head is detected (green) within an dynamic region of interest (blue) to align the camera (red) during approaching

- (a) **Approaching the Selected Wrench:** First, we use the estimated depth and a PBVS approach to move the gripper close to the selected wrench as shown in Fig. 7a, from where it can be tracked.
- (b) **Wrench Head Tracking:** IBVS is used to align the gripper with the wrench before grasping. During alignment, the gripper also approaches the wrench as shown in Fig. 7b. A SVM model trained for the wrench head along with a Kalman Filter is used for detection and tracking.
While approaching, we calculate the ratio between the size of the wrench head and the size of the image frame. As we move closer this ratio increases and we can detect when the gripper is close enough to grasp the wrench as shown in Fig. 7c. A first order low pass filter on the ratio is used to avoid false detection due to movement of the wrench in presence of wind.
- (c) **Grasping:** Once the gripper has approached the wrench, the gripper closes slightly ensuring the wrench is bound within the claws. Then it moves down such that the wrench head is aligned to the rotation axis of the last actuator and does not occlude the camera view. The clamps close with a constant force and the final clamp position is used to detect if the grasping was successful.

7 Valve Manipulation

To fulfill the major part of the manipulation task it is sufficient to simply follow a desired end-effector trajectory. However, the ability to precisely measure and control the applied contact forces and torques demonstrates its usefulness during interaction of the gripper with objects. In particular, it shows its advantages in handling the valve stem with a wrench for both engagement and rotation.

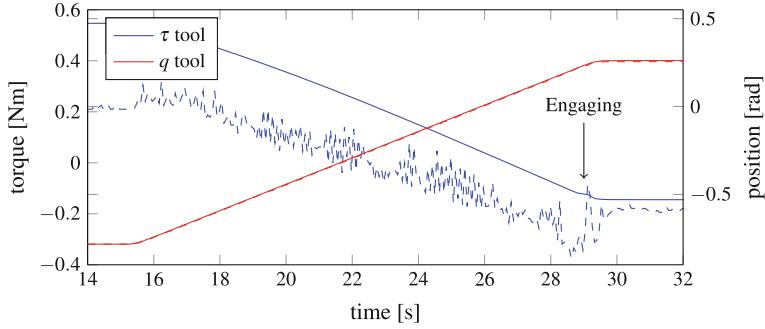


Fig. 8 Torque τ and position q tracking during valve engaging for the manipulator's last joint (tool). The solid lines show the commanded state, the dashed line the actual joint position and torque

7.1 Valve Engaging

Before engaging the wrench with the valve stem, we have to make sure that it is properly gripped. Though in the previous steps we ensure that the wrench is within the gripper, there is a chance that wrench slides down due to its weight. For this purpose we turn the gripper upside down, open the claws slightly to let the wrench head completely slide to the mechanical end stop, and close the gripper again. This ensures that the wrench head is at the correct position for engagement. Subsequently, the engaging procedure is executed.

First the gripper is placed in safe distance to the panel such that wrench head opening is pointing towards the valve stem edge, with an offset of 5 cm to the stem center. Next, we make contact between the wrench and the valve stem by first approaching towards the panel, afterwards to the center of the valve stem, and finally opening the gripper slightly to allow sliding the wrench towards the stem center. To engage, the gripper is moved such that the wrench head is rotating around the stem center. The gripper stays slightly opened during this motion to allow engaging of the wrench and prevent jamming due to induced forces from the manipulator. A successful engagement can be detected by a torque peak measured by the last joint (see Fig. 8). After engaging, the gripper is closed again for the subsequent valve turning.

7.2 Valve Rotation

We have to make sure to stay engaged during a full 360° rotation of the valve. For this purpose the manipulator applies a constant force of 10 N to push the wrench towards the stem center during valve rotation. Additionally, a torque of approximately 5 Nm is required, according to the competition specifications, to operate the valve stem. The resulting desired end-effector force \mathbf{F}_{ee} is directly applied as feed-forward torque

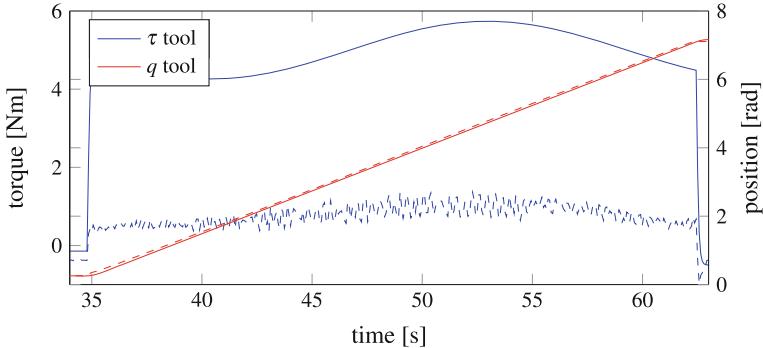


Fig. 9 Torque τ and position q tracking during valve rotation for the manipulator’s last joint. The desired end-effector torque (solid) corresponds to the specified 5 Nm resistance, but the actual required torque (dashed) corresponds to less than 1 Nm

$\tau_{F_{ee}} = J_{ee}(\mathbf{q})^\top \mathbf{F}_{ee}$, where $J_{ee}(\mathbf{q})$ is the end-effector Jacobian. Figure 9 shows the commanded and measured torque of the manipulator’s last joint during valve stem operation with a rotation speed of 0.25 rad s^{-1} . Noticeably, the actual required torque for rotation is less than 1 Nm.

8 Grand Challenge Performance

Both trials during the Grand Challenge showed similar performance. The second attempt lasted approximately 327 s of which 90 s were needed for exploration and navigation until positioned in front of the wrench panel, 95 s for measuring the valve stem and selecting the correct wrench, 110 s for grasping the wrench and engaging it with the valve, and the remaining 32 s for the actual rotation of the valve stem. The main bottleneck during navigation was the slow average driving speed of approximately 0.3 m/s which was owed to the limited turning rate necessary to avoid stick-slip effects between wheels and ground. During manipulation, the end-effector motion was required to be slow and steady for reliable and accurate visual tracking. Especially during grasping, slow approaching was necessary because of the shaking wrenches due to wind gusts.

A common failure source was the loss of target tracking during visual servoing because of changing lighting conditions. To recover from this case the arm moved to the last reference pose and restarted the tracking. Particularly important for successful manipulation is the precise positioning in front of the panel, as the arm with its limited range has to reach both the valve stem and the wrenches. We plan to overcome this limitation in the future through combined arm-base motion.

Overall, we successfully demonstrated the applicability of our system design by accomplishing the full valve operation task during both trials of the grand challenge

of MBZIRC. Our mobile manipulator demonstrated its ability to operate completely autonomously even in the presence of network dropouts and adverse environmental conditions (high temperature, wind). Our modular approach proved valuable by being fast to adapt to a new environment, allowing software faults to be isolated, and making replacements of mechanical components easy. We will continue to use this system for further applications in real-world scenarios and as a research platform.

Acknowledgements This work was supported in part by the Mohamed Bin Zayed International Robotics Challenge, the Swiss National Science Foundation (SNF) and the National Centre of Competence in Research Digital Fabrication. The authors would like to thank Sven Beck, Fabian Günther, and Eris Sako for their support in hardware design and Dario Bellicoso for his helpful advice.

References

1. Bodie, K., Bellicoso, C.D., Hutter, M.: ANYpulator: design and control of a safe robotic arm. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1119–1125. IEEE (2016)
2. Calisti, M., Cianchetti, M., Manti, M., Corucci, F., Laschi, C.: Contest-driven soft-robotics boost: the RoboSoft grand challenge. *Frontiers in Robotics and AI* **3**, 55 (2016)
3. Corke, P.I.: Visual control of robot manipulators—a review. *Visual Servoing* **7**, 1–31 (1993)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 886–893 (2005)
5. Davies, E.: *Machine Vision: Theory, Algorithms, and Practicalities*. Elsevier, New York (1997)
6. 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**(6), 381–395 (1981)
7. Flandin, G., Chaumette, F., Marchand, E.: Eye-in-hand/eye-to-hand cooperation for visual servoing. In: IEEE International Conference on Robotics and Automation (ICRA), vol. 3, pp. 2741–2746 (2000)
8. Guizzo, E., Ackerman, E.: The hard lessons of darpa’s robotics challenge [news]. *IEEE Spectr.* **52**(8), 11–13 (2015)
9. Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Schölkopf, B.: Support vector machines. *IEEE Intell. Syst. Appl.* **13**(4), 18–28 (1998)
10. Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, C.D., Tsounis, V., Hwangbo, J., Bodie, K., Fankhauser, P., Bloesch, M., Diethelm, R., Bachmann, S., Melzer, A., Hoepflinger, M.: ANYmal—a highly mobile and dynamic quadrupedal robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 38–44 (2016)
11. Joachims, T.: *Making Large-Scale SVM Learning Practical*. MIT Press, Cambridge, MA, USA (1999)
12. Kitano, H., Asada, M., Noda, I., Matsubara, H.: RoboCup: Robot world cup. *IEEE Robotics Autom. Mag.* **5**(3), 30–36 (1998)
13. Lima, P.U., Nardi, D., Kraetzschmar, G., Berghofer, J., Matteucci, M., Buchanan, G.: RoCKIn innovation through robot competitions. *IEEE Robotics Autom. Mag.* **21**(2), 8–12 (2014)
14. Pomerleau, F., Colas, F., Siegwart, R., Magnenat, S.: Comparing ICP variants on real-world data sets. *Auton. Robots* **34**(3), 133–148 (2013)
15. Pomerleau, F., Krüsi, P., Colas, F., Furgale, P., Siegwart, R.: Long-term 3D map maintenance in dynamic environments. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 3712–3719 (2014)

16. Riener, R.: The Cybathlon promotes the development of assistive technology for people with physical disabilities. *J. Neuroeng. Rehabil.* **13**(1), 49 (2016)
17. Schneider, F.E., Wildermuth, D., Wolf, H.L.: ELROB and EURATHLON: improving search & rescue robotics through real-world robot competitions. In: International Workshop on Robot Motion and Control (RoMoCo), pp. 118–123. IEEE (2015)
18. Siciliano, B., Caccavale, F., Zwicker, E., Achtelik, M., Mansard, N., Borst, C., Achtelik, M., Jepsen, N.O., Awad, R., Bischoff, R.: EuRoC-The Challenge Initiative for European Robotics. In: International Symposium on Robotics (ISR/Robotik), pp. 1–7. VDE (2014)
19. Suzuki, S., et al.: Topological structural analysis of digitized binary images by border following. *Comput. Vis. Graph. Image Process.* **30**(1), 32–46 (1985)

Towards a Generic Solution for Inspection of Industrial Sites

Marco Hutter, Remo Diethelm, Samuel Bachmann, Péter Fankhauser, Christian Gehring, Vassilios Tsounis, Andreas Lauber, Fabian Guenther, Marko Bjelonic, Linus Isler, Hendrik Kolvenbach, Konrad Meyer and Mark Hoepflinger

Abstract Autonomous robotic inspection of industrial sites offers a huge potential with respect to increasing human safety and operational efficiency. The present paper provides an insight into the approach taken by team LIO during the ARGOS Challenge. In this international competition, the legged robot ANYmal was equipped with a sensor head to perform visual, acoustic, and thermal inspection on an oil and gas site. The robot was able to autonomously navigate on the outdoor industrial facility using rotating line-LIDAR sensors for localization and terrain mapping. Thanks to the superior mobility of legged robots, ANYmal can omni-directionally move with statically and dynamically stable gaits while overcoming large obstacles and stairs. Moreover, the versatile machine can adapt its posture for inspection. The paper additionally provides insight into the methods applied for visual inspection of pressure gauges and concludes with some insight into the general learnings from the ARGOS Challenge.

1 Introduction

Recent advances in environment perception and robot control make autonomous mobile machines more and more applicable for inspection scenarios. Instead of expensive and inflexible instrumentation, mobile surface robots can carry sophisticated sensory equipment to any point of interest in order to conduct inspection or surveillance tasks. If such solutions are available, no humans need to be sent to working places that are dirty or dangerous, and even tedious and repetitive tasks can be conducted day and night with high precision.

M. Hutter (✉) · R. Diethelm · S. Bachmann · P. Fankhauser · C. Gehring · V. Tsounis · A. Lauber · F. Guenther · M. Bjelonic · L. Isler · H. Kolvenbach · K. Meyer · M. Hoepflinger
Robotic Systems Lab, ETH Zurich, Zurich, Switzerland
e-mail: marco.hutter@mavt.ethz.ch

Over the last years, there have been a number of initiatives from research and industry (e.g. EuRoC [16]) that try to bring such solutions from research laboratories to real world sites. The ARGOS Challenge,¹ initiated by TOTAL SA, aims at the application of mobile robotic solutions for offshore oil and gas site inspection. According to [9], the organizers expect a major impact with respect to (*i*) *health, safety and environment* (i.e. *a reduction of risk to personnel, environment and installation* as well as (*ii*) *operation* (i.e. *cost reduction, increase of efficiency and production*). In the ARGOS Challenge, the robots must be able to autonomously navigate on the industrial site and inspect various objects such as pressure gauges, water level gauges, or valve handle positions. They need to analyze the sound of the running pumps in order to identify malfunctioning systems, detect alarm signals, find gas leaks as well as hot spots, and recognize changes that were made on the site (e.g. missing or moved objects). To make the scenario as realistic as possible, the applied robots must satisfy ATEX (explosion protection) and IP (ingress protection) standards. Moreover, during the missions that happen on multiple floors connected by steep stairs, the robots are facing different hurdles such as unexpected obstacles, heavy water falls, strong winds, or humans that are working on the site.

In contrast to specific robotic devices that are already commercially used in tanks, vessels, or pipes² the ARGOS Challenge seeks for very generic robots that can one-to-one take over tasks performed by human specialists. In particular the requirements regarding mobility are demanding such that the few existing solutions like the wheel-based robots MIMROex [11] or SENSABOT [15] are not applicable. To address these issues, four of the five ARGOS Challenger teams selected in 2013 use tracked vehicles [9], while team LIO proposes an innovative solution based on a versatile legged robot. The remainder of this paper provides an insight into the realization of one of the most generic inspection robots.

2 System Description

Team LIO builds upon the modular and lightweight quadrupedal robot ANYmal [7] as transporter platform for inspection (Fig. 1). The legs of this versatile machine are driven by twelve equal series elastic actuator units [6] mounted at the joints. The kinematic structure of the robot is designed to achieve a large mobility allowing to overcome obstacles and stairs as well as for convenient transportation, compact storage, and simple deployment by a single operator. To keep the design as lightweight as possible, most of the structure is manufactured from carbon fibers. For fall protection, the robot features a rollover bar, a Kevlar belly plate, and shock absorbers. Moreover, force sensors in the feet provide haptic feedback of the environment, enabling safe locomotion even in case the robot is completely blind. ANYmal is designed in a hierarchical manner: On joint level, every actuator module is connected over a CAN

¹<http://www.argos-challenge.com>.

²For example, see <http://inspection-robotics.com>.

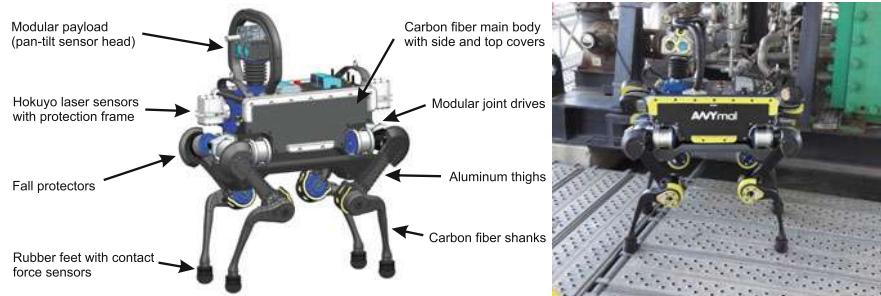


Fig. 1 ANYmal robot extended with a pan-tilt sensor head for inspection

bus and works independently. This allows component-level ingress and ATEX protection as well as fast and simple maintenance in case of hardware failure. On system level, computation is split among three independent computers that are connected through an internal network. The first computer (locomotion) hosts all real-time critical elements required for locomotion control and to interface the joint modules. The second computer (navigation) is responsible for environmental perception, localization, navigation, and mission execution, i.e. all software parts that are required to autonomously operate the robot. The third computer (inspection) runs all algorithms for inspection and detection.

For localization, navigation, and foothold planning, the machine is equipped with two rotating Hokuyo LIDAR sensors in the front and back, which provide detailed scans of the environment and the terrain. Additional wide-angle cameras in the front and back ensure an omni-directional view around the robot.

For inspection, we employ a pan-tilt head containing a high-quality zoom camera with high infrared (IR) sensitivity, a thermal camera, an ultrasonic and a regular microphone, a gas detection sensor, and LED illuminators. Since our robot can move its base in all directions, there is no need to employ the robot with an extension mechanism or arm.

The proposed system can operate fully autonomous with onboard batteries for more than 2.5 h. To extend this lifetime, ANYmal can autonomously dock to recharge the battery and to pressurize the mainbody with Nitrogen for ATEX compliance [8].

3 Autonomous Navigation on Industrial Sites

The ability to autonomously move on industrial sites requires that the robot is able to precisely (and globally) localize, to map its environment, to plan a navigation path, as well as to detect and overcome obstacles.

3.1 Localization and Re-localization

To track its position and orientation, the robot scans its environment with two rotating line LIDAR sensors (Hokuyo UTM-30LX-EW) and matches the resulting 3D point cloud of to the reference map using the iterative closest point (ICP) algorithm [12]. The individual scans (half rotation of the laser, 20'000 points) are dewarped using the local state estimation from IMU and leg kinematics [1]. The ICP algorithm then searches for neighbor points between the two 3D point clouds of the single scan and the map (Fig. 2a) and tries to minimize the sum of all their distances which takes about 0.4 s. The estimated location is then fed back to the systems state estimator as an update measurement. The challenging part in this setup was that the duration of the ICP matching step is variable. To compensate for the potentially old position update, the pose was further propagated using the local state estimation [1].

To converge to the correct solution, ICP requires that the errors of the initial guess of the robots pose are less than 1.5 m in position and 30° in orientation. In case of larger errors (i.e. at initialization or after loss of localization), the robot uses a plane matching algorithm. This algorithm searches for planes in the two 3D point clouds and groups them to all possible triples (see Fig. 2b). A similarity analysis of a scan and a reference plane triple can be done quickly by comparing the interplanar angles. If the plane triples are similar, their relative transformation can be directly computed from the plane parameters. Using this transformation, the scanned point cloud is expressed in the reference frame and a plausibility analysis is done by computing the nearest neighbor ratio. If a certain threshold is met, the according relative transformation between the clouds is taken to derive the robots pose. This approach has proven to work very reliably during all missions and typically took between 20 and 100 s to find the right position. Wrong positives were not encountered in any of the missions.

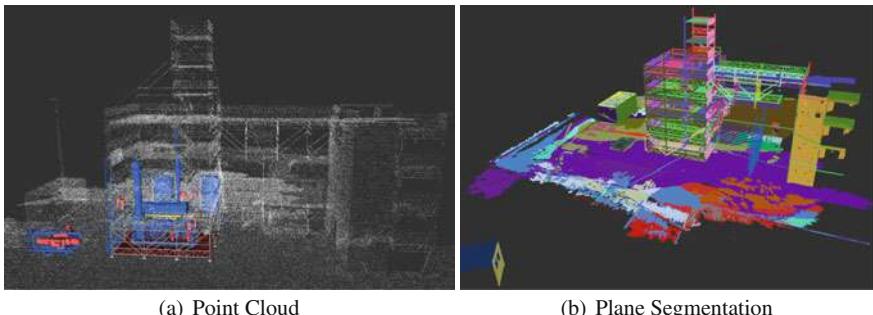


Fig. 2 **a** The ICP algorithm matches every scan acquired by the LIDAR sensor during one half rotation with the known reference map by minimizing the distance between both point clouds. **b** The point cloud of the site is segmented into planes for global localization

3.2 Path Planning and Re-planning

The mission contains two consecutive states for *path planning* and *path following*. The *path planning* requires a start and a goal pose (whereby the start may be the robots current pose) and outputs a path. The *path following* takes this path as input and computes desired forward, sideways and rotational velocities with respect to the robots body frame for the locomotion controller.

For path planning, the robot features two complementary algorithms, namely the *pose graph planner* and the *traversability planner*.

The *pose graph planner* is used for global path planning, when the robot needs to navigate from one pose to another on the site. The pose graph (Fig. 3a), which is created once for the entire site, is a representation of the accessible and safely traversable areas on the reference map. It consists of a tree of nodes with according connections (edges). Since the individual areas are mostly flat, the node is a degenerated pose containing position and yaw information. The type of motion that the robot is able to execute, i.e. the type of gait or climbing maneuver, is encoded in the edge. The *pose graph planner* computes a path by using an A* algorithm on the node tree. This method has several advantages: The planning is very fast (due to the limited dimension of the problem) and the result is completely deterministic. Any two points on the site can be connected by the pose graph planner. If a point does not lie on the pose graph, the entry or exit nodes are determined by closest proximity evaluation. Thanks to the short planning time, the robot can continuously and in real-time re-plan its path independent on the event of blocked paths, changed missions, or emergency situations.

For local path planning, e.g. when an obstacle blocks the global path of the robot, the *traversability planner* [17]³ comes into play. It builds up a map containing the estimated traversability of the environment, which is computed by fusing various

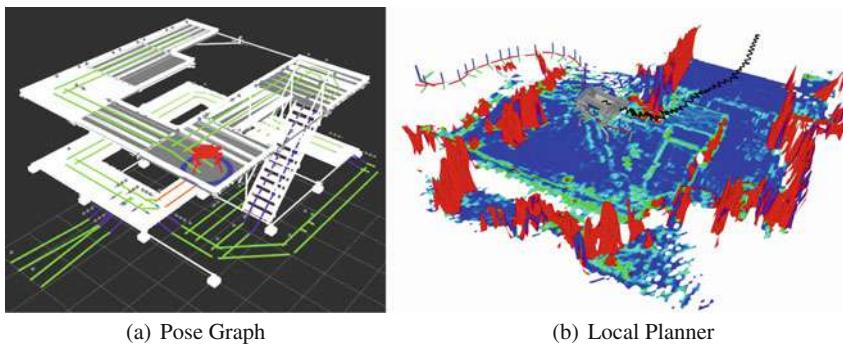


Fig. 3 **a** The pose graph consists of nodes (blue dots) and edges (green lines). **b** While following the nominal trajectory, the traversability planner adapts the path around untraversable areas

³Traversability estimation online available: https://github.com/ethz-asl/traversability_estimation.

filters for slope, roughness, or step heights using the acquired LIDAR data (Fig. 3b). After planning the local path with a sampling based RRT* planner, the mission replaces the blocked segment of the global path with the alternative local path.

3.3 Sense and Move Strategies

In cases where a checkpoint is hidden or badly visible from a selected viewpoint, e.g. due to reflections or bad angle of view, the robot visits an alternative position. To this end, a tool was developed to determine all possible inspection locations for every checkpoint, i.e. for every element that needs to be inspected. From the nominal position and orientation of the checkpoint given by the CAD model of the site, the possible inspection locations are determined. The likelihood of getting a good view on the checkpoint is evaluated as a function of the offset to the nominal robot posture and the angle of view offset (see Fig. 4). The robot then starts with the optimal inspection point (green) and only moves to the others if the confidence for a correct inspection is below a defined threshold.

3.4 Posture Adaptation

At every inspection point, the robot adapts its posture within the kinematically feasible limits in order to get an optimal view of the checkpoint. As displayed in Fig. 5, height and orientation of the body can be adjusted for every inspection situation. In fact, the robot can change height by about 0.5 m, which allows for inspection without an additional arm.

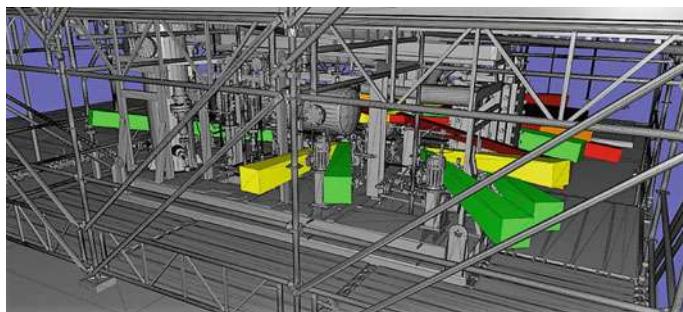


Fig. 4 Possible robot positions to read the checkpoint which are generated beforehand

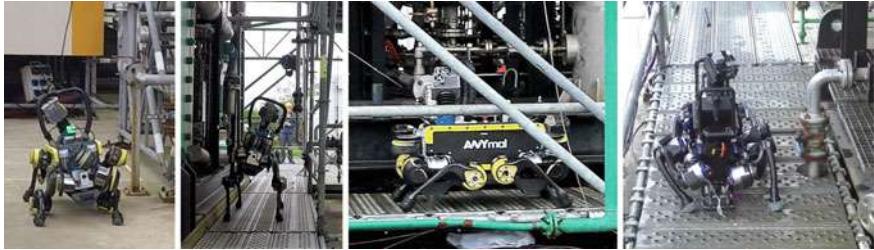


Fig. 5 ANYmal can adapt its posture to get an optimal view of the checkpoint

3.5 Stair Climbing

Ascending and descending stairs with ANYmal is achieved with two different strategies. In the general case, steps are negotiated by stepping regularly with one leg at a time in a walking gait (Fig. 6a). The geometry of the stair such as the step rise and run is either taken from the CAD model or estimated online from the elevation map [5]. Based on this information, the foothold coordinates and whole-body motion is generated using the Free Gait motion control architecture,⁴ where we use pose optimization and spline-based interpolation to automatically synthesize the required climbing motion [4]. In case of extreme conditions such as high inclinations ($>50^\circ$), very slippery ground, or high wind speeds, ANYmal can negotiate the stairs in a turtle like crawling gait as demonstrated during the second competition. Here, the main body lies on the ground, the legs are moved to find the next stable contact holds, and the machine is subsequently pulled upwards (Fig. 6b). Thanks to ANYmal's large range of motion, the legs can be turned overhead to prevent collision with the ground or side rails.



Fig. 6 ANYmal uses the Free Gait motion control architecture [4] to overcome stairs by stepping regularly with one leg a time in a walking gait (a) or by using a turtle like crawling gait (b)

⁴Free Gait online available: https://github.com/leggedrobotics/free_gait.

3.6 Obstacles

3.6.1 Obstacle Detection

Building upon the onboard range measurement from the LIDAR sensors and the robot pose estimation, we create an elevation map⁵ which serves as a reference for obstacle detection [5]. The elevation map is a discrete 2.5D representation of the environment and it is based on the universal grid map library,⁶ a mapping framework for mobile robotics [3]. An example of the visualization of a generated elevation map is shown in Fig. 7.

During operation, the robot updates the elevation map by periodically adding newly perceived point clouds. By comparing the reference elevation map of the site and the updated elevation map, changes in the environment and thus, obstacles are detected. In case of negative obstacles, this task becomes harder since they are defined as the absence of structure which can only be detected by ray-tracing. If a laser point behind an expected structural element is perceived, the robot assumes that the element has disappeared. To robustify the approach against outliers, we only classify clusters as an obstacle if at least several neighbor cells fall in the same region. Since laser data are sparse in far distance, only obstacles closer than 2.5 m are considered for path planning. In case obstacles fall within 2 m distance, the obstacle is additionally tested if it is a human or not.

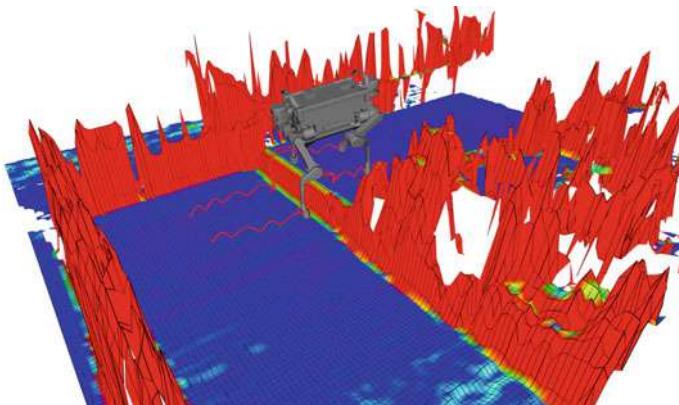


Fig. 7 Visualization of a generated elevation map during locomotion on site

⁵Elevation mapping online available under http://github.com/ethz-asl/elevation_mapping.

⁶Grid map library online available under http://github.com/ethz-asl/grid_map.

3.6.2 Human Detection

The human detection is based on *You Only Look Once* (YOLO) [13] respectively the improved version YOLOv2 [14]. We adapted YOLOv2 to fit in our framework and use the small model of YOLOv2 with the training data from PASCAL VOC 2007 and 2012. Since the entire area around the robot must be covered, we use the front and back fisheye camera. As soon as the robot detects an obstacle inside the guarded space, the coordinates of the objects are transformed from Cartesian space into the image frames of the two cameras. Depending on the size of the obstacle, the image is cropped around the obstacles position to increase the robustness and speed of the human detection. Our version processes images on the central processing unit (CPU) and it only takes few seconds to detect objects.

3.6.3 Obstacle Negotiation

Thanks to its legs and the high range of motion, ANYmal can flexibly climb over various known and unknown obstacles without creating contact with them. Small steps and obstacles (height < 20cm) are overcome by safely stepping on suitable footholds in a static walking gait (Fig. 8, left). Bigger obstacles and high steps (height > 20cm) are negotiated by rotating the legs in an outwards configuration (Fig. 8, center and right), which avoids any potential collision between the robot and the obstacles. With this, the robot can also climb on steps up to approximately 40cm height. In case of known obstacles (Fig. 8, left and center), we defined the desired behavior (leg configuration and step positions) from a database of parameterized motion definitions. Based on our software architecture for motion generation [4], this allows for robust and repeatable obstacle negotiation. In case of unknown obstacles (Fig. 8, right), the elevation map created by the laser scanners is used to determine safe foothold locations. During the negotiation maneuver, the robot automatically adapts to the dimensions of the obstacle.



Fig. 8 ANYmal uses the Free Gait motion control architecture [4] to negotiate known (left and middle) and unknown (right) obstacles. For known obstacles we generated a database of parameterized motion definitions. Unknown obstacles are negotiated by selecting safe footholds from the environmental mapping and optimizing the body motion and posture

4 Inspection

As outlined in the introduction, the robot is able to perform visual, thermal, and acoustic inspection. While a detailed description of all tools would go beyond the scope of this paper, we want to outline the specific approach and results for pressure gauge inspection.

4.1 Camera Alignment, Tracking, Zooming

In order to robustly point the camera at the checkpoint, the robot moves to a pre-computed optimal posture at the inspection point and then determines online the required pan-tilt angles depending on the position of the robot and the checkpoint. After the approximate pan-tilt angles are set and the camera is aligned, the algorithm switches to a tracking mode. Tracking is required to zoom in without losing the checkpoint due to uncertainties in the robots position and position anomalies of the checkpoints. To track the checkpoint, we use a particle filter approach. The measurement updates for the particle filter come from analyzing the image with a histogram oriented gradients (HOG) [2] descriptor.

For a robust detector, a global HOG descriptor is trained with machine learning, namely the SVMlight library, using about 5000 positive and negative samples (Fig. 9). The positive samples are generated by rendering two different manometer models from different view points using blender and adding random background images from the ARGOS site. The negative samples consist only of random background images. Then, for each of those samples, a HOG descriptor vector is computed to train the global descriptor. The detector works with a sliding window approach and

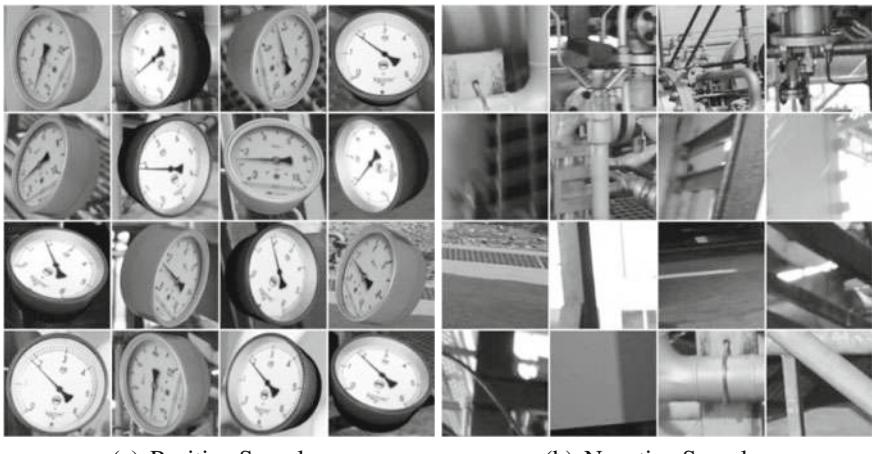


Fig. 9 Positive and negative examples for the machine learning algorithm

different image scales. To speed up the detection process, it is first searched only with the expected manometer size and stopped if at least two overlapping detections exist. If there were no detections or only one, different scales are applied to the image and the detector runs again.

This approach has proven to work very robustly since it does not matter if the pressure gauge has a different dial face. Once the checkpoint is tracked in the image, the camera zooms in until the checkpoint has the optimal size to read.

4.2 Dewarping

Usually it is not possible to face the camera exactly in front of the checkpoint as necessary for accurate reading. In order to dewarp the image by correcting the perspective and rotation of the checkpoint, two complementary approaches are implemented.

First, a given front image of the checkpoint is matched with a given example image by looking for scale-invariant feature transform (SIFT) features [10] that appear in both images. From the matched SIFT features, the homography matrix is computed and used to dewarp the image (see Fig. 10). This method requires an undistorted image of each pressure gauge type.

This SIFT-based approach may fail in situations where the image of the checkpoint differs too much from the given example image such that there are not enough features for image matching. This is typically the case if the angle of view is too large. To overcome this limitation, the algorithm can “manually” dewarp the taken image of the checkpoint with the knowledge of both the cameras orientation and the checkpoints nominal orientation. While this approach works well as long as the checkpoint has the nominal orientation, it is more likely to fail than the SIFT method if the checkpoint does not exhibit its nominal orientation.



Fig. 10 The SIFT features are generated by comparing a given example image (left) with the image of the checkpoint (middle). The visualized homography matrix shows the connection lines between the matched features of both images. This homography matrix is used to generate the dewarped image (right)



Fig. 11 Pressure gauge reading process (from left to right): (1) multiple circle detections, (2) line segment extractions, (3) resulting mean circle with estimated manometer center (green point) and filtered lines (yellow), (4) mean pointer line (green) and resulting read value in red

4.3 Reading

For an accurate reading of the pressure gauge it is important to identify the center of the pressure gauge. As illustrated in Fig. 11, a Hough-based circle detection is first applied to estimate the gauge frame. To limit the search area, the knowledge of the camera tracking is used as it provides an approximate position within the image. Once the circles are estimated, the mean is taken to provide the gauge frame and center of the gauge reading. In a second step, a Hough-based line detection is used to find the pointer of the pressure gauge. The algorithm uses the center to filter the lines and to discard the lines that do not belong to the pointer. The mean of the resulting lines gives the estimated pointer. Finally, the actual pressure value is computed from the known scale of the pressure gauge, the center, and the line angle.

4.4 Evaluation

To verify the robustness of the pressure gauge reading algorithm, a series of measurements from different viewpoints and with different pointer positions was conducted (Fig. 12).

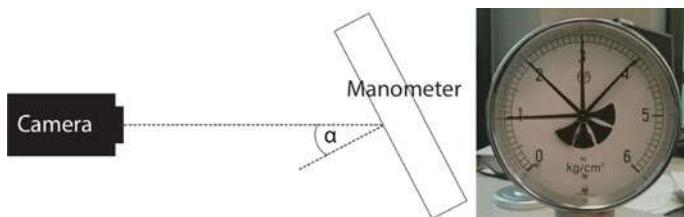


Fig. 12 Experimental setup to identify the maximal possible angle of view

Table 1 Measurements with different angles and pointer positions. 50 measurements per angle and pointer position were obtained

Defined angle α'	0°	10°	20°	40°	45°		50°	
Actual angle α	0°	10°	20°	40°	50°	45°	55°	50°
Pointer position	Measurements							
1.00 kg/cm ² (~90°)	0.96	0.96	0.96	0.96	0.94	0.96	0.92	0.96
1.99 kg/cm ² (~135°)	1.97	1.97	1.99	1.99	<u>2.15</u>	2.00	<u>2.18</u>	2.03
2.98 kg/cm ² (~180°)	2.92	2.96	2.97	2.97	3.01	3.00	3.02	3.02
3.98 kg/cm ² (~225°)	3.98	3.98	3.97	3.98	<u>3.88</u>	3.98	<u>3.86</u>	3.96
SIFT rate	100%	100%	100%	100%	0%	91%	0%	12%

The results displayed in Table 1 proof that the overall robustness of the proposed approach is high. Ignoring the cases where $\alpha \neq \alpha'$ gives an average absolute deviation of 0.013 kg/cm² respectively 0.58° with an allowed maximal deviation of 0.12 kg/cm² or 5.36°. Even the manual dewarping at $\alpha \neq \alpha' = 50^\circ$ provides accurate measurements (only 12% of the images were dewarped by the SIFT algorithm). Overall, manometers can be read within the required accuracy from angles of view between -50 and 50° . The experience during the actual inspection missions supports these results as outstanding inspection accuracy was achieved in all missions.

5 Conclusion

The present paper illustrates the worldwide first attempt of using a versatile legged machine for autonomous inspection on industrial sites. In three consecutive competition on a testing site in Pau, France, team LIO was able to demonstrate the high potential of the proposed solution. For illustration, we collected a video summary of Challenge 2⁷ and Challenge 3.⁸

While all individual tasks regarding system mobility, navigation, and inspection could be entirely fulfilled, the high system complexity entails several potential sources for failure. Firstly, the fact that neither stopping (blocking) nor disabling the joint motors leads to an immediate stop of a legged robot complicates safety consideration. As a result, we encountered during the three competitions several (uncontrolled) falls which the robot all survived. In the future, it is required to fur-

⁷<https://youtu.be/SR5OJ-vklIs>.

⁸<https://youtu.be/2RQDp0Q2vSo>.

ther extend the work on smart emergency behaviors such that legged robots can safely operate even in case of critical software or hardware failure. Secondly, the realistic missions unveiled that robust, reliable, and fast terrain perception under harsh conditions is still challenging. When the robot was moving fast, it was difficult to create terrain maps without any false positive obstacles due to the distorted scans from the motion, reflections on metal and wet surfaces, and rain drops on the laser. Thirdly, despite the superiority in mobility compared to tracked or wheeled vehicles, there is still a lot of potential for improvement of the locomotion skills of legged robots. Once these deficiencies are overcome, we are convinced that legged robots such as ANYmal can find their way into applications like industrial inspection.

