Planning and Localization Using Contacts

Bradley L. Saund

April 2017

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Howie Choset, Co-chair
Reid Simmons, Co-chair
Matt Mason
Arun Srivatsan

Submitted in partial fulfillment of the requirements
for the degree of Masters of Computer Science.

Abstract

Contacting the world can provide stability, support, and sensory information, however many robots avoid contact whenever possible. Contacts present the physical danger of large forces as well as challenging computational issues, but enabling robots to reason about contacts extends the extent to which robots can perceive and act in the world.

This thesis explores both localization and planning where contacts are required. Computational issues arise due to the local nature of contacts, yielding subspaces of intersection thin manifolds, so several common techniques in robotics are adapted to better handle the local effects of contact A particle filter is augmented to update from a precise measurement that would ordinarily eliminate most particles. An efficient information gain metric is defined using these particles to predict useful future measurements. A new dynamics model and associated cost function are created for a robotic arm, which although are not faithful to real dynamics, are conditioned well for use in existing planning methods. This artificial but useful dynamics model is shown in both trajectory optimization and sampled-based planning. The techniques are demonstrated in simulation and physical hardware.

Acknowledgments

This thesis and my work in graduate school would not be possible without the collaboration and guidance of many intelligent people. First, I would like to thank my advisors. Both Howie and Reid have provided me with support and guidance, and most importantly have forced me to justify my theories and guided me to many new concepts in robotics. Their advising styles were different, but worked amazingly well together, and I am a smarter person thanks to their influence.

I would like to thank the senior PhD students who I sat near and from whom I absorbed knowledge. Discussions with Glenn, Arun, and Chao frequently inspired me. In addition, discussions and help from Guillaume, Alex, Matt, Julian, Ky, Elena, Breelyn, and many others have both directly influenced my work and generally improved my robotics knowledge. The former graduate students now at Hebi designed and provided the hardware that first inspired my snake arm planning goals and they have continued to support and improve the hardware.

I also would like to thank Shiyuan Chen, which whom I have extensively collaborated over the course of this thesis and whose work was heavily used in the localization section of this thesis.

I owe so much to my parents for providing me with a scientifically rich childhood, always encouraging me to excel.

And of course I would like to thank the love of my life, Katie Brennan, for agreeing to two years of long distance during this masters program.

Contents

1 Introduction **1**
1.1 Motivation . . . 2
1.1.1 Jigs are More Expensive Than Robots . . . 2
1.1.2 Robots Stay on the Outside . . . 3
1.2 Approach . . . 3
1.2.1 Localization . . . 3
1.2.2 Motion Planning . . . 5

2 Related Work **7**
2.1 Current Methods for Localizing Objects Using Contact Sensors . . . 7
2.1.1 Particle Filters . . . 7
2.1.2 Kalman Filters . . . 8
2.2 Contacts in Planning . . . 9
2.2.1 Bracing, Climbing, and Walking . . . 9
2.2.2 Sample based planning . . . 10
2.2.3 Trajectory Optimization . . . 11

3 Localization using Contacts **13**
3.1 Rigid-Body Object Localization . . . 14
3.1.1 Measurement Model . . . 15
3.1.2 Problems with the standard particle filter . . . 15
3.1.3 Rejection Sampling Method . . . 15
3.1.4 Fast Evaluation of Sampled States . . . 16
3.2 Datum Based Particle Filter . . . 19
3.2.1 Datum Representation . . . 19
3.2.2 Geometric Relationships . . . 20
3.2.3 Independent-State Representation . . . 20
3.3 Predicting Effective Measurement Actions . . . 22
3.3.1 Information Gain . . . 22
3.3.2 Information for Full-State Representation . . . 25
3.3.3 Information for Independent-State Representation . . . 26
3.4 Experiments . . . 26
3.4.1 Robot Results for a Rigid Body . . . 26
3.4.2 Simulation Results for the Datum Particle Filter . . . 28

4 Planning with Contacts for Support **31**
4.1 Robot Dynamics 31
4.2 Trajectory Optimization 32
4.2.1 Auxiliary Variables 33
4.2.2 Smoothing the Cost Function 34
4.2.3 Initialization Challenges 36
4.3 Sample Based Planning 37
4.3.1 Adaptation to Encourage Contacts 37
4.4 Experiments 41

5 Conclusion and Future Work **43**

Bibliography **45**

List of Figures

1.1 Humans using contact to improve reach, accuracy, and stability 1
1.2 Robots working on the outside of parts held firmly in jigs 2
1.3 Workers crawling in the confined spaces of an airplane wing 3
1.4 **(a)**: The robot performing a touch measurement 4

2.1 Constrained Bidirectional Rapidly Exploring Random Tree (CBiRRT) extending the tree onto a measure-zero valid manifold in configuration space 10

3.1 Visualization of the belief of the pose of all sections of the part **(a)**: The prior belief of the poses before localization. The uncertainty of the goal feature is too high to perform the task. **(b)**: The belief of the poses after performing measurements to localize the goal feature. The pose of the goal feature is now known well enough to perform the task. Although the bottom edge (purple) and perpendicular section still have noticeable error, precise localization of these features is not needed. 14
3.2 Comparison of the time and accuracy of the particle filter update step when using Adaptive Bandwidth and Adaptive Sample Size 18
3.3 Visualization of independent-state particle filter **(a)**: Side view of the CAD drawing with dimensions (simplified for clarity). This drawing indicates the nominal distance between the top and bottom edge is 0.23m, with a symmetric tolerance of 5mm. This drawing also defines a hole with a 1cm diameter, and the top edge as its vertical datum and side edge as its horizontal datum. **(b)**: The beliefs of the top (green) and right (blue) edges of the part are shown. The true part location is shown in gray. The measurement (arrow) on the top section partially localized the top edge. For clarity in the image, the belief of the other sections are not shown, and only 50 of the 500 particles are shown. **(c)**: A following measurement (arrow) on the right edge further localizes the part. This measurement provides information on the right edge directly, and the top edge indirectly. . . . 21
3.4 Binning of potential measurement on particles for use in calculating information gain. The three arrows represent the nominal measurement action $\mathcal{M}$ with simulated deviation δ_j. The horizontal lines divide the measurement values into the numbered bins. 23
3.5 Heat maps showing information gain 25
3.6 Translational and rotational error during localization on the physical robot 27

3.7 Comparison of the accuracy of the update step when using full-state particle filter and independent-state particle filter 29

4.1 The visually accurate robot model (left) and the spherical approximation (right) . 32
4.2 Optimizing the shortest path trajectory from an initial path converges to a local best, but possibly misses the globally best solution. 32
4.3 Naive cost function, with sharp changes around contact locations 33
4.4 Contact distance for the end effector section 34
4.5 Arm configuration and simulated (not realistic) contact forces for a variety of arm configurations and continuous contact variables 35
4.6 The cost function for the arm in Figure 4.3, augmented with the auxiallary contact variable 36
4.7 Trajectory optimization finding a local minimum (left) and the global minimum (right) for an arm attempting to reach the blue "X" 37
4.8 Illustration of valid configuration space for an arm potentially supported by contacts 38
4.9 Illustration of Policy RRT extension on the contact manifold 40
4.10 Simulated (left) and actual (right) snake arm robots executing motions planned in a model of an airplane wing section 41

Chapter 1

Introduction

Contacting the world can provide stability, support, and sensory information. Humans benefit from touching the world in situations from blindly detecting an object in the back of the refrigerator, to leaning on a stair railing. In the recent DARPA robotics challenge top robotics research teams from around the world submitted robots to attempt an obstacle course, and during these challenges many of these robots fell on the stairs. Not a single robot used the railing [3]. Even drunk people know to use railings, so there is plenty of room for improvement in robotics.

However, in robotics there are good reasons to avoid contact with the environment. Hitting the world can provide large forces that destabilize or break the robot. Relying on contacts can be dangerous as slight alterations in the world or errors in robot position may drastically alter the contact forces. Instead of firmly grasping a handle, a few centimeters of error have caused robots to embarrassingly grasp air and fall over. Even when working entirely in simulation contacts between rigid bodies cause problems for physics engines.

Contacts introduce non-smooth discontinuities, which presents a challenge for many tools in the roboticists arsenal. The benefits of contacts occur on a measure-zero manifold in the robot's configuration space. For example, hand rests comfortably on a table at a specific configuration of shoulder, elbow, and wrist positions. A slight extension of the elbow while fixing all other joints leads to the hand puncturing the table, while a slight contractions pulls the hand into the air and the table might as well not exist. Random sampling and gradient descent are two techniques

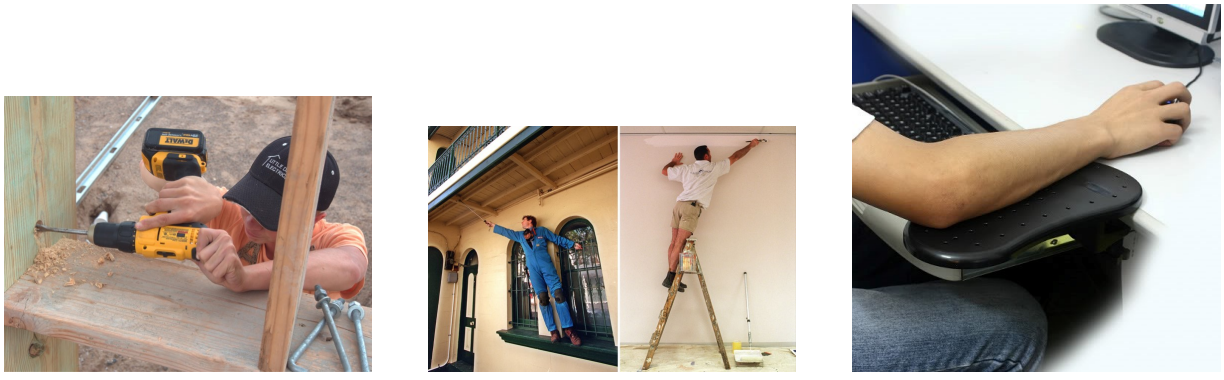


Figure 1.1: Humans using contact to improve reach, accuracy, and stability

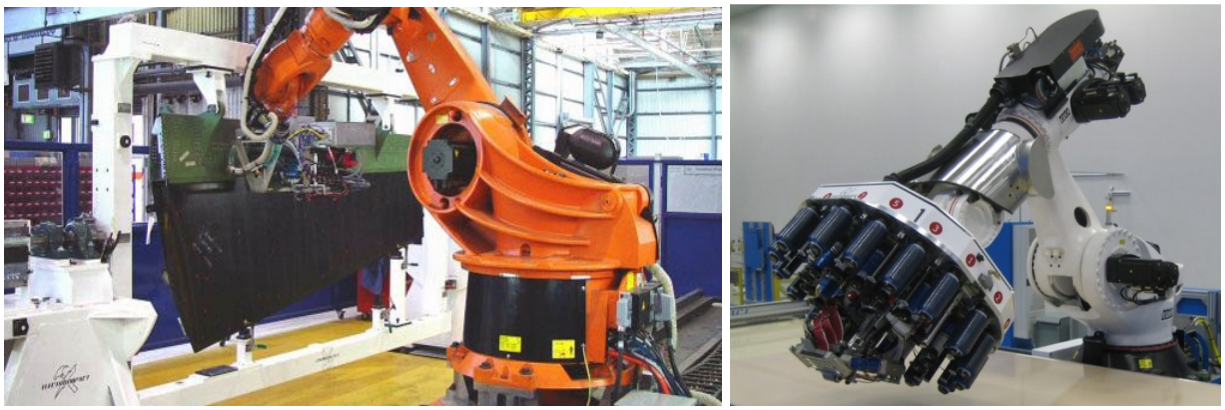


Figure 1.2: Robots working on the outside of parts held firmly in jigs

repeatedly used in robotics, but both have difficulty with these measure-zero contact manifolds. This thesis explores and extends techniques of sampling based and gradient methods to the problems of localization and planning, with the addition of contact forces and measurements.

1.1 Motivation

My personal motivation for the topics covered in this thesis come, in part, from my past experience as a robotics engineering building robots that build airplanes. Currently, large robotic arms equipped with expensive, specialized end effectors perform a variety of tasks for aerospace manufacturing, including drilling holes, inserting fasteners, milling, carbon fiber placement and layup, and sealant application. While there are many potential directions of research to improve these already impressive machines, two striking issues are addressed by my work.

1.1.1 Jigs are More Expensive Than Robots

Robots that perform one type of task on one section of an airplane can cost millions of dollars. However the jig that holds the airplane section while the robot works can cost more than the robot, partically because these jigs may have more actuators and tighter tolerances than the robotic system. When the robot begins work on a new section, before ever making a measurement the jig has already located the part to within a few inches. A few scripted measurement with a probe, programmed by an engineer, are sufficient to fully localize the part to within the needed tolerances. When amortized over many airplanes the per-part cost drops. However, this process is inflexible and poorly suited to low-rate manufacturing.

If the robot could localize the part from a wide range of part configurations, and if the robot could reason about internal assembly tolerances then jigs could be make more cheaply, robots would become more versatile, less part-specific programming would be required, and machining accuracy could improve.

Figure 1.3: Workers crawling in the confined spaces of an airplane wing

1.1.2 Robots Stay on the Outside

Large robotic arms cannot fit in the confined spaces of an aircraft. While a team of people may include workers on both the inside and outside, large robots are limited to just the outside.

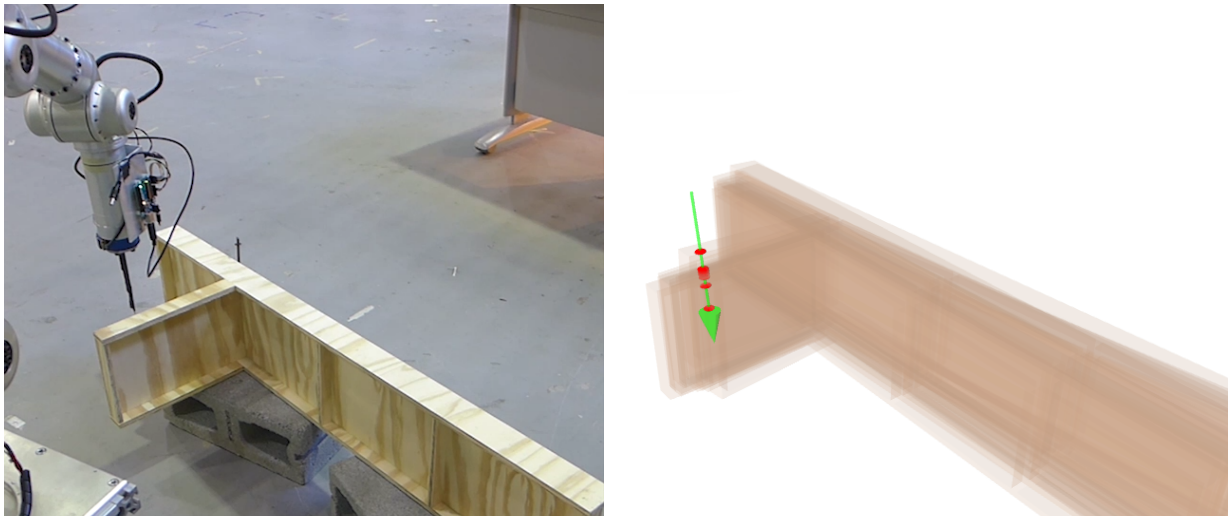
1.2 Approach

The work in this thesis address two separate problems. The first is the task of localizing an object through touch probing. The second is the task of planning a trajectory for a robotic arm that is too long to support its own cantilevered weight, and so must rest on the environment for support.

1.2.1 Localization

In the localization task the robot must estimate the pose X of an object based on a set of measurements $Z_t = \{z_1, ..., z_t\}$ made by probing the object. The robot chooses where to probe, and the measurements are a function of the object and chosen measurement action. A triangular mesh describes the object geometry and a frame is attached to this mesh. The state X is the $SE(3)$ transformation from a fixed world frame to this part frame. This is a 6-dimensional state stored as position (x, y, z) and orientation angles (α, β, γ). Initially it is assumed the geometry is rigid, thus the state can be fully described as the $SE(3)$ configuration of a frame attached to the part, while later later, the approach is generalized to parts with internal uncertainty. A further assumption is that the part is fixed in space relative to a world frame and that probing actions do not perturb the state.

Rather than calculating a single best estimate of the pose, a probability distribution represents the uncertainty in knowledge. Estimating the probability distribution is important both for



(a) (b)

Figure 1.4: **(a)**: The robot performing a touch measurement
(b): The belief of the part location and measurement

choosing effective measurements and knowing when the part is localized sufficiently. For feasibility, this probabilistic belief is represented numerically by a set of points drawn from the true distribution called *particles*. A measurement updates the belief using a *particle filter* [55].

However particle filters behave poorly both when the dimensionallity of the system grows large and when measurements become precise [33]. In the problems considered there is a large initial uncertainty, and achieving high tolerance performance necessitates measurements with low uncertainty. Updating the belief after performing such a measurement will eliminate most particles, leaving a poor representation of the posterior. This problem is called particle starvation.

This thesis first describes a solution to particle starvation during touch localization (section 3.1) developed in collaboration with Shiyuan Chen [47] by using a particle filter with an alternate update procedure that is able to combine accurate measurements with the prior. This particle filter is then generalized to parts with internal uncertainty (section 3.2).

To minimize the time taken to localize the part, the robot should choose informative measurement actions. To achieve this, many actions are sampled and the one with the highest expected *information gain* is selected. Fully predicting this is also computationally expensive, thus previous methods using this metric introduce delays during which the robot pauses between measurements and yet still only samples a small number of actions. Our approach involves a fast approximation for information gain that takes advantage of the discretized belief from the particle filter (section 3.3). This enables the sampling of hundreds of potential measurement actions in a few seconds.

Figure 1.4 shows the experimental setup. A 7-DOF robotic arm equipped with a touch probe performs measurements on an object **(a)**. These measurements are used to update the belief **(b)**. The belief is the used to determine good information gathering future measurement actions.

1.2.2 Motion Planning

The motion planning task considered here involves planning trajectories for a robotic arm through an environment. The robot arm begins at initial joint angles and the goal is to apply torques on the joints to move the end effector of the robot to some specified location in the workspace. When planning, the robot arm kinematics and dynamics, joint torque limits, and the environment are known. Of particular interest is the dynamics model, as it models contact forces at specified locations on the robot.

When not in contact with the environment the robot dynamics obey the manipulator arm equation [38]. However, when in contact, additional forces are applied to the arm obeying a spring model: The force is proportional to the penetration distance, in the direction pushing the arm out of the environment.

In the tasks considered in this thesis, the robot is not capable of reaching the goal point without using contact points, thus the motion planner must make use of contacts. Two approaches are used to plan: trajectory optimization and a sample-based planner. In the trajectory optimization framework a function determines the cost of a trajectory, and an initial trajectory is iteratively improved using gradient descent until a local optimum is achieved. The obvious choice for a cost function would penalize not reaching the goal and penalize joint torques, either through soft or hard constraints. However due to the local nature of contacts, potentially useful contacts that are not yet made have no influence on the cost function, and these solutions will never be discovered. The solution is an augmentation of the dynamics model, allowing contact forces at a distance. This provides a much better conditioned cost function, at the expense of accurately modeling the world. Towards the end of optimization the model parameters are altered to enforce a realistic solution.

When initialized well, this optimization approach produces trajectories that keep joint torques low by making and breaking contacts. However, the cost function is still littered with non-optimal local minima, and escaping these prove difficult. Thus a sample-based planner is employed to generate trajectories that can either directly be used, or that serve as a decent initialization for the optimization approach. Again the thin contact manifold presents problems. Sampled based planners rely on generating trajectories to connect probabilistically sampled configurations, but the thin contact manifold renders naive approaches useless. The solution used augments the extension function of a sampled based planner, causing extensions to make progress towards their goal while staying on the contact manifold as necessary.

Chapter 2

Related Work

2.1 Current Methods for Localizing Objects Using Contact Sensors

Before performing operations on parts for manufacturing, it is crucial that the position and orientation of that part are known to within some tolerance. While there are many methods of localizing a part, this work uses a touch probe to actively gain information through measurements on the part surface. Though it is possible to ignore uncertainty in some tasks, the this work explicitly reasons over the belief distribution of the part's state to address more challening tasks. One approach, the particle filter, maintains this distribution through a finite set of samples. A second approach, the Kalman filter, maintains an analytic expression for the belief distribution. Both approaches require adaptations for use in contact localization.