References

1. Bloesch, M., Hutter, M., Hoepflinger, M., Leutenegger, S., Gehring, C., Remy, C.D., Siegwart, R.: State estimation for legged robots—consistent fusion of leg kinematics and IMU. In: Robotics Science and Systems (RSS), pp. 17–24 (2012)
2. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 886–893 (2005)
3. Fankhauser, P., Hutter, M.: A universal grid map library: implementation and use case for rough terrain navigation. Robot Operating System (ROS)—The Complete Reference, pp. 99–120. Springer, Cham (2016)
4. Fankhauser, P., Bloesch, M., Gehring, C., Hutter, M., Siegwart, R.: Robot-centric elevation mapping with uncertainty estimates. In: International Conference on Climbing and Walking Robots (CLAWAR), pp. 433–440 (2014)
5. Fankhauser, P., Bellicoso, C.D., Gehring, C., Dube, R., Gawel, A., Hutter, M.: Free gait An architecture for the versatile control of legged robots. In: IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), November 2016, pp. 1052–1058. IEEE (2016)
6. Hutter, M., Bodie, K., Lauber, A., Hwangbo, J.: EP16181251—Joint unit, joint system, robot for manipulation and/or transportation, robotic exoskeleton system and method for manipulation and/or transportation (2016)
7. Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, C.D., Tsounis, V., Hwangbo, J., Bodie, K., Fankhauser, P., Bloesch, M., Diethelm, R., Bachmann, S., Melzer, A., Hoepflinger, M.: ANYmal—a highly mobile and dynamic quadrupedal robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2016, pp. 38–44. IEEE (2016)
8. Kolvenbach, H., Hutter, M.: Life extension: an autonomous docking station for recharging quadrupedal robots. In: Field and Service Robots (FSR) (2017) (submitted)
9. Kydd, K., Macrez, S., Pourcel, P.: Autonomous robot for gas and oil sites. In: SPE Offshore Europe Conference and Exhibition, September 2015. Society of Petroleum Engineers (2015)
10. Lowe, D.: Object recognition from local scale-invariant features. In: IEEE International Conference on Computer Vision, pp. 1150–1157. IEEE (1999)
11. Pfeiffer, K., Bengel, M., Buback, A.: Offshore robotics—survey, implementation, outlook. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 241–246 (2011)
12. Pomerleau, F.: Applied registration for robotics. Ph.D. thesis, ETH (2013)

13. Redmon, J., Farhadi, A.: YOLO9000: Better, Faster, Stronger (2016). [arXiv:1612.08242](https://arxiv.org/abs/1612.08242)
14. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
15. Siciliano, B., Caccavale, F., Zwicker, E., Achtelik, M., Mansard, N., Borst, C., Achtelik, M., Jepsen, N.O., Awad, R., Bischoff, R.: EuRoC—the challenge initiative for European robotics. In: International Symposium on Robotics; Proceedings of ISR/Robotik (2014)
16. Staff, J.P.T.: Sensabot: a safe and cost-effective inspection solution. *J. Pet. Technol.* **64**(10), 32–34 (2012)
17. Wermelinger, M., Fankhauser, P., Diethelm, R., Krusi, P., Siegwart, R., Hutter, M.: Navigation planning for legged robots in challenging terrain. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2016, pp. 1184–1189. IEEE (2016)

Foresight: Remote Sensing for Autonomous Vehicles Using a Small Unmanned Aerial Vehicle

Alex Wallar, Brandon Araki, Raphael Chang, Javier Alonso-Mora
and Daniela Rus

Abstract A large number of traffic accidents, especially those involving vulnerable road users such as pedestrians and cyclists, are due to blind spots for the driver, for example when a vehicle takes a turn with poor visibility or when a pedestrian crosses from behind a parked vehicle. In these accidents, the consequences for the vulnerable road users are dramatic. Autonomous cars have the potential to drastically reduce traffic accidents thanks to high-performance sensing and reasoning. However, their perception capabilities are still limited to the field of view of their sensors. We propose to extend the perception capabilities of a vehicle, autonomous or human-driven, with a small Unmanned Aerial Vehicle (UAV) capable of taking off from the car, flying around corners to gather additional data from blind spots and landing back on the car after a mission. We present a holistic framework to detect blind spots in the map that is built by the car, plan an informative path for the drone, and detect potential threats occluded to the car. We have tested our approach with an autonomous car equipped with a drone.

A. Wallar (✉) · B. Araki · R. Chang · D. Rus

Distributed Robotics Lab, CSAIL, Massachusetts Institute of Technology,

32-376 Vassar St, Cambridge, MA 02139, USA

e-mail: wallar@csail.mit.edu

B. Araki

e-mail: araki@csail.mit.edu

R. Chang

e-mail: raphaelchang@mit.edu

D. Rus

e-mail: rus@csail.mit.edu

J. Alonso-Mora

Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2,

Room C-2-310, 2628 CD Delft, Netherlands

e-mail: j.alonsomora@tudelft.nl

1 Introduction

Globally, over 3000 people lose their lives in vehicle-related accidents and over one hundred thousand are injured or disabled on average *every day* [4]. In the United States, over 90% of these accidents are due to human error [1]. This has resulted in the continued development of advanced safety systems by commercial car manufacturers. For example, systems exist to automatically brake in the case of unexpected obstacles [18], maintain a car in a lane at a given speed [6], and alert users of pedestrians, signage, and other vehicles on the roadway [7]. These systems will make our cars safer and eventually autonomous. However, many accidents are due to blind spots, for example when a vehicle takes a turn with poor visibility or when a pedestrian crosses from behind a parked vehicle. In these accidents, vulnerable road users, i.e. pedestrians and bikers, are typically involved and the consequences are dramatic.

We propose to extend the perception capabilities of a vehicle, autonomous or human-driven, with a small Unmanned Aerial Vehicle (UAV) capable of taking off from the car, flying around corners to gather additional data from blind spots and landing back on the car after a mission. Small UAVs are highly mobile and agile, and they are capable of capturing aerial footage autonomously [17]. The quadcopter could use the car as a charging base, while the car could send the quadcopter out on missions to scout ahead and fill in the blind spots in its vision.

A crucial step in enabling a small UAV and an autonomous car to work together is to ensure that there exists an accurate pose transform between the car and the quadcopter. In this paper, we present a method for relative localization using ultra-wideband radios (UWBs) to measure the relative position of the quadcopter relative to the car with an accuracy of less than 14 cm. This information is fused with the internal state estimation to enable the UAV to safely navigate to blind spots and then land back on the car. Furthermore, we have developed a path planning algorithm for remote sensing and have carried out experiments using a quadrotor and a car.

Our method has similarities with multi-robot mapping. For instance [9, 15] combined the maps from a ground and an aerial robot for enhanced exploration and [8] employed a map created by an aerial robot for planning the motion of a ground robot.

Quadrotors have been able to track a moving vehicle using visual techniques such as AprilTag localization [5], optical flow [11], and infrared markers [22]. However, a weakness of visual systems is that visual cues must be in the line of sight of the quadrotor-mounted camera. Moreover, the lighting conditions must be suitable for the cameras. These restrictions limit the usefulness of visual tracking. Meanwhile, GPS tracking, while useful in long-range outdoors scenarios, is not accurate enough for maneuvers such as landing and obstacle avoidance. For example, average accuracy in smartphone GPS receivers is 4.9 m [2].

UWB sensors have in recent years become a popular tool for localization of quadrotors, particularly in indoor GPS-denied environments [12, 14, 19], since they can provide distance measurements with an accuracy of 10 cm and a range of up to 300 m [3]. Indoor localization of quadrotors has been achieved by placing UWB “anchors” around the perimeter of a room and attaching a UWB “tag” on the quadrotor.

Using this technique, [16] and [13] have achieved localization with accuracy on the order of 30 cm.

The problem of relative localization is more challenging because the UWB tag is outside of the perimeter of the anchors. However, in [10], a quadrotor outfitted with four UWBS was able to follow a person carrying a UWB tag in the plane of the quadrotor with less than 10cm mean error. The authors achieved this accuracy using an iterated Extended Kalman Filter. Building off of this work, we used 6 UWB sensors on a car to estimate the 3D position of a UWB on a quadrotor with an average mean error of 13.7 cm. Moreover, unlike in previous systems, we take advantage of the accurate relative transform to use the sensors on the car to plan safe paths for the quadrotor.

1.1 Contribution

This paper presents a method for extending the sensing capabilities of self-driving vehicles by using a small quadrotor to autonomously locate and observe regions occluded to the vehicle and detect potentially unsafe obstacles such as pedestrians or other cars. Our contributions include

- A method for determining the relative transformation between a ground vehicle and a quadrotor using an array of ultra-wideband radios
- An informative planning algorithm that computes collision free paths for the quadrotor relative to the ground vehicle that view occluded regions
- A system that uses the localization and planning algorithms and enables a UAV to position itself and transmit images outside the field of view of the sensors on the car
- Experimental validation using a sensor-equipped Toyota Prius and a Parrot Bebop 2 quadrotor.

1.2 Method Overview

We consider two vehicles.

- A ground vehicle, i.e. the car, which can create a local map of the environment, localize with respect to it and autonomously navigate. We utilize a 2D grid map to represent the free space and obstacles seen by the car. In particular, our vehicle is equipped with a 2D LIDAR.
- A lightweight companion quadrotor equipped with a front facing camera. The drone is able to fly autonomously to/from the ground vehicle and detect obstacles that were originally occluded for the ground vehicle.

Given a laser scan from the ground vehicle, our objective is to: (a) determine which areas of the environment are occluded to ground vehicle, (b) compute a safe path for

the aerial vehicle to observe the occluded areas, and (c) detect unseen obstacles, such as pedestrians, and report them back to the ground vehicle. The quadrotor will take off from the vehicle when it is ready to begin driving and land back on the vehicle when it has parked.

To accurately localize the quadrotor relative to the ground vehicle, we equip the ground vehicle with several UWBS. In the quadrotor, we fuse, via an Unscented Kalman filter (UKF) relative information from a UWB radio with odometry estimates from a down-facing optical flow sensor and an onboard IMU. Our algorithm operates directly on the laser scan from the ground vehicle to find occluded regions. We then employ an anytime sampling-based algorithm to compute a collision free path for the drone that maximizes the occluded area viewed by the quadrotor. To detect obstacles within the occluded areas, we employ a real-time object detecting convolutional neural network [20], which is able to classify and locate objects, such as pedestrians, cars, bicycles, in monocular images. These obstacles are then reported back to the ground vehicle.

Using the ground vehicle's 2D laser scan, we compute the areas it is unable to sense. We employ an anytime sampling-based algorithm to construct a collision free path for the quadrotor that maximizes the total area of the occluded regions it is able to observe. While the quadrotor is executing the planned path, we use a convolutional neural network to classify and detect objects in the quadrotor's field of view, such as pedestrians and cars, and relay this information back to the ground vehicle. The driver of the vehicle is then able to view the quadrotor's camera feed along with the annotated objects. The path is updated if it is no longer collision free due to changes in the laser scan or if a new path is computed that can observe a larger occluded area. A high level overview of the entire process is shown in Algorithm 1.

Algorithm 1 Overview of the Foresight algorithm

```

1:  $\Pi \leftarrow \emptyset$ 
2: while ISRUNNING() do
3:    $x \leftarrow \text{GETQUADROTORCONFIGURATION}()$ 
4:    $L \leftarrow \text{GETLASERSCAN}()$ 
5:    $\mathcal{P} \leftarrow \text{CONSTRUCTBOUNDINGPOLYGON}(L)$ 
6:    $\mathcal{B} \leftarrow \text{COMPUTEBLINDREGIONS}(L)$ 
7:    $\tilde{\Pi} \leftarrow \text{COMPUTECOVERAGEPATH}(x, \mathcal{B}, \mathcal{P})$ 
8:   if  $|\Pi| = 0 \vee \neg \text{PATHCOLLISIONFREE}(\Pi, \mathcal{P}) \vee$ 
       $\text{OBSERVINGAREA}(\tilde{\Pi}, \mathcal{B}) > \text{OBSERVINGAREA}(\Pi, \mathcal{B})$  then
9:      $\Pi \leftarrow \tilde{\Pi}$ 
10:     $\text{SENDPATHTOQUADROTOR}(\Pi)$ 

```

2 Planning for Exploration

Planning a path to observe the blind spots of an autonomous car is broken into following steps. First, using the 2D laser scan from the car, we compute a bounding

polygon. This polygon represents the known free space where the quadrotor can travel. We use the laser scan to determine regions in space where that the car is not able to sense. These regions are called blind regions. A path is then computed for the quadrotor that maximizes the observed area of the blind regions while staying within the bounding polygon for a given time horizon.

The remainder of this section is structured as follows; Sect. 2.1 introduces our formal definition of a laser scan and describes how the bounding polygon is found, Sect. 2.2 describes how the blind regions are computed from the laser scan, and Sect. 2.3 describes the algorithm we developed for computing the exploratory path.

2.1 Finding the Bounding Polygon

The bounding polygon computed using a scan from the 2D LiDAR sensor on the car is used as a conservative representation of the free space in which the quadrotor can travel. Below we provide a formal definition of a laser scan that is used in the rest of the paper.

Definition 1 A laser scan is a sequence of points, $L = \{\mathbf{c} + r_i \cdot [\cos \theta_i, \sin \theta_i]^T : \theta_{\min} \leq \theta_i \leq \theta_{\max}\} \subset \mathbb{R}^2$, where \mathbf{c} is the 2D position of the LiDAR sensor, r_i is the distance from the sensor to the closest obstruction in the θ_i direction, and $[\theta_{\min}, \theta_{\max}]$ is the angular range of the sensor.

From the laser scan, we compute a bounding polygon. The bounding polygon is defined as the minimum area simple polygon that contains all the points in the laser scan. Since the laser scan data is ordered by θ_i from θ_{\min} to θ_{\max} , the bounding polygon can be constructed in one pass with the vertex sequence $\{\mathbf{c}\} \cup L \cup \{\mathbf{c}\}$. Figure 1b shows an example of laser scan data and the corresponding bounding polygon.

2.2 Determining the Blind Regions

Using the laser scan data, we can determine which areas in the environment the car is unable to sense. We call these areas blind regions. The blind region, \mathcal{B} , is the set of points contained within a rectangle with a vertex sequence $\{L_i, L_i + k \cdot \hat{L}_{i,i+1}, L_{i+1}, L_{i+1} + k \cdot \hat{L}_{i,i+1}, L_{i+1}, L_i\}$ where $\hat{L}_{i,i+1}$ is the unit normal for the vector between points L_i and L_{i+1} that points away from the bounding polygon and k is a tuning parameter that contributes to the area of the blind region. In practice we only care for blind regions where $\|L_i - L_{i+1}\|_2 > \delta$ where δ is a tuning parameter because the laser scan consists of a finite number of points with a known angular distance. We will use \mathcal{B} to denote the set of all such regions. Figure 1c shows an example of blind regions found in a laser scan.

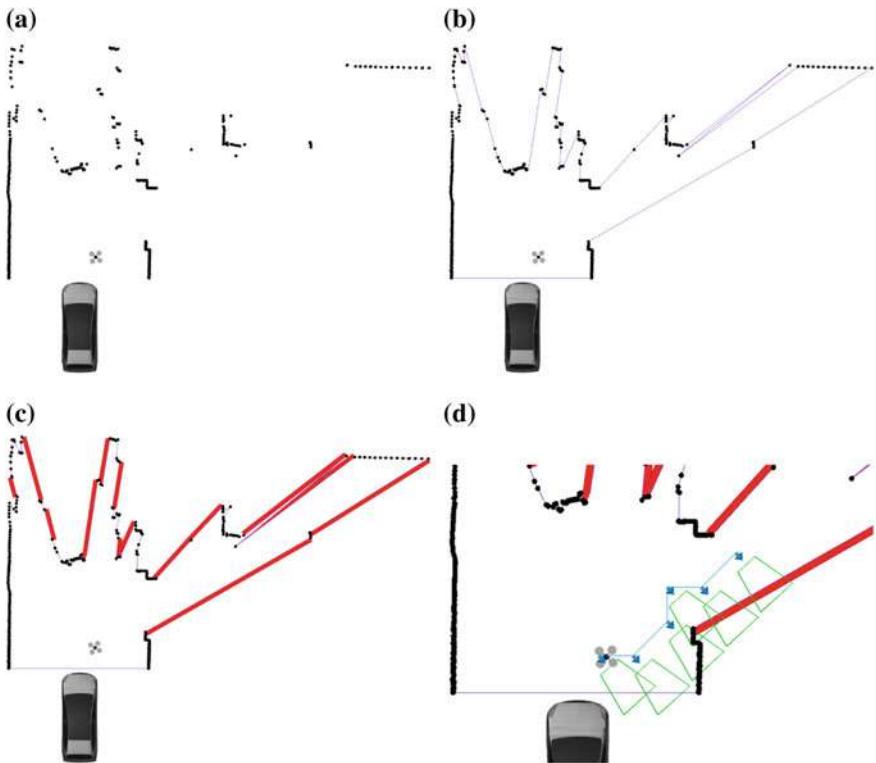


Fig. 1 Plots showing the four stages of the planner. Figure **a** shows the points from the laser scan. Figure **b** shows bounding polygon created from the laser scan. Figure **c** shows the regions occluded to the vehicle in red and Fig. **d** shows the initial plan for the quadrotor to view some of these blind regions

2.3 Computing the Exploratory Path

Using the blind regions, current configuration of the quadrotor, and the bounding polygon, we present an anytime algorithm that computes a collision free path for the quadrotor that maximizes the total observed area of the blind regions within a given time horizon. The algorithm builds a search tree starting from the current configuration of the quadrotor. It expands leaf nodes in descending order of total observed blind region area and only adds new leaf nodes to the search that are contained within the bounding polygon. When a collision free neighbour is propagated, the orientation, $\theta^*(x, \mathcal{B})$, that maximizes the area of the remaining blind region, \mathcal{B} , viewed at that configuration, x , is also added to the search tree. Below we formally define this orientation.

Definition 2 Let $\psi(x, \theta, \mathcal{B})$ be the set of points visible by the quadrotor at position $x \in \mathbb{R}^3$ with orientation θ . Let $\theta^*(x, \mathcal{B}) = \arg \max_{0 < \theta \leq 2\pi} \psi(x, \theta, \mathcal{B})$. For convenience, we define $\psi^*(x, \mathcal{B}) = \psi(x, \theta^*(x, \mathcal{B}), \mathcal{B})$.

As the quadrotor follows the path, the planner is constantly replanning. To avoid oscillating between candidate paths, the quadrotor only follows a new path if its current path is no longer collision free or if the new path has a larger objective value.

Algorithm 2 Path planning for remote sensing UAV (looking around the corner)

Input:

- x_0 : The initial position of the robot, \mathcal{B} : The blind region, \mathcal{P} : The bounding polygon

Output:

- $\Pi \subset \mathbb{R}^3 \times [0, 2\pi]$: A sequence of 3D positions and orientations representing the path

```

1:  $Q \leftarrow \{(x_0, \theta^*(x_0, \mathcal{B}), \mathcal{B} \setminus \psi^*(x_0, \mathcal{B}))\}$ 
2: while  $|Q| > 0$  do
3:    $(x, \theta, \mathcal{B}') \leftarrow \arg \min_{\mathcal{B}' \in Q} \text{AREA}(\mathcal{B}')$ 
4:   if SEARCHTIMEOUTEXPIRED() then
5:      $\Pi \leftarrow \{\}$ 
6:     while HASPARENT( $x, \theta$ ) do
7:        $\Pi \leftarrow \Pi \cup \{x\}$ 
8:        $(x, \theta) \leftarrow \text{PARENT}(x, \theta)$ 
9:     return  $\Pi$ 
10:   for all  $x' \in \text{COLLISIONFREENEIGHBOURS}(x, \mathcal{P})$  do
11:      $\theta' \leftarrow \theta^*(x', \mathcal{B}')$ 
12:      $Q \leftarrow Q \cup \{(x', \theta', \mathcal{B}' \setminus \psi^*(x', \mathcal{B}'))\}$ 
13:      $\text{PARENT}(x', \theta') \leftarrow (x, \theta)$ 
14:    $Q \leftarrow Q \setminus \{(x, \theta', \mathcal{B}')\}$ 
15: return  $\{\}$ 

```

At the start of Algorithm 2, we initialize a priority queue that is used to store the leaf nodes of the search tree. Each node is comprised of the position of the quadrotor, $x \in \mathbb{R}^3$, the orientation of the quadrotor on the Z-axis, $\theta \in [0, 2\pi]$, and the remaining blind region, \mathcal{B} , that is left unobserved after the quadrotor reaches x with orientation θ . Until the search timeout has expired, collision free neighbours of x are added to the search along with their maximizing orientation and remaining unobserved blind regions. Once the search has expired, the path, Π comprised of 3D positions and orientations, that was able to view the largest cumulative blind region area starting from x_0 is returned. Figure 1d shows an example of a path being computed to view the blind regions.

Since the obstacles and blind regions are two dimensional, we fix the altitude of the quadrotor in our experiments. In the future we would like to extend the blind region detection and planning to three dimensions by using a 3D point cloud sensor mounted on the ground vehicle.

2.4 Autonomous Landing

To enable autonomous landing, the quadrotor tries to maintain a static position relative to the car as it drives towards a parking spot. Once the vehicle has stopped, the quadrotor moves directly above the landing platform and proceeds to land on the platform.

3 Relative Pose Localization

From each UWB tag we receive a range measurement r_i in its own frame. From the quadrotor we receive velocity measurements v , a yaw reading ψ_q , and an altitude measurement z_a in an ENU-aligned world frame. The frames and relative transforms of our system are visualized in Fig. 2. Given n UWBs, we define the measurement vector as $z = [r_1, \dots, r_n, v, z_a, \psi_q] \in \mathbb{R}^{n+5}$.

Yaw orientation is calibrated at the start by lining up the quadrotor along the car's x-axis and measuring the yaw offset ψ_{off} between the car and the quadrotor. The yaw of the quadrotor is then given by $\psi = \psi_q - \psi_{\text{off}}$.

One challenge we encountered in estimating the 3D position of the quadrotor was that the distance measurements from the UWBs showed larger errors when the UWB on the quadrotor was out of their plane. We therefore first estimate \hat{p}^{odom} using only use the quadrotor's onboard odometry readings v and z_a as the inputs to a UKF. We then use the estimated height, \hat{p}_z^{odom} with the UWB range measurements r_i to estimate the quadrotor's x-y position, \hat{p}^{xy} . We do this by first projecting each r_i onto the plane of the estimated height of the quadrotor:

$$r_i^{\text{proj}} = \sqrt{r_i^2 - (\hat{p}_z^{\text{odom}})^2}$$

We then find \hat{p}^{xy} by solving the nonlinear least squares optimization

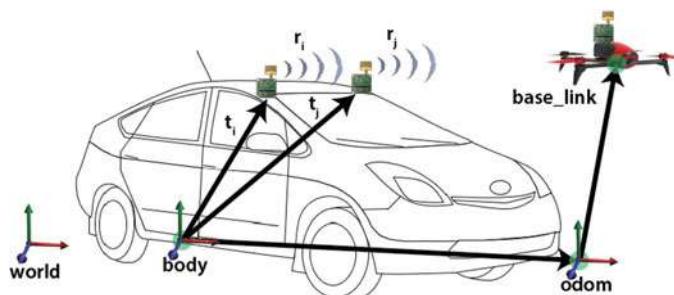


Fig. 2 The frames and measurements of our system

$$h(\hat{\mathbf{p}}^{xy}) = \min_{\hat{\mathbf{p}}^{xy}} \sum_{i=1}^n ((r_i^{\text{proj}})^2 - \|\hat{\mathbf{p}}^{xy} - \mathbf{t}_i^{xy}\|^2)^2$$

We then define the 3D position estimate to be $\hat{\mathbf{p}}^{ls} = [\hat{\mathbf{p}}^{xy}, \hat{z}]$. Next, we combine $\hat{\mathbf{p}}^{ls}$ and $\hat{\mathbf{p}}^{\text{odom}}$ in a second UKF to find a final position estimate $\hat{\mathbf{p}}$. Thus the final state estimate is $\hat{x} = [\hat{\mathbf{p}}, \psi]$.

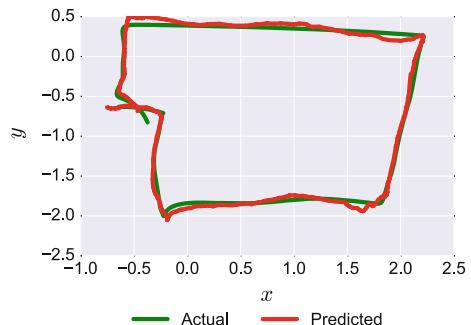
4 Results

In this section we provide experimental results that validate our approach. A video accompanies this submission and is available at [\[21\]](#).

4.1 Localization Accuracy

We tested our localization framework by emulating the car's UWB configuration inside a motion capture system. We placed motion capture markers on the quadrotor and on each UWB sensor. This allowed us to obtain the absolute position of the quadrotor and UWBS in the same coordinate frame. We then flew the quadrotor inside the motion capture system and recorded its predicted position determined by our localization and absolute position using the motion capture markers. We ran 10 tests and were able to obtain an error of 13.7 cm, or 35.9% the length of the quadrotor. Figure 3 shows how our localization compares to the ground truth. The green and red lines respectively show the ground truth and predicted positions of the quadrotor. While our accuracy is less than the system in [10], the UWBS in that study were all in the same plane. The accuracy of out-of-plane position estimation using range measurements is lower than in-plane estimation because of the larger state space. Since we wanted our quadrotor to have the ability to fly beyond the plane of the roof-mounted UWBS, we included additional out-of-plane UWBS which decrease

Fig. 3 Comparison of our localization method with respect to ground truth. Ground truth was supplied by a motion capture system



accuracy compared to having all UWBS in the same plane, but which help to provide greater accuracy for out-of-plane measurements.

4.2 Experimental Setup

For our experiments, we used a Toyota Prius with a SICK LMS1xx mounted on the front of the car and six Decawave TREK1000 UWBS mounted on the roof and front bumper of the car. A platform for the quadrotor to take off and land was attached to the front bumper of the Prius. We used a Parrot Bebop 2 quadrotor with a Decawave TREK1000 mounted on the battery. Figure 4 shows the Toyota Prius and modified Bebop 2 quadrotor used in the experiments.

We also ran tests using an autonomous golf cart as our ground vehicle in two different settings. One setting, shown in Fig. 5d, was artificially created using tall whiteboards as obstacles to mimic an adversarial environment. The second, shown in Fig. 5c, was a more realistic setting with the golf cart approaching an open garage door with blind spots on either side. In both cases, the quadrotor was successfully able to observe the blind spots and relay this information back to the computer on board the golf cart. For the interest of brevity, we will only discuss in detail the experiments using the Toyota Prius.

Our experimental scenario involves a car preparing to leave a garage with a significant blind spot. The car is unable to sense around the corner to determine if there are pedestrians or other cars that may obstruct its path. Our quadrotor takes off from the car's front bumper platform and autonomously flies out of the garage and looks around the corner. The car is then able to leave the garage when there are no more pedestrians detected by the quadrotor. Once the car is ready to return, it backs up into the garage. The quadrotor then follows the car into the garage and autonomously lands on the platform.



Fig. 4 Pictures showing Toyota Prius and Parrot Bebop 2 used in the experiments

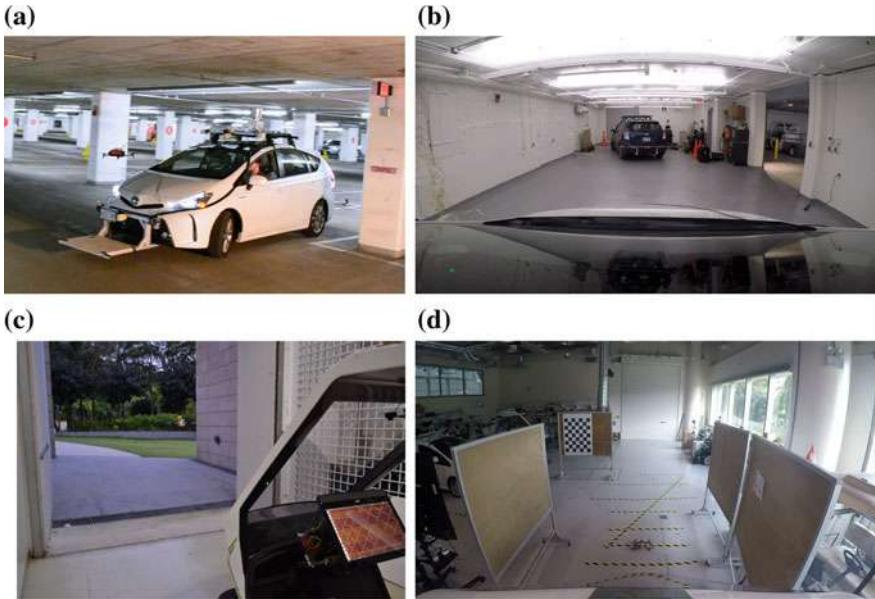


Fig. 5 Snapshots from four experimental settings in which we tested our algorithm. Figures **a** and **b** used a Toyota Prius in a parking lot and a garage, respectively. Figures **c** and **d** used an autonomous golf cart in outdoor and indoor environments

4.3 Experiment with Quadrotor

Under each experimental condition shown in Fig. 5, we conducted multiple tests. In the course of one afternoon we performed 25 tests in the environment shown in Fig. 5b in which the quadrotor successfully took off from the car, followed a path to observe blind spots, and landed back on the car's platform. Each test took around a minute to autonomously look around the corner and land back on the car. In every case, take off, path following, and landing was successfully completed. For the remainder of this section we will detail one representative experiment.

4.3.1 Looking Around the Corner

Figure 6 shows snapshots of the experiment as it progressed. The first column is a third person angle of the Prius and the quadrotor. The second column shows frames from the quadrotor's on-board camera along with object detection and classifications from the convolutional neural net. The third column is a visualization of the sensor data from the car, the bounding polygon, blind regions, and the quadrotor's plan. Each row shows a single snapshot from the experiment.



Fig. 6 Snapshots from the experiment as it progressed. The second column is from the quadrotor’s on board camera. The third column shows a visualization of the planner

The snapshots show that the quadrotor is able to successfully take off from the car, use the laser scan to find the blind regions, and plan a path to look around the corner in the garage. The last row shows that our system is able to detect the pedestrian around the corner and provide the bounding box back to the car.

Note that even though the quadrotor is not equipped with the sensors needed to perform robust 3D obstacle avoidance, it is able to avoid collisions and fly through the open garage door using the laser scan from the car.

4.3.2 Landing on the Car

Once the car is ready to park, the quadrotor is able to autonomously land back on the platform attached to the front bumper. Figure 7 shows snapshots from the experiment



Fig. 7 Snapshots of the quadrotor landing on the car

as the quadrotor followed the car and landed on the platform. The first image in Fig. 7 shows the quadrotor following the car as it backs up into a garage. The second shows the car parked and the quadrotor hovering over the platform. The last image shows the quadrotor after it successfully landed on the platform.

5 Conclusion

In this work we presented a system for using a quadrotor to examine the blind spots of an autonomous car. We developed a path planning algorithm that maximizes visual coverage of blind spots in a 2D laser scan; created an experimental system using UWBs to localize the quadrotor with respect to the car; and performed tests in a variety of environments to verify the effectiveness of our system. Extensions to our work include planning using 3D laser scan data; improving the accuracy of the UWB localization; and applying our system to other problems such as package delivery and formation control. We believe that multi-robot coordination, particularly in the context of an autonomous car and a quadrotor, will become increasingly useful in the future.

References

1. NHTSA traffic safety facts. <https://crashstats.nhtsa.dot.gov/>
2. van Diggelen, F., Enge, P.: The Worlds first GPS MOOC and Worldwide Laboratory using Smartphones. ION Publications (2015)
3. Decawave: <http://www.decawave.com/>. Accessed 07 Apr 2017 (2017)
4. ASIRT: ASIRT association for safe international road travel 2016. (2016)
5. Borowczyk, A., Nguyen, D.T., Nguyen, A.P.V., Nguyen, D.Q., Saussié, D., Ny, J.L.: Autonomous landing of a multirotor micro air vehicle on a high velocity ground vehicle (2016). arXiv preprint [arXiv:1611.07329](https://arxiv.org/abs/1611.07329)
6. Bradley, R.: Tesla Autopilot (2016)
7. Dagan, E., Mano, O., Stein, G.P., Shashua, A.: Forward collision warning with a single camera. In: Intelligent Vehicles Symposium, 2004 IEEE, pp 37–42, (2004). <https://doi.org/10.1109/IVS.2004.1336352>
8. Delmerico, J., Mueggler, E., Nitsch, J., Scaramuzza, D.: Active autonomous aerial exploration for ground robot path planning. IEEE Robot. Autom. Lett. **2**(2), 664–671 (2017)
9. Forster, C., Pizzoli, M., Scaramuzza, D.: Air-ground localization and map augmentation using monocular dense reconstruction. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp 3971–3978 (2013)
10. Hepp, B., Nägeli, T., Hilliges, O.: Omni-directional person tracking on a flying robot using occlusion-robust ultra-wideband signals. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp 189–194 (2016)
11. Herissé, B., Hamel, T., Mahony, R., Russotto, F.X.: Landing a vtol unmanned aerial vehicle on a moving platform using optical flow. IEEE Trans. Robot. **28**(1), 77–89 (2012)
12. Hollinger, G.A., Djugash, J., Singh, S.: Target tracking without line of sight using range from radio. Auton. Robot. **32**(1), 1–14 (2012)

13. Kempke, B., Pannuto, P., Dutta, P.: Harmonium: asymmetric, bandstitched uwb for fast, accurate, and robust indoor localization. In: Proceedings of the 15th International Conference on Information Processing in Sensor Networks. IEEE Press, p. 15 (2016)
14. Liu, H., Darabi, H., Banerjee, P., Liu, J.: Survey of wireless indoor positioning techniques and systems. *IEEE Trans. Syst. Man Cybern. Part C (Applications and Reviews)* **37**(6):1067–1080 (2007)
15. Michael, N., Shen, S., Mohta, K., Mulgaonkar, Y., Kumar, V., Nagatani, K., Okada, Y., Kiribayashi, S., Otake, K., Yoshida, K., Ohno, K., Takeuchi, E., Tadokoro, S.: Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *J. Field Robot.* **29**(5), 832–841 (2012)
16. Mueller, M.W., Hamer, M., D’Andrea, R.: (2015) Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp 1730–1736, IEEE
17. Naegeli, T., Alonso-Mora, J., Domahidi, A., Rus, D., Hilliges, O.: Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization. *IEEE Robot. Autom. Lett.* **PP**(99) (2017)
18. Nohmi, M., Fujikura, N., Ueda, C., Toyota, E.: Automatic braking or acceleration control system for a vehicle. US Patent 4,066,230 (1978)
19. Prorok, A., Martinoli, A.: Accurate indoor localization with ultra-wideband using spatial models and collaboration. *Int. J. Robot. Res.* **33**(4), 547–568 (2014)
20. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
21. Wallar, A., Araki, B., Chang, R., Alonso-Mora, J., Rus, D.: Supporting video material for Foresight (2015). <https://youtu.be/IRhWPcHZmuU>
22. Wenzel, K.E., Masselli, A., Zell, A.: Automatic take off, tracking and landing of a miniature uav on a moving carrier vehicle. *J. Intell. Robot. Syst.* **61**(1), 221–238 (2011)

Dynamic System Identification, and Control for a Cost-Effective and Open-Source Multi-rotor MAV

Inkyu Sa, Mina Kamel, Raghav Khanna, Marija Popović,
Juan Nieto and Roland Siegwart

1 Introduction

Multi-rotor MAV platforms are rotorcraft air vehicles that use counter-rotating rotors to generate thrust and rotational forces. These vehicles have become a very popular research and commercial platform during the past decade. The wide variety of ready-to-fly platforms today is proof that they are being utilized for real-world aerial tasks such as indoor and outdoor inspection [1], aerial photography, cinematography, and environmental survey and monitoring for agricultural applications. The performance of these vehicles has also shown steady improvement over time in terms of flight time, payloads, and safety-related smart-features. However, it is challenging to adapt these commercial platforms for robotic tasks such as obstacle avoidance and path planning [2, 3], object picking [4], and precision agriculture [5]. Because an accurate dynamics model, a low-latency and precise state estimator, and a high-performance controller are required to perform these tasks.

Ascending Technologies provides excellent research grade MAV platforms [6, 7] dedicated to advanced aerial robotic applications. There is also a well-explained software development kit (SDK) and abundant scientific resources. These platforms are ideal for developing aerial robots. However, their relative expensive cost may be a hurdle for researchers and replacing parts in a case of a crash (which can occur in the early development stage) is time-consuming due to the limited number of retail shops.

For vertical take-off and Landing (VTOL) MAVs to become more pervasive, they must become lower in cost and their parts easier to replace. There is an affordable consumer grade VTOL platform shown in Fig. 1. Developers can now access sensor

I. Sa (✉) · M. Kamel · R. Khanna · M. Popović · J. Nieto · R. Siegwart
Autonomous Systems Lab, ETH Zurich, Zurich, Switzerland
e-mail: inkyu.sa@mavt.ethz.ch

M. Kamel
e-mail: mina.kamel@mavt.ethz.ch

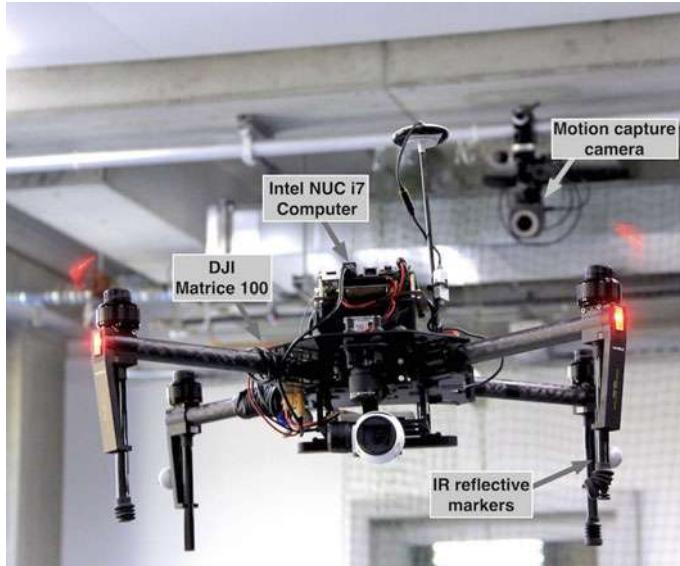


Fig. 1 A commercial Multi-rotor MAV quadrotor platform (Matrice 100). An onboard computer is mounted, and the external motion capture device provides position, velocity, and orientation measurements at 100 Hz for Model Predictive Controller (MPC)

data such as IMU and barometers and send command data to the low-level attitude controller. It is easy and prompt to order parts from local retail stores with short delivery spans. There are, however, difficulties in using these platforms for robotics applications (e.g., lack of essential scientific resources such as attitude dynamics and the structure of underlying autopilot controller). This information is essential for the development of aerial robots. In this paper, we address these gaps by performing system identification using only the built-in onboard IMU. Researchers can perform their system identification with the provided documentation and build a base platform for field and service aerial robots.

The contributions of this system paper are:

- Presenting full dynamics system identification that is employed by a subsequent MPC position controller enabling to build a research grade field and service aerial robotic platform.
- Delivering software packages including modified SDK, linear MPC, and system identification tools and their documentation to the community.

<http://goo.gl/lXRnU8>

The benefit of this paper would be that the proposed techniques can be directly applied to other products such as Matrice 200, or 600 series.

The remainder of this paper is structured as follows. Section 2 introduces state-of-the-art work on MAV system identification and control and Sect. 3 describes the

specification of the vehicle, system identification, and control strategies. We present our experimental results in Sect. 4, and conclude in Sect. 5.

2 Related Work/Background

VTOL MAVs' popularity is gaining momentum both industry and research field. It is necessary to identify their underlying dynamics system and behavior of attitude controllers to achieve good control performance. For a common quadrotor, the rigid vehicle dynamics are well-known [8] and can be modeled as a non-linear system with individual rotors attached to a rigid airframe, taking into account of drag force and blade flapping [9]. In practice, however, the identification of attitude controllers is often a non-trivial task for such consumer products due to the lack of scientific resources.

There are system identification techniques to estimate dynamic model parameters in literature. Traditionally, parameter estimation has been performed offline using complete measurement data obtained from a physical test bed and CAD models [10, 11]. Such pioneer offline methods significantly contributed to the VTOL MAV community in the early development stage. Recently, Burri et al. [12] demonstrated a method for identification of the dominant dynamic parameters of a VTOL MAV using Maximum Likelihood approach. Alternatively, a linear least-squares method is used to estimate parameters from recorded flight data in batch processing manner [13, 14]. We follow a batch-based approach to determine the dynamic parameters of the vehicle from short manual pilots. This allows us to obtain the parameters necessary for MPC using only the onboard IMU and without applying any restrictive simplifying assumptions. Given the identified dynamics model, we use a high-performance state-of-the-art Model Predictive Control (MPC) [15] for horizontal position control.

3 Matrice 100 VTOL MAV Platform and Control

In this section, we present overviews for the hardware platform, software development toolkit, and address coordinate systems, attitude dynamics, and control strategy.