2.1.1 Particle Filters

Particle filters, touch measurements, and maximal information gain measurement selection have all been used in localization tasks. Since their introduction particle filters have been popular due to their ease of implementation and ability to model complex distributions, process models, and measurement models [55]. However, for a measurement with low uncertainty there exists only a thin manifold of states consistent with that measurement, yielding a low probability of any particle existing on that manifold, leading to particle starvation [54]. Handling particle starvation is described in detail in Chapter 3.1.2.

To address particle starvation, Koval introduced the Manifold Particle Filter, using different sampling methods depending on the volume of the space consistent with a measurement [32, 33]. This allows a quick update of the belief when the contact sensor is not in contact with the part, and only requires addressing the harder thin manifold update problem when contact is made. Koval used multiple methods when updating from measurements when on this thin manifold, and shows that rejection sampling requires the fewest restrictions on prior knowledge of the environment, but naive rejection sampling is time consuming. His efficient methods require direct sampling from the contact manifold, which is not feasible for a complex part.

Petrovskaya focused on global localization of objects via touch [41] and introduced the Series

Scaling algorithm to overcome particle starvation. The Series Scaling algorithm adaptively alters the particle density depending on the complexity of the posterior. Multiple passes through the measurement data are used and the precision of the modeled belief is scaled from low to high, avoiding unnecessarily precise estimates in exceedingly unlikely regions of belief space. This is complementary to our approach, and implementing multiple passes on our methods could lead to a faster update process.

Much of the recent work on touch measurements uses a robotic hand with contact sensors. In these works evaluation of both actual and simulated measurements required collision checking between two meshes which is computationally expensive.

Hebert et. al. use geometric (CAD) models of objects such as screwdrivers and door handles as well as the geometric model of their robotic arm to autonomously choose touch actions that localize objects sufficiently to perform everyday tasks, such as grasping and opening doors. Their algorithm greedily selects the next best touch action from a list of candidate actions to maximize information gain [22].

Javdani shows that selecting the next touch to maximize information gain is submodular under assumptions of a static object and an action cost independent of object and robot state, explaining the effectiveness of the greedy approach [25]. This provides a sound theoretical basis for our approach. Javdani demonstrates the computation of information gain is time consuming and proposes an alternative method of hypothesis pruning. Our formulation computes the expected information gain about two orders of magnitude faster, and thus we are able to evaluate many more potential measurement actions and model the belief using more particles.

Recently, there have been a variety of approaches that allow robots to localize objects solely with contact sensors. Different contact sensors have been explored and developed, including basic binary sensors, 6-axis force and torque sensors [12], soft tactile sensor arrays [21], and bio-inspired fingertips [15]. Localization with laser sensors has also been used in the high-precision CNC localization, where a 3D point cloud is acquired in order to estimate the transformation between the actual and planned pose [44]. The localization approach presented here can be generalized to these sensors which can distinguish between contact and free-of-contact states.

2.1.2 Kalman Filters

Alternative approaches to modeling touch localization adapt a Kalman Filter to model the belief distribution. A Kalman filter requires a Gaussian belief of the estimated state, which is inaccurate in touch localization. For example, multi-modal distributions appear when it is ambiguous whether a close edge or far edge was touched.

A Kalman filter also requires a linear measurement models, which does not exist for touch localization, so this too must be approximated. The Extended Kalman Filter performs first-order linear approximations of the measurement model, but this diverges for pose estimation with large initial error [10]. The Unscented Kalman Filter approximates the measurement model through evaluation at many "sigma" points chosen by tuning parameters. As the pose likelihood given a measurement varies by many orders of magnitude in different dimensions, these parameters are sensitive and unreliable. Srivatsan constructs a linear measurement model using two measurements and assuming the correspondence between the probe tip and the touched point on the object [51]. This allows direct use of a Kalman filter with a true measurement model. However,

the true correspondence is unknown, and must be approximated, for example through Iterative Closest Point (ICP) methods.

2.2 Contacts in Planning

Higher level tasks involve frequent making and breaking of contacts: walking requires footstep placement, manufacturing involves part handling, and household tasks need object manipulation. The wide variety of sub-problems lead to different approaches to robotic reasoning and execution of actions with contacts. In some tasks the the robot must decide between a known set of fixed contact locations, while in other tasks the contact space is continuous. Contacts may be a necessity or they may be optional.

In many path planning any contact with the environment is considered a collision and thus valid paths have no interaction with the environment. However environmental contacts can reduce joint torques, damp vibrations, and stabilize an arm. Despite these benefits contacts are usually avoided because they bring algorithmic complexity during the planning stage and physical danger to the robot and environment if executed improperly. The increased planning complexity is due to the contact configurations being a measure-zero manifold in the robot configuration space, rendering naive planning in configuration space ineffective.

2.2.1 Bracing, Climbing, and Walking

The benefits of bracing have been studied since the 1990s. Lew and Book proposed bracing a micro/macro manipulator with small precise arm mounted on the end of a long coarse arm and demonstrated bracing of the macro arm against multiple locations of the environment can damp vibration caused by the micro arm [5] [35]. Hollis and Hammer explored a similar micro/macro robot design and demonstrated $1\mu m$ accuracy, well over an order of improvement compared to their unbraced manipulator [23]. Both of these works did not address the planning problem as both contact locations and robot trajectories were manually specified.

Given a sequence of contact modes for each link, Greenfield computed joint torques to produce desired dynamic behavior and applied this to a climbing snake robot [20]. Bretl et al. [6] developed algorithms for climbing robots that first select contact locations then create collision free trajectories for the robot's limbs between these contact locations.

The recent DARPA Robotics Challenge showcased state of the art walking robots. Rather than solving for a trajectory and contact locations simultaneously some approaches have separated these two problems. The dominant planning architectures hierarchically separated stages of contact planning. Highest, and slowest, in the hierarchy a planner decides the path for a simplified model of the robot through the workspace using heuristic costs. Lower down in the hierarchy, a footstep planner determines the contact locations for feet to best achieve the higher level trajectory. Still further down, another planner determines joint trajectories to move the feet into the requested footstep locations while maintaining stability [11] [3]. Tonneau et al [56] approaches from the opposite direction and first generates a trajectory then considers a discrete set of contacts close to that trajectory.

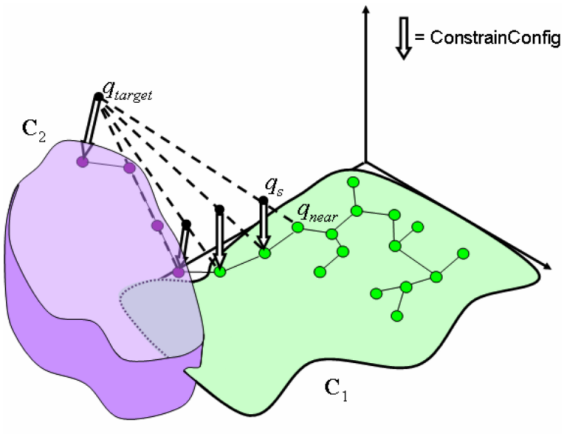


Figure 2.1: Constrained Bidirectional Rapidly Exploring Random Tree (CBiRRT) extending the tree onto a measure-zero valid manifold in configuration space

2.2.2 Sample based planning

Sampling based planners probabilistically probe the configuration space searching for feasible paths. The valid configuration space is never explicitly represented, and all that is needed is a function that determines whether a specific configuration is valid. The Rapidly-exploring Random Tree (RRT) and Probabilistic Roadmaps (PRMs) are sampling based planner which build graphs of valid trajectories (edges) between points in configuration space (nodes). To grow these graphs a new point in configuration space is sampled, and the closest point on the existing graph is extended towards this sampled point, as shown in Algorithm 1. [34]

Algorithm 1 $T = (V, E) \leftarrow \text{RRT}(x_{start})$

1: $T \leftarrow$ InitTree(x_{start})
2: **while** GoalNotReached(T, X_{goal}) **do**
3: $x_{sample} \in X \leftarrow$ Sample()
4: $x_{nearest} \leftarrow$ Nearest(T, x_{sample})
5: T $\leftarrow$Extend(x$_{nearest}, x_{sample}, T$)
6: **end while**

In the most basic implementation, the Extend function constructs a linearly trajectory in configuration space from $x_{nearest}$ towards x_{sample}, and terminates the extension either when x_{sample} is reached or the trajectory enters an invalid region. [34] This extension step requires augmentation to be useful in planning with contacts. Sampling based planners will never sample directly on the measure-zero contact manifold, and linearly extending a contact configuration to a new sampled point will immediately leave the contact manifold, immediately entering an invalid region in configuration space, producing no progress.

Berenson developed a variation of an RRT planner capable of handling measure-zero manifolds by projecting invalid path extensions onto the valid configuration space [4]. The Constrained Bi-directional RRT (CBiRRT) performs a linear extension from $x_{nearest}$ towards x_{sample}

as before, producing an intermediate point x_s. However if x_s is an invalid configuration, a projection function is used to potentially create a valid configuration.

$$x_s^{new} \leftarrow Projection(x_s) \tag{2.1}$$

If x_s^{new} is valid and is making progress toward x_{sample}, the extension algorithm continues. If either the $Projection$ function failed to produce a valid configuration, or the extension toward x_s^{new} moves the trajectory further from x_{sample} then extension is terminated and the RRT algorithm continues with a new randomly sampled point. In CBiRRT the method of projection must be provided by the user.

2.2.3 Trajectory Optimization

In contract to sampling based planners, trajectory optimization addresses the path planning problem by incrementally improving an initial trajectory. Trajectory optimization uses a cost function to assign a cost to every trajectory, and an initial trajectory is repeatedly adjusted to reduce this cost. While a large class of work focuses on trajectory optimization with unknown dynamics, in this thesis it is assumed the dynamics are known, and the guiding approach is to assume the dynamics are locally linear and perform updates to the parameters of the trajectory based on the gradient of the cost function. This approach will converge to local, but not necessarily global, minimal cost trajectories. Contact forces present an initial challenge, as the assumption of locally linear dynamics no longer holds.

Deits et. al. handles the foot placement contacts for walking robots through mixed-integer optimization [11]. In a path for a walking robot there are an integer number of footstep placements, but the trajectories of the feet and robot are continuous. One approach is to fix the number of footsteps and run a purely continuous optimization over the trajectories. This continuous optimization is then repeated for each integer number of footsteps considered plausible. To reduce the number of optimization, their approach assigns a cost to each footstep allowing for faster pruning. This approach works because the integer parameter is relatively simple, i.e. once the right foot steps the only next contact to consider is the left foot.

Both Posa et. al. [43] and Mordatch et. al. [37] handle the discontinuous contacts by smoothing. Both formulations assume a given set of locations on the robot are allowed to make contact with the given environment. Both works write the cost of a trajectory as a function of both joint torques and contact forces, thus the contact forces are a slack variable, and treated as parameters of the trajectory. Of course in reality a robot cannot choose the contact force applied, as the force is generated by electron repulsion as a function of the distance between the environment and the robot. However, this inaccurate physics model smooths the cost function and allows the robot to discover the benefits of contacts. Additional constraints are imposed to ensure the final trajectory is physically feasible.

Posa et. al. [43] is primarily concerned with large impact forces, and writes the problem as a constrained optimization. At each iteration the optimizer alters both the joint torques and the desired contact forces to improve the cost. However this optimization problem has a set of complementarity constraints which enforce that each section of the robot is either in contact and can receive a contact force, or it not in contact.

Mordatch et. al. [37] designed a cost function that continuously models both the benefit and cost of adding contacts and is able to produce trajectories which add and break contacts. This formulates the problem as a purely unconstrained optimization, with all physical constraints being added to the cost function. The parameters of this cost function are both the joint torques and a set of continuous "contact" slack variables, one for each section of the robot that is allowed to make contact. For each section, the contact variable can be intuitively be thought of as trade off between the benefits and the challenges of making contact with the environment. Increasing the contact variable increases the cost as a function of the distance between the robot section and the environment. Increasing the contact variable decreases the cost as it allows contact forces to compensate for joint torques. Of course in reality there is no partial contact, as if a section of a robot is some distance away from the environment, no contact force is applied. However this formulation of force at a distance makes the benefit of adding a contact visible to the gradient of the cost function. To ensure the final trajectory is feasible, the cost of physically infeasible partial contacts is increased dramatically towards the end of the optimization.

Chapter 3

Localization using Contacts

Many robotic tasks require precisely localizing an object, for which tactile sensing is an appealing sensing modality. As motivation, consider touch localization of a partially manufactured part that requires additional machining operations. In order to handle objects with complex shape, prior information of the object shape is used, and most previous work assumes that the geometry (CAD) model will match the real object exactly.

However, during the manufacturing and assembly processes there are tolerances between different sections of the assembly. A *datum* is defined as a geometric constraint within the object that is used as the reference to define the location of one section of the part with respect to another section. The tolerance is the allowed deviation of the actual manufactured dimensions from the nominal designed dimensions. It is assumed a part can be divided into precisely manufactured sections, and the introduced method focuses on handling errors due to imprecise machining over large distances and non-critical components, as well as assembly tolerances.

The introduction of tolerance increases the degrees of freedom (DOFs) of the system, as prior to measurement the true dimensions of the full part are unknown. These internal DOFs can be modeled as transformations with uncertainty between sections of the object. For objects with internal tolerances, perfectly localizing a single datum will not necessarily reduce the uncertainty of the full system sufficiently to perform the desired task. On the other hand, it is usually only necessary to localize a subset of the sections of an object.

In this localization problem, the task is to estimate the pose of a goal feature given multiple measurements obtained through probing. These probing measurements are modeled as a Markov process, where each measurement corresponds to a single action/observation pair. This model eliminates the need to store all of the past measurements. A particle filter numerically stores and updates the belief [55].

Figure 3.1 shows a visualization of the initial (3.1a) and final (3.1b) beliefs of the poses of the sections of the object. The task is to drill a hole, shown as a green cylinder, at a specific location defined by offsets from other sections. Internal tolerances prevent simply treating the entire system as a rigid body.

This chapter first considers rigid-body particle filtering for high-precision localization (section 3.1), which overcomes the particle starvation problem [47]. The datum-based particle filter is then introduced to generalize this rigid-body approach for objects with coupled rigid sections (section 3.2). Two related but different approaches are proposed for this problem. The first ap-

(a) Initial belief

(b) Belief after localization of goal feature (green cylinder)

Figure 3.1: Visualization of the belief of the pose of all sections of the part
(a): The prior belief of the poses before localization. The uncertainty of the goal feature is too high to perform the task.
(b): The belief of the poses after performing measurements to localize the goal feature. The pose of the goal feature is now known well enough to perform the task. Although the bottom edge (purple) and perpendicular section still have noticeable error, precise localization of these features is not needed.

proach maintains a single particle filter system that stores the full joint distribution of the coupled datums, while the second approach simplifies the relationships by assuming independence between the distribution of the internal transformations and the pose of the sections, and models the system using separate coupled particle filters.

This chapter then generalizes the technique of choosing informative measurement actions to accommodate objects with coupled sections. To achieve this, many potential measurement actions are sampled and the action with the highest expected information gain is selected. Fully predicting the information gain over the continuous belief is computationally expensive, so similar to other approaches [25] this work involves a fast approximation for information gain that takes advantage of the discretized belief from the particle filter (section 3.3). This chapter concludes with evaluations of the rigid body approach on a real part, and the datum localization on a simulated part (section 3.4).

3.1 Rigid-Body Object Localization

During my thesis I collaborated heavily with Shiyuan Chen on the task of autonomous localization. The rigid body particle filter developed in this section is mostly his work. The datum-based particle filter described later relies heavily on the framework and methods developed for rigid-body localization, and this section provides an overview of the relevant details [47]. The task is to determine the pose of an object by choosing and performing touch measurements, given the geometry of the object and prior belief over the distribution of poses.

The geometry of the object to be localized is stored in a STL file using a triangular mesh, defined in the *part frame*. The pose of the object can then be defined as the transformation between this *part frame* and the *world frame* of the workspace. The object is assumed to be fixed in the workspace during measurement and localization, thus the configuration will not change during the localization process.

In order to estimate the true distribution of the pose, each particle in the particle filter represents a single potential pose ${}^{i}x \in SE(3)$ of the object. For a rigid body, the pose includes both translational dimensions (x, y, z) and rotational Euler angles (α, β, γ). The state is a 6D vector $(x, y, z, \alpha, \beta, \gamma)$ in the configuration space. The particle filter updates based on a set of measurements $M_t = \{m_1, ..., m_t\}$ made by the robot directly on the object.

3.1.1 Measurement Model

A measurement action, $\mathcal{M}$ is defined by a start point $\mathcal{A}_p$ for the probe and a linear trajectory vector $\mathcal{A}_v$ both in $\mathbb{R}^3$. The measurement value m is the distance the probe travels in the direction of $\mathcal{A}_v$ until contact is made. The point of contact can then be recovered by $A_p + m \frac{\mathcal{A}_v}{||\mathcal{A}_v||}$. The entire information obtained from the measurement t is $z_t = \{\mathcal{M}_t, m_t\}$. Measurement error exists due to sensor error and robot uncertainty.

3.1.2 Problems with the standard particle filter

A common method of updating particles based on a measurement is importance sampling [54]. In importance sampling each particle is weighted by the probability of the measurement conditioned on the state that particle represents. This is usually followed by resampling, where particles are redrawn from the set of weighted particles with probability proportional to their weights. The effectiveness of importance sampling relies on the existence of multiple particles consistent with the measurement, such that inconsistent particles will have low weights and be unlikely to be resampled, but a sufficient number of particles will be resampled to model the true belief of the state.

Importance sampling tends to break down in situations with accurate measurements and low densities of particles. This is because when a measurement is consistent with the true prior belief yet no particles are consistent with the measurement, a situation called *particle starvation*, resampling will yield a set of particles that does not model the true posterior belief. A more accurate sensor measurement is consistent with a smaller volume of state space, thus a higher density of particles is required. For higher dimensional state spaces and more accurate sensors the number of particles required becomes prohibitively computationally expensive. This leads to the counter-intuitive result that particle filters tend to perform worse as measurement accuracy increases [33].

3.1.3 Rejection Sampling Method

An alternate approach is to use rejection sampling. Rejection sampling does not require a high density of particles to avoid particle starvation and failure of rejection sampling is far easier to notice and resolve. Most importantly, we can increase the limits of state dimensionality and measurement accuracy that can be handled efficiently.

Rejection sampling generates independent samples from a density f by sampling from a different distribution g. A constant M is determined such that $f(x) \leq Mg(x) \; \forall \; x$. A sample x^* drawn from $g(x)$ is accepted with probability $f(x^*)/Mg(x^*)$ and rejected otherwise. The

process is repeated until the desired number of samples has been accepted. We wish to sample particles from the posterior $bel(x_{t+1})$ but cannot do so directly. Instead we sample from our continuous prior belief $bel(x)$ and possibly reject based on the measurement.

Broad Particles: We first reconstruct the continuous prior belief by broadening each of the particles. We apply a *Gaussian kernel* to the particles, with the kernel covariance proportional to the covariance of the particle states. This reduces particle starvation, as even if no prior particle is consistent with the measurement, the continuous belief generated fills in the gaps between particles.

3.1.4 Fast Evaluation of Sampled States

States sampled from this continuous prior are then rejected if they are inconsistent with the measurement. While the formulation of rejection sampling allows us to model complicated measurement error, we implement a binary measurement model. We reject all sampled particles where the measured point is sufficiently far from all faces of the object. We define "sufficiently far" as more than 3 standard deviations of the sensor measurement noise. As a low uncertainty measurement will accept only a thin manifold in state space, the probability of sampling a particle consistent with the measurement may be low, and a lot of sampling may be required, therefore we desire the rejection process to be fast.

To reduce the computational cost per sampled state we use discretized space, known as a *distance field* [14], to precompute and cache the minimum unsigned distance $D_f(p)$ from point p in voxelized space to the object surface $\partial S \subseteq \mathbb{R}^3$:

$$D_f(p) = \min_{q \in \mathbb{R}^3}(||p - q|| + f(q)) \tag{3.1}$$

$$f(q) = \begin{cases} 0, & \text{if } q \in \partial S \\ \infty & \text{otherwise.} \end{cases} \tag{3.2}$$

As the object is fixed during the localization process, voxelization can be done for the entire piece based on the given CAD mesh model in the precomputation step.