We define 2 right-handed frames following standard Robot Operating System (ROS) convention: world $\{\mathcal{W}\}$ and body $\{\mathcal{B}\}$ shown in Fig. 2. The x-axis in $\{\mathcal{B}\}$ indicates forward direction for the vehicle, y-axis is left, and z-axis is up. We use Euler angles; roll (ϕ), pitch (θ), and yaw (ψ) about x, y, z-axes respectively for the RMS error calculation and visualization purposes. Quaternions are utilized for any computational processes. The defined coordinate systems and notations are used over the rest of paper.¹

¹Image source from <http://goo.gl/7NsbmG>.

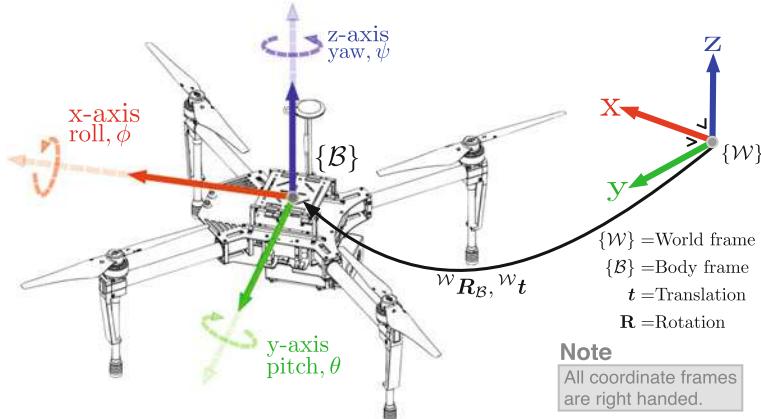


Fig. 2 Coordinate systems definition. ${}^W t$ is the 3×1 translation vector w.r.t $\{W\}$. ${}^W R_B$ is 3×3 matrix and rotates a vector defined w.r.t $\{B\}$ to a vector w.r.t $\{W\}$

3.1 Aerial Platform and Its SDK

The general hardware specification of Matrice 100 is well documented, this section only highlights our findings. The vehicle is a quadrotor and has 650 mm diagonal length. It uses N1 flight controller, but the information regarding the device is not disclosed to the public. The variety of sensing data can be accessed using SDK through serial communication, such as IMU, GPS, barometer, and magnetometer. The SDK enables to access most functionalities and supports cross-platform development environments. We use the onboard SDK with the Robot Operating System (ROS) wrapper, but there is a fundamental issue for sending control commands with this protocol. The manufacturer uses ROS services to send commands that are strongly not recommended.² It is a blocking call that should be used for triggering signals or quick calculations. If data transaction (hand-shaking) remains as a failure for some reasons (e.g., poor WiFi connection), it blocks all subsequent calls. Small latency ≈ 10 ms in control commands makes a huge difference in the resultant performance. We thus modify the SDK to send direct control commands via serial communication.

Common control commands through a transmitter are pitch, roll angles (rad), yaw rate (rad/s), and thrust (N). However, the vertical stick input of the platform is velocity (m/s) that permits easier and safer manual flight. To address this different, we use a classic PID vertical position controller alongside linear MPC horizontal position controller. Interestingly, there is no trim buttons on the provided transmitter.; instead, the N1 autopilot has auto-trim functionality (e.g., position mode) that balances attitude by estimating horizontal velocity. This feature allows easier and safer manual piloting but introduces a constant offset position error for controlling. This needs

²<http://wiki.ros.org/ROS/Patterns/Communication>.

to be properly compensated. We estimate the balancing point where the vehicle's motion is minimum (hovering) and then adjust the neutral position to the estimated balancing point. If there is a change in an inertial moment (e.g., mounting a new device or changing the battery position), the balancing position has to be updated.

Another interesting aspect of the autopilot is the presence of relatively large dead zone that is the range close to the neutral value. Within this zone, the autopilot ignores all input commands, and no API is supported to set this. This function is also useful for a manual pilot since the vehicle should not react to small inputs yielded by tremor of hands but significantly degrades the performance of the controller. We determine this by sweeping control commands around dead zone area and detecting the control inputs when any motion is generated. Although this task is difficult with a real VTOL platform due to its fast and naturally unstable dynamics, we use the manufacturer's hardware-in-loop simulator (DJI Assistant 2)³ that enables to receive input commands from the transmitter. After determine the dead zone, we simply compensate it by adding the estimated offset to control commands if they lie within the dead zone range. Details of the modifications and tutorial are given in <http://goo.gl/vSCQjg>.

3.2 Dynamic Systems Identification

In this section, we present full dynamics system identification resulting from the simulator and experiments. We record input and output data; Virtual RC commands and attitude response while manual flight on an onboard computer.

3.2.1 Input Commands Scaling

Prior to performing system identification, it is necessary to identify the relation between Virtual RC (actual control commands) and the corresponding attitude measurements from the IMU. This can be determined by linearly mapping with the maximum/minimum angles ($\pm 30^\circ$), however there is small error in practice. This can be caused by a variety of sources such as unbalanced platform, small dynamics modelling error. We estimate these parameters using nonlinear least-squares optimization such that

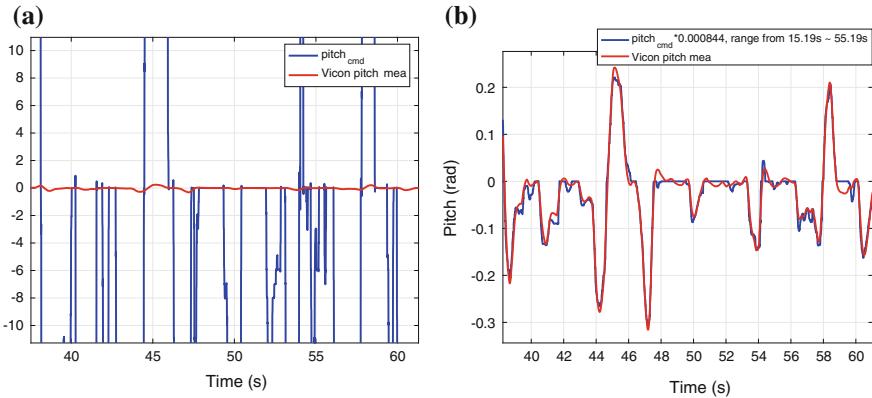
$$\lambda^* := \underset{\lambda}{\operatorname{argmin}} \sum_{k=1}^T \|z_k^{\text{Meas}} - \lambda u_k^{\text{cmd}}\|^2 \quad (1)$$

where T denotes the number of samples used for optimization. λ is 4×1 vector containing roll, pitch, and yaw rate scaling parameters, $[\lambda_\phi, \lambda_\theta, \lambda_\psi, \lambda_{\dot{z}}]^T$. z_k^{Meas} is

³<http://goo.gl/Pk5kTL>.

Table 1 Virtual RC scaling parameters

Scale param	Experiment	Simulator
λ_ϕ	8.65×10^{-4}	8.35×10^{-4}
λ_θ	8.44×10^{-4}	8.23×10^{-4}
$\lambda_{\dot{\psi}}$	2.24×10^{-3}	3.23×10^{-3}
λ_z	2.65×10^{-3}	3.02×10^{-3}

**Fig. 3** u_ϕ scaling parameter estimation. **a** and **b** are before and after the scaling

4×1 of $[\phi, \theta, \dot{\psi}, \dot{z}]^T$ obtained from IMU and a motion capture device at 100 Hz. u_k^{cmd} is also 4×1 vector of Virtual RC commands $[u_\phi, u_\theta, u_{\dot{\psi}}, u_z]^T$. Note that it is difficult to measure vertical velocity using IMU, but there are three options; (1) motion capture device, (2) ultrasonic, barometer, and IMU fusion for vertical velocity estimation from the onboard flight controller, (3) using the simulator. The first approach is the most accurate but the third is useful and produces similar results as shown Table 1. The input commands and output measurements are aligned with cross-correlation to remove the delay between the two signals. This is acceptable since we estimate the signal magnitude. Figure 3a shows original input command in blue and angle measurement in red. (b) displays results after scaling using estimated parameters from Table 1.

3.2.2 Roll and Pitch Attitude Dynamics

Our linear MPC controller [15] requires first order attitude dynamics for position control and the second order for the disturbances observer. We estimate the dynamics by recording the input and output at 100 Hz and logged two sets of a dataset for model training and validation.

We assume a low-level flight controller that can track the reference roll, ϕ^* , and pitch, θ^* , angles with first order behavior. The first order approximation provides sufficient information to the MPC to take into account the low-level controller behavior. We thus utilize classic system identification techniques such that:

$$\frac{y(s)_\phi}{u(s)_\phi} = \frac{3.544}{s + 2.118}, \quad \frac{y(s)_\theta}{u(s)_\theta} = \frac{3.827}{s + 2.43}$$

$y(s)_\phi$ and $u(s)_\phi$ are IMU measurement and input commands in continuous-time space. The time constants for roll and pitch are, $\tau_\phi = 0.472$, $\tau_\theta = 0.472$ and DC gains are $k_\phi = 1.673$, $k_\theta = 1.575$.

The identified first order dynamic models in continuous time space are discretized in MPC and will be addressed in the next Sect. 3.3. We also performed second order dynamic system identification exploited by disturbances observer, and the dynamic models are

$$\frac{y(s)_\phi}{u(s)_\phi} = \frac{26.37}{s^2 + 5.32s + 27.04}, \quad \frac{y(s)_\theta}{u(s)_\theta} = \frac{28.86}{s^2 + 6.00s + 27.45}$$

Their gain, k_ϕ , damping, ζ_ϕ , and natural frequency, ω_ϕ are presented in Table 2. Figure 4 shows measured attitude, estimated first and second order dynamics. The models fit close to the measurement (dotted line), and they are utilized by controllers presented in next Sect. 3.3.

3.2.3 Yaw and Height Dynamics

The input commands of yaw and height are rates, $u_{\dot{\psi}}$, $u_{\dot{z}}$, and the desired references are orientation, ψ^* and position, z^* . This implies there are controllers that track the desired yaw rate, $\dot{\psi}^*$, and height velocity, \dot{z}^* . Their first order dynamics are

$$\frac{y(s)_{\dot{\psi}}}{u(s)_{\dot{\psi}}} = \frac{5.642}{s + 5.268}, \quad \frac{y(s)_{\dot{z}}}{u(s)_{\dot{z}}} = \frac{3.342}{s + 2.99}$$

Table 2 Matrice 100 identified dynamics summary

	ϕ	θ	$\dot{\psi}$	\dot{z}
1st order	$\tau_\phi = 0.472$	$\tau_\theta = 0.472$	$\tau_{\dot{\psi}} = 0.161$	$\tau_{\dot{z}} = 0.334$
	$k_\phi = 1.673$	$k_\theta = 1.575$	$k_{\dot{\psi}} = 1.057$	$k_{\dot{z}} = 1.118$
2nd order	$k_\phi = 0.975$	$k_\theta = 1.052$	$k_{\dot{\psi}} = 1.079$	$k_{\dot{z}} = 1.024$
	$\zeta_\phi = 0.512$	$\zeta_\theta = 0.573$	$\zeta_{\dot{\psi}} = 1.898$	$\zeta_{\dot{z}} = 0.718$
	$\omega_\phi = 5.200$	$\omega_\theta = 5.239$	$\omega_{\dot{\psi}} = 23.448$	$\omega_{\dot{z}} = 4.985$

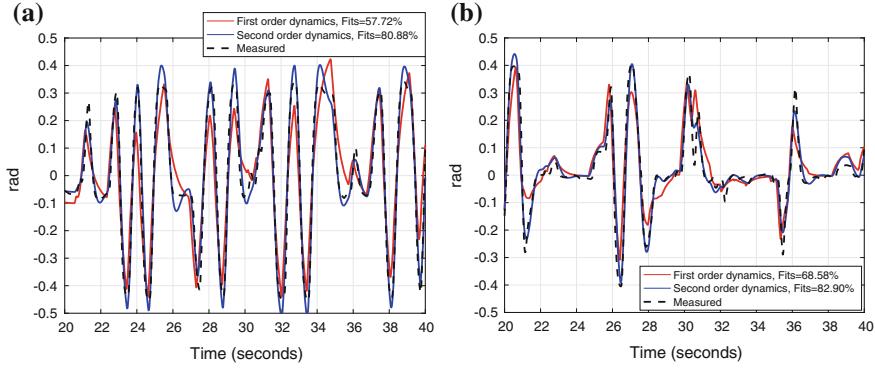


Fig. 4 Measured and predicted pitch (a) and roll (b) angles for manual flight. The black denotes the measured angle, and the blue represents the model response

Similarly, the second order dynamics for $\dot{\psi}$ and \dot{z} are identified as

$$\frac{y(s)\dot{\psi}}{u(s)\dot{\psi}} = \frac{593.3}{s^2 + 89.0s + 549}, \quad \frac{y(s)\dot{z}}{u(s)\dot{z}} = \frac{25.43}{s^2 + 7.16s + 24.8}$$

$y(s)\dot{\psi}$ is obtained from the built-in IMU, gyro measurement along the z-axis, and $y(s)\dot{z}$ is provided by a motion capture device. It is also feasible to identify the height dynamics by utilizing vertical velocity estimation from N1 flight controller.

3.3 Linear-MPC for Horizontal Control

The control of the lateral motion of the vehicle is based on a Linear Model Predictive Control (MPC) [15]. The vehicle dynamics are linearized around the hovering condition. We define the state vector to be $x = (x, y, v_x, v_y, \mathcal{W}\phi, \mathcal{W}\theta)$ and the control input vector to be $u = (\mathcal{W}u_\phi, \mathcal{W}u_\theta)$. We also define the reference state sequence as $X_{\text{ref}}^T = [x_{\text{ref},0}^T, \dots, x_{\text{ref},N}^T]^T$, the control input sequence as $U = [u_0^T, \dots, u_{N-1}^T]^T$, and the steady state input sequence $U_{\text{ref}} = [u_{\text{ref},0}^T, \dots, u_{\text{ref},N-1}^T]^T$. $x_{\text{ref},k}$, $u_{\text{ref},k}$ are the k^{th} reference state and control input. Every time step, the following optimization problem is solved:

$$\begin{aligned} \min_{U, X} & \sum_{k=0}^{N-1} ((x_k - x_{\text{ref},k})^T Q_x (x_k - x_{\text{ref},k}) \\ & + (u_k - u_{\text{ref},k})^T R_u (u_k - u_{\text{ref},k}) \\ & + (u_k - u_{k-1})^T R_\Delta (u_k - u_{k-1})) \\ & + (x_N - x_{\text{ref},N})^T P (x_N - x_{\text{ref},N}) \end{aligned} \quad (2)$$

$$\begin{aligned} \text{subject to } & x_{k+1} = Ax_k + Bu_k + B_d d_k; \\ & d_{k+1} = d_k, \quad k = 0, \dots, N-1 \\ & u_k \in \mathbb{U} \\ & x_0 = x(t_0), \quad d_0 = d(t_0). \end{aligned}$$

where $Q_x \succeq 0$ is the penalty on the state error, $R_u > 0$ is the penalty on control input error, $R_\Delta \succeq 0$ is a penalty on the control change rate and P is the terminal state error penalty. The \succeq operator denotes positive definiteness of a matrix.⁴ d_k is the estimated external disturbances. Note that the attitude angles ϕ, θ are rotated into the inertial frame to get rid of the vehicle heading ψ .

A high-performance solver has been generated to solve the optimization problem (2) using the FORCES⁵ framework. The solver is running in real-time for a prediction horizon of $N = 20$ steps. Moreover, to achieve an offset-free tracking, the external disturbances d_k has to be estimated and provided to the controller each time step. These disturbances include external forces (the wind for instance) and also a modeling error. The disturbances are estimated using an augmented Extended Kalman Filter (EKF) based on the second order dynamics model identified from the previous section. For the remaining axes control (i.e., height and yaw), we use standard PID controllers.

4 Experimental Results

In this section, we present implementation details; hardware and software setup and control performance evaluation through experiments.

4.1 Hardware Setup

Matrice 100 quadcopter carries an Intel NUC 5i7RYH (i7-5557U, 3.1 GHz dual cores, 16 GB RAM), running Ubuntu Linux 14.04 and ROS Indigo onboard. A Vicon motion capture system consisting of 6 IR cameras provides 6 DOF pose of the quadcopter target at 100 Hz, used for control. The quadcopter is also equipped with a flight controller, N1, embedded an onboard IMU which provides vehicle orientation, acceleration, and angular velocity at 100 Hz to the computer via 921,600 bps USB-to-serial communication. Control commands are also transmitted at 100 Hz through the serial bus.

⁴https://en.wikipedia.org/wiki/Positive-definite_matrix.

⁵<http://embotech.com/FORCES-Pro>.

The total system mass is 3.3 kg and WiFi is used for communicating with the quadcopter using a ground control laptop via ROS and a customized onboard SDK ROS interface. The ground station, Vicon server, and onboard computer are time synchronized via chrony time sync software package. The vehicle carries 0.92 kg payload with the onboard computer, and a gimbal camera. 6 cells LiPo battery, 22.2 V, 4500 mAh powers the vehicle and the total flight time is around 14 mins with a small angle of attack $\approx \pm 20^\circ$.

4.2 Software Setup

We integrate the system using ROS as shown in Fig. 5. Each box represents a ROS node and runs at 100 Hz. The Vicon server publishes position, orientation, translational and angular velocity as denoted $[p, q, \dot{p}, \dot{q}]$ using `ros_vrpn_client`. The data is subscribed by the Multi-Sensor Fusion (MSF) framework [7] to filter noisy measurement and to compensate for a delay. The ground station sets either a goal pose as denoted p^* , position, and q^* , orientation or N sequences, $[p_{1:N}^*, q_{1:N}^*]$, planned by the trajectory generator [12].

The SDK runs on the onboard computer and receives IMU measurements, $[\Phi, \dot{\Phi}, {}^B a]$, orientation, angular velocity, acceleration in B coordinate from the N1 flight controller. It also sends the calculated control commands to the attitude controller.

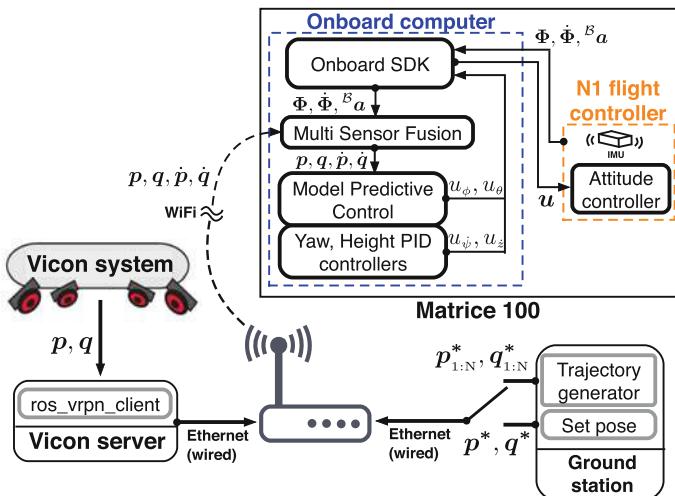


Fig. 5 Software packages implementation using ROS

Table 3 Control performance summary

	Hovering		Step response		Trj. following		Unit
Pose	0.039	0.041	0.394	0.266	0.080	0.103	m
x	0.021	0.027	0.259	0.127	0.058	0.066	m
y	0.016	0.015	0.295	0.226	0.043	0.059	m
z	0.029	0.026	0.034	0.059	0.035	0.053	m
Roll	0.392	1.044	—	—	—	—	deg
Pitch	0.618	0.697	—	—	—	—	deg
Yaw	1.087	1.844	1.141	2.165	1.539	2.876	deg
Duration	15–75	15–75	20–120	20–120	30–80	20–70	s
Wind	—	11–11.5	—	11–11.5	—	11–11.5	m/s

4.3 Experiments Setup

For control performance evaluation, we conduct real-world experiment: hovering, step response, and trajectory following. To demonstrate, the robustness of the controllers, we generate wind disturbances using a fan that has 260 W and 300 m³/min air flow. This produces 11–11.5 m/s disturbance at the hovering position measured by an anemometer. Each task has two results, i.e., with/without wind disturbances.

We use root-mean-square (RMS) error metric between the reference and actual position and orientation measured by a motion capture device for performance evaluation. Euclidian distance is used for 3D pose RMS error calculation. Table 3 summarizes experimental results.

4.4 Control Performance Evaluation

We present quantitative control performance evaluation using RMS error while performing 3 tasks, i.e., hovering, step response, and trajectory following, and qualitative results for step response and trajectory following.

4.4.1 Experiments Results for Hovering

Figure 6 shows hovering results (position and orientation) without any wind disturbances (a) and (b) and with disturbances (c) and (d). Noticeable areas are magnified due to the small scale of plots. We can clearly see the presence of wind disturbances affect to control performance. Especially, the variation in yaw and attitude are significant since a fan is located at South-East of the vehicle. As shown in Table 3, we achieve competitive results while hovering. Interestingly, the position errors for x, y,

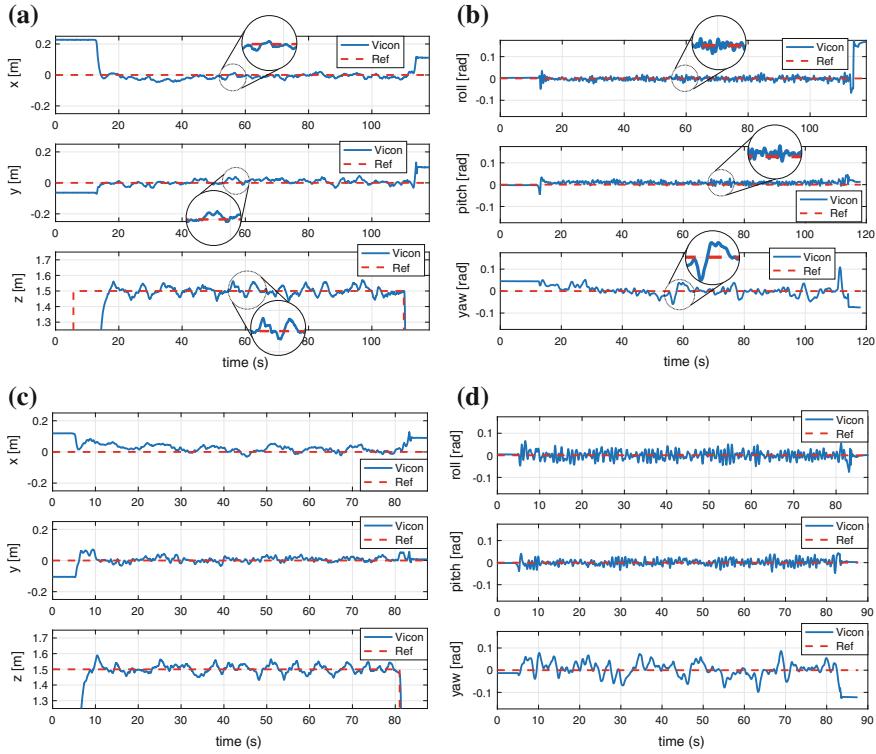


Fig. 6 Hovering control performance without/with wind disturbances

and z are consistent even with wind disturbances, 11–11.5 m/s. The force acting on the platform in the wind is too small to push the 3.3 kg flying platform.

4.4.2 Experiments Results for Step Response

As oppose to the advantage of strong resistance to external disturbances, the downside for the heavy platform is a slower response. Figure 7 shows step response plots without wind disturbances (a), with wind (b). A goal position is manually chosen to excite all axes. The peaks in roll and pitch are caused by tilting toward the direction of maneuver, so we ignore them in RMS calculation. The results from Table 3 show a large control error in both x and y . Slow response causes accumulating error while the vehicle reaches to a reference goal. Note that the RMS error of x -axis in windy condition (i.e., 0.127 m) is much smaller than without wind (0.259 m). This is mainly due to the vehicle travels only 2 m for the former whereas it flies 6 m for the latter.

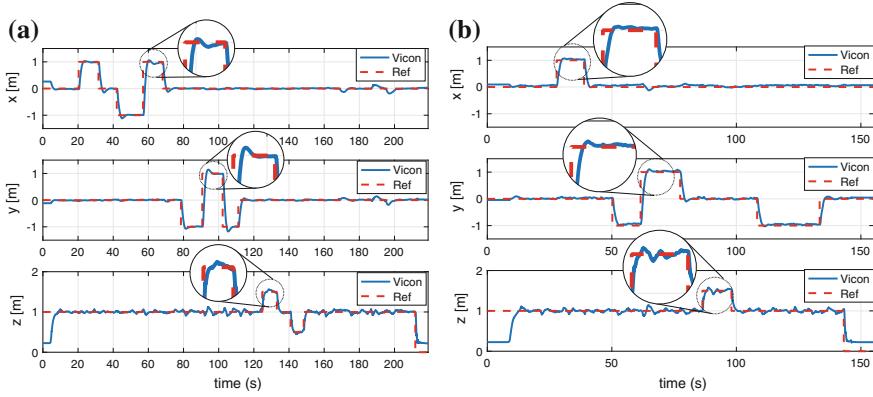


Fig. 7 Step response control performance without (a) and with (b) wind disturbances

4.4.3 Experimental Results for Trajectory Following

We use [2] to generate a smooth polynomial reference trajectory as shown in Fig. 9. Even though hovering and step response tasks explicitly demonstrate control performance and essential functionalities for VTOL MAVs, trajectory following is also a significant task for many robotic applications such as obstacle avoidance, and path planning. Figure 8 shows trajectory following results without wind disturbances (a) and with the wind (b). It can be seen that the platform tracks the reference well in both conditions. RMS errors are also presented in Table 3.

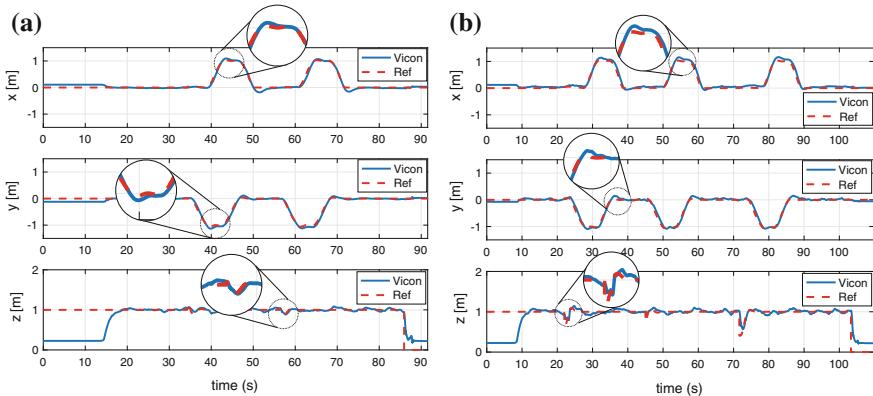


Fig. 8 Trajectory following control performance without (a) and with (b) wind disturbances

4.4.4 Qualitative Results

We present two qualitative results for trajectory following and step response. Figure 9 illustrates the planned trajectory (red) and the vehicle position (blue) obtained from a motion capture device. The left column is without the wind, and the right is in windy condition. Note that a fan is located around 3 m away from the hovering position along the South-East direction as illustrated. Each row is top and side views. It is clearly seen that the trajectory is shifted to the wind direction (positive x and y-direction). Figure 10 shows motions of step response (a) and trajectory following (b). For the step response, it can be seen that the vehicle builds up moments by tilting toward the goal direction and decelerates by tilting into the opposite direction when it approaches. For the trajectory following, the vehicle accurately follows 1×1 square shown in Fig. 10b.

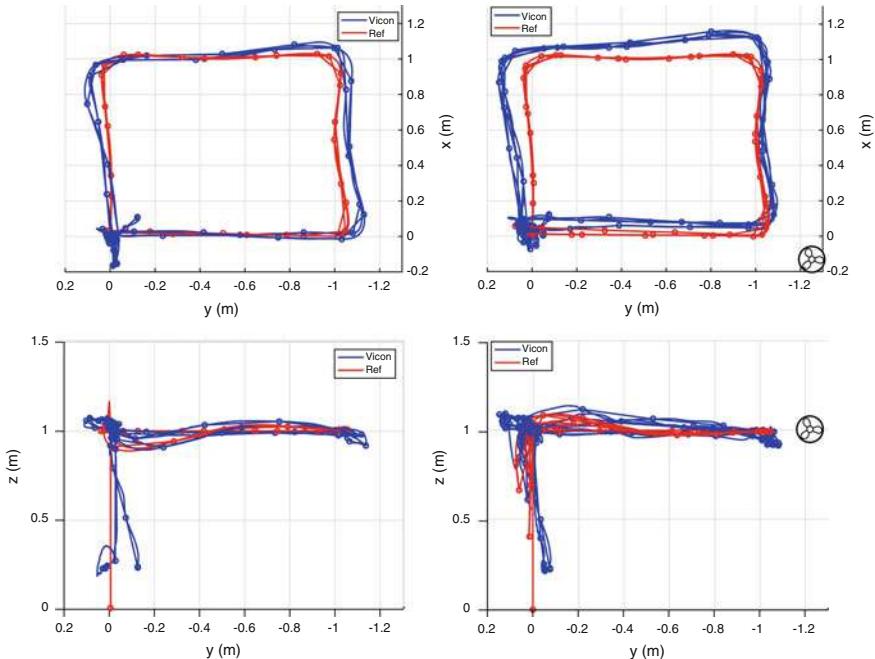


Fig. 9 3D pose during trajectory following without (left column) and with (right column) wind disturbances

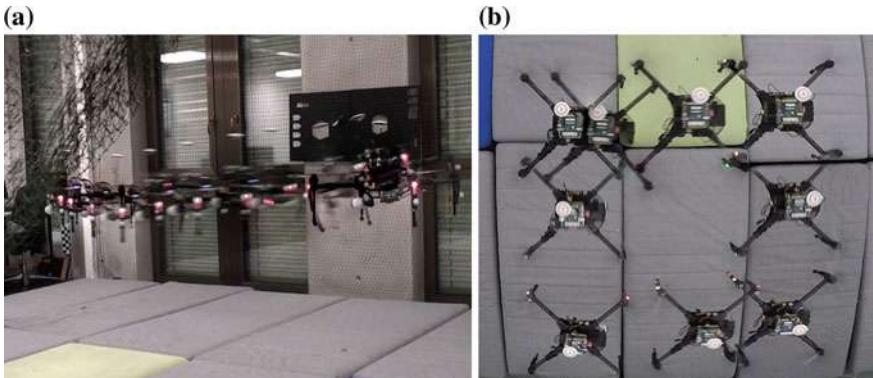


Fig. 10 Vehicle motion while step response (a) and trajectory following (b)

5 Conclusions

We have presented control performance of a cost-effect VTOL MAV platform using classic system identification techniques and the-state-of-the-art MPC controller. The essential information for developing the VTOL MAV robotic platform such as attitude dynamics are addressed. The applied controller performance are evaluated through experiments while executing hovering, step response and trajectory following. Using this platform has many advantages such its low-cost, high payload, ease of use and the ready availability of replacement parts, user-friendly interface, powerful SDK, and a large user community. Our experiences and findings are returned to the community through open documentation and software packages to support researchers having their high-performance MAV platform for diverse field-service aerial robot applications.

Acknowledgements This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644227 and No 644128 from the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number 15.0029 and 15.0044.

References

1. Sa, I., Corke, P.: Vertical infrastructure inspection using a quadcopter and shared autonomy control. In: The International Conference on Field and Service Robotics (2012)
2. Burri, M., Oleynikova, H., Achtelik, M., and Siegwart, R.: Real-time visual-inertial mapping, re-localization and planning onboard MAVs in unknown environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2015)
3. Popovic, M., Hitz, G., Nieto, J., Sa, I., Siegwart, R., Galceran, E.: Online informative path planning for active classification using UAVs. IEEE Int. Conf. Robot. Autom. (2016)

4. Mellinger, D., Lindsey, Q., Shomin, M., Kumar, V.: Design, modeling, estimation and control for aerial grasping and manipulation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2011)
5. Zhang, C., Kovacs, J.: The application of small unmanned aerial systems for precision agriculture: a review. *Precis. Agric.* (2012)
6. Achtelik, M., Weiss, A., Siegwart, R.: Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments. In: Proceedings of the IEEE International Conference on Robotics and Automation (2011)
7. Weiss, S., Scaramuzza, D., Siegwart, R.: Monocular-SLAM-based navigation for autonomous micro helicopters in GPS-denied environments. *J. Field Robot.* (2011)
8. Bouabdallah, S.: Design and control of quadrotors with application to autonomous flying. Ecole Polytechnique Federale de Lausanne, Ph.D. Thesis (2007)
9. Mahony, R., Kumar, V., Corke, P.: Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robot. Autom. Mag.* (2012)
10. Pounds, P., Mahony, R., Corke, P.: Modelling and control of a large quadrotor robot. *Control Eng. Pract.* (2010)
11. Hoffmann, M., Waslander, S., Tomlin, C.: Quadrotor helicopter trajectory tracking control. In: AIAA Guidance, Navigation and Control Conference and Exhibit (2008)
12. Burri, M., Nikolic, J., Oleynikova, H., Achtelik, M., Siegwart, R.: Maximum likelihood parameter identification for MAVs. *IEEE Int. Conf. Robot. Autom.* (2016)
13. Sa, I., Corke, P.: System Identification, Estimation and Control for a Cost Effective Open-Source Quadcopter. *IEEE Int. Conf. Robot. Autom.* (2012)
14. Tischler, M., Remple, R.: Aircraft and rotorcraft system identification. In: AIAA Education Series (2006)
15. Mina, K., Thomas, S., Kostas, A., Roland, S.: Model Predictive Control for Trajectory Tracking of Unmanned Aerial Vehicles Using Robot Operating System. Springer Press (2016) (to appear)

AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles

Shital Shah, Debadeepa Dey, Chris Lovett and Ashish Kapoor

Abstract Developing and testing algorithms for autonomous vehicles in real world is an expensive and time consuming process. Also, in order to utilize recent advances in machine intelligence and deep learning we need to collect a large amount of annotated training data in a variety of conditions and environments. We present a new simulator built on Unreal Engine that offers physically and visually realistic simulations for both of these goals. Our simulator includes a physics engine that can operate at a high frequency for real-time hardware-in-the-loop (HITL) simulations with support for popular protocols (e.g. MavLink). The simulator is designed from the ground up to be extensible to accommodate new types of vehicles, hardware platforms and software protocols. In addition, the modular design enables various components to be easily usable independently in other projects. We demonstrate the simulator by first implementing a quadrotor as an autonomous vehicle and then experimentally comparing the software components with real-world flights.

1 Introduction

Recently, paradigms such as reinforcement learning [1], learning-by-demonstration [2] and transfer learning [3] are proving a natural means to train various robotics systems. One of the key challenges with these techniques is the high sample complexity—the amount of training data needed to learn useful behaviors is often prohibitively high. This issue is further exacerbated by the fact that autonomous vehicles are often unsafe and expensive to operate during the training phase. In order to

S. Shah (✉) · D. Dey · C. Lovett · A. Kapoor
Microsoft Research, Redmond, WA, USA
e-mail: shitals@microsoft.com

D. Dey
e-mail: dedey@microsoft.com

C. Lovett
e-mail: clovett@microsoft.com

A. Kapoor
e-mail: akapoor@microsoft.com

seamlessly operate in the real world the robot needs to transfer the learning it does in simulation. Currently, this is a non-trivial task as simulated perception, environments and actuators are often simplistic and lack the richness or diversity of the real world. For example, for robots that aim to use computer vision in outdoor environments, it may be important to model real-world complex objects such as trees, roads, lakes, electric poles and houses along with rendering that includes finer details such as soft shadows, specular reflections, diffused inter-reflections and so on. Similarly, it is important to develop more accurate models of system dynamics so that simulated behavior closely mimics the real-world.

AirSim is an open-source platform [4] that aims to narrow the gap between simulation and reality in order to aid development of autonomous vehicles. The platform seeks to positively influence development and testing of data-driven machine intelligence techniques such as reinforcement learning and deep learning. It is inspired by several previous simulators (see related work), and one of our key goals is to build a community to push the state-of-the-art towards this goal.

2 Related Work

While an exhaustive review of currently used simulators is beyond the scope of this paper, we mention a few notable recent works that are closest to our setting and has deeply influenced this work.

Gazebo [5] has been one the most popular simulation platforms for the research work. It has a modular design that allows to use different physics engines, sensor models and create 3D worlds. Gazebo goes beyond monolithic rigid body vehicles and can be used to simulate more general robots with links-and-joints architecture such as complex manipulator arms or biped robots. While Gazebo is fairly feature rich it has been difficult to create large scale complex visually rich environments that are closer to the real world and it has lagged behind various advancements in rendering techniques made by platforms such as Unreal engine or Unity.

Other notable efforts includes Hector [6] that primarily focuses on tight integration with popular middleware ROS and Gazebo. It offers wind tunnel tuned flight dynamics, sensor models that includes bias drift using Gaussian Markov process and software-in-loop using Orocos toolchain. However, Hector lacks support for popular hardware platforms such as Pixhawk and protocols such as MavLink. Because of its tight dependency on ROS and Gazebo, it's limited by richness of simulated environments as noted previously.

Similarly, RotorS [7] provides a modular framework to design Micro Aerial Vehicles, and build algorithms for control and state estimation that can be tested in simulator. It is possible to setup RotorS for HITL with Pixhawk. RotorS also uses Gazebo as its platform, consequently limiting its perception related capabilities.

Finally, jMavSim [8] is easy to use simulator that was designed with a goal of testing PX4 firmware and devices. It is therefore tightly coupled with PX4 simulation

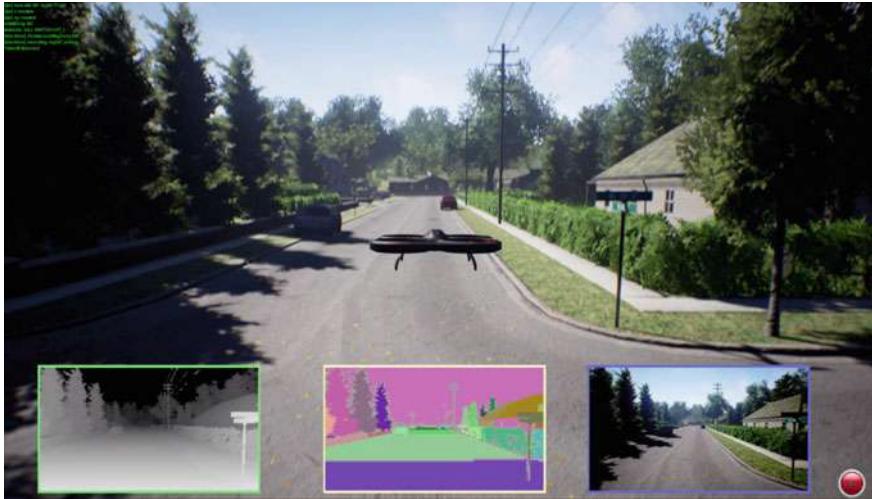


Fig. 1 A snapshot from AirSim shows an aerial vehicle flying in an urban environment. The inset shows depth, object segmentation and front camera streams generated in real time

APIs, uses albeit simpler sensor models and utilizes simple rendering engine without any objects in the environment.

Apart from these, there have been many games like simulators and training applications, however, these are mostly commercial closed-source software with little or no public information on models, accuracy of simulation or development APIs for autonomous applications (Fig. 1).

3 Architecture

Our simulator follows a modular design with an emphasis on extensibility. The core components includes environment model, vehicle model, physics engine, sensor models, rendering interface, public API layer and an interface layer for vehicle firmware as depicted in Fig. 2.

The typical setup for an autonomous aerial vehicle includes the flight controller firmware such as PX4 [9], ROSFlight [10], Hackflight [11] etc. The flight controller takes desired state and the sensor data as inputs, computes the estimate of current state and outputs the actuator control signals to achieve the desired state. For example, in case of quadrotors, user may specify desired pitch, roll and yaw angles as desired state and the flight controller may use sensor data from accelerometer and gyroscope to estimate the current angles and finally compute the motor signals to achieve the desired angles.