Voxelization: Voxelization is the key part to transform the mesh model to axis-aligned discretized space, which can be stored and accessed easily as a standard array. The array form of the CAD model can greatly facilitate the computation of the distance field, as described below. Each voxel is assumed to be a cube in 3D space. A fast 3D Triangle-Box Overlap method [2] is used to label the voxels that overlap the mesh triangles of the object surface. The voxel map is then mapped to a binary-valued 3D array $f(q)$, where each value is either 0 or ∞ depending on whether the corresponding voxel overlaps the object surface.

Voxelized Distance Field: The computation of distance field $D_f(p)$ takes the input of the computed binary array $f(q)$ (Eq. 3.2), and a linear-time algorithm for 3D distance field construction [14] is then used. The resulting distance field is also stored in an array for constant time access during the evaluation of sampled states.

Fast Evaluation: Different configurations result in different poses of the object in the workspace, which makes it difficult to compute the distance field directly in the world frame. Instead, the computation of the voxel map and distance field is relative to the object frame, where the object is assumed fixed during the entire localization. Each measurement M_t in the workspace is then

transformed into the part frame, where the transform $T(x_{t+1})$ comes from the pose of the sampled state x_{t+1}. Therefore, by transforming back to the object frame, all measurements on this same object can share the same distance field, where the minimal unsigned distance $dist_u(M_t, S)$ between each measurement M_t and the object $S(x_{t+1})$ can be obtained directly:

$$dist_u(M_t, S(x_{t+1})) = D_f(T(x_{t+1})^{-1} M_t) \tag{3.3}$$

The signed distance $dist(M_t, S(x_t))$ between the probe and the object can be obtained from the unsigned distance, as shown in Eq. 3.4, by checking whether the voxel is inside or outside of the object. For the manifold shape object, ray-casting is applied from the corresponding voxel in a certain direction: the voxel is inside of the object only if the number of intersections between the ray and mesh is odd

$$dist(M_t, S(x_t) = \begin{cases} dist_u(M_t, S(x_t) - r_p, & \text{if } M_t \notin S \\ -dist_u(M_t, S(x_t) - r_p, & \text{otherwise.} \end{cases} \tag{3.4}$$

The unsigned distance $dist(M_t, S)$ is always 0 in the ideal case, however, when evaluating a sampled state, only those that satisfy $|dist(M_t, S)| > T_d$ will be rejected, where T_d is the tolerance selected according to the measurement uncertainty of the touch probe and the robot. If the distance is within the tolerance, ray-casting is then applied to check intersections from the start point $\mathcal{A}_p$ along the path vector $\mathcal{A}_v$ in order to determine whether the path is free of collision with other parts of the object.

When the measurement is very accurate, in order to sample enough particles from the prior belief, a large number of states will get rejected, which makes the ray-casting for all sampled states computationally expensive. Instead, early rejection is applied using a greater distance $dist_u(M_t, S) + r_p$ before the computation of signed distance.

Suppose that the true distribution is given by a discrete multinomial distribution with k different bins, it can be shown that with probability $1 - \delta$, the KL-divergence is less than or equal to ϵ when the sample size n is given by [16]:

$$n = \frac{1}{2\epsilon} \chi^2_{k-1,1-\delta} \tag{3.5}$$

$$\approx \frac{k-1}{2\epsilon} \left(1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}} z_{1-\delta} \right)^3 \tag{3.6}$$

where $\chi^2_{k-1,1-\delta}$ is the upper $1 - \delta$ quantile of χ^2-distribution with $k - 1$ degrees of freedom, and $z_{1-\delta}$ is the upper $1 - \delta$ quantile of the normal distribution.

Therefore, the number of particles n can be adjusted according to the number k of bins with support, as shown in Eq. 3.3. The bins are implemented as a multidimensional grid with fixed size in the configuration space. During sampling, the number k of occupied bins is counted whenever the newly sampled state falls into an empty bin. The current sample size is counted and each increment of k will result in an update of desired sample size n. When the actual number of particles reaches the desired value or a predefined maximal limit, whichever is smaller, the sampling process finishes.

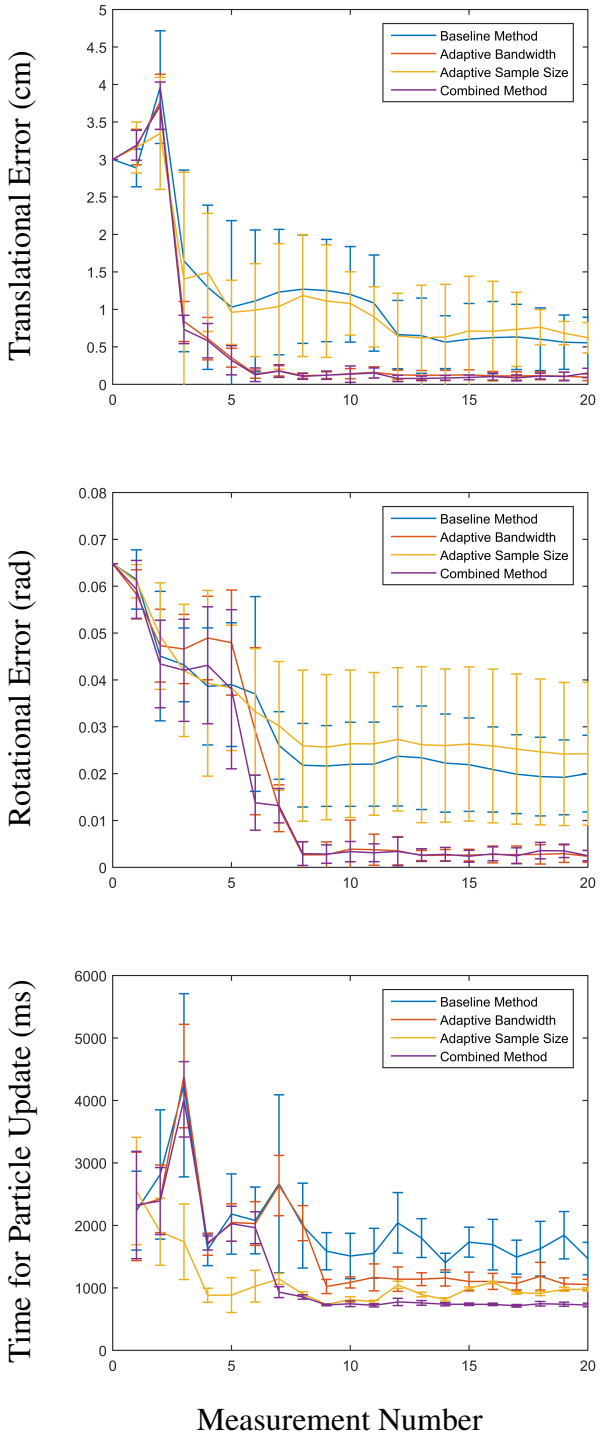


Figure 3.2: Comparison of the time and accuracy of the particle filter update step when using Adaptive Bandwidth and Adaptive Sample Size

Figure 3.2 shows the comparison between the particle filters with different improvements that are mentioned above. The simulation uses a mesh model with 39444 triangles. The set of measurements used for each method is fixed and predefined. The baseline method represents the particle filter that only implements adaptive voxelization while using fixed kernel bandwidth $h = 0.0035$ and fixed number of particles ($n = 500$). Adaptive Bandwidth improves the convergence accuracy significantly compared to the baseline method, while Adaptive Sample Size generally leads to poorer convergence with faster speed. The combined method applies all of the improvements above. It can achieve both faster and more accurate convergence compared to the other methods.

3.2 Datum Based Particle Filter

The particle filter localization method presented above assumes that the object matches its CAD model exactly. However, this is usually not the case, due to tolerances in the manufacturing and assembly processes. To handle manufacturing deviations, features on parts are not located with respect to the part frame, but with respect to datums, (edges, surfaces, and holes) on the actual "as built" part. Incorporating the notion of datums, and their relationships, adds complexity because the relationship between the datum and the CAD model contains uncertainty. Thus, measuring one section of the assembly provides only uncertain updates to other sections, dependent on the specified tolerances. The following formulation treats these as semi-rigid parts, where each complete part is composed of rigid *sections*, coupled through a probabilistic distribution of transformations connecting the section frames. The datum based particle filter is introduced to allow updates on the belief of all sections of a part using the prior distribution of coupling transformations, and a measurement on a single section.

3.2.1 Datum Representation

The formulation introduced here treats the overall part as composed of separate, known sections. The problem is to precisely localize some feature which cannot be measured *directly* (e.g. a location to drill a hole) with respect to given datums (other sections). To localize the goal feature, certain datums must be localized in certain dimensions. For instance, Figure 3.3a shows a hole feature referenced to the top and right edges datums of the part. The true part configuration is shown in gray in 3.3b and 3.3c. In this example, it is necessary to localize the top edge's vertical position and orientation, but not its in-page or horizontal position. Similarly, the right edge only must be localized horizontally.

Two approaches are now introduced; the first explicitly represents the joint probability distribution between the sections, and the second stores separate, independent probability distributions for each section. Figures 3.1 and 3.3 visualize the independent-state particle filter. The full-state particle filter produces similar images.

The rest of this chapter uses the following notation. X_t^k is the set of N particles representing the belief of section k at time step t. Frequently t is omitted when implicit. Each particle is a configuration for a single section $X^k = \{{}^j x^k\}_{j=1}^N$. The omission of k indicates all necessary particles to represent the belief of the part: $X = \{X^k\}$ and $x = \{x^k\}$.