During simulation, the simulator provides the sensor data from the simulated world to the flight controller. The flight controller outputs the actuator signals which

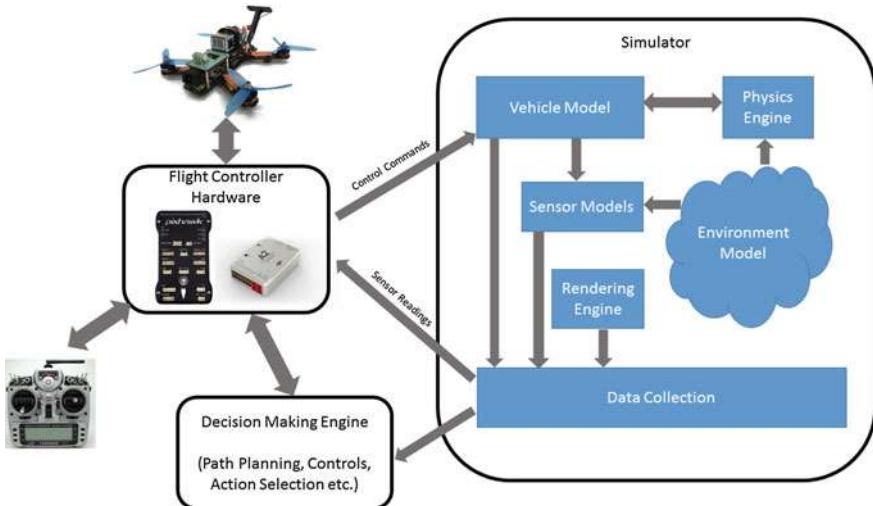


Fig. 2 The architecture of the system that depicts the core components and their interactions

is taken as input by the vehicle model component of the simulator. The goal of the vehicle model is to compute the forces and torques generated by the simulated actuators. For example, in case of quadrotors, we compute the thrust and torques produced by the propellers given the motor voltages. In addition, there may be forces generated from drag, friction and gravity. These forces and torques are then taken as inputs by the physics engine to compute the next kinematic state of bodies in the simulated world. This kinematic state of bodies along with the environment models for gravity, air density, air pressure, magnetic field and geographic location (GPS coordinates) provides the ground truth for the simulated sensor models.

The desired state input to the flight controller can be set by human operator using remote control or by a companion computer in the autonomous setting. The companion computer may perform expensive higher level computations such as determining next desired waypoint, performing simultaneous localization and mapping (SLAM), computing desired trajectory etc. The companion computer may have to process large amount of data generated by the sensors such as vision cameras and lidars which in turn requires that simulated environments have reasonable details. This has been one of the challenging areas where we leverage recent advances in rendering technologies implemented by platforms such as Unreal engine [12]. In addition, we also utilize the underlying pipeline in the Unreal engine to detect collisions. The companion computer interacts with the simulator via a set of APIs that allows it to observe the sensor streams, vehicle state and send commands. These APIs are designed such that it shields the companion computer from being aware of whether its being run under simulation or in the real world. This is particularly important so that one can develop and test algorithms in simulator and deploy to real vehicle without having to make additional changes.

The AirSim code base is implemented as a plugin for the Unreal engine that can be dropped in to any Unreal project. The Unreal engine platform offers an elaborate marketplace with hundreds of pre-made detailed environments, many created using photogrammetry techniques [13] to generate reasonably faithful reconstruction of real-world scenes.

Next, we provide more details on the individual components of the simulator.

3.1 Vehicle Model

AirSim provides an interface to define vehicle as a rigid body that may have arbitrary number of actuators generating forces and torques. The vehicle model includes parameters such as mass, inertia, coefficients for linear and angular drag, coefficients of friction and restitution which is used by the physics engine to compute rigid body dynamics.

Formally, a vehicle is defined as a collection of K vertices placed at positions $\{\mathbf{r}_1, \dots, \mathbf{r}_k\}$ and normals $\{\mathbf{n}_1, \dots, \mathbf{n}_k\}$, each of which experience a unitless vehicle specific scalar control input $\{u_1, \dots, u_k\}$. The forces and torques from these vertices are assumed to be generated in the direction of their normals. However note that the positions as well as normals are allowed to change during the simulation.

Figure 3 shows how a quadrotor can be depicted as a collection of four vertices. The control input u_i drives the rotational speed of the propellers located at the four vertices. We compute the forces and torques produced by propellers using [14]:

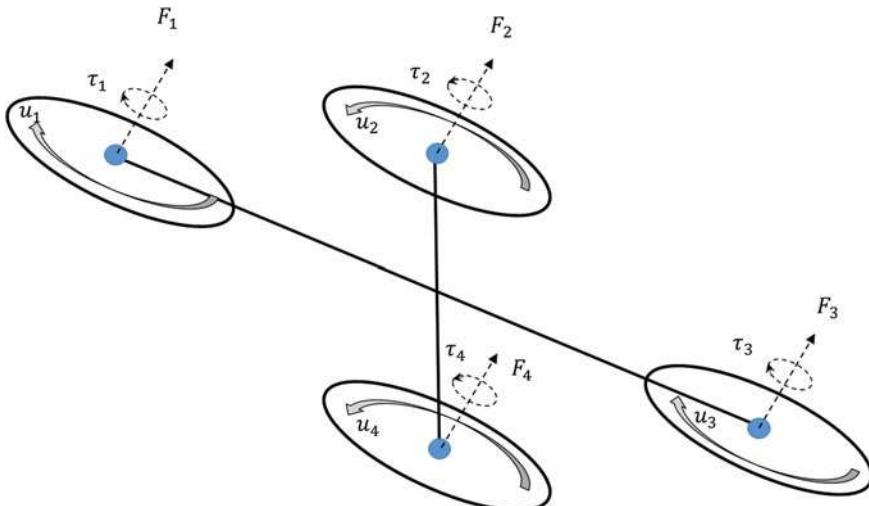


Fig. 3 Vehicle model for the quadrotor. The four blue vertices experience the controls u_1, \dots, u_4 , which in turn results in the forces $\mathbf{F}_1, \dots, \mathbf{F}_4$ and the torques τ_1, \dots, τ_4

$$\mathbf{F}_i = C_T \rho \omega_{max}^2 D^4 u_i \quad \text{and} \quad \tau_i = \frac{1}{2\pi} C_{pow} \rho \omega_{max}^2 D^5 u_i.$$

Here C_T and C_{pow} are the thrust and the power coefficients respectively and are based on the physical characteristics of the propeller, ρ is the air density, D is the propeller's diameter and ω_{max} is the max angular velocity in revolutions per minute. By allowing the movements of these vertices during the flight it is possible to simulate the vehicles with capabilities such as Vertical Take-Off and Landing (VTOL) and other recent quadrotors that change their configuration in flight.

The vehicle model abstract interface also provides a way to specify the cross sectional area in body frame that in turn can be used by physics engine to compute the linear and angular drag on the body.

3.2 Environment

The vehicle is exposed to various physical phenomena including gravity, air-density, air pressure and magnetic field. While it is possible to produce computationally expensive models of these phenomena that are very accurate, we focus our attention to models that are accurate enough to allow a real-time operation with hardware-in-the-loop. We describe these individual components of the environment below.

3.2.1 Gravity

While many models use a constant number to model the gravity, it varies in a complex manner as demonstrated by models such as GRACE [15]. For most ground based or low altitude vehicles these variations may not be important; however, it is fairly inexpensive to incorporate a more accurate model. Formally, we approximate the gravitational acceleration g at height h by applying binomial theorem on Newton's law of gravity and neglecting the higher powers:

$$g = g_0 \cdot \frac{R_e^2}{(R_e + h)^2} \approx g_0 \cdot \left(1 - 2 \frac{h}{R_e}\right).$$

Here R_e is Earth's radius and g_0 is the gravitational constant measured at the surface.

3.2.2 Magnetic Field

Accurately modeling the magnetic field of a complex body such as Earth is a computationally expensive task. The World Magnetic Model (WMM) model [16] by National Oceanic and Atmospheric Administration (NOAA) is one of the best known magnetic models of Earth. Unfortunately, the most recent model WMM2015 is fairly complex and computationally expensive for real-time applications.

We implemented the tilted dipole model where we assume Earth as a perfect dipole sphere [17, pp. 27–30]. This ignores all but the first order terms to derive magnetic field estimate using the spherical geometry. This model allows us to simulate variation of the magnetic field as we move in space as well as areas that are often problematic such as polar regions. Given a geographic latitude θ , longitude ϕ and altitude h (from surface of the earth), we first compute the magnetic co-latitude θ_m using:

$$\cos \theta_m = \cos \theta \cos \theta^0 + \sin \theta \sin \theta^0 \cos(\phi - \phi^0).$$

where θ^0 and ϕ^0 denote the latitude and longitude of the true magnetic north pole. Then, the total magnetic intensity $|B|$ is computed as:

$$|B| = B_0 \left(\frac{R_e}{R_e + h} \right)^3 \sqrt{1 + 3 \cos^2 \theta_m}$$

Here B_0 is the mean value of the magnetic field at the magnetic equator on the Earth's surface, θ_m is the magnetic co-latitude and R_e is the mean radius of the Earth. Next, we determine the inclination α and declination β angles using:

$$\tan \alpha = 2 \cot \theta_m \quad \text{and} \quad \sin \beta = \begin{cases} \sin(\phi - \phi^0) \frac{\cos \theta^0}{\sin \theta_m}, & \text{if } \cos \theta_m > \sin \theta^0 \sin \theta \\ \cos(\phi - \phi^0) \frac{\cos \theta^0}{\sin \theta_m}, & \text{otherwise.} \end{cases}$$

Finally, we can compute the horizontal field intensity (H), the latitudinal (X), the longitudinal (Y) and the vertical field (Z) components of the magnetic field vector as follows:

$$H = |B| \cos \alpha \quad Z = |B| \sin \alpha \quad X = H \cos \beta \quad Y = H \sin \beta.$$

3.2.3 Air Pressure and Density

The relationship between the altitude and the pressure of the Earth's atmosphere is complicated due to the presence of many distinct layers, each with its own individual properties. First we compute Standard Temperature T and Standard Pressure P using 1976 U.S. Standard Atmosphere model [18, Eqs. 1.16 and 1.17] for altitude below 51 km and switch to the model in [19, Table 4] beyond that up to 86 km. Then, the air density is $\rho = \frac{P}{R \cdot T}$ (where R is the specific gas constant.)

3.3 Physics Engine

The kinematic state of the body is expressed using 6 quantities: position, orientation, linear velocity, linear acceleration, angular velocity and angular acceleration.

The goal of the physics engine is to compute the next kinematic state for each body given the forces and torques acting on it. We strive for an efficient physics engine that can run its update loop at high frequency (1000 Hz) which is desirable for enabling real-time simulation scenarios such as high speed quadrotor control. Consequently, we implement a physics engine that avoids the extra complexities of a generic engine allowing us to tightly control the performance and make trade-offs that best meet our requirements.

3.3.1 Linear and Angular Drag

Since the vehicle moves in the presence of air, the linear and the angular drag has a significant effect on the dynamics of the body. The simulator computes the magnitude $|\mathbf{F}_d|$ of the linear drag force on the body according to the drag equation [20]:

$$|\mathbf{F}_d| = \frac{1}{2} \rho |\mathbf{v}|^2 C_{lin} A.$$

Here C_{lin} is the linear air drag coefficient, A is the vehicle cross-section and ρ is the air density. This drag force acts in the direction opposite to the velocity vector \mathbf{v} .

Computing the angular drag for arbitrary shape remains complex and computationally intensive task. Many existing physics engines use a small but often an arbitrary damping constant as a substitute for computing actual angular drag. We provide simple but better approximations to model the angular drag.

Consider an infinitesimal surface area ds in the extremity of the body experiencing the angular velocity ω . As the linear velocity $d\mathbf{v}$ experienced by ds is given by $\mathbf{r}_{ds} \times \omega$, we can now use the linear drag equation for ds [21, pp. 160–161]:

$$|\mathbf{dF}| = \frac{1}{2} \rho |\mathbf{r}_{ds} \times \omega|^2 C_{lin} ds, \text{ where direction of } \mathbf{dF} \text{ is } -\mathbf{r}_{ds} \times \omega.$$

Now, the drag torque is computed by integrating over the entire surface: $\tau_d = \int_S \mathbf{r}_{ds} \times \mathbf{dF}$. To simplify the implementation, we approximate the body of the vehicle as set of connected faces which further can be approximated as a rectangular box for the purpose of evaluating the integral.

3.3.2 Accelerations

In addition to the drag forces and torques, we also need to consider the forces \mathbf{F}_i and the torques τ_i present on the vehicle at the vertex located at \mathbf{r}_i relative to center of gravity (see Sect. 3.1). We thus compute the net force and torque as:

$$\mathbf{F}_{net} = \sum_i \mathbf{F}_i + \mathbf{F}_d \quad \text{and} \quad \tau_{net} = \sum_i [\tau_i + \mathbf{r}_i \times \mathbf{F}_i] + \tau_d.$$

We obtain the linear acceleration by applying Newton's second law and then adding gravity vector to compute the net acceleration, $\mathbf{a} = \mathbf{F}_{net}/m + \mathbf{g}$. The angular acceleration in body frame is given by Euler's rotation equation: $\alpha = I^{-1} \cdot (\tau_{net} - (\omega \times (I \cdot \omega)))$, where, I is the inertia tensor and ω is angular velocity, both in body frame.

3.3.3 Integration

We update the position \mathbf{p}_{k+1} of the body at time $k + 1$ by integrating the velocity and the initial position \mathbf{p}_0 . The first order integration algorithms such as Euler method diverges quickly with unbounded error although very simple to implement. In our implementation we use Velocity Verlet algorithm instead of Runge Kutta for its computationally inexpensiveness and stability while still being second order method [22]. Formally,

$$\mathbf{v}_{k+1} = \mathbf{v}_k + \frac{\mathbf{a}_k + \mathbf{a}_{k+1}}{2} \cdot dt \quad \mathbf{p}_{k+1} = \mathbf{p}_k + \mathbf{v}_k \cdot dt + \frac{1}{2} \cdot \mathbf{a}_k \cdot dt^2$$

The angular velocity is updated in similar manner as linear velocity however updating orientation isn't straight forward. One of the approach is to maintains the orientation as a rotation matrix that is updated every time step. However this causes a slow drift which must be corrected by orthonormalization at regular intervals which is expensive. Alternative approach is to maintain rotations as much more efficient quaternions which are also numerically stable and trivially normalizable. One of the problem, however, is that the orientation quaternion is maintained in the world frame while the angular velocity is maintained in the body frame in our framework. To update the orientation, we first compute the angle-axis pair $(\alpha_{dt}, \mathbf{u})$ where α_{dt} is the angle traversed around unit vector \mathbf{u} . We can compute the angle $\alpha_{dt} = |\omega| \cdot dt$ and axis by $\mathbf{u} = \omega / |\omega|$. This allows us to compute equivalent change in quaternion \mathbf{q}_{dt} representing the change in orientation in time dt . As noted before, \mathbf{q}_{dt} is in body frame while \mathbf{q}_k in world reference frame. The problem now remains that of adding \mathbf{q}_{dt} to \mathbf{q}_k to obtain \mathbf{q}_{k+1} which can be proven to given by relationship $\mathbf{q}_{k+1} = \mathbf{q}_k \cdot \mathbf{q}_{dt}$.

3.3.4 Collisions

Unreal engine offers a rich collision detection system optimized for different classes of collision meshes and we directly use this feature for our needs. We receive the impact position, impact normal and penetration depth for each collision that occurred during the render interval. Our physics engine uses this data to compute the collision response with Coulomb friction to modify both linear and angular kinematics [23].

3.4 Sensors

AirSim offers sensor models for accelerometer, gyroscope, barometer, magnetometer and GPS. All our sensor models are implemented as C++ header-only library and can be independently used outside of AirSim. Like other components, sensor models are expressed as abstract interfaces so it is easy to replace or add new sensors.

3.4.1 Barometer

To simulate barometer, we compute ground truth pressure using the detailed model of atmosphere (Sect. 3.2.3) and model the drift in the pressure measurement over time using Gaussian Markov process [24] for more realistic behavior in long flights. Formally, if we denote the current bias factor as b_k then the drift is modeled as:

$$b_{k+1} = w \cdot b_k + (1 - w) \cdot \eta, \text{ where: } w = e^{-\frac{dt}{\tau}} \text{ and } \eta \sim N(0, s^2).$$

Here τ , is the time constant for the process and set to 1 h in our model. η is a zero mean Gaussian noise with standard deviation that can be selected using the data available in [25]. This pressure p is then added with white noise drawn from zero mean Gaussian distribution with standard deviation set from datasheet of the sensor (such as MEAS MS56112). Finally we convert the pressure to altitude using barometric formula used by the sensor's driver:

$$h = \frac{T_0}{a} \left[\left(\frac{p}{p_0} \right)^{-\left(\frac{a \cdot R}{g}\right)} - 1 \right],$$

here T_0 is the reference temperature (15°C), $a = -6.5 \times 10^{-3}$ is the temperature gradient, g and R are the gravity and the specific gas constants, p_0 is the current sea level pressure and p is the measurement.

3.4.2 Gyroscope and Accelerometer

Gyroscope and accelerometers constitute the core of the inertial measurement unit (IMU) [26]. We model these by adding white noise and bias drift over time to the ground truth. For gyroscope, given the true angular velocity in body frame ω , we compute the measurement ω^{out} as,

$$\omega^{\text{out}} = \omega + \eta_a + b_t, \quad \text{where } \eta_a \sim N(0, r_a) \text{ and}$$

$$b_t = b_{t-1} + \eta_b, \quad \text{where } \eta_b \sim N\left(0, b_0 \sqrt{\frac{dt}{t_a}}\right).$$

Here parameters r_a , bias b_0 and the time constant for bias drift t_a can either be obtained from Allan variance plots or from datasheets. Accelerometer output is computed in the similar manner except that we must first subtract gravity from the true linear acceleration in the world frame and then convert the result to the body frame before we add bias drift and noise.

3.4.3 Magnetometer

We use the tilted dipole model for Earth's magnetic field Sect. 3.2.2, given the geographic coordinates to compute the components of the ground truth magnetic field in body frame and add the white noise as specified in the datasheet.

3.4.4 Global Positioning System (GPS)

Our GPS model simulates latency (typically 200 ms), slower update rates (typically 50 Hz) and horizontal and vertical position error estimate decay rates to simulate gaining fix over time. The decay rate is modeled using first order low pass filter individually parameterized for horizontal and vertical fix.

3.5 Visual Rendering

Since advanced rendering and detailed environments have been a key requirement for AirSim we chose Unreal Engine 4 (UE4) [12] as our rendering platform. UE4 offers several features that made it an attractive choice including it being an open source and available on Linux, Windows as well as OSX. UE4 brings some of the cutting edge graphics features such as physically based materials, photometric lights, planar reflections, ray traced distance field shadows, lit translucency etc. Figure 1 shows a screen-shot from AirSim which highlight near photo-realistic rendering capabilities. Further, Unreal's large online Marketplace has various pre-made elaborate environments, many of which are created using photogrammetry techniques.

4 Experiments

We perform experiments primarily to evaluate how close the flight characteristic of a quadrotor flying in real-world is to that of a simulation of the same vehicle in AirSim. We also evaluate some of our sensor models against the real-world sensors.

Hardware Platform: Real-world flights were performed with the Pixhawk v2 flight controller mounted on a Flamewheel quadrotor frame, together with a Gigabyte

5500 Brix running Ubuntu 16.04. The sensor measurements were recorded on the Pixhawk device itself. We configured the simulated quadrotor in AirSim using the measured physical parameters and simulated sensor models configured using sensor data sheets. The AirSim MavLinkTest application was used to perform repeatable offboard control for both the real-world and the simulated flights.

Trajectory Evaluation: We fly the quadrotor in the simulator in two different patterns: (1) trajectory in square shape with each side being 5 m long (2) trajectory in circle shape with radius being 10 m long. We then use exact same commands to fly the real vehicle. For both the simulation and the real-world flights, we collect location of the vehicle in local NED coordinates along with timestamps.

Figures 4c and b shows the time series of locations in simulated flight and the real flight. Here, the horizontal axis represents the time and the vertical axis represent the off-set in X and Y directions. We also compute the symmetric Hausdorff distance between the real-world track and the track in simulation. We found that the simulation and real-world tracks were fairly close both for the circle (Hausdorff distance between simulated and real-world: 1.47 m) as well as the square (Hausdorff distance between simulated and real-world: 0.65 m).

We also present visual comparison for this experiment for the circle and the square patterns in Fig. 4a and d respectively. The simulated trajectory is shown with a purple line while the real trajectory is shown with a red line. We can observe that qualitatively the trajectories tracked by both the real-world and the simulated vehicle are close. The small differences may have been caused by various factors such as integration errors, vehicle model approximations and mild random winds.

Sensor Models: Besides evaluating the entire simulation pipeline we also investigated individual component models, namely the barometer (MEAS MS5611-01BA), the magnetometer (Honeywell HMC5883) and the IMU (InvenSense MPU 6000). Note that the simulated GPS model is currently simplistic, thus, we only focus on the three more complex sensor models. For each of the above sensors we use the manufacturer specified datasheets to set the parameters in the sensor models.

- **IMU:** We measured readings from the accelerometers and gyroscope as the vehicle was stationary and flying. We observed that while the characteristics were similar when the vehicle was stationary (gyro: simulated variance $2.47e - 7 \text{ rad}^2/\text{s}^2$, real-world variance $6.71e - 7 \text{ rad}^2/\text{s}^2$, accel.: simulated variance $1.78e - 4 \text{ m}^2/\text{s}^4$, real-world variance $1.93e - 4 \text{ m}^2/\text{s}^4$), the observed variance for an in-flight vehicle was much higher than the simulated one (accel.: simulated $1.75e - 3 \text{ m}^2/\text{s}^4$ vs. real-world $9.46 \text{ m}^2/\text{s}^4$). This is likely in real-world the airframe vibrates when the motors are running and that phenomenon is not yet modeled in AirSim.
- **Barometer:** We raised the sensor periodically between two fixed heights: ground level and then elevated to 178 cm (both in simulation and real-world). Figure 5a shows both the measurements (green is simulated, blue is real-world) and we observe that the signals have similar characteristics. Note that the offset between the simulated and the real-world pressure is due the difference in absolute pressure in the real-world and the one in the simulation. There is also a small increase in

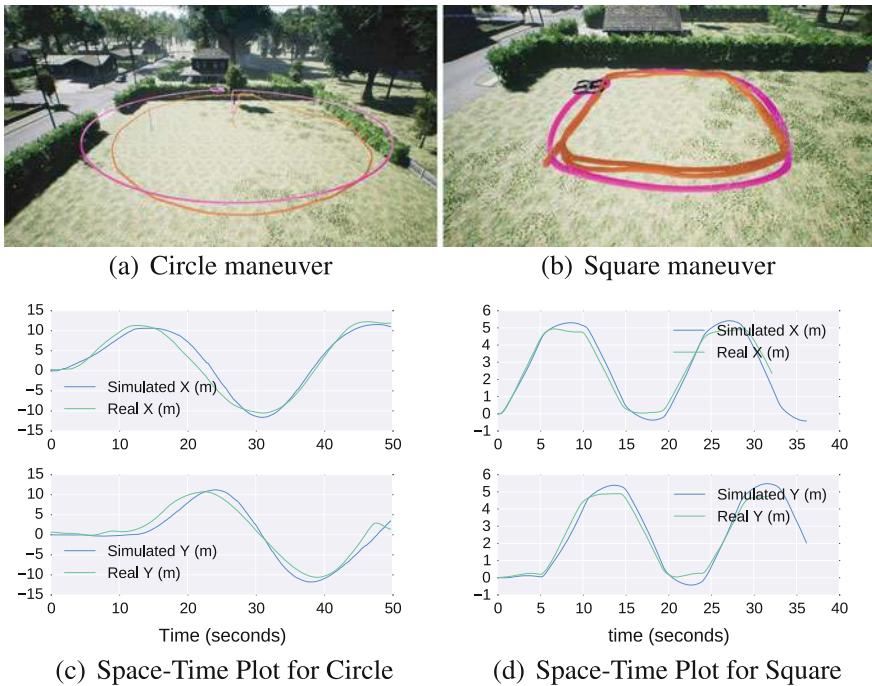


Fig. 4 Evaluating the differences between the simulated and the real-world flight. In top figures, the purple and the red lines depict the track from simulation and the real-world flights respectively

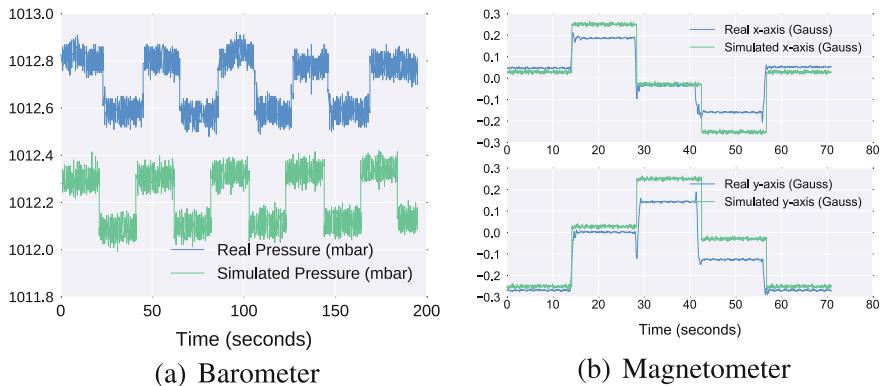


Fig. 5 Figure 5a and b show that barometer and the magnetometer characteristics in simulation closely resemble that of the real world

the middle due to a temperature increase, which wasn't simulated. Overall, the characteristics of the simulated sensor matches well to the real sensor.

- **Magnetometer:** We placed the vehicle on the ground and then rotated it by 90° four times. Figure 5b shows the real-world and the simulated measurements and highlight that they are very similar in characteristic.

5 Conclusion and Future Work

AirSim offers hi-fidelity physical and visual simulation that allows to generate large quantity of training data cheaply for building machine learning models. AirSim API design allows developing algorithms against simulator and then deploy them without change on real vehicles. The core components of AirSim including physics engine, vehicle models, environment models and sensor models are designed to be independently usable with minimal dependencies outside of AirSim and are easily extensible. AirSim is inspired by the goal of developing reinforcement learning algorithms for the autonomous agents that can operate in the real world.

The task of mimicking the real-world in *real-time simulation* is a challenging endeavor. There are a number of things that can be improved. Currently we do not simulate richer collision response or advanced ground interaction models which may be possible in future by implementing our physics engine interface for NVIDIA PhysX and utilizing features such as physics sub-stepping. Also we do not simulate various oddities in camera sensors except those directly available in Unreal engine. We plan to add advanced noise models and lens models in future. The degradation of GPS signal due to obstacles is not simulated yet which we plan to add using ray tracing methods. We also plan to add more advanced wind effects and thermal simulations for fixed wing vehicles. Our extensibility APIs have been designed with above future work in mind and can also be used to realize other vehicle types.

References

1. Kober, J., Bagnell, J.A., Peters, J.: Reinforcement learning in robotics: a survey. *Int. J. Rob. Res.* **32**(11), 1238–1274 (2013). <https://doi.org/10.1177/0278364913495721>
2. Bagnell, J.A.: An invitation to imitation. Technical Report, CMU ROBOTICS INST (2015)
3. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. *J. Big Data* **3**(1), 9 (2016). <https://doi.org/10.1186/s40537-016-0043-6>
4. Shah, S., Dey, D., Lovett, C., Kapoor, A.: Airsim open source platform at github. <https://github.com/Microsoft/AirSim> (2017)
5. Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: IROS (2004)
6. Meyer, J., Sendobry, A., Kohlbrecher, S., Klingauf, U., Von Stryk, O.: Comprehensive simulation of quadrotor uavs using ros and gazebo. In: SIMPAR, pp. 400–411. Springer (2012)
7. Furrer, F., Burri, M., Achtelik, M., Siegwart, R.: Rotorsa modular gazebo mav simulator framework. In: Robot Operating System (ROS), pp. 595–625. Springer (2016)

8. Babushkin, A.: Jmavsim. <https://pixhawk.org/dev/hil/jmavsim>
9. Meier, L., Tanskanen, P., Fraundorfer, F., Pollefeyns, M.: Pixhawk: A system for autonomous flight using onboard computer vision. In: ICRA, pp. 2992–2997. IEEE (2011)
10. Jackson, J., Ellingson, G., McLain, T.: Rosflight: a lightweight, inexpensive mav research and development tool. In: ICUAS, pp. 758–762 (2016). <https://doi.org/10.1109/ICUAS.2016.7502584>
11. Hackflight: Simple quadcopter flight control firmware and simulator for c++ hackers. <https://github.com/simondlevy/hackflight>
12. Karis, B., Games, E.: Real shading in unreal engine 4. In: Proceedings Physically Based Shading Theory Practice (2013)
13. Moore, H.: Creating assets for the open world demo (2015)
14. Brandt, J., Deters, R., Ananda, G., Selig, M.: Uiuc propeller database, university of illinois at urbana-champaign. <http://m-selig.ae.illinois.edu/props/propDB.html> (2015)
15. Tapley, B., Ries, J., Bettadpur, S., Chambers, D., Cheng, M., Condi, F., Poole, S.: The ggm03 mean earth gravity model from grace. In: American Geophysical Union, G42A-03 (2007)
16. Chulliat, A., Macmillan, S., Alken, P., Beggan, C., Nair, M., Hamilton, B., Woods, A., Ridley, V., Maus, S., Thomson, A.: The US/UK world magnetic model for 2015–2020 (2015). <https://doi.org/10.7289/V5TB14V7>
17. Lanza, R., Meloni, A.: The Earth's Magnetism: An Introduction for Geologists. Springer Science & Business Media (2006)
18. Stull, R.: Practical Meteorology: An Algebra-based Survey of Atmospheric Science. University of British Columbia (2015)
19. Braeunig, R.: Atmospheric models. <http://www.braeunig.us/space/atmmodel.htm> (2014)
20. Taylor, J.: Classical Mechanics. University Science Books (2005)
21. Nakayama, Y., Boucher, R.: Introduction to Fluid Mechanics. Butterworth-Heinemann (1998)
22. Herman, R.: A first course in differential equations for scientists and engineers. <http://people.uncw.edu/hermannr/mat361/ODEBook/> (2017)
23. Hecker, C.: Physics, Part 3: Collision Response. Game Developer Magazine (1997)
24. Sabatini, A.M., Genovese, V.: A stochastic approach to noise modeling for barometric altimeters. Sensors (Basel, Switzerland) **13**(11), 15,692–15,707 (2013)
25. Burch D., B.T.: Mariner's Pressure Atlas: Worldwide Mean Sea Level Pressures and Standard Deviations for Weather Analysis. Starpath School of Navigation (2014)
26. Woodman, O.J.: An introduction to inertial navigation. Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory (2007)

Design and Development of Tether-Powered Multirotor Micro Unmanned Aerial Vehicle System for Remote-Controlled Construction Machine

Seiga Kiribayashi, Kaede Yakushigawa and Keiji Nagatani

Abstract In Japan, several types of natural disasters such as floods, earthquakes, and volcanic eruptions have occurred and will likely occur in the future. Therefore, civil engineering works are required for restoration after such natural disasters, and teleoperated construction machines have been developed to facilitate such works. During the operation of teleoperated construction machines, images from various viewpoints e.g., an image from the perspective of the machines or that from the side of the bucket is essential for carrying out tasks efficiently. However, in the case of the initial response to natural disasters, it is difficult to use dedicated, conventional camera-equipped vehicles and fixed cameras on external towers to obtain such perspective images, particularly within a month after the disaster. Therefore, in this research, we propose a tether-powered multirotor micro unmanned aerial vehicle (MUAV) system to obtain images from various perspectives for the operator of a teleoperated construction machine. The features of the proposed system are (1) high voltage for transmitting electric power through thin tether, (2) tension control of the tether in vibration and inclined conditions, and (3) wired VDSL communication between the MUAV and the helipad. In this paper, we introduce the design and implementation of the proposed system. In addition, we report the results of the field test of the tethered MUAV mounted on a construction machine.

1 Introduction

In Japan, several natural disasters have occurred and will likely occur even in the future. In the case of large-scale catastrophes caused by earthquakes, heavy rain, or volcanic eruptions, civil-engineering work with construction machinery is required for restoration. However, the restoration work at a disaster site is quite dangerous for operator, and a secondary disaster may occur at the site. Therefore, to ensure the safety of the operators, unmanned construction technology has been developed and used in practical situations in Japan. As a representative example, the Tele-earthwork system

S. Kiribayashi (✉) · K. Yakushigawa · K. Nagatani
Tohoku University, 6-6-10, Aramaki-Aoba, Sendai, Japan
e-mail: seiga@frl.mech.tohoku.ac.jp

[1] developed teleoperated construction machinery to excavate volcanic products and to construct mud-control dams after the eruptions of Mt. Unzen-Fugen that occurred in the 1990s. Such a system was also used for restoration work in the 2011 Great East Japan Earthquake [2].

Although this system worked effectively in long-term restoration work, it is difficult to make use of such dedicated machines in the initial response of natural disasters, particularly within 1 month after the disaster. This is because (1) teleoperated construction machines require much preparation before operation, (2) the number of teleoperated construction machines is small, and (3) it takes a considerable amount of time to transport the machines to the disaster site. A teleoperation technology for “general” construction machinery is required for the initial response to disasters.

As general construction machines are used for various constructions in several places in Japan, their transportation is easy, and the number of machines is large. However, teleoperation cannot be performed with general construction machines. Therefore, much research and developments have been carried out to mount hardware on the cockpit of a general-construction machine and enable its remote control [3–5]. With the help of the above technologies, operators can control a general construction machine by teleoperation. However, they typically use on-vehicle cameras only. It is well-known that visual information obtained from various viewpoints is effective for the teleoperation of construction machines, e.g., the image of a bird view of the machines or that from the side of the bucket. Therefore, in restoration works at

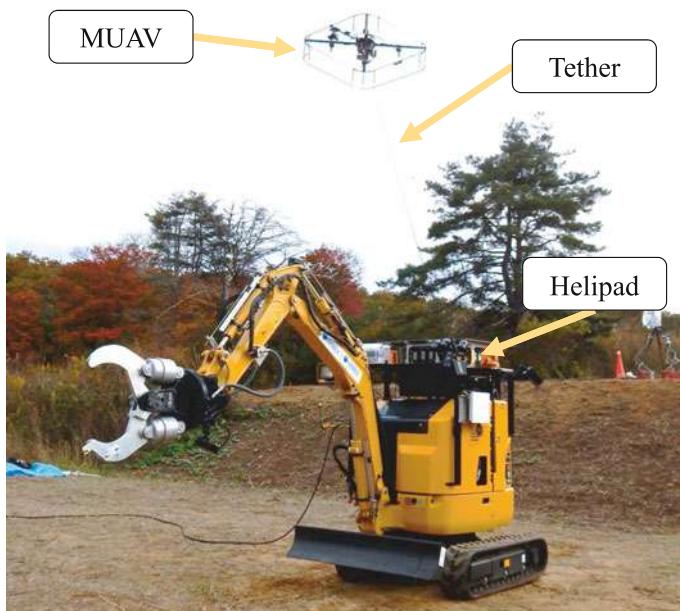


Fig. 1 Developed system on a teleoperated small construction machine

Mt. Unzen-Fugen, dedicated camera-equipped vehicles and fixed cameras on external towers were used to obtain images from various viewpoints. However, it is difficult to install them for the purpose of an initial response within a month of the occurrence of a natural disaster. In addition, there are limitations to the arrangement of the cameras.

As a solution to the aforementioned issues, we have been researching and developing a system that use a multirotor micro unmanned aerial vehicle (MUAV) as an external camera carrier to obtain images from various viewpoints. The system is consisting of a tether-powered multirotor MUAV and a helipad that have a tension controllable winch to windup a tether.

By installing the tether-powered multirotor MUAV system on a general-construction machine and mounting conventional teleoperation hardware on the cockpit, it is possible to realize an instant teleoperated construction machine that can obtain images from various viewpoints by itself. Figure 1 shows our recent version of the tether-powered multirotor MUAV and helipad on a construction machine.

In this paper, we present our research on and development of the proposed system and report the results of our field tests on the actual construction machine.

2 Tether Powered Multirotor MUAV System

2.1 *Proposal*

The most important information for the teleoperation of a construction machine is visual information. In teleoperated restoration works at Mt. Unzen-Fugen, the operator uses not only images from the perspective of the cockpit, but also images from various perspectives obtained from dedicated camera-equipped vehicles and fixed cameras on external towers [1]. The images from the latter can compensate for the blind spot in the cockpit view, and it enables the operators to directly grasp the motion of the construction machine. In particular, the images are used to grasp a details of the distance between the operating device (e.g., bucket) and the target (e.g., volcanic rocks).

For developing images from various viewpoints, Sato et al. developed a system with multiple on-vehicle fisheye cameras [6]. It is effective for an initial response because it does not require any external camera installation in the target environment. Nevertheless, blind spots exist on the image obtained from the construction machine. Moreover, a typical construction machine has an arm and projections that cause blind spots.

Therefore, in this research, we aimed to develop a system to use a multirotor MUAV as an external camera carrier to obtain images from various viewpoints. It is now matured technologies to obtain images by MUAV from the air. However, there still exist some problems with this system, and one of these problems is its short flight time. Therefore, we have been researching and developing a system for the

use of a tether-powered multirotor MUAV. The following are the advantages of the proposed MUAV:

1. Within the tether length, the multirotor MUAV can stay at arbitrary positions to obtain alternate viewpoints without being affected by the rough terrain.
2. The flight time of the tether-powered MUAV is much longer than that for a typical MUAV because a power-feeding tether is used to supply electric power from the helipad.
3. In case the multirotor MUAV is out of control, it flies only within the range of its tether length. Thus, the damage to the environment can be minimized.
4. It helps secure pinpoint landings of the multirotor MUAV by forcefully winding the tether.

There are some researches on tether-powered multirotor MUAVs [7, 8], and some have been recently used in practical applications [9]. However, the flight range of the tethered MUAV has a limitation because of its tether length. In order to allow an external camera carrier to obtain images from various viewpoints, we have developed a tethered MUAV helipad that can be mounted on a construction machine. The proposed system is effective for an initial response for disaster because the system enables long-time flight, and it requires no external camera installation to teleoperate a construction machine in the target environment.

There exist some problems in the realization of the system, and one of these problems is large vibrations in the construction machines. Therefore, we organized the problem and implemented countermeasures as described in the next subsection.

2.2 Challenges of the System

The tether-powered multirotor MUAV system has three major challenges to install it to the construction machine as we proposed.

The first challenge is the flight range. A typical objective of tether-powered multirotor MUAVs in general use is a fixed-point observation, and the MUAV is not required to move dynamically. However, for teleoperation of the construction machines, the viewpoint of the MUAV should be changed based on the task. For example, to take an image of an excavator, the MUAV had better locate at directly above of the machine for its navigation, or the MUAV had better locate at the side of the bucket when it excavates the ground. In addition, when the MUAV fly horizontal, the power-feeding tether may slack and come into contact with the environment. The greater the tension in the tether, the lesser is the looseness in the tether becomes. However, the thrust required for the MUAV's flight increases, and the controllability of the MUAV decreases. Therefore, appropriate tension control of the tether is required.

The second challenge is a power source. A typical power-feeding tether system uses an AC power supply from the ground station. However, in the proposed system, an independent power source is required to fly the tether-powered multirotor MUAV

because the helipad that works as typical ground station is mounted on the construction machine. In addition, a lightweight, thin tether is required to reduce the load on the MUAV. In this case, it requires a high-voltage supply, but the MUAV requires a voltage step-down circuit if use a high-voltage, which implies an additional payload to drive propellers' motors. Considering the above trade-off, it is necessary to select an appropriate power supply method.

The third challenge is the vibration and inclination of the construction machine. As the power for a general construction machine is obtained from an engine, the construction machine vibrates, which also affects the equipment installed on the machine. Furthermore, the target environment is natural uneven terrain. Thus, the vibration and inclination caused by the machine navigation on such rough terrain affect the equipment of the machine. Therefore, it requires a system that is robust against vibration and inclination.

In keeping with the above challenges, we consider that the helipad has several development factors. Therefore, in the first stage of our development, we developed a novel helipad for a tether-powered multirotor MUAV that can work on construction machine. We used a conventional airframe as the multirotor MUAV, and controlled it manually under direct visual.

3 Development of the System

The tether-powered multirotor MUAV is controlled manually based on the flight controller inside the MUAV and the radio control transmitter. The tension of the tether is controlled independently using a winch mounted on the helipad. The operators control both the systems and camera gimbals mounted on the MUAV. Figure 2 is a block diagram of the system. In this paper, we do not describe in detail the camera gimbals' system. In the following subsections, we introduce the "power system", "winch with controllable tether tension", and "wireless communication".