Algorithm 2 Independent-State Particle Filter

Input: number of particles N and number of sections n
Input: sets of particles $\mathbf{X_t} = \{X_t^k\}_{k=1}^n$
Input: observation m_t^i
Input: meshes $S = \{S_k\}_{k=1}^n$,
Input: transformations $\{p(T_i^k)\}_{k=1}^n$
Output: particles $\mathbf{X_{t+1}} = \{X_{t+1}^k\}_{k=1}^n = \{^j x_{t+1}^k\}$

build distance field $D_f(p)$ for section S_i
for $k = 1, ..., n$ **do**
    $j \leftarrow 1$
    while $j \leq N$ **do**
        $x \sim p(x^k | X_t^k)$
        $T_i^k \sim p(T_i^k)$
        $\tilde{x} \leftarrow T_i^k \times x$
        $dist \leftarrow D_f(T(\tilde{x})^{-1} M_t^i)$
        if $dist \leq \xi$ **then**
            $^j x_t^k \leftarrow x$
            $j \leftarrow j + 1$
        end if
    end while
end for

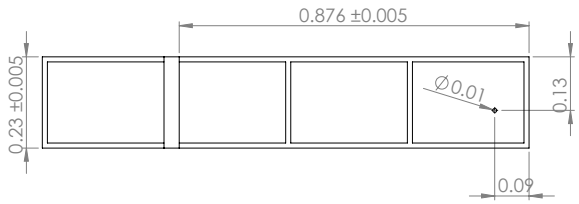
3.2.2 Geometric Relationships

Geometric relationships are defined between two or more part sections. The existence of the tolerance introduces uncertainty to these relationships, which are modeled as a distribution of transformations in the configuration space between the pose of each section. More generally, the conditional probability $p(x_t^k | \mathbf{x_t})$ represents the belief of the pose of section k given the poses of the other sections.

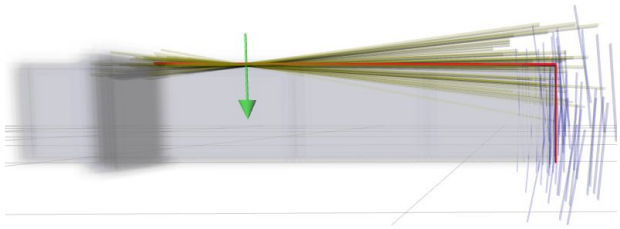
A measurement is made on a single section at each step by a touch probe. Let the measurement on the section k at step t be $\mathcal{M}_t^k$, then the posterior of section k is $p(x_t^k | \mathbf{x_t}, M_t^k)$. In the following algorithms, the section that a measurement contacts is known. This assumption is reasonable if the uncertainty of the prior belief is small compared to the physical size of each section, and measurement are not chosen on the boundary between sections. If this assumption does not hold, localization can be performed for the whole object using the methods of section 3.1 to get better estimate, before considering the object as a combination of coupled sections.

3.2.3 Independent-State Representation

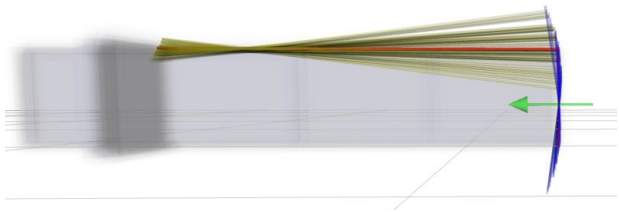
My approach is to maintain the probability distribution for each section separately. Instead of using a full high-dimensional particle filter for the full object, individual 6-dimensional particle filters are used for each individual section under the approximation that the belief over transformations between sections are fixed and independent. While this loses information compared



(a) CAD



(b) Update from a measurement on the top datum



(c) Update from another measurement on the right datum

Figure 3.3: Visualization of independent-state particle filter
(a): Side view of the CAD drawing with dimensions (simplified for clarity). This drawing indicates the nominal distance between the top and bottom edge is 0.23m, with a symmetric tolerance of 5mm. This drawing also defines a hole with a 1cm diameter, and the top edge as its vertical datum and side edge as its horizontal datum.
(b): The beliefs of the top (green) and right (blue) edges of the part are shown. The true part location is shown in gray. The measurement (arrow) on the top section partially localized the top edge. For clarity in the image, the belief of the other sections are not shown, and only 50 of the 500 particles are shown.
(c): A following measurement (arrow) on the right edge further localizes the part. This measurement provides information on the right edge directly, and the top edge indirectly.

with the full joint belief, in practice this loss is acceptable.

As in the rigid body particle filter described in 3.1, a sample in the particle filter for section k represents a $SE(3)$ pose of the geometry of section k. The transformation information between different sections are defined explicitly. The prior belief on the transformation from section k to section j is $bel(T_k^j)$, which is a distribution over $SE(3)$ transformations. A measurement on a single section updates all individual particle filters related through a defined transformation distribution. Given a measurement M^k on the section k, the updated belief becomes $p(x_{t+1}^j | T_k^j, M^k)$ for a related section j.

The update to the belief $bel(x_t^k)$ given a measurement performed on section k itself is identical to the particle filter update for the rigid object. Since T_k^k is the identity with probability 1, the updated belief can be written as:

$$bel(x^k) = p(x^k | T_k^k, M^k) = p(x^k | M^k) \tag{3.7}$$

For a section j that references section k ($k \neq j$), each new sample x_{t+1}^j is drawn from the prior of its corresponding particle filter j. x_{t+1}^j is then transformed from the frame of section j to the frame of section k:

$$\tilde{x}^k = {}^i T_j^k \times x^j \tag{3.8}$$

where ${}^i T_j^k \sim bel(T_j^k)$ is a sampled transformation from the distribution $bel(T_j^k)$. As the measurement was performed on section k, the geometry of section k is used rather than j when computing the consistency with the measurement. The sampled particle is accepted with probability $p(M_t | \tilde{x}^k)$. The above process is repeated until the desired number of particles have been accepted (shown in Algorithm 2).

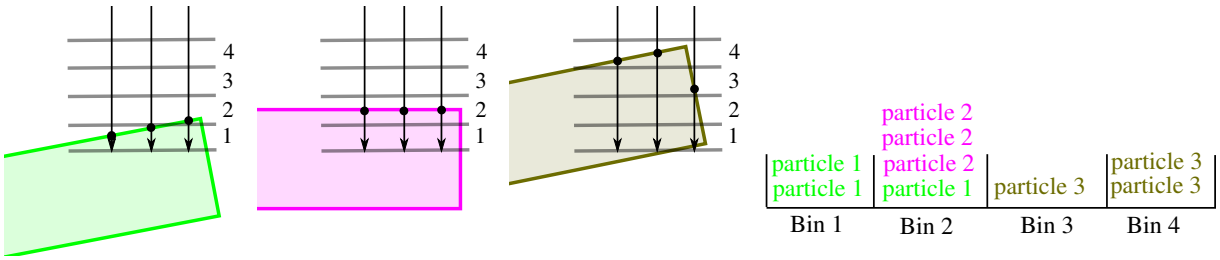
3.3 Predicting Effective Measurement Actions

Performing measurements is expensive, so it is important to choose the measurement action that provides the most *information gain* on the goal feature. Each action is treated as a probabilistic decision over a set of particles approximating the belief of the goal feature. This is an approximation for the information gain for the underlying, continuous belief distribution. The best measurement may not be on the goal feature, and it may be impossible to even measure the goal feature directly. This formulation predicts the information gain on the goal feature for both a measurement directly on the goal feature, or indirectly for a measurement on datums or other sections of the part.

3.3.1 Information Gain

Given X^G, a set of particles representing the belief of the goal feature, the *information gain* from a measurement action $\mathcal{M}$ is defined as the expected reduction of entropy.

$$IG(X^G | \mathcal{M}) = H(X^G) - H(X^G | \mathcal{M}) \tag{3.9}$$



(a) Part in configuration: particle 1 (b) Part in configuration: particle 2 (c) Part in configuration: particle 3 (d) Binned Measurements

Figure 3.4: Binning of potential measurement on particles for use in calculating information gain. The three arrows represent the nominal measurement action $\mathcal{M}$ with simulated deviation δ_j. The horizontal lines divide the measurement values into the numbered bins.

$H(X^G)$ is the entropy of the particles and $H(X^G|\mathcal{M})$ is the entropy of the particles conditioned on the measurement action.

The entropy of a discrete distribution of states depends only on the probabilities of each state occurring.

$$H(X^G) = -\sum_i w_i \log w_i \tag{3.10}$$

where w_i is the weight of particle i.

To calculate the conditional entropy, $H(X^G|\mathcal{M})$, the measurement action $\mathcal{M}$ is simulated on the part distribution. Performing a measurement action yields a continuous distribution of measurement value. $\mathcal{W}$ samples are drawn from this distribution for each particle:

$$m_{i,j} = Simulate(\{\mathcal{M} + \delta_j\}, {}^i X^G) + \eta_j \tag{3.11}$$

$$j \in \{1, 2, \ldots, \mathcal{W}\} \tag{3.12}$$

$$i \in \{1, 2, \ldots, N\} \tag{3.13}$$

where δ_j is the deviation from the nominal measurement action, $Simulate$ computes the value for a measurement action applied to the part in a specific configuration, and with η_j as measurement noise.

Measurement Simulation

To predict the measurement value obtained from a measurement action: $p(m_j)$, the measurement uncertainty is again approximated by a discrete sampling of the continuous distribution. For a single measurement action, and for each particle a sampling of measurement values is drawn from the distribution of measurement values that would be obtained if that particle was the true state.

The sensor measurement will indicate a distance z traveled along the measurement action $\mathcal{A}$ until reaching the part. We start by examining the distance from the start point along the vector

until the first intersection with the part. Though a crude approximation of the true measurement value, the benefit of this model is that given a measurement action and part pose, the measurement value can be calculated as the intersection of a ray and a triangular mesh. Due to their heavy use in computer graphics, ray-mesh intersection algorithms have been heavily optimized and can be computed in parallel.

Measurement Width: While a ray is infinitely thin, the touch probe's spherical tip has a non-zero diameter, and thus will cast a cylinder rather than a ray. The true value returned by our sensor is the smallest distance until any contact with the part. Ray casting approximates the measurement cylinder by discrete uniformly spaced rays on the cylinder exterior, with the measurement value as the lowest ray-mesh intersection distance from this set.

Measurement Error: Error is caused both by inaccurate start positions and orientations due to robot positioning error, as well as inaccuracies in the sensor. In the most extreme cases error may cause a measurement to move from barely hitting an edge to completely missing the part. Thus it is clear neither adding a constant error term, nor a dependent Gaussian error will accurately model the error.

Instead discrete general method models this error. For each measurement action we make many simulated measurements where we perturb the initial conditions according to an error model for the robot and perturb the measured value according to a model of the sensor. Because our ray-mesh intersection method is cheap, the additional cost these extra simulations add is acceptable.