3.1 Power System

A multirotor MUAV requires an electric power source. For example, the typical MUAV developed in this study (quad-rotor MUAV with 15-in. propellers, of approximately 3.0 kg) requires 400 W for hovering, and 800 W for moving or dealing with a disturbance, in our experience. The required electric power mainly depends on the weight and the rotors' diameter. Therefore, even if the MUAV itself is improved, it will not be possible to reduce the power consumption drastically. When such a large electric power is used, the power loss in the power-feeding tether is critical. Based on Ohm's law, the loss due to the electric resistance in the power-feeding tether is proportional to the square of the current. Therefore, to obtain the same electric

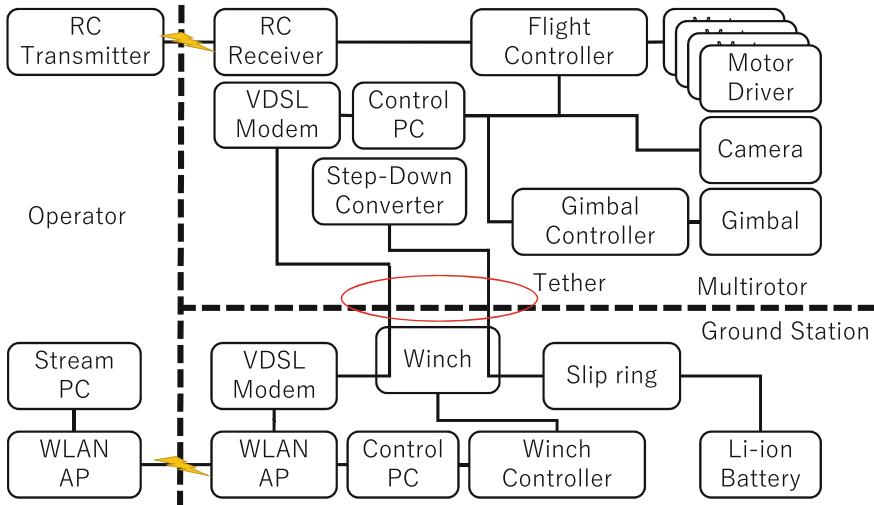


Fig. 2 Block diagram of the tether powered multirotor MUAV system

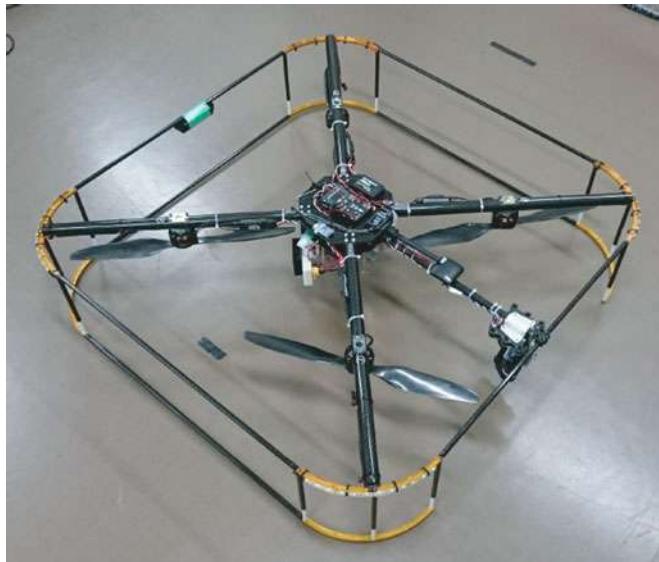
power, a high-voltage and low-current system is required to be configured for the power supplied through the tether.

On the multirotor MUAV side, we use a voltage step-down converter that allows an input between DC200 V and 420 V. A continuous output of 600 W is possible for one converter. Therefore, we use two converters in parallel to obtain an output of 1200 W. The converter was selected by the considerations of the weight and availability. The voltage step-down converter is installed on the side opposite to the camera gimbals to balance the center of gravity, and the converter is cooled by a downstream flow from the rotors. Figure 3a shows an appearance of the MUAV, Fig. 3b shows the setup of the voltage step-down converter on the MUAV, and Figure 3c shows the converter itself.

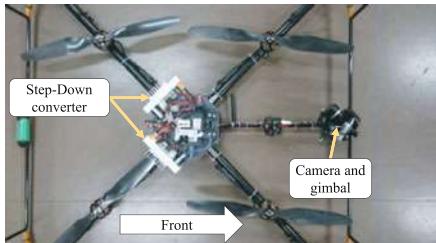
The weight of each module is 160 g, and the total weight of the two converters is lighter than the weight of the batteries normally used for its flight. The weight of the MUAV before installing any equipment is about 2.5 kg, and after installed, it is about 3.2 kg. The payload capacity of the MUAV is over 3 kg, thus the MUAV have more than 1 kg payload even consider about tension and weight of a tether.

Next, we describe the power system in the helipad side. Typically, a commercial power source was used to handle a large power consumption in previous researches. However, in a disaster environment, it is impossible to use such a commercial power source. Therefore, a small-sized power source that has a sufficient power capacity is required to be mounted on the construction machine. In addition, the power source is required to be used in the vibration and inclination conditions.

There were two realistic candidates for the supply of 800 W of electricity: the battery type and gas-powered generator type power supply. In the case of the gas-powered generator type, it is difficult to estimate the remaining working time and to



(a) Multirotor MUAV with the voltage step-down converter.



(b) Bottom side view of the multirotor MUAV.



(c) Voltage step-down converter

Fig. 3 Developed tether powered multirotor MUAV

use it in the inclined condition. Therefore, we chose the battery type power supply in our implementation.

In the case of the battery type, there are two methods to obtain a high voltage to tether power feeding: (1) boosting the voltage from a battery with a voltage converter and (2) using a series connection of batteries. In the former case, it is possible to increase the choices of batteries. However, conversion loss occurs in the converter. In the latter case, there is no conversion loss, but the total voltage fluctuates greatly depending on the remaining battery power. As mentioned above, the chosen voltage step-down converter installed on the MUAV allows a wide-range input of between DC200 V and 420 V. Therefore, we chose the latter method.

According to the required conditions—low weight and possibility of use in the inclined condition—we did not choose lead batteries but lithium batteries. Specifically, lithium ion battery packs (23.1 V, 127 Wh, 62KSP545483-2, Hitachi Maxell, Ltd.)

are chosen. This includes the circuit used for the estimation of the remaining battery level. In this research, we connected twelve packs in serial and use as a large battery unit (277.2 V, 1,524 Wh). It enabled over 3 h of operation, i.e., hovering, for our multirotor MUAV.

3.2 Winch with Controllable Tether Tension

To enable the appropriate control of the tether tension, we developed a winch with a controllable tether tension located on the helipad.

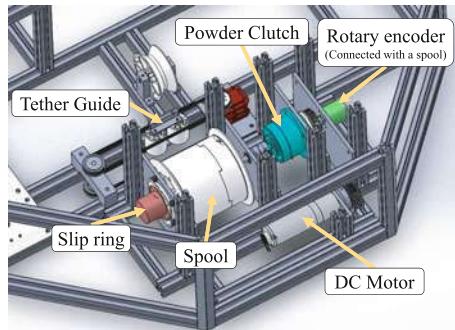
When both endpoints of a string-like object are fixed at any two points, the object forms a catenary curve. Our tether is sufficiently thin (approximately 5 mm outer diameter) and soft, such that the tether also shapes a catenary curve. When the lower side of the tether is fixed at the helipad, and the upper side of the tether is connected to the multirotor MUAV, the angle formed by the tether and helipad is expressed as a function of the tether tension. Therefore, the helipad can control the shape of the tether by controlling the tether tension, and it can avoid the contact of the tether with the surrounding objects.

The general tension control uses feedback control based on a tension measurement. The tension measurement is typically conducted by measuring the displacement of a movable pulley to which a spring is connected. However, when the acceleration is applied to the measurement device, it measures the total of the tether tension and acceleration. Furthermore, when the helipad is in an inclined condition, gravity acceleration is affected, and a measurement error occurs. Therefore, it is difficult to apply the typical feedback control method to our system.

To solve the above problem, we chose to use a powder clutch that can specify the arbitrary torque with the open loop control, instead of a tension measurement of the tether. Figure 4a shows a CAD model of our winch that includes a powder



(a) Developed Helipad.



(b) CAD image of the winch.

Fig. 4 Winch with controllable tether tension on helipad

clutch. The powder clutch uses magnetic powder, and it transmits torque from the motor to the spool according to the current. Once the torque control of the winch is realized, the tension of the tether is calculated based on the spool torque and spool radius. To estimate the tether tension accurately, an estimation of the spool radius, which changes according to the extended tether length, is very important. Therefore, in this research, we developed a mechanism to wind up the tether densely to estimate the winding position of the tether. With this mechanism, the helipad can accurately generate an arbitrary tension in the tether at any time, even under the conditions of vibration and inclination.

3.3 Communication System

In our system, control PCs are located at both the MUAV and helipad, and both the PCs should communicate with each other. On the other hand, it is necessary to establish a wireless communication between the operation room and the construction machine for teleoperation. To secure the wireless bandwidth, it was decided that the communication between the MUAV and the helipad should be wired.

The weight of the wires for the communication significantly affects the payload of the MUAV, and therefore, we chose a VDSL communication system that can be realized with only two metal lines. We mounted the VDSL modems on both the multirotor MUAV and the helipad and enabled a mutual conversion of the VDSL and Ethernet. The control signals for the flight and camera gimbals are sent from the helipad to the MUAV through the VDSL communication.

All the control signals are gathered in the control PC on the helipad, and the PC communicates with the operator's PC on the wireless LAN. Based on the proposed communication system, various external communication devices can be used, and their integration with other systems on the construction machine becomes easy.

4 Field Test

In November 2016, we integrated our system on the teleoperated construction machine developed in the Impulsing Paradigm Change through Disruptive Technologies Program (ImPACT) and conducted an operation verification test of the proposed system. The objective of the test of the ImPACT Program was to confirm the efficient work made possible using a teleoperated construction machine in the case of a disaster situation, and the objective of the test of us was to provide images from various viewpoints to the operators by our proposed system. In this test, tether length is limited to 12 m for safety reasons. Also, our developed system fly over an hour continuously and battery capacity remains over 60% after the test.



Fig. 5 Test flight overview images in case of MUAV's middle flight altitude

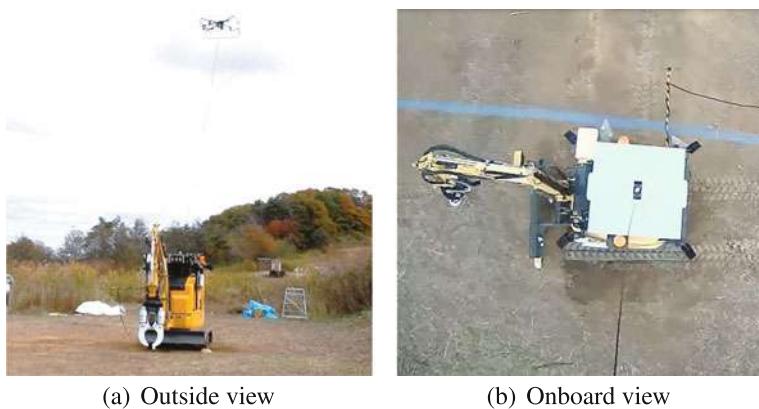


Fig. 6 Test flight overview images in case of MUAV's high flight altitude

Figures 5, 6 and 7 shows the photographs of the test. The photographs in the left column are taken from the outside, and the photographs in the right column show the images obtained from the multirotor MUAV. Figure 6 show the outside view and obtained image from the high flight altitude of the MUAV. Figure 7 show the outside view and obtained image from the low flight altitude and long horizontal length. In the last case, the tether is not slackened or tightened too much because of the appropriate tension control by the winch.

The above results showed that the developed system worked as expected without the occurrence of any problems.



Fig. 7 Test flight overview images in case of MUAV's low flight altitude and long horizontal distance from helipad

5 Conclusion

In this paper, we introduced the design and implementation of a tether-powered multirotor MUAV system to obtain images from various viewpoints for the teleoperation of a construction machine. The features of the system are as follows:

1. high voltage battery is used to transmit electric power through thin cables,
2. a powder clutch is used for the winch to enable tension control in the vibration and inclined conditions, and
3. a VDSL communication system is used for communication between the multirotor MUAV and the helipad.

Finally, we conducted an operation verification test of the proposed system to provide images from various viewpoints to the operator.

In future works, we intend to evaluate this system in detail and develop a relative positioning method for the MUAV based on the tether information to realize autonomous MUAV flights.

Acknowledgements This research was conducted as part of the Impulsing PAradigm Change through Disruptive Technologies Program (ImPACT) led by the Council for Science, Technology, and Innovation.

References

1. Minamoto, M., Nakayama, K., Aokage, H., Sako, S.: Development of a tele-earthwork system. In: Proceedings of Automation and Robotics in Construction XI, pp. 269–275 (1994)
2. Eiji, E., Kensuke, K., Masaharu, I., Takayuki, E.: Use of construction machinery in earthquake recovery work. Hitachi Rev. **62**(2) (2013)
3. 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: IEEE International Conference on Humanoid Robots, pp. 1–6 (2013)

- ence on Robotics and Automation, 2002. Proceedings. ICRA'02, vol. 3, pp. 2246–2252. IEEE (2002)
4. Yokoi, K., Nakashima, K., Kobayashi, M., Mihune, H., Hasunuma, H., Yanagihara, Y., Ueno, T., Gokyuu, T., Endou, K.: A tele-operated humanoid robot drives a backhoe in the open air. In: 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003. IROS 2003. Proceedings, vol. 2, pp. 1117–1122. IEEE (2003)
 5. Sasaki, T., Kawashima, K.: Remote control of backhoe at construction site with a pneumatic robot system. *Autom. Constr.* **17**(8), 907–914 (2008)
 6. Sato, T., Moro, A., Sugahara, A., Tasaki, T., Yamashita, A., Asama, H.: Spatio-temporal bird's-eye view images using multiple fish-eye cameras. In: 2013 IEEE/SICE International Symposium on System Integration (SII), pp. 753–758. IEEE (2013)
 7. Choi, S.Y., Choi, B.H., Jeong, S.Y., Gu, B.W., Yoo, S.J., Rim, C.T.: Tethered aerial robots using contactless power systems for extended mission time and range. In: 2014 IEEE Energy Conversion Congress and Exposition (ECCE), pp. 912–916. IEEE (2014)
 8. Papachristos, C., Tzes, A.: The power-tethered UAV-UGV team: a collaborative strategy for navigation in partially-mapped environments. In: 2014 22nd Mediterranean Conference of Control and Automation (MED), pp. 1153–1158. IEEE (2014)
 9. CyPhy. Parc. <http://cyphyworks.com/parc/>

Human-Robot Teaming: Concepts and Components for Design

**Lanssie Mingyue Ma, Terrence Fong,
Mark J. Micire, Yun Kyung Kim and Karen Feigh**

Abstract In the past, robots were used primarily as “tools for humans.” As robotics technology has advanced, however, robots have increasingly become capable of assisting humans as partners, or peers, working together to accomplish joint work. This new relationship creates a new host of interdependencies and teamwork questions that need to be addressed in order for human-robot teams to be effective. In this paper, we define communication, coordination, and collaboration as the cornerstones for human-robot teamwork. We then describe the components of teaming, including agent abilities, taskwork, metrics, and peer-to-peer interactions. Our purpose is to enable system designers to understand the factors that influence teamwork and how to structure human-robot teams to facilitate effective teaming.

1 Introduction

The role of robots in human-robot (HR) teams has begun to shift from an extension tool to a peer-like teammate that is able to assist with and complete joint tasks [8, 13]. Human-robot teams are groupings of humans and robotic systems who communicate, coordinate and collaborate together to perform a joint activity [8, 14]. The role of robots as teammates in human teams will allow for more collaboration and performance on joint activities. For example, robots in space exploration environments are

L. Mingyue Ma (✉) · K. Feigh

Georgia Institute of Technology, North Ave NW, Atlanta, GA 30332, USA
e-mail: lanssie.ma@gatech.edu

K. Feigh

e-mail: karen.feigh@gatech.edu

T. Fong · M. J. Micire · Y. K. Kim

IRG NASA Ames, Moffett Blvd, Mountain View, CA 94035, USA
e-mail: terry.fong@nasa.gov

M. J. Micire

e-mail: mark.j.micire@nasa.gov

Y. K. Kim

e-mail: yunkyoung.kim@nasa.gov

placed in a variety of applications where humans cannot operate alone and require assistance [2, 13, 14]. Research geared towards studying the effect this newer perspective has on teaming design and approaches in Human-Robot Interaction (HRI) will be beneficial.

As NASA moves toward future deep space missions, taskwork will be given to human-robot teams for mission safety and success [13, 14]. Longer space missions are more dangerous to humans as astronauts must operate autonomously from ground. Astronauts are limited by time and physical constraints and will need to rely on robot systems to assist in joint tasks that require both agents to be involved. To effectively complete joint activities, however, requires an understanding of how interdependencies between team members affect the execution of tasks. Creating and structuring an effective team and assigning work to each agent in the team is imperative to mission success and proper task execution [31].

Developing HRI by focusing on team performance for joint activities is imperative to developing better systems. Systems must evaluate the changing capabilities of each team member and their distinct roles in the team composition. While previous research and surveys of HRI describe different methodologies on developing human-robot systems, there is still a lack of convergence on HR teaming design.

This paper surveys various ways to build and construct HR teams. The goal of this survey is to provide system designers better insight into HR teaming, particularly regarding methods that can be used for HR team design. We first elaborate on the background of teams and prior research in HR teaming. Next, we describe how designers should consider taskwork, agent abilities, design metrics, and dependencies when composing HR teams. Lastly, we discuss open issues in HR teaming that warrant additional research.

2 Motivation

The central concept behind this paper is the concept of a team—interdependent members who share a common goal, have common ground, and trust in between them [13–15]. Teams are structurally organized include members who have their expertise and background; each member brings their own skills and background to the team. Teamwork is a fluid, context-dependent activity. Teamwork is composed of a variety of factors; the combination and effective of which can greatly affect the structure of teamwork [10]. High-performing teams operate with dynamic skillsets, including anticipation and prediction, to handle more complicated scenarios and workflow [24, 28]. While teamwork might seem to be an obvious detail in HRI design, system's designers must consider a variety of factors that affect the structure of teamwork from team compositions to the resulting dependencies.

Neither HR team design nor effective ways of measuring its success have been well defined in the literature. Prior research in developing human-robot teams has been widely vetted from human factors perspective [30]. Moreover, there has been little to no translation of HR teaming theory into real-world application. To close

the gap between theory and practice, system designers need to develop effective teamwork designs, methods, and protocols.

Creating effective HR teams is challenging because robotic capabilities are continually advancing—leading to better physical abilities, cognition, and awareness. Despite these advances, we expect that robots will always have limitations, particularly when faced with anomalies, edge cases, and corner cases. Humanoid robot appearances can disillusion humans as they appear human-like but lack human capabilities [23]. Humans face difficulty in creating mental models of robots and managing their expectations for their robot companion’s behavior and performance. Robots struggle to recognize human intent, which causes incorrect or poorly timed responses, as well as slow and jittery interaction. These issues result in an unnatural and inefficient teamwork with high human workload [14, 23]. Future HR teams need to consider robots as trustworthy team members despite their limits.

Given the difficulties of creating human-robot teams, it is clear placing a well-designed robot with a human to complete a task is not enough to ensure good teamwork and task execution. Design for HR teams must understand the context of human-robot relationships and the dependencies that form as they work together [25]. In joint activity, understanding interdependencies between team members will reinforce better human-machine systems design decisions [25]. Future HR teams will need to understand how teammates can communicate, coordinate, and collaborate effectively for mission success.

3 The Components of Teamwork

3.1 *Communication*

Communication is the expression or exchange of information between two (or more) parties [5, 18, 37]. For example, robots requesting help communicate: (1) getting attention, (2) alerting that help is needed, and (3) requesting help [5]. Any robotic cue provides pertinent information about its state and current action. While humanoid robots have a larger breath of communication means, non-humanoid robots are more goal driven and require more planning to take human collaborators into consideration [5]. Even robots that are capable of speech or text (on screen) lose the subtlety, tone, and context of human speech.

Communication can extend between various pairings in teams, between robots through a shared network, or to humans through various means. To do the latter, they require more signal types to be more informative and develop richer intuition. Signals are very limited in content (up to a few bits), but they are capable of conveying awareness, intent, and state. Numerous mechanisms support this by combining redundancy with emphasis through auditory, gaze, gesture, and motion. Signals can involve lights, sounds, haptic feedback, and more depending on the system. Language

is highly extensive and conveys a high level of detail, however, whether language is specific or general is dependent on task, domain and other factors [5].

3.2 Coordination

Coordination is the harmonious functioning of the group or ensuring that two or more people or groups can work together properly throughout the mission [12, 37]. It requires integration of activities and responsibilities for resources to be used efficiently [23]. Coordination requires cooperating with foresight and planning to set, organize, and monitor activity. Effective coordination requires (1) Common Ground, where mutual knowledge supports joint activity, (2) Directability, assessing and modifying individual actions within a joint activity, and (3) Intepredictability, being able to predict what others will do [23, 25]. These traits are more measurable and support the goal of teammates that can do work together successfully.

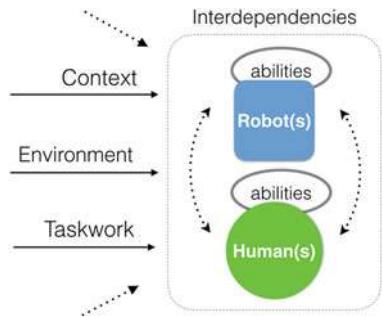
3.3 Collaboration

Collaboration is a joint activity involving two (or more) parties working collectively to achieve a common goal [9, 23, 25]. Joint work requires multiple members working together to achieve a shared objective and is dependent on communication and coordination. Members share of knowledge, intention, and goals between themselves —like two agents working together to build a bookshelf. Collaborative tasks can be tightly or loosely coupled, as well as planned or spontaneous [20, 23, 40]. In tightly coupled work, each member's actions depends on each other. In loosely coupled work, members engage in complementary actions towards a common goal. The difference between planned and spontaneous work is acutely dependent on the environment, situation, task, and more. In scavenger hunt (a collaborative task), teammates have a common goal (find all items) but do so separately.

4 Considerations for Designing Human-Robot Teams

It is essential to consider a variety of factors to design an effective HR team. These include the pitfalls that HR teams designers should keep in mind pitfalls HR teams can fall into, the task work scope for the team to complete, limitations of and support for team members, metrics for evaluating teams, and how to facilitate effective teamwork. These following sections discuss the importance of these considerations for taskwork, team composition, and structuring teamwork. Figure 1 shows these factors outside the team and between the agents. The context of the mission, the environment the team is in, and the tasks to complete are external factors that impact the design

Fig. 1 Factors that are influences on Human-Robot Teaming from external to internal within a team



of a team to fully capable of completing the work. Humans and robots have varying abilities, and the dependencies that arise between these agents as team members creates interdependencies within the team. Human-robot teams must have excellent communication, coordination, and collaboration in order to work effectively together amongst these factors of influence.

4.1 Pitfalls for Human-Robot Teams

A variety of macro and micro factors like the human(s), robot(s), remote users, controls, environments, and task context, affect the relationships between team members and the resulting teamwork [36].

Barriers of Communication High time delays in communication or even team members' inability to understand each other can cause collapses in teamwork. Barriers are a result of unintuitive or improper modes of communication when processing human attention, predicting actions, and understanding intent from others [35].

Inefficient Collaboration Team members with different goals work divergently resulting in delays in task completion and poor quality of work. Users who fail to check for qualifying capabilities or lack training and proficiency cause improper handoffs or transfer of control between users [36].

Poor Coordination A mismatch of individual abilities can cause poor cooperation when trying to coordinate activity, especially in cases where team members have gaps in their capabilities or uncomplimentary skills. A poor team composition results in a lack of trust between the members that can be detrimental to the coordination [31].

Bad Leadership A leader lacking leadership can steer a team into disarray. In imperfect scenarios, an inadequate leader may struggle to redirect the team or adjust the taskwork goals.

Lack of Management Adjacent to bad leadership is poor management, which can cause teams to become confused or lose sight of the work to be done. Managing

the team also requires managing the team's environment, continuously planning, and ensuring that the team is on track.

Workload Issues Task demands, temporal demands, and task structure cause workload issues [22]. The complexity of the mission and the consequences of its failure need to be considered while designing HR teams. The task constraints and pressure, task interruptions and ill-defined tasks are setbacks that can confuse team members [22].

Forgetting Environmental Factors Visibility, complexity, uncertainty and stressors factor in team interactions with the environment and each other [22]. Failure to consider the context-dependent factors result in teaming that is unprepared to work in its environment.

4.2 *Taskwork for Teams*

Taskwork is a key component of HR team design consideration as it is the breakdown of work teams need to complete to achieve their goal. Taskwork is also situational and context-dependent; the flow of teamwork and operator workload is dependent on it [22, 40]. Taskwork can determine how teams should be structured and put together; completing it requires varying levels of user performance, fault management, multiple team members, and for some tasks, multitasking. Scheduling and dependencies on other system components also constrain how taskwork can be assigned and managed. The type of taskwork and its criticality are important to review. For example, the individual actions exploring planetary surfaces vary and depend on the density of route waypoints, hazards, and obstructions. Properly valuing taskwork's complexity and fully scoping will lay out the groundwork for designing an HR Team to complete it [22, 25, 40].

4.3 *The Structure of Teams*

The structure, or composition, of an HR team, helps construct effective teamwork and is determined by the physical makeup of a team, as well as each individual's abilities [39, 40]. Team structure involves finding the proper team size to fit the mission goals and teammates who have complimentary skill sets to coordinate and collaborate effectively. The ratio of humans to robots is also important to consider, be it homogeneous (human-only, robot-only) or heterogeneous (mix of humans and robots) [39]. Team composition shapes interactions that vary based on the ratio from one human, one robot; one human, robot team; one human, multiple robots; human team, one robot; multiple humans, one robot; human team, robot team; human team, multiple robots; and multiple humans, robot team [40].

4.3.1 Agent Abilities and Team Abilities

Each agent's abilities and the overall team capabilities should compliment each other. Robot abilities are based on factors of the work they are given and their capabilities depend on the concept of capacity [25].

Capacity is the total set of inherent things (e.g., knowledge, skills, abilities, and resources) that an entity requires to competently perform an activity individually. (p. 47)

This concept of capacity implies a limit to a robot's capacity—specifically, their ability to extend their capabilities beyond engineered abilities and to possessing a certain level of autonomy or cognition. Autonomy for robots has been previously defined from a variety of perspective, from teleoperation to ‘full autonomy’. Desai wrote the resulting interaction between HR teams is dependent on this level of robot autonomy [6]. Goodrich defines autonomy as the measure of neglect a system can take—the more autonomous, the fewer interactions [18–20]. Brum’s sliding scale of autonomy was a large step in the understanding robot autonomy is not simply a single discrete mode, but adjustable depending on circumstances [3]. While there are many varieties of robot autonomy, it is clear that successful HR teams take shifting context-dependent autonomy into consideration for team design [14, 18].

Autonomy is a relative concept and a robot's autonomy should be defined respective to another system's autonomy [25]. One step further is recognizing autonomy is a function of the team's capacity together, not just a robot's cognition agents alone. We frame Team Autonomy to be an HR team's capability to operate as a single unit. HR team designers should understand this parallel concept of autonomy from a robot's shifting autonomy and the overall Team Autonomy.

4.4 Measuring Human-Robot Team Performance

Metrics are central to good design as they describe how to measure team design [32]. Metrics for team performance can be considered qualitative or quantitative, but identifying the best metrics is essential to assess effective teamwork. Working effectively as a team is benchmarked not only on the success of completing the mission but also the quality of teamwork.

Evaluating HR teams can be from a (1) high-performance perspective, through quantitative analysis and formal methods, (2) user perspective, through the usability, overall effectiveness, and satisfaction, or (3) team perspective through fluidity, team workload, or interaction between team members [29, 41]. Quantitative metrics measure efficiency and productivity by time to complete tasks, idle time for each member, and the total mission time [1]. Qualitative metrics assess social measures, team flexibility, adaptability, and robustness or resilience to errors in the environment. Several authors with different perspectives have identified metrics to measuring HRI through task, common, or interaction metrics [1, 32]. Steinfeld and Fong compare three common metrics that emphasize the HR team and interactions [32].

System performance This group is comprised of *quantitative performance* which evaluates effectiveness (% of mission completed) and efficiency (time taken to complete a task), *subjective ratings*, where indirect and direct factors impact effectiveness and *mixed-initiative utilization*, which includes the effort to regulate the control and interaction efforts had between robots and humans through interruptions, requests for help, etc. [32].

Operator performance This focuses on the system operator, considering *situational awareness* which impacts decision-making, performance and handling dynamic tasks, *workload*, which assesses multidimensional workload balancing techniques and operator stress, and *mental model accuracy*, which tracks how much the interface affects user performance [32].

Robot performance This examines *self awareness*, which is a degree by which a robot could assess itself through knowing its capacities, monitoring itself through tasks, and recovering from faults, *human awareness*, which is how a robot may be able to perceive humans and predict and read human behaviors, and *autonomy*, whereby a robot can function as a unit comparatively to a measure [32].

While these common metrics assess the components of a team, they do not identify a team metric. System performance describes how to quantitatively assess the overall system, but does not consider the interaction that can be measured amongst the team members. From a task perspective, Steinfeld and Fong continue to dive into “task metrics” which cover the span of robot capabilities and provide a more granular taxonomy on appropriate metrics for specific robot task performance (navigation, perception, management, manipulation, and social) [32]. Shah et al. describes quantitative cognitive and physical interaction metrics that can be gleaned from instances of active communication, coordination and collaboration within an HR team [1].

Physical Interaction metrics Physical interaction can be easily taken from systems through various metrics such as response time, availability, proximity of physical interaction, and duration of physical interaction.

Cognitive Interaction metrics Cognitive interaction is more difficult to measure but depends highly on the system used to evaluate HRI. These metrics include information exchange and assessment, decision and action selection, inherent lag, and command specifications as primary metrics to capture.

Freedy et al.’s *Collaborative Mixed-Initiative System* describe qualitative measurements allow for team assessment through evaluating how teammates predict each other’s actions, collaborate together and develop trust between members [15].

1. **Measures of Performance:** These measures are observable and come from the operators’ task skills, strategies, steps or procedures used to accomplish tasks. These consists of human team, UV control, and human/robot team processes. The HR team process consists specifically of shared mental models, human to robot ratio, level of trust, behavior acceptance and observability, human-robot coordination, and mixed initiative efficiency [15].
2. **Measures of Effectiveness:** These measures note the ‘goodness’ of the quality and execution of tasks but depend on environment and luck. This condenses down

Metric	Individuals	Team	Quantitative	Qualitative
Common Metrics	●	●	●	●
Task Metrics	●		●	●
Interaction Metrics		●	●	
Mixed Initiative Measures of Performance		●	●	●
Mixed Initiative Measures of Effectiveness		●	●	●

Fig. 2 Comparison of four metrics as team or individual focused, and quantitative versus qualitative

into mission and team behavioral effectiveness (collective conformity, latency, decision quality, and comparative performance amongst teams) [15].

The metrics above have distinct and different focuses on components of HRI and HR teaming. These differences and similarities are captured in the Fig. 2. HR teaming should use metrics that capture both the perspective of the team while balancing quantitative and qualitative metrics.

5 Structuring Effective Teamwork

This section brings together the previous investigation in workload and agent capabilities to demonstrate how to structure teams. Teaming is a dynamic structure where changing agent roles, task definition and requirements, environment context and circumstances affect how the team does work. The structure of the team affects when (pre, during, post), where, and how robots do their work.

First, system designers need to consider the roles each member will play and any occurring interdependencies as team members work together. Fong's collaborative control points to the importance of humans as collaborators instead of controllers [11]. Scholtz et al. notes five roles that humans take on when interacting with robots: supervisor, operator, teammate(peer), mechanic, and bystander [30]. Members' roles may shift as circumstances change, which reshapes the team structure and dynamics. Interactions change depending on the roles humans and robots take on in a team; these greatly impact how to foster teamwork effectively [30, 40]. A natural progression as humans and robots work closely is the occurrence of interdependence [20, 25]. Interdependencies and relationships structure teamwork from influences of

the external (environment, taskwork, and context) forces, and the internal (abilities, capabilities) [25].

interdependence describes the set of complementary relationships that two or more parties rely on to manage requires (hard) or opportunistic (soft) dependencies in joint activity. (p.47)

With that in mind, teamwork can be structured in the following ways:

1. **A “Play”** or sequence structure assigns work to users to follow the “play” (i.e. military plays, football plays). These types of plans are pre-thought out and simply need to be carried out with some room for mild adaptation given a non-nominal circumstance [16]. Plays provide a breakdown of the task into several appropriate metrics like time to complete and task allocation, offering a full blo-byblow description. This level of detail can fit certain types of task where the mission may be very specific and distinct. Overall, a play can be a strategy but if too constrained, it is not flexible to handle variances in scenarios.
2. **Function Allocation (FA)** first decides the work to be done then allocates that work to the agents in the system [38]. FA asks the questions who can do what task and describes how to make that decision; it determines the best fit for a user to a task before or during real-life execution. FA can be evaluated through modeling and simulation to vet a series of work allocations, using varying analytics to assess the best distributions of tasks. This process can be methodical, and has opportunity to use formal modeling tools to quantitatively analyze results. This type of work is best for evaluating teams when early-in-design to fully explore the potential combinations of teamwork.
3. **Bidding** allows agents bid on clearly delineated tasks [7]. Agents are responsible for task allocation and bid considering their availability, skill set, and time to complete the task. Team designers determine winners from variety of measures based on preference and appropriateness. This structure works well for teams with agents that have specific abilities and bidding measures are clearly defined.
4. **Interdependencies** implies both robot autonomy and the interface should be designed together to manage dependencies [25]. Given taskwork and team capabilities, work is assigned effectively for joint activities through interdependence requirements (observability, predictability, and directability) [11, 25]. Human-robot teams must be able to observe and understand their teammates actions, predict their teammates next moves, and direct each other to do work [27]. Teams cognition is key here in order to have the same shared mental model to facilitate communication and coordination [33].

The following examples of HR team scenarios show various types of team structures, ratios of humans to robots, and use cases of teams in a space exploration application. These structures demonstrate the breadth of work HR can accomplish [4, 14]. In Fig. 3, we show these four examples as the team works through the workload and the changing dynamics of the team.

1. **Robotic Scouting** involves a human remotely controlling a rover to scout planetary surfaces for scientific and technical progress. In this example, there is a 1:1

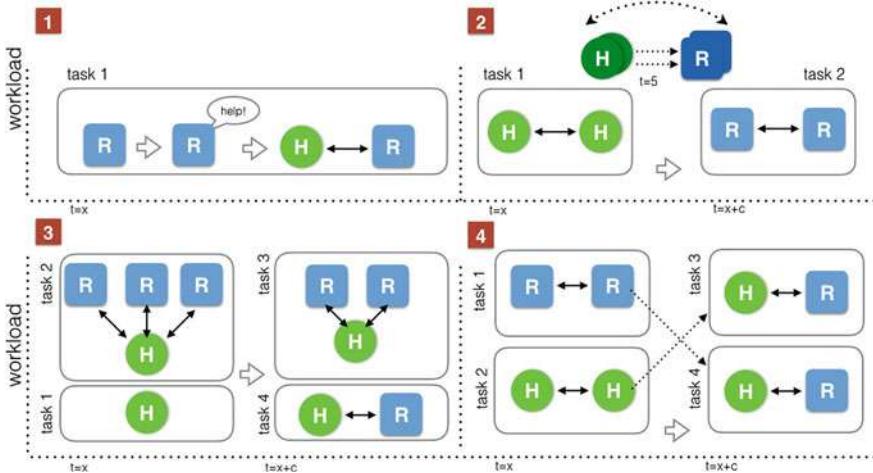


Fig. 3 Four different types of team structures that are possible build from a variety of factors

communication line between the human and robot. The work is completed before humans take action as a precursor to human workload. Additionally, the robot can work alone but needs to have strong capabilities in maneuvering and fault recovering from environmental factors. These two members must have good coordination and communication to efficiently explore and inform the most impactful information [14].

- Ex. 1** For a single task, a robot works in solitary scouting mission until it requests for human help or alert it has completed the task. The robot requests for help on a task and requires human intervention to work closely on a joint task. It interacts through communicating its progression through the task and completes the task 1:1 with the human thereafter. The robot here would require high capacity for communication, coordination, and collaboration.
- Robotic Followup** involves humans working first, then robots following up after to further investigate and finish the task. There may be a transfer of control between the humans and robots collaborating loosely together to finish the subsequent work. Teaming here revolves around coordinating this transfer and a follow-up of completing the task, see Fig. 3 [14].
 - Ex. 2** Two humans complete task 1 and communicate with the two-robot team the remainder of the task. The robot team then complete task 2. At the transition phase, communication of the remainder of the task and coordination in shifting authority of the task will impact the mission success and metrics.
 - Multiple Astrobees with Crew**, up to three Astrobees are capable of flying through the International Space Station and assist astronauts on board with re-

search activities. Members can remotely direct the bees to do specific actions (like navigation), or command them through a series of sequences (similarly to ‘plays’) to work directly with the crew [4]. Work between astronauts and Astrobees ranges from tightly compacted to more dispersed, see Fig. 3.

- a. **Ex. 3** Shifting abilities and dynamic task requirements can alter groupings within a team of two humans and three robots [8]. At $t = x$, one human is working with three bees and another human is working alone. At some time $t = x + c$ (where c is the time it takes to complete tasks 1 and 2), one of the robot bees moves to the other human to take on task 4 together while the remaining bees stay to finish task 3. Here the successful coordination and collaboration allows the flexibility of the bee to traverse between human teammates and take on different team tasks.
4. **Planetary Exploration** with HR teams may be far off in the future but understanding the implications of close-knit teamwork on team success is important for homologous real life scenarios. Fong et al. provides several real-life examples of closely coupled HR teams co-located and responsible for a multitude of joint tasks [14]. These robots are highly dexterous and capable of assorted collaborative tasks. They are remotely controlled or somewhat autonomous and have strong skills in communication through voice and gesture recognition. Their capabilities enable collaboration between HR teammates and provide insight for culminating interdependencies from close conduct.
 - a. **Ex. 4** At $t = x$, a robot team and a human team work loosely coupled on tasks 1 and 2 respectively. At $t = x+c$, the teams shift partners into two single HR teams and work on the tasks 3 and 4 in tightly coupled teams. Here, teamwork is highly dynamic and the modification of teams is necessary to coordinate effectively [8]. Robots and humans need complimentary skillsets and adapt to new scenarios and unpredictable contexts easily.

6 Future of Human-Robot Teaming

This paper discusses various factors that impact building effective teamwork in HR teams by investigating the components of teaming of taskwork, capabilities, dependencies. However, future work can continue to provide insight on this topic by exploring the effect of communication, coordination, and collaboration on team performance. As the need for effective HR teams grows, analyzing these design considerations will allow for these heterogeneous teams to advance from a team perspective. Agent capabilities, taskwork breakdowns, metrics for valuable measurements, team autonomy and teamwork structuring are all areas that have room to expand our understanding of HR teaming. The goal of this paper is to highlight that designing for human-robot teams is important for effective teamwork. Building successful teams goes beyond considerations for individuals and extends to the team’s combined ca-

pabilities and deepening our knowledge of team member relationships. Designing teams where robots are not tools, but peers, is a start to designing effective teamwork and structuring future HR teams to be more successful in their endeavors.

Acknowledgements This work was supported by a grant from the NASA Human Research Program #NNX17AB08G. Partial support was provided by a DARPA/NASA effort to model and predict workload in human-machine systems.