Bins

$H(X^G|\mathcal{M})$ is calculated by dividing the continuous values $m_{i,j}$ into discrete bins, b_k. The conditional entropy of this measurement action is then:

$$H(X^G|\mathcal{M}) = \sum_k p(b_k)\ H(X^G|b_k) \tag{3.14}$$

where $p(b_k)$ is the prior probability that this measurement will fall into bin b_k and $H(X|b_k)$ is the entropy of the particles within bin b_k. The likelihood of a bin is computed by summing the weights of the measurements in that bin. Defining the weight of the bin, W_k as:

$$W_k = \sum_{i,j} \mathbb{1}(m_{i,j} \in b_k) \cdot w_i \tag{3.15}$$

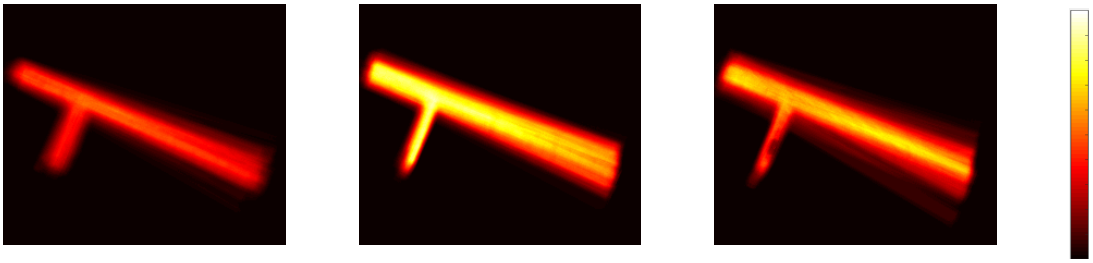
then:

$$p(b_k) = \frac{W_k}{\sum_{i,j} w_i} \tag{3.16}$$

$$= \frac{W_k}{\mathcal{W}} \tag{3.17}$$

Given a bin, the probability of a specific particle is:

$$p({}^iX|b_k) = \frac{\sum_j \mathbb{1}(m_{i,j} \in b_k) \cdot w_i}{W_k} \tag{3.18}$$



(a) Initial Belief (b) 1 Measurement (c) 2 Measurements

Figure 3.5: Heat maps showing information gain

Then the entropy of the bin can be calculated:

$$H(X|b_k) = -\sum_i p({}^iX|b_k)\log(p({}^iX|b_k)) \tag{3.19}$$

Figure 3.4 visualizes this binning process. Three measurement actions (arrows) $\mathcal{M} + \delta_j$ are simulated on three configurations of the part (${}^ix$ = particle i). The intersection between the simulated measurement action and the part determines the measurement value $m_{i,j}$. These measurement values are sorted into bins.

Figure 3.5 visualizes the densely computed information gain for a grid of parallel rays as the "temperature" in a heat map. Initially **(a)** probing into the page is likely to miss the part, so information gain is low for all these measurements and we instead probe sideways. After this first measurement **(b)** the uncertainty is reduced, and probing into the page is more likely to hit the part and provide information. Once this second measurement is performed **(c)**, another measurement in the same place will not provide more information.

Adaptations are made to this process to accommodate the two representations as described next:

3.3.2 Information for Full-State Representation

The full-state representation does not maintain a set of particles over just the goal feature, but rather each particle, X, represents the full $6 \times n$ state. Using these full-state particles for X^G above provides a good metric for localizing every section of the part, but a poor metric for localizing the goal feature. This metric would often suggest to perform measurements on non-datum features that are irrelevant to the location of the goal feature. The error in this metric is due to

$$H(X^G|\mathcal{M}) \neq H(X|\mathcal{M}) \tag{3.20}$$

One approach is to incorporate domain knowledge when designing the full-state representation, by including only the relevant datums necessary for a particular task. Then, any information on this limited full state will be reduction of uncertainty of at least one datum required for the task.

An approach that does not require pruning irrelevant sections from the full state involves combining particles that produce similar configuration for the goal feature. To calculate $H(X^G|\mathcal{M})$

	Measurement Error (cm)	Init. Position Uncertainty (cm)	Init. Angular Uncertainty (rad)	Final Position Error (cm)	Final Angular Error (rad)	Average Computation (s)
Physical Robot	0.05	3	0.08	0.22 $\pm$0.09	0.0047 $\pm$0.002	7.0 $\pm$0.3
Simulation of Robot	0.05	3	0.08	0.17 $\pm$0.09	0.005 $\pm$0.003	7.0 $\pm$0.3
Simulation of Accurate Robot	0.01	3	0.08	0.03 $\pm$ 0.02	0.0004 $\pm$0.0002	7.8 $\pm$0.3

Table 3.1: Results of Experiments

using particles X, measurement actions are used to sort the particles into bins as in described in section 3.3.1. Then, particles that produce sufficiently similar goal feature configurations are treated as identical particles when computing entropy, by combining these into groups L.

$$p({}^{L}X|b_k) = \frac{\sum_{i,j} \mathbb{1}(m_{i,j} \in b_k)\mathbb{1}(i \in L) \cdot w_i}{W_k} \tag{3.21}$$

$$H(X|b_k) = -\sum_{L} p({}^{L}X|b_k) \log(p({}^{L}X|b_k)) \tag{3.22}$$

These group can be constructed by discretizing the space of possible configurations for the goal feature. An issue which this thesis does not address is the balance of a requiring a reasonably small number of particles while maintaining sufficient density for this descretization of goal feature configuration to produce meaningful group sizes.

3.3.3 Information for Independent-State Representation

The independent-state representation does maintain the set of goal feature particles, X^G, however additional steps are needed when computing Eq. 3.11. When simulating $\mathcal{M}$, the robot will measure some section, $\mathcal{S}$, of the part. Simulating the measurement using $X^{\mathcal{S}}$ is straightforward, but leads to computing $IG(X^{\mathcal{S}}|\mathcal{M})$, which is not the desired metric $IG(X^G|\mathcal{M})$.

The independent-state representation makes the approximation that the distribution of transformations between sections are fixed and independent, and this approximation is used to achieve the desired metric. A temporary set of particles $\tilde{X}^{\mathcal{S}}$ is created by sampling transforms $T_G^{\mathcal{S}}$ and applying these transforms to X^G. $\tilde{X^{\mathcal{S}}}$ is used in Eq. 3.11 to generate sample measurement values $\tilde{m}_{i,j}$, which are used in the calculation for bin entropy $H(X|b_k)$. While the independence approximation could be used again in the calculation of bin probabilities $p(b_k)$, this approximation is not needed. Measurements $m_{i,j}$ calculated using $X^{\mathcal{S}}$ are used to calculated $p(b_k)$.

3.4 Experiments

Experiments were performed in simulation and on a physical system to validate the particle filter and measurement selection process for a rigid body. Futher experiments performed in simulation validate the Datum Particle Filter and measurement selection for parts with internal uncertainty.

3.4.1 Robot Results for a Rigid Body

For validating the approach for a rigid body, a robot equiped with a touch probe was used to localize a model part from an airplane. The touch probe consisted of a 0.6mm spherical tip

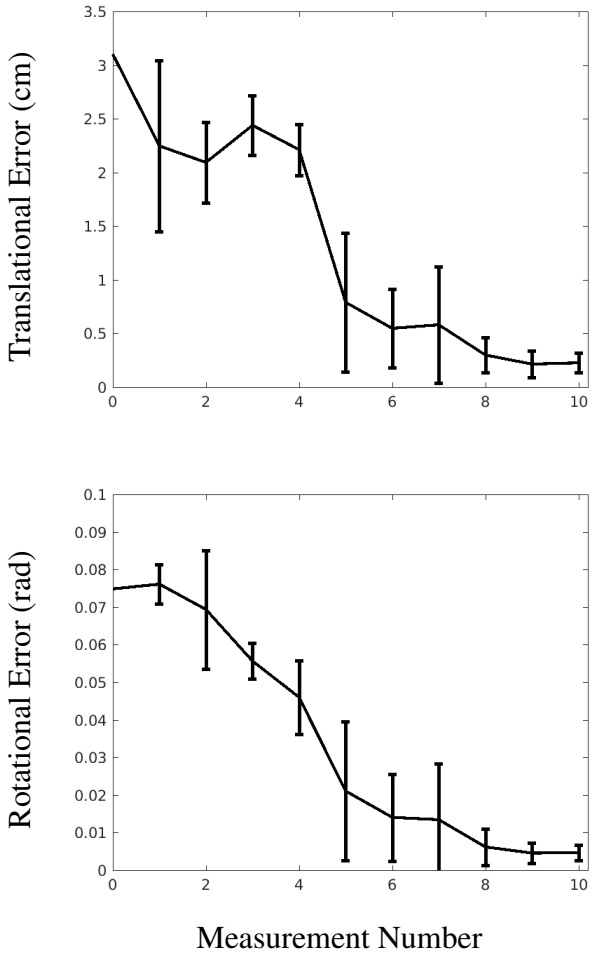


Figure 3.6: Translational and rotational error during localization on the physical robot

mounted to a 100mm rod attached to a 6-D JR3 force/torque sensor. Although contacts in any direction were possible, the sensor was significantly more accurate with the contact force parallel to the shaft, and measurements actions were always chosen to place the contact force in this direction. Under these conditions our sensor exhibited a 1mm repeatability.

After selecting from the set of candidate actions the robot executed the measurement action by first following a trajectory to the start point $\mathcal{A}_p$, designed to avoid collision with the part. The end effector was then controlled to move in a straight line in the direction of the measurement vector $\mathcal{A}_v$ until contact was detected. To improve measurement accuracy a double-touch was implemented: the robot first moved the probe tip at a higher velocity (1cm/s) until contact, backed off slightly, then probed at a much slower velocity (2mm/s). The robot then retreated to a safe point while the particle filter updated and the next action was selected.

We performed 15 localization trials with identical initial beliefs and true part location. Figure 3.6 shows the translational and rotational error of the mean of the belief distribution compared with our best measurement of the true state, averaged over all trials. The error bars show one standard deviation. Summaries are shown in Table 3.1.