References

1. Arnold, J.A.: Towards a framework for architecting heterogeneous teams of humans and robots for space exploration. Doctoral dissertation, Massachusetts Institute of Technology (2006)
2. Breazeal, C.: Social interactions in HRI: the robot view. *IEEE Trans. Syst. Man Cybern Part C (Appl. Rev.)* **34**(2), 181–186 (2004)
3. Bruemmer, D.J., Dudenhoeffer, D.D., Marble, J.L.: Dynamic-autonomy for urban search and rescue. In: AAAI Mobile Robot Competition, pp. 33–37 (2002)
4. Bualat, M., Barlow, J., Fong, T., Provencher, C., Smith, T., Zuniga, A.: Astrobee: developing a free-flying robot for the International Space Station. In: AIAA SPACE 2014 Conference and Exposition, vol. 4643 (2015)
5. Cha, E., Matari, M., Fong, T.: Nonverbal signaling for non-humanoid robots during human-robot collaboration. In: 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pp. 601–602. IEEE (2016)
6. Desai, M., Yanco, H.A.: Blending human and robot inputs for sliding scale autonomy. In: IEEE International Workshop on Robot and Human Interactive Communication, ROMAN 2005, pp. 537–542. IEEE (2005)
7. Dias, M.B., Goldberg, D., Stentz, A.: Market-based multirobot coordination for complex space applications. In: The 7th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS) (2003)
8. Dias, M.B.: Traderbots: a new paradigm for robust and efficient multirobot coordination in dynamic environments. Robotics Institute, 153 (2004)
9. Drury, J.L., Scholtz, J., Yanco, H.A.: Awareness in human-robot interactions. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 1, pp. 912–918. IEEE (2003)
10. Fiore, S.M.: Interdisciplinarity as teamwork: how the science of teams can inform team science. *Small Group Res.* **39**(3), 251–277 (2008)
11. Fong, T.: Collaborative control: a robot-centric model for vehicle teleoperation. Technical Report CMU-RI-TR-01-34. Ph.D. dissertation, Robotics Institute, Carnegie Mellon University (2001)
12. Fong, T., Nourbakhsh, I., Dautenhahn, K.: A survey of socially interactive robots. *Robot. Auton. Syst.* **42**(3), 143–166 (2003)
13. Fong, T., Nourbakhsh, I.: Interaction challenges in human-robot space exploration. *ACM Interact.* **12**(2), 42–45 (2005)
14. Fong, T., et al.: Space telerobotics: unique challenges to human-robot collaboration in space. *Rev. Hum. Factors Ergon.* **9**(1), 6–56 (2013)
15. Freedy, A., DeVisser, E., Weltman, G., Coeyman, N.: Measurement of trust in human-robot collaboration. In: International Symposium on Collaborative Technologies and Systems, 2007. CTS 2007, pp. 106–114. IEEE (2007)
16. Funk, H., Goldman, R., Miller, C., Meisner, J., Wu, P.: A playbook for real-time, closed-loop control. In: Proceedings of the 1st Annual Conference on Human-Robot Interaction (2005)
17. Funk, H., Goldman, R., Miller, C., Meisner, J., Wu, P.: A Playbook TM for Real-Time, Closed-Loop Control (2006)

18. Goodrich, M.A., Crandall, J.W., Stimpson, J.L.: Neglect tolerant teaming: issues and dilemmas. In: Proceedings of the 2003 AAAI Spring Symposium on Human Interaction with Autonomous Systems in Complex Environments, pp. 24–26 (2003)
19. Goodrich, M.A., Olsen, D.R.: Seven principles of efficient human robot interaction. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 4, pp. 3942–3948. IEEE (2003)
20. Goodrich, M.A., Schultz, A.C.: Human-robot interaction: a survey. *Found. Trends Hum.-Comput. Interact.* **1**(3), 203–275 (2007)
21. Groom, V., Nass, C.: Can robots be teammates?: benchmarks in human-robot teams. *Interact. Stud.* **8**(3), 483–500 (2007)
22. Hooey, B., Kaber, D., Adams, J., Fong, T., Gore, B.: The underpinnings of workload in unmanned vehicle systems. *IEEE Trans. Hum.-Mach. Syst.* (2017) (in press)
23. Hoffman, G., Breazeal, C.: Collaboration in human-robot teams. In: Proceedings of the AIAA 1st Intelligent Systems Technical Conference, Chicago, IL, USA (2004)
24. Hollenbeck, J.R., DeRue, D.S., Guzzo, R.: Bridging the gap between I/O research and HR practice: improving team composition, team training, and team task design. *Hum. Resource Manage.* **43**(4), 353–366 (2004)
25. Johnson, M., Bradshaw, J.M., Feltovich, P.J., Van Riemsdijk, M.B., Jonker, C.M., Sierhuis, M.: Coactive design: designing support for interdependence in joint activity. *J. Hum.-Robot Interact.* **3**(1), 2014 (2014)
26. Kim, M., Oh, K., Choi, J., Jung, J., Kim, Y.: User-centered HRI: HRI research methodology for designers. In: Mixed Reality and Human-Robot Interaction, pp. 13–33. Springer Netherlands (2011)
27. Klien, G., Woods, D.D., Bradshaw, J.M., Hoffman, R.R., Feltovich, P.J.: Ten challenges for making automation a “team player” in joint human-agent activity. *IEEE Intell. Syst.* **19**(6), 91–95 (2004)
28. Salas, E., Cooke, N.J., Rosen, M.A.: On teams, teamwork, and team performance: discoveries and developments. *Human Factors* **50**(3), 540–547 (2008)
29. Scholtz, J.: Evaluation methods for human-system performance of intelligent systems. In: Proceedings of the 2002 Performance Metrics for Intelligent Systems (PerMIS) Workshop, Gaithersburg, MD. National Institute of Standards and Technology (2002)
30. Scholtz, J.: Theory and evaluation of human robot interactions. In: Proceedings of the 36th Annual Hawaii International Conference on System Sciences. IEEE (2003)
31. Scholtz, J., Consolvo, S.: Toward a framework for evaluating ubiquitous computing applications. *IEEE Pervasive Comput.* **3**(2), 82–88 (2004)
32. Steinfeld, A., Fong, T., Kaber, D., Lewis, M., Scholtz, J., Schultz, A., Goodrich, M.: Common metrics for human-robot interaction. In: Proceedings of the 1st ACM SIGCHI/SIGART Conference on Human-Robot Interaction, pp. 33–40. ACM (2006)
33. Sycara, K., Sukthankar, G.: Literature review of teamwork models, p. 31. Carnegie Mellon University, Robotics Institute (2006)
34. Thrun, S.: Toward a framework for human-robot interaction. *Hum.-Comput. Interact.* **19**(1–2), 9–24 (2004)
35. Thomaz, A., Hoffman, G., Cakmak, M.: Computational human-robot interaction. *Found. Trends Robot.* **4**(2–3), 105–223 (2016)
36. Tsui, K.M., Yanco, H.A.: Design challenges and guidelines for social interaction using mobile telepresence robots. *Rev. Hum. Factors Ergon.* **9**(1), 227–301 (2013)
37. Pedersen, L., Kortenkamp, D., Wettergreen, D., Nourbakhsh, I., Korsmeyer, D.: A survey of space robotics. In: Proceedings of the 7th International Symposium on Artificial Intelligence, Robotics and Automation in Space (2003)
38. Pritchett, A.R., Kim, S.Y., Feigh, K.M.: Measuring human-automation function allocation. *J. Cogn. Eng. Dec. Making* **8**(1), 52–77 (2014)
39. Yanco, H.A., Drury, J.L.: A taxonomy for human-robot interaction. In: Proceedings of the AAAI Fall Symposium on Human-Robot Interaction, pp. 111–119 (2002)

40. Yanco, H.A., Drury, J.: Classifying human-robot interaction: an updated taxonomy. In: 2004 IEEE International Conference on Systems, Man and Cybernetics, vol.3, pp. 2841–2846. IEEE (2004)
41. Yanco, H.A., Drury, J.L., Scholtz, J.: Beyond usability evaluation: analysis of human-robot interaction at a major robotics competition. *Hum.-Comput. Interact.* **19**(1–2), 117–149 (2004)

An Analysis of Degraded Communication Channels in Human-Robot Teaming and Implications for Dynamic Autonomy Allocation

**Michael Young, Mahdieh Nejati, Ahmetcan Erdogan
and Brenna Argall**

Abstract The quality of the communication channel between human-robot teammates critically influences the team's ability to perform a task safely and effectively. In this paper, we present a nine person pilot study that investigates the effects of different degradations of that communication channel, and within three shared-autonomy paradigms that differ according to how and at what level control is partitioned between the human and the autonomy. Accordingly, the rate and granularity of the human input differs for each shared-autonomy paradigm. We refer to each paradigm according to the input expected from the user, namely high-level, mid-level and low-level control paradigms. We find three primary insights. First, interruptions in the signal transmission (dropped signals) decrease safety and performance in modes where continuous and high-bandwidth inputs from the human are expected. Second, decreased transmission frequency offers a trade-off between safety and performance for low-level and mid-level control paradigms. Lastly, noise alters the safety of high-level input since the user is not continually correcting the signal. These insights inform us when to shift autonomy levels depending on the quality of the communication channel, which can vary with time. Knowing the ground truth of how the signal was degraded, we evaluate a recurrent neural network's ability to classify whether the communication channel is experiencing lowered transmission frequency, dropped signals or noise, and we find an accuracy of 90% when operating with low-level commands. Combined with the key insights, our results indicate that a framework to dynamically allocate autonomy between the user and robot could improve overall performance.

M. Young (✉) · M. Nejati · A. Erdogan · B. Argall
Rehabilitation Institute of Chicago, Chicago, IL, USA
e-mail: mikesyoung@u.northwestern.edu

M. Nejati
e-mail: m.nejati@u.northwestern.edu

A. Erdogan
e-mail: ahmetcan.erdogan@northwestern.edu

B. Argall
e-mail: brenna.argall@northwestern.edu

M. Young · M. Nejati · A. Erdogan · B. Argall
Northwestern University, Evanston, IL, USA

1 Introduction

In recent years, the world increasingly relies on human-robot teams to perform various functions in defense, human assistance and field operations. In these teams, the human operator interacts differently with the robotic system depending on the task. In some cases, the user issues low-level commands where they are in charge of the majority of control. In others, the user provides high-level commands, such as goals, which the robot works towards achieving. Indeed, there are a multitude of control levels in between and the level is typically set before the team sets out to accomplish a given task. However, there are many scenarios in which performance might improve if the control allocation shifted online between the two entities.

One reason that autonomy levels might benefit from shifts is signal degradation. In the domain of user-operated assistive robots, such as a robotic wheelchair, the commands issued by the user may degrade due to human motor impairment, fatigue or pain. There is a parallel to field robotics where the user operates a robot at a distance. In this case, the signal may degrade due to barriers between the robot and operator or environment changes such as severe weather. In both scenarios, operators are susceptible to distraction or work overload that may affect performance and transmit through the control signal. Other reasons to shift autonomy may include hardware issues or changes in the environment that prevent either the robot or the user from providing reliable control signals. For example, a person using a powered wheelchair may move from indoors to a busy sidewalk where more moving obstacles are present and the subject can no longer avoid collisions independently.

In the domain of assistive robots, the signals provided by motor-impaired users in many ways mirror those of compromised communication channels: the user signals are often noisy due to artifacts left by their impairment (noise), limb weakness may result in undetectable commands by the interface (dropped signal) and the rate at which the user provides commands may also vary due to factors like fatigue and pain (transmission frequency).

To study how shared-autonomy performance changes with signal degradation on the communication channel between the human and the autonomy, we conduct a nine person pilot study to inform future decisions on how autonomy should be allocated. In the study, subjects use three levels of control to perform daily-life tasks with a robotic wheelchair while we modulate the signal to simulate real-world challenges. Furthermore, we assess the feasibility of detection of a degraded communication channel so that we can switch autonomy automatically when necessary. The end goal is to provide a dynamic autonomy allocation framework that will improve the safety and performance of human-robot teams in both field and service robot applications.

2 Related Work

Here we review related literature on autonomy allocation. Much of the literature develops paradigms for determining beforehand which autonomy level to use [2, 11]. These take into account factors such as task criticality, task accountability and environment complexity. However, these static, *a priori* methods are often not robust to the varying mental load of the user and changing environment.

Other works [15–17] investigate physiological parameters, using sensors such as EEG and ECG, as an indication of a user’s cognitive load. (Cognitive load can play a critical role in successful teleoperation and control of robots [9].) The physiological parameters are used to indicate when the autonomy should change control levels. For the domain of these works (pilots and military), to expect the physiological signals is reasonable, as soldiers and military pilots already wear highly instrumented and sensorized gear. For assistive robotics however, it is unlikely that we would have access to such signals due to both fiscal constraints and user preference.

Trust between the user and robot has drawn interest from researchers as a metric to allocate autonomy. In the field of human-robot interaction, several studies [4, 7, 8] outline key factors, such as feedback, environment and age, that influence a human’s trust in automation, which ultimately affects the team’s performance. In other work, researchers calculate trust in both the robot and the human through performance-based metrics [13] and by comparing the autonomy signal and the user input [3]. Trust shows promise as a factor to influence autonomy allocation, and we expect other metrics may also play a role, such as communication channel quality.

Most similar to our work, Choi et al. [5] perform a virtual experiment where subjects navigate a mobile robot through an obstacle course while noise is added at a specific section of the map. The users operate in three modes (1) full teleoperation (2) goal selection and (3) manual switching between goal selection and full teleoperation. They demonstrate that a dynamic allocation of autonomy controlled by the user (mode 3) outperforms the other two modes when faced with a distraction task.

Our work differs from the state of the art in several ways. We conduct an experiment using a physical system and modulate the signal with three degradation schemes: dropped signals, transmission frequency and noise. This helps us to identify when performance or safety has declined, and we furthermore do so for multiple levels of (static) autonomy allocation. From these results, we gather insights for a dynamic autonomy allocation framework and demonstrate a signal degradation detection technique to be used within said framework.

3 Methods and Design

The focus of this study is human-robot teams in which the robot is jointly controlled by robotics autonomy and a human operator. In these scenarios, the *robot* relies on the control commands from the human operator and its own sensor readings.

The quality of the *communication channel* which relays this information between the operator and the robot impacts the team's interaction. If degraded, it can obstruct the transmission of information needed for successful task completion.

The human operator can control the robot with different levels of command granularity: from low-level commands using teleoperation, through increasingly higher levels of commands until (nearly) full autonomy. While low-level commands give the human operator more control over the minutia of task execution, higher-level commands may be all that are practical when the communication channel is degraded—for example, due to increased distance of communication, human fatigue or other external factors.

The purpose of this study is to investigate the effect of various degradations of the signal coming from the human and how this changes with various control levels (autonomy allocations)—that is, which control levels are invariant or particularly susceptible to a given signal degradation. We design an experiment to investigate this scenario in a controlled laboratory environment. Towards this aim, nine subjects control the navigation of a mobile robot to multiple goal locations. The robot is commanded with three different levels of control granularity, while the signal is artificially modulated to capture different features of communication channel quality. The task performance, safety, control signals and human attention are monitored during task execution. The following subsections elaborate on details of the design and protocol.

3.1 Control Level

Humans command mobile robots using different control levels with varying degrees of command granularity, typically dictated by the task, environment and/or the user's cognitive load. The user signal might encode *low-level* control commands—for example, the speed and direction at every instance in the trajectory. Commonly, for low-level control the user operates the mobile robot using an interface like a joystick with some visual information provided by their own eyes, on-board cameras or a sensorized environment. In other formulations, the operator may provide *mid-level* control commands—for example, discrete longer-duration actions such as turn right or go forward. Such commands might be provided via switches, button presses or voice, to name a few. In *high-level* control, the operator provides even higher level information—for example, the human might indicate a task or goal, through selections on a screen or natural language, for example.

In this study, we consider three levels of shared control:

- 1. Low-level Control (C_L):** Using a PS3 controller *joystick*, the user provides a continuous stream of linear and angular velocities to control the robot. The autonomy steps in only to prevent collisions [1], and the execution trajectory otherwise is determined by the human operator.

2. **Mid-level Control (C_M):** Using the PS3 controller *buttons*, the user provides discrete directional commands: such as “turn left” and “forward”. The autonomy executes these commands, taking care also to avoid obstacles.¹
3. **High-level Control (C_H):** The user provides end goals or waypoints for the robot to navigate towards via a point-and-click visual interface using RVIZ.² An autonomous path planner calculates a safe trajectory from the current robot pose to the human-provided target pose, while avoiding obstacles.

3.2 Signal Modulation

The quality of signals received from the human by the robotic system depends on the quality of the communication channel between them, which can be influenced by human and environmental factors. In various scenarios involving a mobile robot, the signals may be sent over wireless networks. The wireless signals can be affected by many external factors such as weather, electrical interference, radio frequency interference and distance, to name a few.

Signal quality can be quantified according to different properties such as the signal frequency, transmission frequency and noise. In this study, we replicate these factors in a controlled setting, where three signal properties are *individually* modulated. For each of the following signal properties, we test three different *levels* of signal modulation by changing the threshold values: low modulation, moderate modulation and a high level of modulation. The thresholds which determine the modulation settings were chosen empirically during the experiment design phase.

1. **Dropped Signals:** Every input signal is assigned a random number η sampled from a Gaussian distribution $\eta \sim \mathcal{N}(0, 1/3)$. If η is greater than a preset threshold, the corresponding input signal is dropped. (In our implementation, the three thresholds were [0.6, 0.5, 0.4].) The result is *lost* information.
2. **Transmission Frequency:** The rate ρ at which the robot receives the user’s command over the communication channel is varied, within a preset range. (In our implementation, the three values of ρ were [5, 10, 15] Hz.) The result is a *delay* in the receipt of information.
3. **Noise:** A random value ε is sampled from a zero-mean Gaussian distribution, with a different variance σ^2 for each combination of control level and modulation level. The noise is implemented differently depending on the control level:
 - a. Low-level control (C_L): ε is continuously added to the control signal at low, moderate and high levels of variance. (In our implementation, $\sigma^2 = [0.6, 0.8, 1.0]$.)

¹Note the primary differences between C_L and C_M are the discrete input and the rate of input.

²RVIZ is a 3D visualization tool distributed with the Robot Operating System (ROS).

- b. Mid-level control (C_M): If ε is greater than the preset noise threshold, one of the commands is chosen randomly. (In our implementation, the thresholds were [1.0, 0.92, 0.85] and $\sigma^2 = 1/3$.)
- c. High-level control (C_H): ε is multiplied by a distance value d dictated by the task, and $d \cdot \varepsilon$ is added to the goal position provided by the user. (In our implementation, d is set at [6, 8, 10] cm and $\sigma^2 = 1$.)

Participants perform each task under 10 experimental conditions: three modulation levels for each of three signal properties (dropped signals, transmission frequency, noise), plus an unmodulated (clean) signal.

3.3 Experimental Setup and Tasks

We use a robotic powered wheelchair in this experiment. This wheelchair, shown in Fig. 1, is a commercially available Permobil wheelchair that we retrofit with a laser scanner, RGB-depth sensor and on-board computer. These components plus our

Fig. 1 Robotic wheelchair



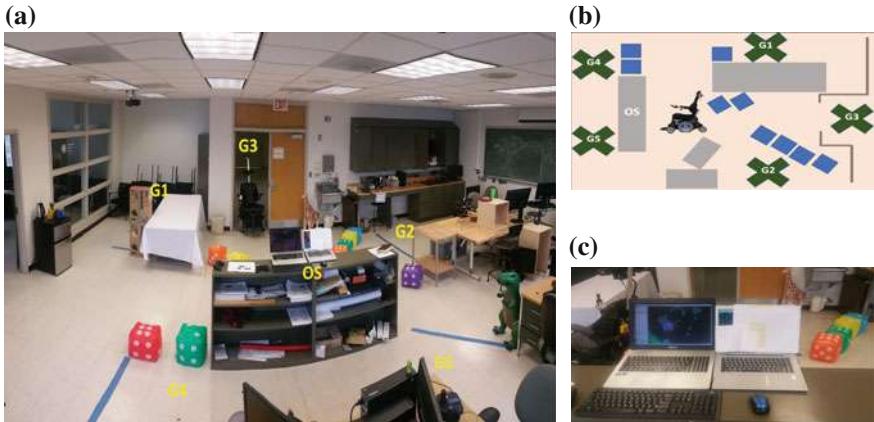


Fig. 2 Experimental setup. **a** Task layout: (G1) Docking station, (G2) Turn in place for orientation, (G3) Doorway traversal, (G4) Left turn (wide), (G5) Right turn (tight), (OS) Operator Station. **b** Layout diagram showing five goals. **c** Operator station

software suite provide additional autonomous capabilities such as doorway detection, obstacle avoidance and path-planning, to name a few.

The test environment is located in the Assistive and Rehabilitation Robotics Laboratory at the Rehabilitation Institute of Chicago. The setup consists of an obstacle course as shown in Fig. 2. The operator is positioned at the operator station³ (Fig. 2c) and can clearly observe the robotic wheelchair for each task.

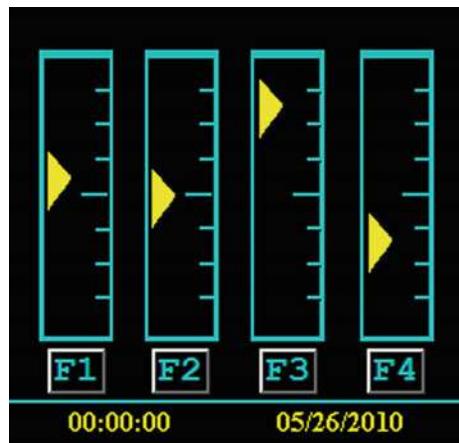
Wheelchair tasks are chosen for measuring task performance, safety and control signals. A distraction task is also included for measuring human attention.

Wheelchair Tasks. The tasks are selected to cover a range of commands and non-trivial control strategies. In the domain of assistive powered wheelchairs, some of the challenging daily tasks include obstacle avoidance, navigating through tight spaces and correcting orientation for a desired pose. The following five tasks are selected from the Wheelchair Skills Test (WST) [14] and illustrated in Fig. 2b: (G1) Dock at a table (G2) Turn in place for orientation correction (G3) Doorway traversal (G4) Wide left turn and (G5) Tight right turn.

The goals are located such that the distance traversed from the center of the room to each of the five goal positions is equal. Achieving the above goals requires careful maneuvering around obstacles and controlling the linear and angular velocity of the robot's trajectory.

As seen in Fig. 2a, blue tape lines on the floor mark the goals. In order for the goal to be considered as successful, the wheelchair frame needs to fully cross the

³We have the subjects stand at a static operator station, instead of riding the wheelchair, in order to allow for the assessment of subject attention using an established distraction task [12] that is well-studied within the human factors literature. This task requires the subject to monitor a screen and interact with a keyboard, which was an overly cumbersome setup to have onboard the wheelchair.

Fig. 3 Distraction Task

blue line. The operator is located at the operator station and has full visibility of the wheelchair for all five goals.

Distraction Task. To measure and evaluate the operator's cognitive workload and strategic behavior, we include a distraction task in the experiments. Distraction tasks such as this are commonly employed in psychophysiological studies.

We use the U.S. Air Force Multi-Attribute Task Battery (US AF_MATB) software developed and distributed publicly by the U.S. Air Force Research Laboratory [12] and well-studied throughout the human factors literature. For this study, the “Gauges” subtask from System Monitoring is selected (Fig. 3). In normal operating behavior, the yellow gauge indicator fluctuates within one tick of the center gauge. A malfunctioning gauge goes beyond this normal operating range. The user's task is to monitor the gauges and send a correcting signal when a gauge has malfunctioned by pressing the corresponding key (i.e. F1, F2, F3 or F4). The speed of the gauges and the rate of malfunction are tunable.⁴ All other adjustable parameters were kept at default settings.

3.4 Procedure

The experimental protocol and consent form was approved by Northwestern's Institutional Review Board (IRB). The full session lasted for approximately two hours.

Participants. Nine consenting able-bodied adults (age range: 21–28) participated in the experiments. The subjects included those with varying levels of skill and experience with robotic devices: from no experience to regular usage.

⁴For this study, we use the following System Monitoring Subtask Basic Parameters: (a) Gauge Speed Lower Limit = 2, (b) Gauge Speed Upper Limit = 4, (c) Correct Fault Identification Pause = 10 and (d) Gauge Malfunction Timeout = 10. We use the keyboard as the only input option.

Protocol. The participants were introduced to the mobile robot and given an overview of the experiment and the wheelchair tasks. They were shown each task and given complete instructions on what constituted a completed goal. Then they were introduced to the distraction task. They were instructed on the normal operating behavior of the gauge and what was considered a malfunction. They were shown how to respond appropriately and given time to practice monitoring and operating the distraction task. It was stressed to the users that they should treat both wheelchair and distraction tasks with equal importance. The session began after the participant became familiar with each task and the nature of the experiment. They were informed that their control input may be randomly varied, but they were not given the details about what features of the signal would be varied or how.

Each session consisted of *three sections* corresponding to the three control levels: C_L , C_M , and C_H . For each of three control levels, 30 trials were performed, covering all 10 combinations of modulation type-level (3 modulation types \times 3 modulation levels + 1 clean run) with 3 tasks executed per combination. (Which 3 tasks were randomly assigned and balanced, such that across subjects each combination was performed the same number of times for each task, and within a given subject across all combinations each task was performed the same number of times.) The order of the control levels, modulations and modulation levels was randomized and balanced across participants in order to minimize bias due to fatigue.

Each *section* of the experiment consisted of *two phases*: (1) an instruction phase and (2) a test phase. In the instruction phase, the participant was shown how to use the control level for the current section of the experiment and allowed time to become familiar with its operation. This time varied for each participant. After the participant was comfortable, the 30 trials of the test phase began. For each trial, the subjects were given their next goal after the completion of the current one. Subjects were not aware of which modulation setting was applied to their control signals. For safety, collision avoidance remained on at all times.

Metrics. In accordance with the literature on assistive and mobile robotics, we chose two metrics:

1. **Performance:** Calculated as the time from task initiation until the goal was reached. This metric is important for scenarios where the objective is to optimize time, for example crossing a busy road in a timely manner.
2. **Safety:** Calculated as the average distance from the closest obstacle to the robot at each time-step of the task execution. This metric is useful when physical safety is a priority; for example, operation in a crowd where the user and those around them are safer the farther the wheelchair is from any person or object.

4 Experiment Results

This section highlights key results from our pilot experiment for each metric. Statistical analysis is performed using analysis of variance (ANOVA) where group labels

are either modulation level or shared-control paradigm. If statistical significance is found ($p < 0.05$), a pairwise t-test is performed and results are indicated in Figs. 4, 5 and 6. For all plots, * denotes $p < 0.05$, ** $p < 0.01$ and *** $p < 0.001$.

Dropped Signals. Looking at performance for control level C_L , we notice that dropping the signal significantly alters a user's ability to complete a task within a reasonable time frame, shown in Fig. 4a. Namely, a statistically significant difference in task time is found between the low and high modulation levels ($p < 0.05$). Figure 4b shows that safety is also significantly compromised with a dropped signal ($p < 0.05$).

Conversely, dropped signals do not appear to significantly affect control levels C_M and C_H . This suggests that when the signal is degraded by drop, the autonomy should switch away from C_L since both safety and performance are compromised. The results further suggest that an appropriate threshold, above which the amount of dropped signal is considered too damaging for C_L , should be set between the low and moderate modulation levels.

Channel Frequency. The results of lowered channel frequency suggest some trade-offs between safety and performance. Figure 5a indicates C_L provides the best task time performance across all modulation levels, and significantly so for highly delayed signals ($p < 0.05$). However, Fig. 5b shows that C_M is safer as users tend to operate farther from obstacles, significantly so at moderate modulation levels ($p < 0.01$).

Signal Noise. Signals degraded by noise can play a role in the safe operation of the robot. Safety is more compromised across all noise levels in C_H compared to both C_L and C_M . While this difference is not statistically significant, it likely would have practical implications. On average, the robot moves 3.6 cm closer to obstacles when operating under C_H with noise at all levels, as shown in Fig. 6b. For context, the ADA requires doorways to be only 11 cm wider than our Permobil wheelchair,

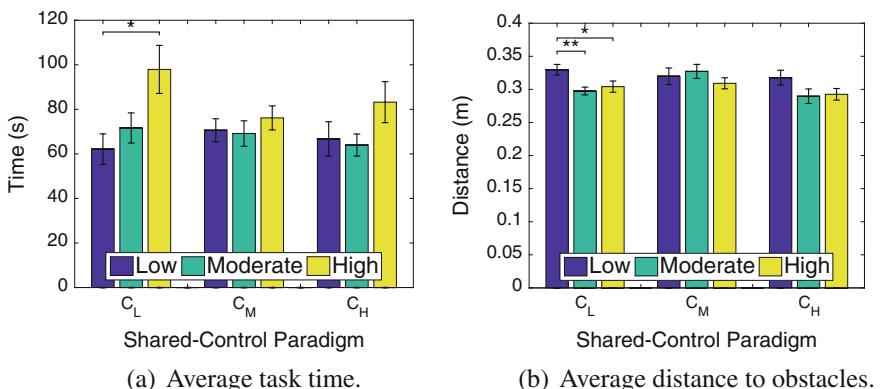


Fig. 4 Effect of modulating the dropped signal. Data from three modulation levels (Low (blue), Moderate (green), High (yellow), Sect. 3.2) for each of three shared-control paradigms (C_L , C_M , C_H). Plots show (a) task time and (b) distance to obstacles, averaged over all tasks and subjects

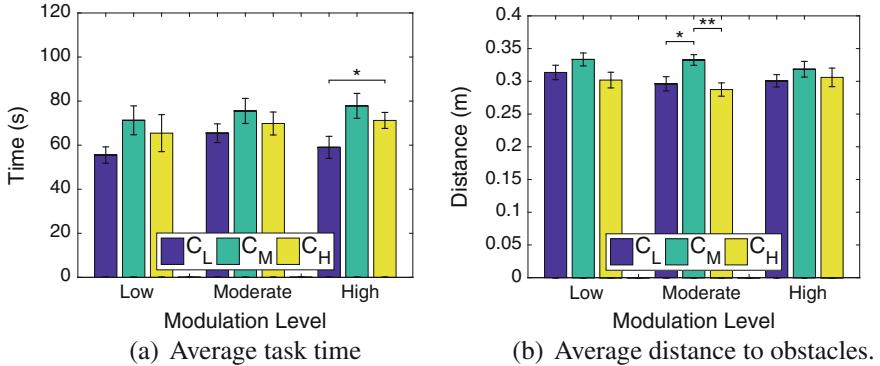


Fig. 5 Effect of modulating channel frequency. Data from three shared-control paradigms (C_L (blue), C_M (green), C_H (yellow)) for each of three frequency modulation levels (Low, Moderate, High, Sect. 3.2). Plots show (a) task time and (b) distance to obstacles, averaged across all tasks and subjects

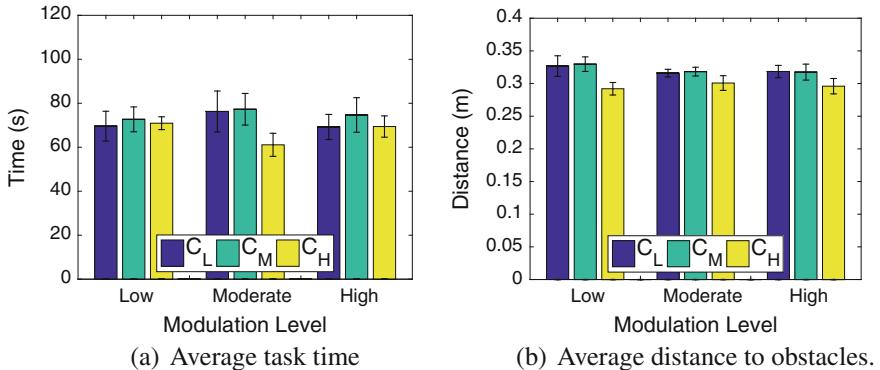


Fig. 6 Effect of modulating signal noise. Data from three shared-control paradigms (C_L (blue), C_M (green), C_H (yellow)) for each of three noise modulation levels (Low, Moderate, High, Sect. 3.2). Plots show (a) task time and (b) distance to obstacles, averaged across all tasks and subjects

so 3.6 cm can be significant. Somewhat surprisingly, noise otherwise appears to have little effect on performance or safety for control levels C_L and C_M .

Distraction Task. The results of ANOVA on the distraction task performance do not indicate any statistical significance across control levels or modulation levels. Across all paradigms, the percent correct gauge responses of triggered faults is 62.0% \pm 26.4%.

5 Signal Degradation Detection

The results of the experiment suggest that the signal quality influences both the performance and safety of the user differently in each control level. In real-world scenarios, operating in a degraded state could lead to a mission-critical failure or compromise the safety of a patient. Thus, it is important to allocate control and autonomy appropriately in real-time. The first step is to detect the quality of the signal. This section outlines the use of specific machine learning techniques to classify the state of the signal's degradation.

When in C_L or C_M the robot receives from the user only motion commands. In C_H , the robot receives only a goal. If the communication channel has degraded at all, the robot would need to detect the channel quality using only this information.

We choose a recursive neural network (RNN) structure using long short-term memory cells (LSTM) that classifies the user's commands over a set time period as either clean, noisy, dropped or lowered transmission frequency. The RNN LSTM is chosen because of its ability to retain information about the previous state or input. Moreover, results in speech processing show that a bidirectional LSTM (BLSTM) structure which shares information about future states, improves the classification rate [6]. We test both in our analysis (since there is additional computational complexity associated with the BLSTM). In our implementation, a snapshot of the signal from the human—continuous velocity commands, discrete motion commands or goal positions, depending on the control level—for a designated number of samples is the input to the RNN, which outputs a classification of the signal's state.

To obtain unbiased results, we use three-fold cross validation where, in each fold, 6 subjects (randomly balanced) are used to train and the 3 remaining subjects to test. The reported accuracy in Table 1 is the average of the three models from the cross validation. The data is split into samples of 30 consecutive points in time and then randomized for both training and testing. We also ensure the data is split using approximately equal amounts of all classes. The algorithm used for training is Adam [10] with a maximum of 200 epochs. This process is repeated for each control level. Since the signal type is different for each control level and robot will always know its current control level, it is necessary and reasonable to train separate networks.

When operating under C_L , it is critical to determine when the signal is dropping because of the significant decrease in safety and increase in average task time. In Table 1, both the unidirectional and bidirectional LSTM achieve a classification accuracy between 70 and 80% when classifying all 4 possible signal states.

The results, however, indicate that primary source of error is false positives between the clean and lowered transmission frequency samples—the network could not differentiate reliably between the two. Since for C_L lowered transmission frequency does not appear to affect performance or safety (Fig. 5), both transmission frequency and clean modulations can be bundled into a single class, which increases the accuracy to ~90% for both models. This 10% error might further be reduced by taking an ensemble approach or the mean over several time intervals rather than a single 30 sample segment, for example.

Table 1 Prediction accuracy results for single-layer 128-cell LSTM and BLSTM architectures

Architecture	Classes	Accuracy (%)		
		C_L	C_M	C_H
LSTM	4	73.1	27.7	26.2
BLSTM	4	75.8	30.8	28.7
LSTM	3	89.8	46.8	47.7
BLSTM	3	89.8	46.7	52.5

When predicting degradation in control levels C_M and C_H , the network could not accurately differentiate between the different signal degradation types. On average, it achieved a classification accuracy of around 25% (where random performance is also 25%). The user inputs are at a lower frequency in these two control levels, which causes the data to be sparse and have long periods of time without a command. This sparsity and time between commands is likely the primary issue with this approach. Thus, other methods will need to be explored in the future to determine when the signal has degraded when in control levels C_M and C_H .

In summary, we have developed a general model able to classify the signal of users whose data has not yet been observed, which performs well under low-level shared-control paradigms. Also, we see that the bidirectional model does not provide much improvement in the 3-class formulation. Therefore, if computational power is a limiting factor, the unidirectional model provides comparable results.

6 Insights for a Dynamic Autonomy Allocation Framework

In human-robot teams, mobile in particular, signal quality and human attention, awareness and workload changes constantly. Thus, it is vital that the robot can detect when the user is hindered or if the autonomy cannot succeed in performing the desired task. With this knowledge, control can be allocated in real-time to either the human or robot, or some mix of the two. Knowing that the prediction of signal degradation type is feasible for a low-level control command, we will use the results of the experiment to provide insight into when and how autonomy should be allocated for use in a dynamic autonomy allocation framework.

When the communication channel is dropping signals, the human-robot team should shift away from a low-level shared-control paradigm. In the low-level paradigm, the operator can continuously correct their commands to adjust the robot's behavior. The more often the signal is dropped, the less often the user can correct the behavior of the robot, leading to both performance and safety decreases (Fig. 4). Based on our results, the autonomy should shift when the dropped rate surpasses a threshold between the low and moderate modulation amounts.

A reduction in transmission frequency requires the design to prioritize either safety or performance. Our results show that low-level control provides the best performance when the channel frequency drops a lot to the high modulation frequency level. Conversely, safety is significantly improved by operating at moderate levels in a mid-level shared-control paradigm. If a reduced channel frequency is detected and the level is known, the autonomy can shift between C_M and C_L . If the level is unknown, the designer will need to prioritize safety or performance to decide which paradigm to use.

The autonomy should shift from high-level control when the communication channel is noisy. Lack of continuous correction may also have impacted the safety in the high-level shared-control paradigms when afflicted by a noisy communication channel. Here, the user provides only a goal for the mobile robot, and noise may place the goal closer to an obstacle. Moreover, we found that noise did not affect performance or safety in C_L (despite many subjects expressing a less enjoyable experience). Thus, if avoiding hazards is a critical component of the function of the robot, detecting noise and moving to a paradigm that allows for more operator correction may prove helpful.

7 Conclusion

The experimental results demonstrate the need for a framework that can dynamically allocate autonomy between the user and robot to optimize both performance and safety. Based on an analysis of the data, some control levels are explicitly better than others under certain degraded states of the communication channel between the human and the robot. We hypothesize that the rate at which a user can send corrective signals—which is dictated by the specific shared-control paradigm—explains these findings. Additionally, the experimental results suggest that a designer may need to choose between safety and performance when the transmission frequency is lowered. The results provide insight that can inform the design of a framework to dynamically adjust the control level when the quality of the signal changes in real-time. The first step in the design of such framework is to identify the degradation state of the signal. When in lower-level control, RNN LSTMs can reliably predict the state of the communication channel. However, more work is needed for classifying the signal in the other shared-control paradigms. This work lays the foundation for a framework that will be able to optimize the safety and performance of patients using assistive devices as well as human mobile robot teams.

Acknowledgements This work was supported by grant from U.S. Office of Naval Research under the Award Number N00014-16-1-2247, which we gratefully acknowledge. The authors would also like to thank Enid Montague for her guidance with the distraction task.

References

1. Argall, B.D.: Modular and adaptive wheelchair automation. In: Proceedings of International Symposium on Experimental Robotics (ISER) (2014)
2. Beer, J., Fisk, A.D., Rogers, W.A.: Toward a framework for levels of robot autonomy in human-robot interaction. *J. Hum.-Robot Interact.* **3**(2), 74 (2014)
3. Broad, A., Schultz, J., Derry, M., Murphrey, T., Argall, B.: Trust adaptation leads to lower control effort in shared control of crane automation. *IEEE Robot. Autom. Lett.* **2**(1), 239–246 (2017)
4. Chen, J.Y., Barnes, M.J.: Human-agent teaming for multirobot control: a review of human factors issues. *IEEE Trans. Hum.-Mach. Syst.* **44**(1), 13–29 (2014)
5. Chiou, M., Stolk, R., Bieksaite, G., Hawes, N., Shapiro, K.L., Harrison, T.S.: Experimental analysis of a variable autonomy framework for controlling a remotely operating mobile robot. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3581–3588 (2016)
6. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **18**(5), 602–610 (2005)
7. Hoff, K.A., Bashir, M.: Trust in automation integrating empirical evidence on factors that influence trust. *Hum. Factors: J. Hum. Factors Ergon. Soc.* **57**(3), 407–434 (2015)
8. 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)
9. Kaber, D.B., Onal, E., Endsley, M.R.: Design of automation for telerobots and the effect on performance, operator situation awareness, and subjective workload. *Hum. Factors Ergon. Manuf.* **10**(4), 409–430 (2000)
10. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
11. Lodwich, A.: Differences between industrial models of autonomy and systemic models of autonomy. [arXiv:1605.07335](https://arxiv.org/abs/1605.07335) (2016)
12. Miller, W.D.J.: The U.S. Air Force-Developed Adaptation of The Multi-Attribute Task Battery for the Assessment of Human Operator Workload and Strategic Behavior. Air Force Research Laboratory (2010)
13. Saeidi, H., Wang, Y.: Trust and self-confidence based autonomy allocation for robotic systems. In: Proceedings of IEEE Conference on Decision and Control (CDC), pp. 6052–6057 (2015)
14. Wheelchair Skills Program: Wheelchair Skills Test (WST) Version 4.2 Manual (2013)
15. Yang, S., Zhang, J.: An adaptive human-machine control system based on multiple fuzzy predictive models of operator functional state. *Biomed. Signal Process. Control* **8**(3), 302–310 (2013)
16. Yoo, H.S., Lee, P.U., Landry, S.J.: Detection of operator performance breakdown as an automation triggering mechanism. In: Proceedings of IEEE/AIAA Conference on Digital Avionics Systems Conference (DASC), pp. 3D3–1 (2015)
17. Zhang, J.H., Qin, P.P., Raisch, J., Wang, R.B.: Predictive modeling of human operator cognitive state via sparse and robust support vector machines. *Cogn. Neurodyn.* **7**(5), 395–407 (2013)

LEAF: Using Semantic Based Experience to Prevent Task Failures

Nathan Ramoly, Hela Sfar, Amel Bouzeghoub and Beatrice Finance

Abstract Using service robots at home is becoming more and more popular in order to help people in their life routine. Such robots are required to do various tasks, from user notification to devices manipulation. However, in such complex environments, robots sometimes fail to achieve one task. Failing is problematic as it is unpleasant for the user and may cause critical situations. Therefore, understanding and preventing failures is a challenging need. In this paper, we propose LEAF, an experience based approach to prevent task failure. LEAF relies on both semantic context knowledge through ontology and user validation, allowing LEAF to have an accurate understanding of failures. It then uses this new knowledge to adapt a Hierarchical Task Network (HTN) in order to prevent selecting tasks that have a high risk of failure in the plan. LEAF was tested in the Hadaptic platform and evaluated using a randomly generated dataset.

1 Introduction

Nowadays, we are facing an emergence of use of robots and smart homes. As there is a growing need for domestic health-care, in particular for elderly people, service robots provide a welcomed help. During their everyday routine, such robots perform various tasks, from reminding the user to take his/her medicine to using some devices. However, in home environment, multiple problems can block the robots' task plan.

N. Ramoly (✉) · H. Sfar · A. Bouzeghoub
SAMOVAR, Telecom SudParis, CNRS, Paris-Saclay University, Evry, France
e-mail: nathan.ramoly@telecom-sudparis.eu

H. Sfar
e-mail: hela.sfar@telecom-sudparis.eu

A. Bouzeghoub
e-mail: amel.bouzeghoub@telecom-sudparis.eu

B. Finance
DAVID Laboratory, University of Versailles Saint-Quentin-en-Yvelines,
Versailles, France
e-mail: beatrice.finance@uvsq.fr

For example, it may encounter a breakdown or have to difficulty to properly understand the context. In a nutshell, the robot is likely to encounter Failure Situations in its plan.

Overcoming failures is a common problem in robotics. Most works tackle this issue by reacting to it [8, 17]. In those cases, whenever the robot fails, it understands the cause and tries to find an alternative solution by generating a new plan. However, doing so is energy and time consuming [6] and it delays the reach of the goal. By proactively avoiding a failure instead of reacting, the robot can be quicker and more efficient to reach its goal, which is essential for domestic health-care application. To do so, the robot has to understand the cause of the failure and adapt its planning when it encounters them again. Some works have addressed this issue [9, 16], but the constraints of the home environment induce more challenges. For such application, it is essential to satisfy the user needs and to consider various and highly semantic data.

In this paper, we propose a solution to learn failure causes, evaluate them and prevent further failure called LEAF (Learning, Evaluating and Avoiding Failures). LEAF aims to learn failures' causes from previous encountered situations in order to prevent repeating them in future plans. It uses reasoning on semantic based context knowledge as well as a user validation to ensure the efficiency and accuracy of cause identification. LEAF also adjusts the planning phase to generate failure-free plans.

The main contributions of the paper are as follows:

1. A semantic model to represent and store situations that enables reasoning and interoperability with other systems.
2. A method to extract the causes of failures that includes the user in the loop to guarantee the quality of causes identification.
3. An improvement of HTN planner that takes into account the detected causes and to select safer sub-plans (i.e. avoiding task failures).

The remaining of the paper is divided as follows. Section 2 illustrates our needs through a scenario. Section 3 reviews the related works and points out their limits. Section 4 presents definitions and notions required for the understanding of our work. The contributions are described in Sect. 5, while the experiments are addressed in Sect. 6. Finally Sect. 7 concludes the paper.

2 Motivating Scenario

In order to motivate our proposal, let us consider this scenario:

Scenario 1: Nono, a service robot, operates in the home of a user named Katleen. Her house is equipped with various sensors, including motion sensors, microphones and Katleen's smart phone. One of the main role of Nono is to remind Katleen to take her medicine. Based on a schedule, the robot decides when it must vocally alert Katleen. Whenever it has to do so, it generates a plan to achieve this task. Nono has two options to alert Katleen. Firstly, if it knows the location of Katleen, it can

Table 1 History of task ‘vocal alert’

Sit.	Status	Start	End	Observed context data (failure causes are bold , <i>inferred data are italic</i>)
S_1	Failure	18/03/17 18:30:50	18/03/17 18:40:55	(katleen isLocatedIn livingroom), (katleen isDoing music), (livingroom hasSoundLevel 5db), (nono hasVolumeLevel 30db)
S_2	Failure	18/03/17 20:10:30	18/03/17 20:15:30	(katleen isLocatedIn livingroom), (katleen isDoing tv), (livingroom hasSoundLevel 75db), (nono hasVolumeLevel 30db), (katleen vocUnreachTo nono)
S_3	Success	19/03/17 10:30:50	19/03/17 10:34:15	(katleen isLocatedIn livingroom), (katleen isDoing tv), (livingroom hasSoundLevel 40db), (nono hasVolumeLevel 30db)
S_4	Failure	19/03/17 11:00:50	19/03/17 11:03:55	(katleen isLocatedIn livingroom), (katleen isDoing music), (livingroom hasSoundLevel 20db), (nono hasVolumeLevel 30db)
S_5	Success	19/03/17 15:06:35	19/03/17 15:08:55	(katleen isLocatedIn bedroom), (katleen isDoing reading), (livingroom hasSoundLevel 25db), (nono hasVolumeLevel 30db)
S_6	Failure	20/03/17 10:15:50	20/03/17 10:18:00	(katleen isLocatedIn livingroom), (katleen isDoing phoning), (livingroom hasSoundLevel 65db), (nono hasVolumeLevel 30db), (katleen vocUnreachTo nono)

go directly to her and talk to her. This is the prior solution as it is direct and has impact on the user. Secondly, if it doesn’t know her location, it can send a message to her phone. For both cases, the user is expected to provide an acknowledgement. However, sometimes Nono tries but fails to vocally alert the user. This can be due to various causes. Let us consider two examples of failure situations: (1) Katleen is listening to music through her phone with headsets. Therefore, she may not hear the voice of the robot; (2) the room may be filled with noise, for example from television or phone discussions. Hence, if the volume of the robot is set to a low value, Katleen may not hear Nono. After executing several times the task ‘vocal alert’, Nono obtains a history of situations for this task, represented in Table 1.

The aim of this work is to prevent the failure situations. To achieve this, we aim to identify causes, using history, in order to prevent task execution when failure causes are observed again. In scenario 1, Nono shall be able to identify context data (katleen isDoing music) (katleen vocUnreachTo nono) as failing causes for the task ‘vocal alert’. Thus, if a situation $S = (\text{katleen isLocatedIn livingroom}), (\text{katleen isDoing music}), (\text{livingroom hasSoundLevel 20db}), (\text{nono hasVolumeLevel 30db})$ occurs, the robot would understand that the task ‘vocal alert’ is likely to fail, thus would opt for the text message solution. By doing so, Nono avoids wasting time going to the user and trying to alert Katleen.

3 Related Work

Understanding and explaining plans and failures is an active research direction. In this section, we discuss some of the works that are related to our proposition.

The work proposed by Hanheide et al. [8] aims to provide a global planning solution for robots operating in an open and uncertain environment, such as a smart home. They address the issue of explaining task failure. To do so, they compare the actual and expected context observations. The idea is to discover what particular unexpected data caused the failure of the task. To do so, the robot generates a dedicated plan and relies on a diagnostic knowledge. Afterwards, the robot is able to make a new plan that avoids the identified causes. Although this technique is able to adapt the plan by understanding the missing elements, it does not learn from previous failures in order to prevent the occurrence of future failures.

The proposition of Sariel and Kapotoglu [9, 16] shares similar objective from our proposal. To the best of our knowledge, it is the main work about learning causes from previous experiences to prevent future failure. In this approach, based on previous failed situations, the robot is able to determine failure causes and avoid using tasks that are expected to fail. To do so, the authors use Inductive Logic Programming (ILP), an experiential learning framework that builds an experience by deriving hypothesis from failure situations. Context observations are stored and labeled as success or failure. The hypotheses are then adjusted and associated with a probability. A low probability implies there was a lot of ambiguity. With these hypotheses, the planning process is adjusted to prevent future failing situations. This technique improves the efficiency of planning for the object manipulation case study.

However, such a solution faces numerous issues. First, this solution relies on a simple model that does not include any reasoning. Reasoning would allow to infer further data from already acquired context data. For instance, in scenario 1, if the volume difference between the robot and the ambient noises is too low, the user can not be vocally reached. Sariel’s solution however, does not consider ‘relations’ between context data as possible failure. Secondly, depending on the situations available in the history, this solution can be biased. For example, it may identify two context data as cause while only one actually explains the failure. Lastly, by encountering redundant, yet non related to failure, context data, the solution of Sariel et al. can identify wrong causes of failure.

To overcome these limitations, we propose a system, named LEAF, to identify failures. Firstly (i), LEAF relies on an ontological representation, that enables reasoning on context data. Furthermore (ii), it considers each context data independently to prevent bias. In fact, by doing so, non related context data are not associated to potential real cause, as it is in [9, 16]. Indeed, if a situation can be explained by multiple context data, each of them is considered as a potential cause. Moreover, (iii) it relies on user validation. By relying on user’s feedback, our solution ensures the quality of the identification of the failure causes. Finally, (iv) it provides an adjustment of the HTN planner to take into account the causes in the planning process. The

overall proposition is detailed in Sect. 5. Additionally, the next section introduces some background and definitions for a better understanding.

4 Background

4.1 Ontology

An ontology is generally defined as representation of a shared conceptualization of a particular domain. It can easily be shared across people and application systems. It relies on the Resource Description Framework (RDF) [11]. An ontology is described through a set of RDF triples (*subject, predicate, object*). The set of all triples can be seen as a graph where nodes are concepts or instances and vertices are predicates among them. Using ontologies has multiple benefits, such as the inference of further data. In our work, we use the ontology in order to model Context Data (CD, Definition 1), Situation (ST, Definition 2), and Causes of Failure (CE, Definition 3).

4.2 Definitions

In this section, we define the concepts required for comprehension of our work.

Definition 1 CONTEXT DATA (CD): a context data is a piece of information about the environment provided by robots' sensors, environment's sensors (smart devices) or a knowledge base. Formally, it is a 4-tuple (*subject, predicate, object, t*). *subject* is an entity of the environment (i.e. user, robot or thing (physical object)), *object* is a context data about the *subject*, *predicate* is the relation between the *subject* and the *object*, and *t* is the timestamp of observation of this contextual data.

A CD is modeled as a RDF triple annotated with *t*. We distinguish two types of CD: High level CD and Low level CD. Low level CD are numeric observables generated from sensors; while high level CD are symbolic observables at the appropriate level of abstraction to make sense.

Property 1 *An activity is a high level CD where subject can be either a user or a robot and predicate is equal to “isDoing”.*

Example 1 In Table 1, for S_1 , (katleen isDoing music) is a high level context data obtained through a complex activity recognition process. While (katleen isLocatedIn livingroom) is a low level context data given by, possibly, a single sensor.

Definition 2 SITUATION (ST): A situation is a set of CD that have occurred at a given time interval. In this work, we are interested in the situation during one robot's task. A situation can be seen as a ‘snapshot’ of the state of the environment during

one task. Formally, we define a situation as a 5-tuple $(\{CD_i\}, status, task, t_s, t_e)$. $\{CD_i\}$ is a set of context data captured during a time interval $[t_s, t_e]$; $task$ is the task the robot was doing during this situation; $status$ is the outcome of the execution of the $task$, it is set to “null” when the situation is created and is set either to “success” or “failure” after the task execution; t_s is the start time of the execution of the $task$ and t_e is the end time of its execution. In the rest of the paper, we refer to situations as either *failure situation* or *success situation* when their status are respectively equal to failure or success.

Definition 3 CAUSE (C): A cause is a CD that fully or partially explains a failure situation. Formally, we define a cause as a couple (CD, ST) where CD is the context data that causes the failure of the situation ST .

All observed situations are stored in an history H . We denote the history for one task t as H_t . We denote all the causes of one task t as C_t . Each task relies on an ontology O_t that includes both H_t and C_t .

Example 2 let us consider this situation: $\{(katleen, isLocatedIn, kitchen, 15 : 01 : 10), (katleen, isDoing, music, 15 : 3 : 44)\}$, Failure, Alert, 15:00:52, 15:03:00). This is a failure situation caused by the ‘music’ activity of the user.

4.3 Planner

In this work, we rely on the Hierarchical Task Network (HTN) planner. HTN aims to find a *solution* to a *planning problem* by decomposing *tasks* into *subtasks*. HTN relies on two types of *tasks*: *primitive tasks* and *compound tasks*. *Compound tasks* are realized by *subtasks*, while *primitive tasks* are ‘ready-to-run’ non-decomposable tasks. The result of the planning process is called *solution*, and is a totally ordered set of *primitive tasks*. A *method* indicates how to decompose a compound task a sequence of *subtasks*, primitive or compound, based on preconditions. HTN planning consist in selecting a method for each compound task. The selection is perform by checking the validity of method’s preconditions. For example in scenario 1, if Katleen’s location is known, meaning there is a predicate (katleen isLocatedIn room), the robot can use the ‘vocal alert’ branch. For a more detailed overview and definition of HTN please refer to dedicated works [5]. HTN was selected for various reasons. Firstly, it offers good performance by having a reduced search space compared to other planners, such as STRIPS-like solutions [3]. Moreover, it is a popular planner as it is used for several applications [5, 14], particularly in robotics [10, 13, 15, 17]. Furthermore By using HTN, LEAF has the possibility to be easily integrated with some of these works.

5 Proposition

In this section, each component of our contribution, LEAF, is presented. It relies on three independent steps, depicted in Fig. 1, namely:

1. **Acquiring Situations**: The robot acquires situations and stores them in the history whenever a task ends.
2. **Extracting Failure Causes**: Causes are extracted based on the history and validated by the user.
3. **Enhancing Planner**: The extracted causes are then taken into account to enhance the planning process through an upgraded HTN.

5.1 Acquiring situations

The first step of our approach is to acquire the current situation. Whenever the robot finishes a task t , it stores the corresponding situation in an ontology. This ontology was designed from a previous work [1] and enhanced with new concepts and relations, such as *Activity*. The observed CD are inserted in the ontology under the Context Data concept. An exemplary representation of a subset of a situation can be found in Fig. 2. The property of the *Situation* concept are also stored in the ontology (not represented in Fig. 2)

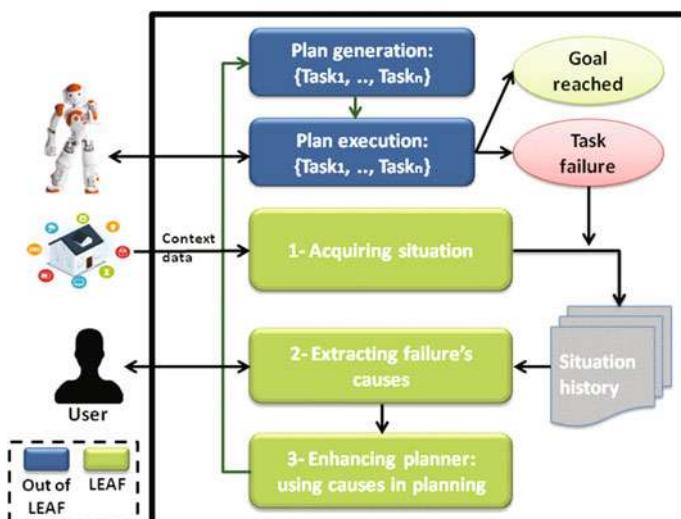


Fig. 1 The architecture of LEAF

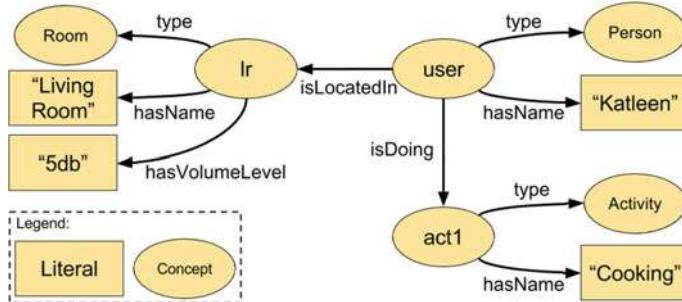


Fig. 2 Ontological representation of a subset of \$S_1\$

Once the situation is acquired, a rule based reasoning is applied to infer new data. It is essential to have a complete understanding of the context. These rules are provided by an expert. For example, let us consider situation \$S_2\$ in Table 1 where (katleen vocUnreachTo noho) is inferred by applying the following rule using Jena's formalism¹:

```
ruleVocUnreachable: (?user isLocatedIn ?room) ∧ (?room hasSoundLevel ?sndLvl)
∧ (?robot hasVolumeLevel ?robotLvl) ∧ difference(?sndLvl ?robotLvl ?soundDiff)
∧ greaterThan(?soundDiff 25) → (?user vocUnreachTo ?robot)
```

This rule states that if a user is in a room that is too loud for the current speaking volume of the robot, then the user cannot be reached vocally by the robot.

After applying these rules, more contextual data are generated. The enriched situation is then stored in the history \$H\$.

5.2 Extracting Failing Causes

Whenever a task fails, the robot tries to identify the causes of the failure.

In our approach, based on the history of situations and the previous identified causes, the robot identifies possible causes and asks for confirmation from the user. Through the involvement of the user in the process, we aim to provide high quality learning that is compliant with the user. The process of extracting the failing causes is composed of three steps: (1) Selecting data to be validated by the user: the robot selects context data (possible causes) that are to be validated by the user according to to the robot's current needs for the learning process. (2) Requesting user validation: the robot requests the user to provide validation about the selection of context data.

¹<https://jena.apache.org/documentation/inference/>.

(3) Getting user feedback: the robot receives the user feedbacks and updates its knowledge accordingly. The following subsections are reviewing each steps.

5.2.1 Selecting Context Data to Be Validated by the User

In order to build its experience, the robot requires user validation concerning possible causes. The first step is to identify what are the possible causes to be validated. In other words, the objective is to extract some context data that may be possible causes of the failure. This selection is important to quickly identify the causes without wasting user's validations. We consider that the robot should ask only a few validations from the user in order not to disturb him/her. This selection of causes has two modes: cold start and warm process. They are described below.

Cold start:

LEAF is subjected to the problem of cold start or cold boot. In fact, despite the main process relying on the history and previously identified causes, it also needs a procedure to start the learning process without any prior knowledge. Thus, the selection of context data to be validated is particular for the first encountered situations. The cold start procedure is applied once a minimum number of failing situations are encountered to a task t . In our experiments, we launched the procedure after 3 failure situations. The principle of the cold start procedure is to compute a ‘causality’ score for each context data based on its occurrence in the situations belonging to the history of the task. For one given context data cd , the ‘causality’ score $score_{cd}$ is computed as expressed in Eq. 1.

Let $Ssucc_{cd}$ be the set of all successful situations in the history H_t that contains the context data cd : $Ssucc_{cd} \subset H_t, \forall S \in Ssucc_{cd}$ where $cd \in S$. Let $Sfail_{cd}$ be the set of all failure situations in H_t : $Sfail_{cd} \subset H_t, \forall S \in Sfail_{cd}$ where $cd \in S$. Consequently, $|Ssucc_{cd}|$ and $|Sfail_{cd}|$ are respectively the number of success situations and the number of failure situations containing the cd , and $|Sfail_{cd} + Ssucc_{cd}|$ is the total number of situations in H_t containing cd . Finally, the $score_{cd}$ is computed as follows:

$$score_{cd} = (|Sfail_{cd}| - |Ssucc_{cd}|) / (|Sfail_{cd} + Ssucc_{cd}|) \quad (1)$$

For example, let us suppose that the robot has just encountered the situation S_4 in Table 1 and has previously encountered S_1 , S_2 and S_3 . The context data cd_1 = (katleen isLocatedIn livingroom), by applying Eq. 1, will have the causality score: $score_{cd_1} = (3 - 1) / (3 + 1) = 0.5$, since there was 3 failure situations and 1 success situation. On the other hand, the piece of context data cd_2 = (katleen isDoing music), which is a failure cause for the current situation, will have a causality score: $score_{cd_2} = (2 - 0) / (2 + 0) = 1$. The context data with the highest causality scores are then selected to be checked by the user. In the previous example, cd_2 would be selected over cd_1 for validation.

Warm process:

Once the cold start procedure is executed, the robot has some initial experience and can use the new knowledge to identify new causes. Whenever the robot is facing a

failure situation, it has two options for selecting context data to be validated as a cause by the user: either already identified causes or not registered context data that may be novel causes. The first option consolidates its knowledge while the other allows to explore new possible causes. It is important to remember that the number of validations performed by the user is limited. Therefore, this process can be described as a **multi-armed bandit problem** [12]. Multi-armed bandit solvers aim to maximize reward by efficiently choosing between exploration, i.e. using resources to explore new possibilities of gain, or exploitation, i.e. ensure gain by using resources from reliable sources. In our case of study, checking an already encountered cause corresponds to the exploitation phase, while checking a new possible cause corresponds to the exploration.

Our proposition is to use a multi-armed bandit approach to select the cause to be asked to the user. The selection of the strategy to choose depends on the context. In fact, if the robot is used to succeed the execution of a particular task and it fails to achieve this task for the first time, this means that probably there is a new cause of failure. Hence, the robot should prioritize exploration of a new cause. On the other hand, if the robot fails multiple times in a row, it implies the robot's current knowledge of causes is not accurate, hence the robot should focus on adjusting its knowledge and exploit.

In this work, we are using a variation of R-UCB [2], that is an improvement of the Upper Confidence Bound (UCB) algorithm [4]. UCB is a well known solution for tackling multi-armed bandit problems. It allows to select the context data with the higher upper confidence bound when exploiting. In other words, when the robot observes multiple causes, it selects the most relevant one by applying the following formula described in Eq. 4. In UCB, the selection between exploration and exploitation is performed randomly by following a fixed rate. R-UCB improves the UCB by adapting the exploration/exploitation rate according to the current ‘risk’. In R-UCB, the selection rate ε is dynamic and depends on the risk: the higher the risk is, the less the exploration is performed. In this work, we use the notion of reliability instead of the risk. The more the robot has previously failed, the lower the reliability is. If the reliability is low, more exploitation is required as the current knowledge of cause is not good enough to prevent failure. LEAF uses the Eq. 2 for computing ε :

$$\varepsilon = \varepsilon_{max} - (1 - R) * (\varepsilon_{max} - \varepsilon_{min}) \quad (2)$$

where R is the reliability that represents the success rate of a task t over the past N situations. R is computed through Eq. 3:

$$R = nbr\text{Succ}_N / N \quad (3)$$

where $nbr\text{Succ}_N$ is the number of successful situations in the past N situations in H_t . For instance, for $N = 4$, with the H_t presented in Table 1, thus considering S_6 , S_5 , S_4 and S_3 , we obtain $R = 2/4 = 0.5$. By using this process, LEAF is able to efficiently balance exploration and exploitation. **When exploring**, a random context data in the situation, that was not previously identified as a cause, is selected. **When**

exploiting, the R-UCB algorithm selects the context data with the highest upper confidence bound d_{cd} . d_{cd} represents the confidence in the selection of cd according to its current CB, its occurrence and the number of feedback already provided by the user. A high d_{cd} means cd needs to be checked in priority. It is computed as follows:

$$d_{cd} = CB_{cd} * \sqrt{\log(F_t)/N_{cd}} \quad (4)$$

where CB_{cd} is the current causality belief of context data cd (see Sect. 5.3.3 for belief computation), F is the number of failure situations for the current task t , and N_{cd} is the number of feedback provided by the user for causality of context data cd (for task t). For instance, let us consider that the robot is in the situation $S_{12} = (\text{katleen isLocatedIn livingroom}), (\text{katleen isDoing music})$ with the history H_t presented in Table 1 and $cd = (\text{katleen isLocatedIn livingroom})$; let us assume the user provided feedback eight times for cd , $N_{cd} = 8$, and that the resulting causality belief is $CB_{cd} = 0.75$, in that case: $d_{cd} = 0.75 * \sqrt{\log(4)/8} = 0.21$. The context data cd with the highest d_{cd} are the best candidate to be user checked.

5.2.2 Getting User Feedback

Once LEAF has determined what are the context data that require user validation about their causality, it requests feedbacks from the user. We consider five different user's answers ordered by confidence: {‘Yes’, ‘Probably’, ‘Partially’, ‘Possibly’, ‘No’}; each is respectively associated to a belief value: {1.0, 0.75, 0.5, 0.25, 0.0}. Based on these feedbacks, the asked context data is associated to a **causality belief** CB and stored in a knowledge base. This causality belief is set as follows: Let B be the belief value from the user's answer. If the context data is new and does not have causality belief CB_{cd} yet, its causality belief is set to B . If the context data was already identified as a cause and has a causality belief, the latter is updated using Eq. 5:

$$\text{new } CB = (\text{old } CB * N + B) / (N + 1) \quad (5)$$

where $\text{new } CB$ is the adapted causality belief of the context data and N is the number of previous user answers for this context data (for task t). Let us suppose that the robot asks the user for validation about the context data $cd_a = (\text{katleen isDoing music})$ after the failure of the situation S_4 . The user answers ‘Probably’, cd_a was not identified as a cause yet, thus $CB_{cd_a} = 0.75$ (cold start process starts after at least 2 failures). Now, let us suppose that the robot has encountered a similar situation some time later. However, this time the user is more confident and answers ‘Yes’. Hence, the causality belief of cd_a is updated as $CB_{cd_a} = (0.75 * 1 + 1) / (1 + 1) = 0.875$.

After analyzing user's answers, the robot has identified causes and associated each one with a causality belief. These beliefs are used in the extracted belief, where they can be seen as ‘rewards’. In addition, they are used to adapt the planning process in order to prevent further failures.

5.3 Checking Current Context

Once the causes are identified and associated to a causality belief, they are used in the planning process—HTN in our case. The objective is to take into account these causes in the planning process: if failure causes for a task t are observed, then t should be avoided in the plan. To do so, HTN was enhanced with an altered planning processes. Indeed, whenever HTN is decomposing, it not only checks the preconditions, but also the failing of subtasks. When a method has to be selected for a compound task ct in the planning phase, the following process is executed. We denote M_{ct} the set of all method realizing ct .

The first step of the process is to merge failure causes C_{t_i} of all subtasks t_i of method m . The merging is quite simple: all the identified causes of subtasks $t_i \in m$ are extracted from O_{t_i} and put in a set C_m that carries all failure causes of the method m . If a context data is a failure cause for multiple tasks in m , the maximum CB is selected. Please note that compound methods are not taken into account: they are tackled when the planning algorithm decomposes them. For example, regarding the task ‘alert’, the robot has two methods: m_1 = go to user, vocal alert and m_2 = phone alert. The robot has the history presented in Table 1, through the cause extraction process, for task ‘vocal alert’. Two causes were identified: cd_a = (katleen isDoing listeningMusic) and cd_b = (katleen vocUnreachTo nono) with $CB_{cd_a} = 0.875$ and $CB_{cd_b} = 1.0$. Thus, in that case C_{m_1} contains cd_a and cd_b .

Once C_m are generated for each possible method m in M_{ct} , methods’ conditions are checked. Methods whom conditions are not verified are excluded, the remaining set is labeled M_{ct}^* . The remaining methods’ failure causes are then checked. The idea is to check if the method would succeed in the current context. Let W be the current context (world). And let C_{obs} the set of currently observed causes $C_{obs} = W \cap C_m$. Let A be a constant. For each method $m \in M_{ct}^*$, a confidence value is computed through a following logistic differential presented in Eq. 6:

$$conf_m = 1 - 1/(1 + e^{-ln(A * \sum_{cd \in C_{obs}} CB_{cd})}) \quad (6)$$

Equation 6 allows to compute a $[0, 1]$ confidence value based on any number of causality belief. With a sum of causality belief of 0, the function return 1, meaning the task can be safely executed. The confidence then quickly decrease as the sum of causality belief increases and has an asymptote at 0. Thanks to this function, one cause with a high causality belief is enough to compute a low confidence, thus preventing execution. The more the causality is high, the lower the confidence is, this enables the comparison of tasks’ confidences. For example, if the robot is trying to generate a plan in situation S_{13} = (katleen isLocatedIn livingroom), (**katleen isDoing music**), for $A = 0.75$ confidence value of m_1 is: $conf_{m_1} = 1 - 1/(1 + e^{-ln(1*0.875)}) = 0.6$. In situation S_{14} (katleen isLocatedIn livingroom), (**katleen isDoing music**), (**katleen vocUnreachTo nono**), where two causes are observed, thus the task is more unlikely to succeed, for $A = 0.75$ confidence value of m_1 is: $conf_{m_1} = 1 - 1/(1 + e^{-ln(1*(0.875+1.0))}) = 0.41$.

Having the confidence values for each computed method in M_{ct}^* , the method with the highest confidence value is selected for decomposition. Within the standard HTN, the first method matching the conditions would have been selected. However, a final checking is done before decomposing. In fact, if the confidence value is below 25%, meaning there is 75% of chance the robot won't succeed, the decomposition is aborted. As all other methods are less reliable, the compound task ct is marked to be non-executable. Thus, HTN will backtrack and try to find another option. If a method is successfully selected, HTN can continue its process. In the end, a plan is generated and HTN has checked the possibilities of failures, minimizing the risk of encountering a failing situation.

6 Experiments

6.1 Implementation

LEAF has been implemented in Java and relies on Jena² to manage ontologies and reasoning. The core code of LEAF is available on github.³ The implementation is independent from any robots and/or smart environment and holds to the description in Sect. 5. Nevertheless, we integrated LEAF on a Nao robot [7] using ROS (Robot Operating System) Indigo.⁴ Nao is a small robot that is able to walk, identify persons and talk with users. It gathered user validation through vocal interaction thanks to the NaoQi API. The robot was equipped with a DHTN [15] planner. As a proof of concept, we tested our scenario in the Hadaptic⁵ platform, that includes a modular room and various sensors. Videos can be found online⁶ (Fig. 3).

6.2 Evaluation

In order to evaluate LEAF learning capabilities, it was applied on a randomly generated dataset. The principle of these evaluations is to simulate various situations and evaluate how LEAF is able to learn the causes and prevent failure situations. The evaluation process is the following. First, a situation is generated and associated with the expected results of tasks' execution. LEAF is then asked to check if a given task will fail or not in this situation. LEAF's answer is compared to the expected result: if it is correct, the situation is tagged as successful; otherwise, the situation is tagged

²<https://jena.apache.org/>.

³<https://github.com/Nath-R/LEAF>.

⁴<http://wiki.ros.org/indigo>.

⁵<http://hadaptic.telecom-sudparis.eu>.

⁶<http://nara.wp.tem-tsp.eu/what-is-my-work-about/leaf/>.

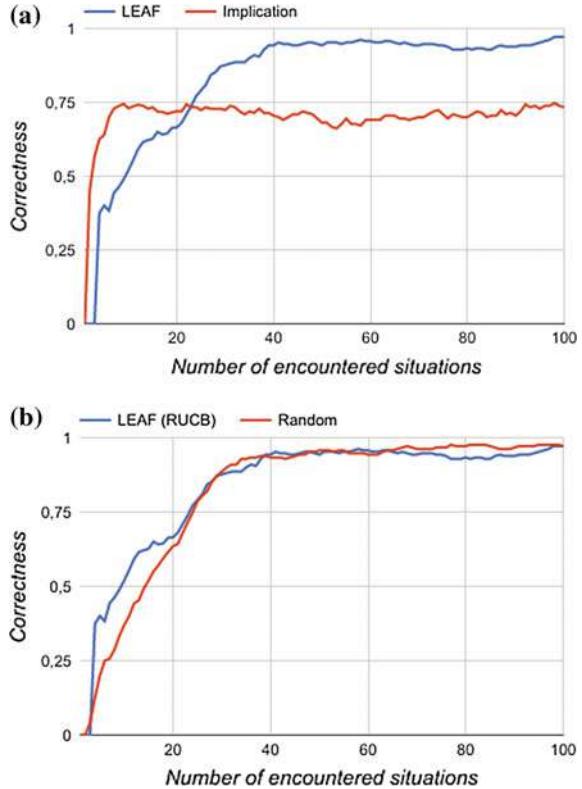
as failure and LEAF starts the learning process: it determines possible causes to be validated by the user and gets the feedbacks from a virtual user (no human in the simulation process). The process is then repeated on a new situation. The situation generation process consists in feeding the live situation with random, yet controlled, context data such as user's location, object position or user's activity. In our experiments, we considered the Scenario 1 and Scenario 3 possible failure causes. We also created 2 ontological rules, further could have been created, but would have been irrelevant in these experiments. One evaluation run is composed of 100 generated situations. For each step, the correctness is computed as the number of true positive and true negative answers based on the total number of answers. There was 20 runs conducted (for each variant, see below) and the average results are presented in Fig. 4. Experiments are divided in two parts: (a) comparing LEAF to the literature, (b) Evaluating RUCB contribution.

LEAF was compared to a *state-of-the-art* like approach in Fig. 4a. We used an implication based approach similar to the one proposed by Sariel et al. [9, 16]. It infers implication between data and task's outcomes based on previous encountered situations without the user being in the loop. Results show that the *state-of-the-art* approach learns quicker and reach its asymptote after 10 situations with a correctness of 75%. In the meantime, LEAF achieves only 50% of correctness for the

Fig. 3 Picture of the realized tests



Fig. 4 Correctness of task's risk evaluation according to the number of situations encountered. **a** Comparison to literature: LEAF versus implication based approach. **b** Comparison for CD validation: LEAF (RUCB) versus randomly selected cause



same amount of learning data. Although LEAF is slower to learn, it reaches a better correctness after 22 situations and reaches its asymptote after 40 situations with a correctness of 95%. Such a difference can be explained by the validation phase. In fact, it needs to have enough user feedbacks in order to reach efficient results. In these evaluations, three validations were asked per failing situation. This explains the slow increase compared to the implication based approaches. Nevertheless, by asking the user, LEAF reaches extremely precise results: when trained, LEAF outclassed the state-of-the-art approach by almost 20%. In fact, the implication based learning process suffers of all the limits described in Sect. 3. These experiments underline that LEAF is efficient once trained, but needs to be improved in order to learn quicker. Using a higher number of user validations when the robot is not experienced is a possible way to solve this shortcoming.

Determining what to ask to the user is essential for the efficiency of LEAF. Particularly, LEAF selects between exploration and exploitation, in other word, exploring new causes or improving already identified ones. RUCB (used by LEAF) was compared to a random data selection. Results presented in Fig. 4b show that RUCB allows for a quicker learning in comparison a randomly selected data: it is 10% more correct between 4 and 16 situations. After 20 situations, the two approaches obtain similar

correctness and, as expected, have the same asymptote around 95% (when experienced, both approaches are similar). RUCB efficiently selects the causes to validate, thus it learns quicker than the random based approach as it does not ‘waste’ questions to user. It proves RUCB to be pertinent to reach decent correctness faster.

7 Conclusion

In this paper, we presented a novel approach to prevent encountering failure situations and thus enhancing the efficiency of domestic robots. Our solution is based on a live learning process that analyses the failures. We described how failing situations were modeled and how failing causes were extracted using user validation. We enhanced HTN by enabling it to check failing risk when decomposing, ensuring sub-tasks success when executed. Furthermore, we implemented and tested our approach with a Nao robot and the Hadaptic smart platform. Results underline the validity of our contribution, however further experimentations in real case scenarios using the Evident⁷ platform, a fully equipped smart apartment, are to ought be performed.

References

1. Al-Moadhen, A., Qiu, R., Packianather, M., Ji, Z., Setchi, R.: Integrating robot task planner with common-sense knowledge base to improve the efficiency of planning. *Procedia Comput. Sci.* **22**, 211–220 (2013)
2. Bouneffouf, D.: Drars, a dynamic risk-aware recommender system. Ph.D. thesis, Institut National des Télécommunications (2013)
3. Fikes, R.E., Nilsson, N.J.: Strips: a new approach to the application of theorem proving to problem solving. *Artif. Intell.* **2**(3), 189–208 (1972)
4. Garivier, A., Moulines, E.: On upper-confidence bound policies for switching bandit problems. In: International Conference on Algorithmic Learning Theory, pp. 174–188. Springer (2011)
5. Georgievski, I., Aiello, M.: An overview of hierarchical task network planning (2014). [arXiv:1403.7426](https://arxiv.org/abs/1403.7426)
6. Ghezala, M.W.B.: Compréhension dynamique du contexte pour l'aide à l'opérateur en robotique. Ph.D. thesis, Institut National des Télécommunications (2015)
7. Gouaillier, D., Hugel, V., Blazevic, P., Kilner, C., Monceaux, J., Lafourcade, P., Marnier, B., Serre, J., Maisonnier, B.: Mechatronic design of nao humanoid. In: IEEE International Conference on Robotics and Automation, 2009 (ICRA'09), pp. 769–774. IEEE (2009)
8. Hanheide, M., Göbelbecker, M., Horn, G.S., Pronobis, A., Sjöö, K., Aydemir, A., Jensfelt, P., Gretton, C., Dearden, R., Janicek, M., et al.: Robot task planning and explanation in open and uncertain worlds. *Artif. Intell.* (2015)
9. Kapotoglu, M., Koc, C., Sariel, S.: Robots avoid potential failures through experience-based probabilistic planning. In: 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO), 2015, vol. 2, pp. 111–120. IEEE (2015)
10. Lallement, R., De Silva, L., Alami, R.: Hatp: An htn planner for robotics (2014). [arXiv:1405.5345](https://arxiv.org/abs/1405.5345)

⁷<https://evident.telecom-sudparis.eu>.

11. Lassila, O., Swick, R.R., et al.: Resource description framework (rdf) model and syntax specification (1998)
12. Mahajan, A., Teneketzis, D.: Multi-armed bandit problems. Foundations and Applications of Sensor Management pp. 121–151 (2008)
13. Milliez, G., Lallement, R., Fiore, M., Alami, R.: Using human knowledge awareness to adapt collaborative plan generation, explanation and monitoring. In: The Eleventh ACM/IEEE International Conference on Human-Robot Interaction, pp. 43–50. IEEE Press (2016)
14. Nau, D.S., Au, T.C., Ilghami, O., Kuter, U., Murdock, J.W., Wu, D., Yaman, F.: Shop2: An htn planning system. *J. Artif. Intell. Res. (JAIR)* **20**, 379–404 (2003)
15. Ramoly, N., Bouzeghoub, A., Finance, B.: Context-aware planning by refinement for personal robots in smart homes. In: Proceedings of ISR 2016: 47st International Symposium on Robotics, pp. 1–8. VDE (2016)
16. Sariel, S., Yildiz, P., Karapinar, S., Altan, D., Kapotoglu, M.: Robust task execution through experience-based guidance for cognitive robots. In: International Conference on Advanced Robotics (ICAR), 2015, pp. 663–668. IEEE (2015)
17. Weser, M., Off, D., Zhang, J.: Htn robot planning in partially observable dynamic environments. In: IEEE International Conference on Robotics and Automation (ICRA), 2010, pp. 1505–1510. IEEE (2010)

State Estimation and Localization for ROV-Based Reactor Pressure Vessel Inspection

Timothy E. Lee and Nathan Michael

Abstract A vision-based extended Kalman filter is proposed to estimate the state of a remotely operated vehicle (ROV) used for inspection of a nuclear reactor pressure vessel. The state estimation framework employs an overhead, pan-tilt-zoom (PTZ) camera as the primary sensing modality. In addition to the camera state, a map of the nuclear reactor vessel is also estimated from a prior. We conduct experiments to validate the framework in terms of accuracy and robustness to environmental image degradation due to speckling and color attenuation. Subscale mockup experiments highlight estimate consistency as compared to ground truth despite visually degraded operating conditions. Full-scale platform experiments are conducted using the actual inspection system in a dry setting. In this case, the ROV achieves a lower state uncertainty as compared to subscale mockup evaluation. For both subscale and full-scale experiments, the state uncertainty was robust to environmental image degradation effects.

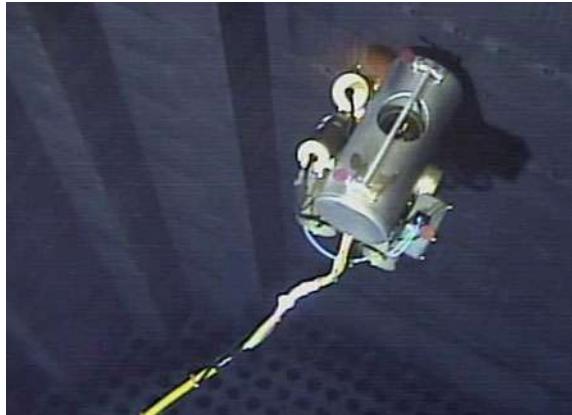
1 Introduction

We propose a vision-based state estimation and localization framework to enable submersible robots to conduct inspection of nuclear reactor pressure vessels. The framework is formulated as an extended Kalman filter (EKF) that is robust to sensor degradation and image corruption that may occur due to environmental effects, such as radiation and color attenuation. The proposed framework relies on a pan-tilt-zoom (PTZ) camera, fixed with respect to the vessel frame, that is autonomously controlled. To model the reactor vessel, we propose the use of a sparse map that concisely represents the vessel geometry as a series of planes and landmarks. The map is assumed to be known in advance with limited uncertainty arising from differences

T. E. Lee (✉) · N. Michael
Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh,
PA 15213, USA
e-mail: timothyee@cmu.edu

N. Michael
e-mail: nmichael@cmu.edu

Fig. 1 This submersible robot is used to inspect reactor pressure vessels. Note the planar structure of the vessel and the geometric features located on the walls and floor. The robot is equipped with three red fiducial markers that are used for pose estimation. A tether is used to transmit robot control signals from the control station



between engineering blueprints and construction. The map is estimated in the state to enable corrections based on projections of vessel landmarks (points and lines) in the camera image space.

A submersible robot that is used to inspect a nuclear reactor pressure vessel is shown in Fig. 1. The robot and vessel are monitored by an external PTZ camera, which is the primary sensing modality of the framework. The key advantage of using this camera is its zoom capability. High optical resolution images of the scene are still obtained via zoom despite the camera being positioned relatively farther from the reactor, which mitigates the adverse effects of radiation. Indeed, the existence of radiation in this environment restricts the use of other sensors for the framework. Significant radiation exposure excludes the use of localization sensors with sensitive electronics, such as inertial measurement units or depth sensors. Indeed, radiation-sensitive electronics can degrade and fail with exposure of a few krad [21], which is below the expected exposure dosage in this setting. The underwater setting excludes the use of GPS, and water attenuation excludes sensing depth using projective infrared light without the use of an external light source [25]. However, the underwater setting lacks turbidity, so vision-based perception is viable.

The use of vision for underwater robots has been studied both in laboratory experiments and in deployed field robots. Although submersible robots can utilize a range of sensing modalities [15], radiation exposure from the nuclear reactor restricts us to considering systems where vision is the primary sensing modality. A visual SLAM formulation with pose-graph optimization was utilized to construct a texture-mapped, three-dimensional model of a ship hull for inspection purposes [14]. An EKF state estimation formulation that includes vision and inertial measurements was found to be successful in underwater navigation of a submersible robot [22]. Another study demonstrated a localization solution for a AUV using acoustic sensors and visual odometry [7]. Our use of structural landmarks is similar to previous work in localizing an underwater robot in a structured environment using only visual perception

(an onboard camera) [4], but our study differs in that robot localization is achieved through a fixed, external camera.

Deployed field inspection robots that utilize vision have conducted subsurface bridge inspection [19] and ship hull inspection [11], with sonar imaging as the primary inspection modality in these cases. In the domain of nuclear reactor inspection, the use of cameras and robotics for inspection has been studied [18, 20]. A previous study estimated the x - and y -position and yaw angle of a submersible robot within a reactor vessel by observing eight LEDs located on the vehicle with an external camera (primarily using a depth sensor for the z -position) [5]; our work differs by estimating both the robot and camera pose with six degrees of freedom.

Regarding PTZ cameras, we note that the movement of a small unmanned system with a pan-tilt camera has been estimated using an EKF [8]. This study estimates the projection of the system in the camera image space, not in three dimensions as our framework does. Jain and Neumann [12] employ an EKF to estimate the pose and focal length of a PTZ camera.

2 System Overview

The robotic inspection system consists of a submerged PTZ camera that monitors a ROV operating in a reactor pressure vessel. The robot is equipped with three fiducial markers. Figure 2 illustrates the system and depicts three distinct reference frames:

1. the *body frame* $\{B\}$, located at the robot center of mass;
2. the *external camera frame* $\{E\}$, located at the optical center of the camera; and
3. the inertial *world frame* $\{W\}$, which is the reference frame for the vessel map.

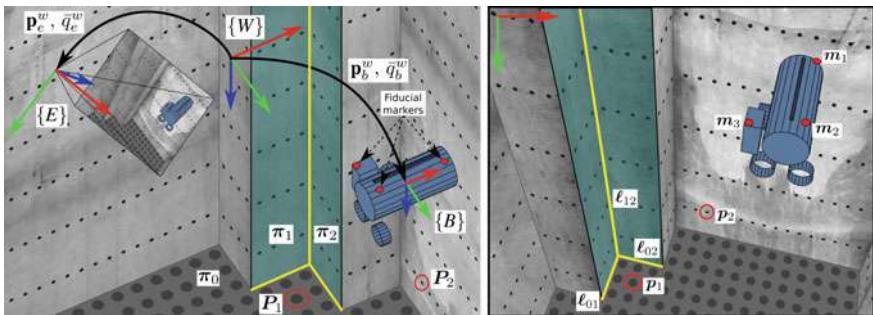


Fig. 2 System representation and landmarks: (left) in three dimensions; (right) in the two dimensional image space of the external camera. The intersections of vessel planes π_i and π_j yield Plücker lines that project as lines ℓ_{ij} . Similarly, three-dimensional points P_i project to two-dimensional points p_i in the image space

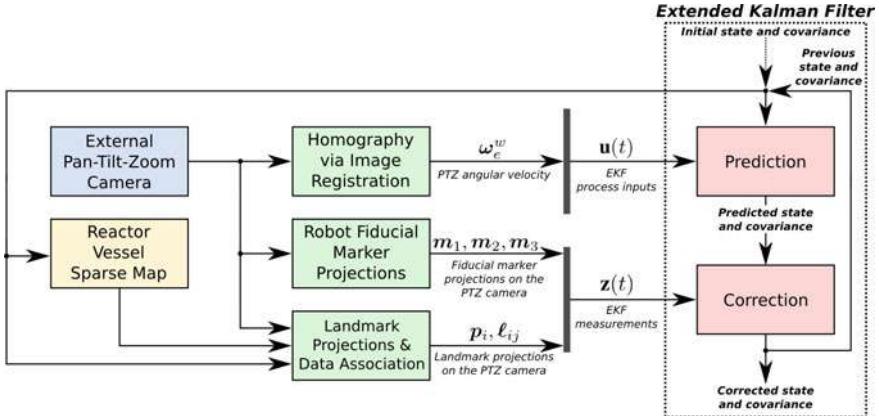


Fig. 3 The system diagram of the state estimation framework

The robot pose, camera pose, and camera focal length are estimated using an EKF. To account for uncertainties in the vessel geometry, the map representation of the vessel is also estimated in the state. The resulting framework is shown in Fig. 3.

The remainder of this section will address system models and methods: the external camera (Sect. 2.1), the submersible robot (Sect. 2.2), the map representation of the vessel (Sect. 2.3), a method for camera rotation inference via homography-based image registration (Sect. 2.4), and a method for incorporation of landmark projections into the sparse map (Sect. 2.5). The EKF formulation detailed in Sect. 3 will leverage these models and methods to enable ROV state estimation.

2.1 External PTZ Camera

A PTZ camera is utilized to monitor the robot and vessel during infrastructure inspection and is mounted to the vessel, external to the robot. The camera is controlled via visual servoing such that the robot is always in view with reasonable magnification. The PTZ images are used for inference of camera rotation (Sect. 2.4), camera-to-robot localization (Sect. 2.2), and camera-to-world localization using projections of structural landmarks in the image space (Sect. 2.5). We assume pinhole projection as the underlying camera model that relates a three-dimensional point in homogeneous coordinates $\tilde{\mathbf{P}} \sim [\mathbf{P}^T, 1]^T$ to its image projection in homogeneous coordinates $\tilde{\mathbf{p}} \sim [\mathbf{p}^T, 1]^T$, where R and \mathbf{t} represent the transformation from the point frame to the camera frame:

$$\tilde{\mathbf{p}} \sim K[\mathbf{R} | \mathbf{t}] \tilde{\mathbf{P}} \quad (1)$$

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

2.2 Submersible Robot and Fiducial Markers

The submersible robot (Fig. 1) is equipped with three fiducial markers to enable pose estimation from the marker projections in the camera image space. These projections ($\mathbf{m}_i = [u_i, v_i]^T$, $i = \{1, 2, 3\}$) provide corrections between the external camera and the robot frames. The markers are detected using the K-means clustering algorithm [1] and assigned based on the estimated robot pose.

The position of the markers ($\tilde{\mathbf{M}}_i^b$) with respect to the body frame $\{B\}$ is static and known from robot measurements. The marker positions provide the visual scale that is necessary to infer the three-dimensional pose of the robot from the marker projections. These projections arise in the external camera image space as follows:

$$\tilde{\mathbf{m}}_i \sim K [R_w^e | \mathbf{t}_w^e] T_b^w \tilde{\mathbf{M}}_i^b \quad (3)$$

In this model, T_b^w is the rigid body transformation matrix that relates points expressed in the body frame to the world frame, calculated from the robot pose estimate $(\mathbf{p}_b^w, \bar{q}_b^w)$. Similarly, the extrinsic calibration matrix $[R_w^e | \mathbf{t}_w^e]$ is determined from the pose estimate of the external camera $(\mathbf{p}_e^w, \bar{q}_e^w)$.

2.3 Sparse Map from Structural Elements

As shown in Fig. 1, the characteristic geometric appearance of the reactor pressure vessel structure can be described as a series of intersecting planes, with landmarks that exist on these planes. These three-dimensional geometric entities (planes and points) form two types of landmarks in the image space: lines and points. The vessel geometry is specified in the world frame $\{W\}$.

Each plane $\pi = [\bar{n}^T, d]^T \in \mathbb{R}^4$ is described by a unit normal vector \bar{n} and distance d . The three-dimensional line that arises from the intersection of two adjacent planes, π_i and π_j , is represented in Plücker coordinates $\mathcal{L}_{ij} = \pi_i \wedge \pi_j$, where $\mathcal{L} \in \mathbb{P}^5$.

The infrastructure contains landmarks, which are engineered structural elements such as relief holes, cavities, or bolts that can be represented as a three-dimensional point, \mathbf{P} . Specifically, we note the prominence of repeated point elements such as flow holes on the reactor core floor (Fig. 1, *c.f.* Fig. 5) and bolts (*c.f.* Fig. 8). These landmarks exist on a plane, as represented by the constraint $\pi \cdot \tilde{\mathbf{P}} = 0$.

2.4 Homography-Based Inference of Camera Rotation

To infer the change of the external camera rotation, a homography-based methodology is used that leverages image registration. The pixel coordinates of successive images from the external camera are mapped by a homography [23]:

$$\tilde{\mathbf{x}}' = H \tilde{\mathbf{x}} \quad (4)$$

where $\tilde{\mathbf{x}}$ is the homogeneous pixel coordinates, i.e., $\tilde{\mathbf{x}} = [u, v, 1]^T$. Because the external camera does not translate, image pixel displacements do not depend on scene structure [6]. Specifically, this homography H is the infinite homography H_∞ induced by the plane of rotation at infinity [10]. Between frames i and j , this homography has one of two structures, H_{static} and H_{rot} , depending on whether the camera is static or rotating, respectively:

$$H_{\text{static}} = I_{3 \times 3} \quad (5)$$

$$H_{\text{rot}} = K R_{ij} K^{-1} \quad (6)$$

The homography, H , is calculated via intensity-based image registration between consecutive frames. The resulting camera rotation R_{ij} is then used to drive the process model of the external camera in the state estimation framework.

2.5 Projections of Structural Elements

The projections of structural elements that compose the map are observable from the external camera image space (Fig. 2). Landmarks (points and lines) are used for correcting the state estimate by identifying and associating them in the image space. These projections are utilized for localization between the robot and the world, as well as for localization between the external camera and the world.

Landmarks are identified and associated first using feature detection based on geometric shape. Lines are detected using the Canny filter [3] and the probabilistic Hough transform [17]. Points are detected using blob detection and, in the case of circular elements, the Hough transform [26]. After detection, data association is performed by first projecting map elements into the image space, and comparing them against candidate detected landmarks. The closest detected landmark (within a heuristic threshold) is then associated to a projected landmark.

Three-dimensional points and their projections in the image space are related by

$$\tilde{\mathbf{p}}_i \sim K [R_w^e | \mathbf{t}_w^e] \tilde{\mathbf{P}}_i^w \quad (7)$$

For lines, the Plücker line $\mathcal{L}_{ij} = \boldsymbol{\pi}_i \wedge \boldsymbol{\pi}_j$ formed from the intersection of adjacent planes defined in the world coordinate frame are projected in the external camera image space as a line $\boldsymbol{\ell}_{ij} \in \mathbb{P}^2$:

$$\boldsymbol{\ell}_{ij} \sim \mathcal{K} L_w^e \mathcal{L}_{ij}^w \quad (8)$$

The matrix L_w^e is the rigid displacement matrix for lines, and \mathcal{K} is the perspective projection matrix for lines [2].

3 EKF Methodology

We estimate the non-linear state of the system (robot, camera, and map) using a discrete extended Kalman filter (EKF) with quaternions [16] to obtain a recursive state estimate. This state estimation framework leverages the algorithms described in Sect. 2 (Fig. 3) and requires as priors the position of the fiducial markers on the robot, the vessel geometry as obtained from blueprint specifications within a limited degree of uncertainty, and the camera radial and tangential distortion parameters, obtained via a previously known calibration. We assume from this point forward that the external camera images are unwarped following a transformation to reverse the lens distortion. To initialize the filter, we optimize the initial position of the external camera and focal length to minimize the reprojection error between the observed and predicted vessel landmarks. Thereafter, the initial pose of the robot may be estimated from the marker projections from constrained optimization.

3.1 System Parameterization

The non-linear state of the inspection system $\mathbf{X}(t)$ is estimated via an extended Kalman filter with inputs $\mathbf{u}(t) \in \mathbb{R}^3$ and a variable number of measurements $\mathbf{z}(t)$, depending on the scene. The system state encompasses the state of the inspection system (robot and external camera) $\mathbf{X}_S(t)$ and the map of the vessel \mathbf{X}_M :

$$\mathbf{X}(t) = [\mathbf{X}_S(t)^T, \mathbf{X}_M^T]^T \quad (9)$$

3.1.1 Inspection System State

The state of the system is represented by

$$\mathbf{X}_S(t) = [\mathbf{p}_b^w(t)^T, \bar{q}_b^w(t)^T, \mathbf{p}_e^{wT}, \bar{q}_e^w(t)^T, \mathbf{f}_e(t)^T]^T \quad (10)$$

where $\mathbf{p}_b^w(t) = [x_b^w(t), y_b^w(t), z_b^w(t)]^T$ and $\bar{q}_b^w(t)$ are the position and orientation (respectively) of the robot with respect to the world frame, $\mathbf{p}_e^w = [x_e^w, y_e^w, z_e^w]^T$ and

$\bar{q}_e^w(t)$ are the (static) position and orientation of the external camera with respect to the world, and $\mathbf{f}_e(t) = [f_x(t), f_y(t)]^T$ is the vector of focal length components.

3.1.2 Map State

The world structure is represented by a map in the state estimate \mathbf{X}_M that encompasses the vessel planes and three-dimensional landmarks that exist on these planes. Each plane $\pi = [\bar{n}^T, d]^T$ is described by the unit normal vector \bar{n} and distance d .

We propose a minimal representation for utilizing a map of the reactor in the EKF framework by extending the geometric representation described in Sect. 2.3. Assuming that the walls of the vessel are orthogonal to the floor, planes are specified by their rotational degree of freedom (θ) about the world z -axis and translational degree of freedom (d). Therefore, the unit normal for each wall is $\bar{n} = [\cos \theta, \sin \theta, 0]^T$. For the floor of the vessel, only the height of the vessel h is needed in the state, for $\bar{n} = [0, 0, 1]^T$ and $d = -h$. Therefore, if the vessel consists of N walls, $2N + 1$ parameters are needed for the planar structure of the vessel.

Although landmark points are three-dimensional, they all must exist on a plane. To enforce the coplanarity of a landmark with its associated plane, a point is represented by two translational degrees of freedom within in the plane, δ_1 and δ_2 , relative to the point $-\bar{n}d$, which is the point on the plane closest to the origin of the world frame $\{W\}$. These represent the two-dimensional position of the landmark in the two-dimensional subspace of \mathbb{R}^3 formed by π . When considering \bar{n} as one axis of an orthonormal basis of the plane, the other two axes are $\bar{v}_1 = [-\sin \theta, \cos \theta, 0]^T$ and $\bar{v}_2 = [1, 0, 0]^T$ for walls, and $\bar{v}_1 = [1, 0, 0]^T$ and $\bar{v}_2 = [0, 1, 0]^T$ for the floor.

A landmark's three-dimensional position in the world frame $\{W\}$ can be recovered from its coincident plane and two-dimensional position within this plane:

$$\mathbf{P} = -\bar{n}d + \delta_1\bar{v}_1 + \delta_2\bar{v}_2 \quad (11)$$

It follows from Eq. 11 that the coplanarity constraint of the landmark, $\pi \cdot \tilde{\mathbf{P}} = 0$, is always satisfied for any choice of θ , d , δ_1 , or δ_2 .

With this minimal geometric representation, the map state $\mathbf{X}_M \in \mathbb{R}^{2N+1+2L}$ for N planes and L points is as follows:

$$\mathbf{X}_M = [\theta_1, d_1, \dots, \theta_N, d_N, h, \delta_{1,1}, \delta_{1,2}, \dots, \delta_{L,1}, \delta_{L,2}]^T \quad (12)$$

3.2 Process Models

The system process model characterizes the temporal evolution of the state. The process input, $\mathbf{u}(t)$, consists of the angular velocity of the external camera with respect to its reference frame, $\omega_e^w(t)$. The entire process model expressed in continuous time is

$$\dot{\mathbf{p}}_b^w(t) = 0 \quad (13)$$

$$\dot{\mathbf{q}}_b^w(t) = 0 \quad (14)$$

$$\dot{\mathbf{p}}_e^w = 0 \quad (15)$$

$$\dot{\mathbf{q}}_e^w(t) = \frac{1}{2} Q(\boldsymbol{\omega}_e^w(t)) \bar{\mathbf{q}}_e^w(t) \quad (16)$$

$$\dot{\mathbf{f}}_e(t) = 0 \quad (17)$$

$$\dot{\boldsymbol{\pi}} = 0 \quad (18)$$

$$\dot{\mathbf{P}} = 0 \quad (19)$$

These continuous time update equations are converted into discrete time using Euler discretization. We model the robot's state evolution as being driven forward by a random walk. For the external camera process, the position of the external camera \mathbf{p}_e^w is static. The external camera rotates with an average angular velocity $\boldsymbol{\omega}_e^w(t)$ determined from the output of the homography (Sect. 2.4). $Q(\boldsymbol{\omega}(t))$ is the quaternion kinematic matrix [13] that relates angular velocity in a body-referenced frame and quaternion orientation to quaternion rate. Lastly, the map is static as it is defined in the world frame $\{W\}$.

3.3 Measurement Models

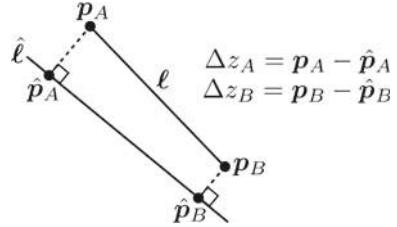
All system measurements $\mathbf{z}(t)$ consist of projections into the external camera image space. The measurements can be categorized into two types: 1) $\mathbf{z}_b^e(t)$, relating the robot body frame $\{B\}$ to the external camera frame $\{E\}$; and 2) $\mathbf{z}_e^w(t)$, which relates the external camera frame $\{E\}$ to the world frame $\{W\}$:

$$\mathbf{z}(t) = [\mathbf{z}_b^e(t)^T, \mathbf{z}_e^w(t)^T]^T \quad (20)$$

The body-to-external-camera measurements, $\mathbf{z}_b^e(t)$, are determined through robot fiducial marker detection (Sect. 2.2):

$$\mathbf{z}_b^e(t) = [\mathbf{m}_1(t)^T, \mathbf{m}_2(t)^T, \mathbf{m}_3(t)^T]^T \quad (21)$$

Fig. 4 Quantifying error between a predicted line $\hat{\ell}$ and an observed line segment ℓ . Points \hat{p}_A and \hat{p}_B are the closest points on line $\hat{\ell}$ to points p_A and p_B , respectively, that define line segment ℓ



Projections of structural elements (Sect. 2.5) provide observations for external-camera-to-world localization and robot-to-world localization. While the number of marker corrections is fixed while the robot is in view, the number of landmark corrections will vary depending on the scene. All measurements assume $\sigma = 3$ noise.

The predictions for these measurements, $\hat{\mathbf{z}}(t)$, utilize an ideal projective camera model as detailed in Sect. 2. For points, the correction model is simply the predicted landmark projection in the image space. For lines, we adopt the line error formulation as shown in Fig. 4, which is based on the distance from each point of the detected line to its nearest point on the predicted line [24].

4 Results

We conduct experiments to demonstrate the correctness, accuracy, and robustness of the state estimation framework. Experimental datasets are representative of actual infrastructure for which the framework was designed. We pursue experiments of two different types: (1) camera experiments with a subscale mockup infrastructure system; and (2) platform experiments using the inspection system with a to-scale reactor vessel mockup.

The robustness of the state estimate is assessed against speckling, which is radiation-induced chromatic image noise (Fig. 5). Speckling is characterized by the random occurrence of clusters of pixels to become activated with a high color intensity that persists for only one frame. A probabilistic speckling model was quantified using 549 frames from an inspection dataset with speckling. Table 1 shows the distribution for the number of speckles per frame, n_{frame} (normalized by total number of pixels) and size of the speckle in pixels, s_{size} . All experimental datasets were processed twice: (1) “clean” (no speckling); and (2) “degraded,” with artificial speckling and color attenuation to emulate the environmental image effects that are expected when deployed in a nuclear reactor vessel.

Fig. 5 Speckling (radiation-induced chromatic image noise) observed during a reactor pressure vessel inspection. Speckle clusters are circled in yellow for ease in viewing

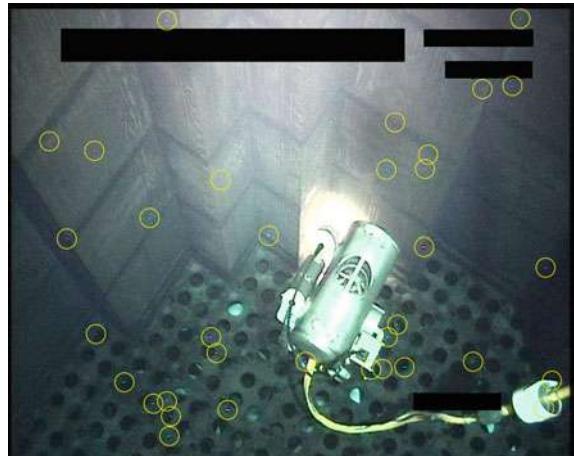


Table 1 Speckling model

Parameter	Value
$n_{frame} \sim \mathcal{N}(\mu, \sigma^2)$	
μ	6.4655e-5
σ^2	3.0493e-10
$s_{size} \sim Cat(K_i, p_i)$	
$i = \{1, \dots, 6\}, K_i = i$	
p_1	0.4034
p_2	0.4087
p_3	0.1021
p_4	0.0542
p_5	0.0200
p_6	0.0116

4.1 Camera Experiments with a Subscale Mockup Structure

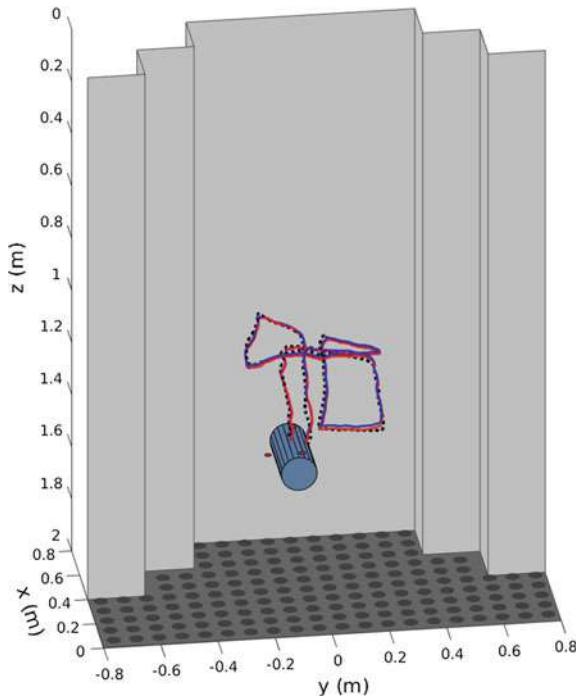
We perform camera experiments using a subscale mockup system (Fig. 6) that is designed to replicate the geometry of a generic reactor pressure vessel on a smaller scale. A frame with markers is used as a mockup for the submersible robot. An Axis V5915 PTZ camera is used for the external camera. We use a Vicon motion capture system to obtain ground truth pose measurements of the robot. We calibrate the external camera assuming a projective pinhole model and radial and tangential distortion coefficients using the Kalibr calibration toolbox [9].

In this experiment, the robot is translated in a motion that is representative of inspection robot motion. The results of state estimate are shown in Fig. 7 and Table 2. From ground truth, we calculate that the framework has mean-squared error (MSE)



Fig. 6 Subscale mockup system: (far left) experimental setup shown with the subscale mockup of a reactor pressure vessel; (middle left) external camera; (middle right) subscale mockup of the inspection robot; (far right) image of the subscale system from the external camera. Note the complete lack of visual texture on the structure

Fig. 7 Path of the robot relative to the subscale mockup structure. Shown are the estimated paths for the clean (red) and degraded (blue) cases. The ground truth path (black) is from motion capture



in position of under $2.9e-4 \text{ m}^2$ in x , $3.1e-4 \text{ m}^2$ in y , $1.7e-3 \text{ m}^2$ in z . Using ZYX Tait-Bryan Euler angles, the angular position MSE is under $3.8e-4 \text{ rad}^2$ in roll, $1.3e-3 \text{ rad}^2$ in pitch, and $6.7e-4 \text{ rad}^2$ in yaw. Qualitatively, good agreement is observed between the ground truth and estimated paths as shown in Fig. 7, and the state estimate is shown to be robust to images degraded by environmental effects.

The uncertainty of the state estimate is also shown in Table 2. The $\pm 3\sigma$ uncertainty for the robot within the xy -plane is under 4.7 cm and under 10 cm vertically. For the external camera, the lateral $\pm 3\sigma$ uncertainty is 0.5 cm and 4.2 cm vertically.

Table 2 Accuracy and uncertainty for subscale experiments

Parameter	Value (Clean)	Value (Degraded)
Accuracy (MSE), m^2 or rad^2		
x_b^w	2.7515e-4	2.8637e-4
y_b^w	3.0282e-4	3.0809e-4
z_b^w	1.6192e-3	1.6982e-3
θ_e^w	3.6473e-4	3.7681e-4
ϕ_e^w	1.1025e-3	1.2869e-3
ψ_e^w	6.0344e-4	6.7328e-4
Uncertainty ($\pm 3\sigma$), m or rad		
Robot, $\{B\}$		
x_b^w	0.0435	0.0434
y_b^w	0.0174	0.0175
z_b^w	0.0961	0.0960
θ_b^w	0.1361	0.1358
ϕ_b^w	0.1101	0.1118
ψ_b^w	0.0527	0.0520
External camera, $\{E\}$		
x_e^w	0.0028	0.0037
y_e^w	0.0021	0.0034
z_e^w	0.0393	0.0424
θ_e^w	0.0053	0.0056
ϕ_e^w	0.0048	0.0051
ψ_e^w	0.0162	0.0172

The relatively higher error and uncertainty in the z -direction for both the robot and the external camera is a direct result of the subscale vessel. The subscale vessel by design has no landmarks or visual texture on the walls, which is representative

**Fig. 8** External camera images from platform testing: (left) clean and (right) degraded with artificial environmental image effects (speckling and color attenuation)

of the most challenging types of reactor vessels for this system. In contrast with this vessel, the to-scale mockup shown in Fig. 8 contains landmarks on the walls that improve localization in the z -direction. For this reason, we expect that observing wall landmarks or the top edge of the reactor will improve the uncertainty in this dimension. Nonetheless, we note that the error and uncertainty in the xy -plane are still suitable for coarse localization of the robot within the vessel.

4.2 Platform Experiments with Inspection System

Next, we demonstrate the capability for estimating the state of the inspection system platform. We perform motion experiments with the inspection system platform hoisted by a crane and translated relative to a (to-scale) reactor vessel quarter mockup (Fig. 8).

Fig. 9 Estimated path of the inspection robot relative to a quarter mockup of a reactor vessel for the clean (red) and degraded (blue) cases

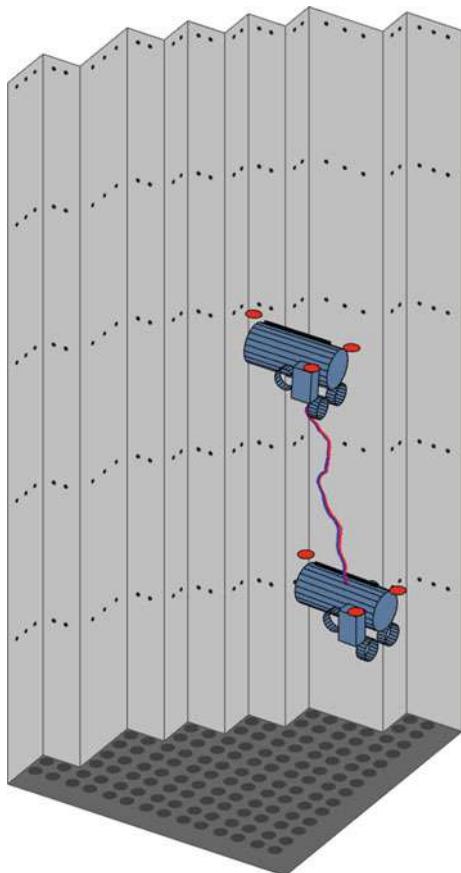


Table 3 Uncertainty for platform experiments

Parameter	Value (clean)	Value (degraded)
Uncertainty ($\pm 3\sigma$), m or rad		
Robot, $\{B\}$		
x_b^w	0.0085	0.0084
y_b^w	0.0161	0.0162
z_b^w	0.0241	0.0237
θ_b^w	0.0687	0.0727
ϕ_b^w	0.0395	0.0406
ψ_b^w	0.0214	0.0225
External camera, $\{E\}$		
x_e^w	0.0031	0.0031
y_e^w	0.0036	0.0036
z_e^w	0.0166	0.0190
θ_e^w	0.0035	0.0039
ϕ_e^w	0.0033	0.0035
ψ_e^w	0.0092	0.0097

We present the results for a 26 s test where the robot was translated vertically by approximately 1.42 m. The external camera rotates during this experiment to keep the ROV in view. Figure 9 shows the estimated path of the robot for this test. As shown in Table 3, the state estimation framework estimates the pose of the robot with sufficient uncertainty for coarse localization of the robot within the vessel. Specifically, we note that within the xy -plane the robot uncertainty ($\pm 3\sigma$) is under 1.8 and 2.4 cm vertically. For the external camera, the $\pm 3\sigma$ uncertainty is under 0.5 cm within the xy -plane and under 1.9 cm vertically, with total rotational uncertainty (in terms of Euler angles) to be approximately 0.01 rad.

We note that the uncertainty estimates were lower overall for the platform experiments as compared to the subscale mockup experiments, due to utilizing wall landmarks on the vessel mockup. Additionally, as in the subscale mockup experiments, we observe that the framework is robust to image degradation effects, with little significant effect on state uncertainty. Although ground truth position data is not available for this experiment, cross-referencing the estimated position against images from a camera installed on-board the robot suggests good agreement between the actual and estimated path.

5 Conclusion and Future Work

In this work, we have proposed a state estimation and localization framework designed for coarse localization of a submersible robot within a nuclear reactor

pressure vessel that primarily utilizes a PTZ camera. We have proposed a map representation for reactor pressure vessels that models the structure as a series of orthogonal planes, with structural points of interest that exist on the planes. The intersection of vessel planes project to lines in the camera image space. These lines, as well as the points of interest on the planes, serve as landmarks for correcting the state estimate. The rotational motion of the camera is inferred from intensity-based homography.

We have shown that the proposed framework is suitable for coarse localization by conducting two types of experiments. First, we confirmed the accuracy of the filter for localizing the robot with respect to a challenging vessel with no visual texture or wall landmarks. Second, we validated the framework using the actual inspection system, showing that the estimated path uncertainty ($\pm 3\sigma$) is under 1.8 cm in the xy -plane and 2.4 cm vertically. For the camera, the position uncertainty ($\pm 3\sigma$) is under 0.5 cm in the xy -plane and under 1.9 cm vertically, with total rotational uncertainty ($\pm 3\sigma$) of about 0.01 rad. We verified that our framework is robust to the environmental image effects (speckling and color attenuation) that are expected to degrade the system sensing when operating in the reactor vessel.

Our current work has shown the capability of this framework for state estimation and localization to enable ROV-based inspections of nuclear reactor vessels. For future work, we will investigate how image-based registration could also be used to model variations in zoom setting of the PTZ camera. Additionally, we will pursue an automated EKF initialization procedure that bootstraps the filter with minimal effort from inspection personnel. We will also pursue testing of our framework in real-time during an inspection of a reactor pressure vessel.

Acknowledgements We gratefully acknowledge support from Westinghouse Electric Company, LLC.

References

- Arthur, D., Vassilvitskii, S.: K-means++: the advantages of careful seeding. In: Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, pp. 1027–1035 (2007)
- Bartoli, A., Sturm, P.: The 3d line motion matrix and alignment of line reconstructions. *Int. J. Comput. Vis.* **57**, 159–178 (2004)
- Canny, J.: A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **6**, 679–698 (1986)
- Carreras, M., Ridao, P., García, R., Nicosevici, T.: Vision-based localization of an underwater robot in a structured environment. In: Proceedings of the IEEE International Conference on Robot and Automation, vol. 1, pp. 971–976 (2003)
- Cho, B.H., Byun, S.H., Shin, C.H., Yang, J.B., Song, S.I., Oh, J.M.: Keprov: Underwater robotic system for visual inspection of nuclear reactor internals. *Nucl. Eng. des.* **231**(3), 327–335 (2004)
- Collins, R.T., Tsin, Y.: Calibration of an outdoor active camera system. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1 (1999)
- Corke, P., Detweiler, C., Dunbabin, M., Hamilton, M., Rus, D., Vasilescu, I.: Experiments with underwater robot localization and tracking. In: Proceedings of the IEEE International Conference on Robot and Automation, pp. 4556–4561 (2007)

8. Doyle, D.D., Jennings, A.L., Black, J.T.: Optical flow background estimation for real-time pan/tilt camera object tracking. *Measurement* **48**, 195–207 (2014)
9. 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, pp. 1280–1286 (2013)
10. Hartley, R., Zisserman, A.: *Multiple View Geometry in Computer Vision*. Cambridge University Press (2003)
11. Hover, F.S., Eustice, R.M., Kim, A., Englot, B., Johannsson, H., Kaess, M., Leonard, J.J.: Advanced perception, navigation and planning for autonomous in-water ship hull inspection. *Int. J. Robot. Res.* **31**(12), 1445–1464 (2012)
12. Jain, S., Neumann, U.: Real-time camera pose and focal length estimation. *IEEE International Conference on Pattern Recognition*, vol. 1, pp. 551–555 (2006)
13. Kelly, J., Sukhatme, G.S.: Visual-inertial sensor fusion: localization, mapping and sensor-to-sensor self-calibration. *Intl. J. Robot. Res.* **30**(1), 56–79 (2011)
14. Kim, A., Eustice, R.: Pose-graph visual slam with geometric model selection for autonomous underwater ship hull inspection. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1559–1565 (2009)
15. Kinsey, J.C., Eustice, R.M., Whitcomb, L.L.: A survey of underwater vehicle navigation: recent advances and new challenges. In: IFAC Conference of Manoeuvering and Ctrl of Marine Craft, vol. 88 (2006)
16. LaViola, J.J.: A comparison of unscented and extended kalman filtering for estimating quaternion motion. In: Proceedings of the American Control Conference, vol. 3, pp. 2435–2440. IEEE (2003)
17. Matas, J., Galambos, C., Kittler, J.: Robust detection of lines using the progressive probabilistic hough transform. *Comput. Vis. Image Underst.* **78**(1), 119–137 (2000)
18. Mazumdar, A., Lozano, M., Fittery, A., Asada, H.H.: A compact, maneuverable, underwater robot for direct inspection of nuclear power piping systems. In: Proceedings of the IEEE International Conference on Robot and Automation, pp. 2818–2823 (2012)
19. Murphy, R.R., Steimle, E., Hall, M., Lindemuth, M., Trejo, D., Hurlebaus, S., Medina-Cetina, Z., Slocum, D.: Robot-assisted bridge inspection. *J. Intell. Robot. Syst.* **64**(1), 77–95 (2011)
20. Odakura, M., Kometani, Y., Koike, M., Tooma, M., Nagashima, Y.: Advanced inspection technologies for nuclear power plants. *Hitachi Rev.* **58**(2), 82–87 (2009)
21. Shea, H.R.: Effects of radiation on mems. In: Proceedings of SPIE, vol. 7928, pp. 79,280E-1–79,280E-13 (2011)
22. Shkurti, F., Rekleitis, I., Scaccia, M., Dudek, G.: State estimation of an underwater robot using visual and inertial information. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5054–5060 (2011)
23. Sinha, S.N., Pollefeyns, M.: Pan-tilt-zoom camera calibration and high-resolution mosaic generation. *Comput. Vis. Image Underst.* **103**(3), 170–183 (2006)
24. Sola, J., Vidal-Calleja, T., Civera, J., Montiel, J.M.M.: Impact of landmark parametrization on monocular ekf-slam with points and lines. *Int. J. Comput. Vis.* **97**(3), 339–368 (2012)
25. Tsui, C.L., Schipf, D., Lin, K.R., Leang, J., Hsieh, F.J., Wang, W.C.: Using a time of flight method for underwater 3-dimensional depth measurements and point cloud imaging. In: IEEE OCEANS Conference, pp. 1–6 (2014)
26. Yuen, H., Princen, J., Illingworth, J., Kittler, J.: Comparative study of hough transform methods for circle finding. *Image Vis. Comput.* **8**(1), 71–77 (1990)