3.4.2 Simulation Results for the Datum Particle Filter

The following experiments validate both the full-state representation approach and independent-state representation approach in simulation. The software was implemented in ROS using C++. These experiment simulate a specific task which is common in manufacturing: localizing a target location, defined by datums, to drill a hole on an object. The datum-based particle filter was simulated on a structural component used in aircraft. This is the same object as used in the original rigid-body particle filter paper[47], with adaptations made to allow internal degrees of freedom. The object is composed of 5 precisely manufactured sections, and tolerance between sections was determined by engineering drawings (for precisely defined features), and educated guesses (for loosely defined relationships).

Measurement Selection

The target hole is localized by measuring its referenced datums. Specifically, the pose of the hole feature (green cylinder) in figure 3.1 is defined by an offset distance from the top and right sections shown in figure 3.3a, and the axis of the hole is orthogonal to the front plane. The hole does not exist yet, and thus cannot be measured directly. In order to localize the hole location precisely without direct measurement on the hole, it is assumed that the transformations between the target location and its defining datum sections have very small uncertainty along some dimensions, e.g. the vertical distance between the center of the hole and the top plane.

At each step, the measurement is simulated using ray-mesh intersection algorithms[47]. Potential measurement actions are sampled in the workspace. The information gain for each measurement is calculated based on the current estimated pose of each section. For each measurement performed, candidate actions are evaluated until 500 actions with non-zero information gain have been modeled. Only the measurement action with the largest expected information gain is "performed" in simulation and used to update the belief.

Simulation Results

Both proposed approaches are compared in similar settings. A total of 20 measurements are simulated during each trial. For the full-state representation, a maximum of 800 particles are used. For the independent-state representation, a maximum of 500 particles are used for each section. The simulation uses 5 mesh models for different sections. After each update, the average estimated pose of the hole is computed by averaging the hole poses produced from all particles.

Figure 3.7 shows the comparison between the full-state particle filter and independent-state particle filter. Translational error and rotational error are defined between the estimated pose of the hole and its true state. From the simulation, the errors of the estimated pose decreases rapidly for both approaches after each new measurement is applied on the system.

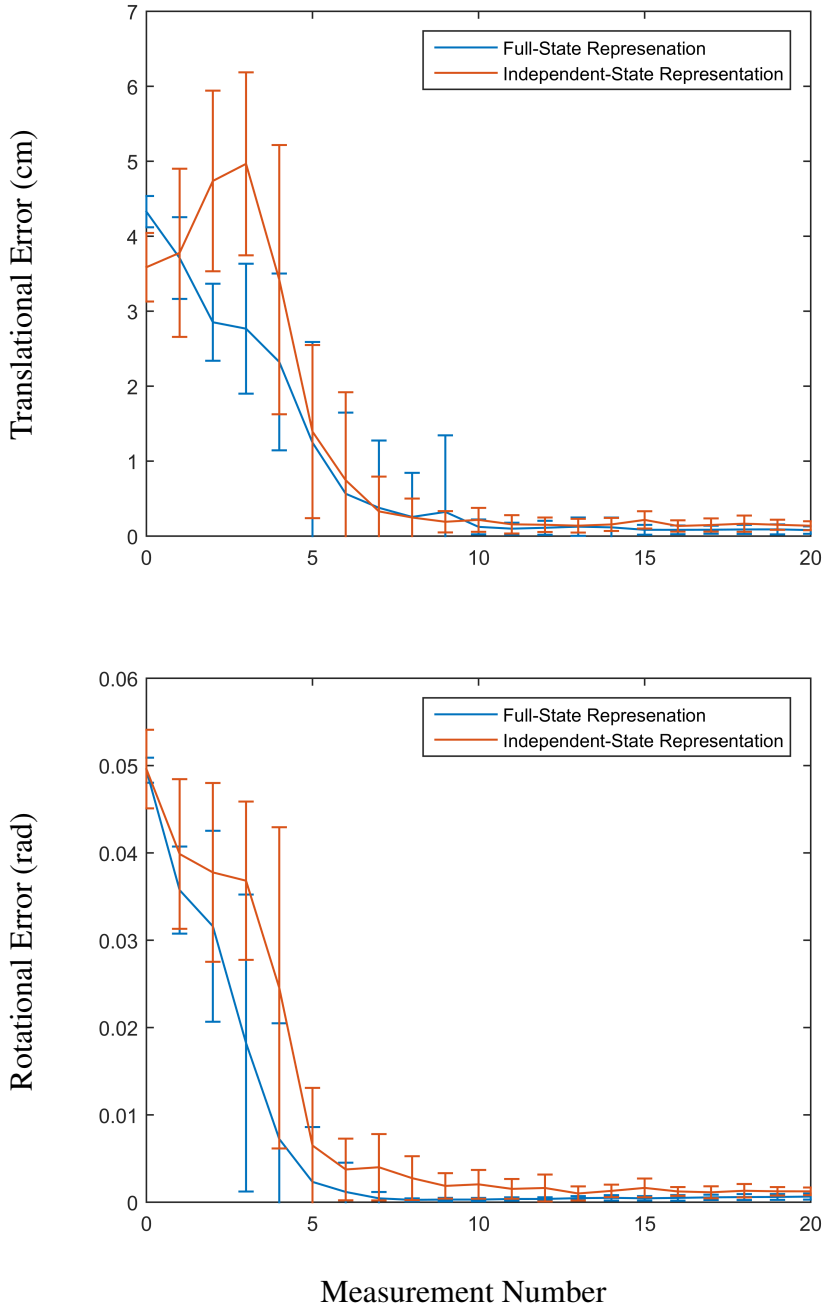


Figure 3.7: Comparison of the accuracy of the update step when using full-state particle filter and independent-state particle filter

Chapter 4

Planning with Contacts for Support

To reach into confined spaces robot arms must be thin, long, and maneuverable. A thin robot may have small actuators with limited torque such that the joints cannot support the weight of the outstretch cantilevered arm. However confined spaces are filled with places to rest for support, and contact forces can compensate for low joint torque. Robot motion planners that take advantage of support extend the reach and improve stability for long thin arms.

This chapter considers motion planning for a robot arm that may experience contact forces. The motion planner's goal is to output a trajectory of joint torques that will move the robot from an initial configuration to a configuration with the end effector in a specified location. Each joint can only apply a limited torque, and the trajectory must respect these limits. The arm experiences forces due to gravity, inertial, friction, and also interaction with the environment, and this dynamics model is described in detail in section 4.1. Contact forces can be helpful by balancing out gravity, or detrimental by blocking a desired motion. Contact forces prove difficult due to their large but local effect.

Section 4.2 describes a solution to the motion planning task that uses trajectory optimization and is strongly based on Contact Invariant Optimization [37]. To better conidition the problem for optimization, the dynamics are augmented to create smooth gradients around contacts. Section 4.3 describes a sampled-based planner capable of using contact forces, and again alterations are made to handle the thin contact manifold.

4.1 Robot Dynamics

When not in contact with the environment the robot model follows the manipulator arm equation [38]:

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + N(\theta, \dot{\theta}) = \tau \tag{4.1}$$

M and C are determined through the robot's known mass characteristics. N includes both gravity and frictional terms. In practice M, C, and N do not need precise estimation, because closed loop controllers on the robot can compensate for errors.

The robot arm has defined sections which may be in contact with the environment. For simplicity in calculations, these contact sections are approximated by spheres, shown in Figure 4.1.

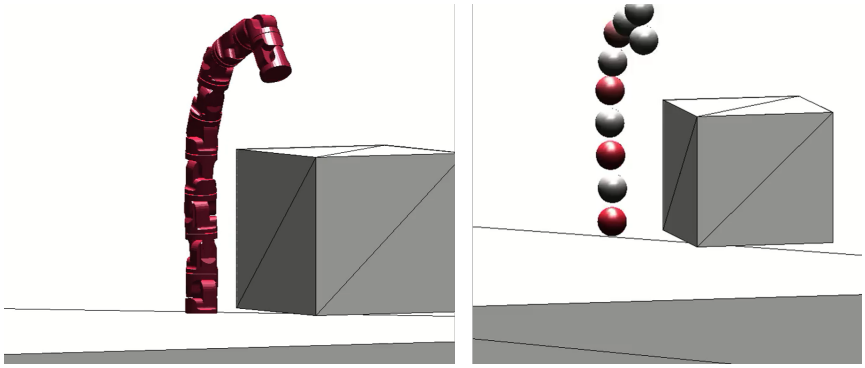


Figure 4.1: The visually accurate robot model (left) and the spherical approximation (right)

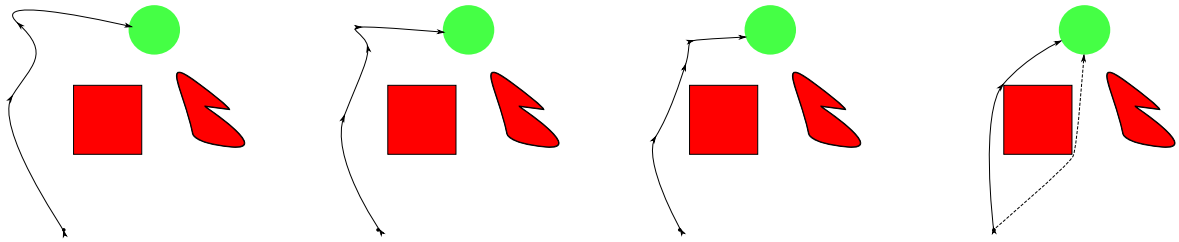


Figure 4.2: Optimizing the shortest path trajectory from an initial path converges to a local best, but possibly misses the globally best solution.

The environment is modeled as a triangular mesh, and contact occurs when a contact section intersection the mesh. The contact force is determined by a spring model, and so the magnitude is proportional to the penetration distance, and pushing the section out of the environment. The physical robot and environment are both constructed of metal, so while a large spring constant provides a more accurate model this also causes numerical instabilities. Each trajectory is simulated as a series of discrete time steps, and a large spring constant will result in giant forces for even the slight penetration this discretization introduces.

4.2 Trajectory Optimization

Trajectories for a robotic arm can be created through trajectory optimization, a process that progressively improves the cost of a trajectory. A trajectory s is a discrete list of robot states s_t at increasing points in time, where the state contains both the joint angles and joint velocities, though for sufficiently slow trajectories the velocities may be ignored. To transition between timesteps, joint torques are calculated through inverse dynamics, and these torques must remain below the actuator torque limit.

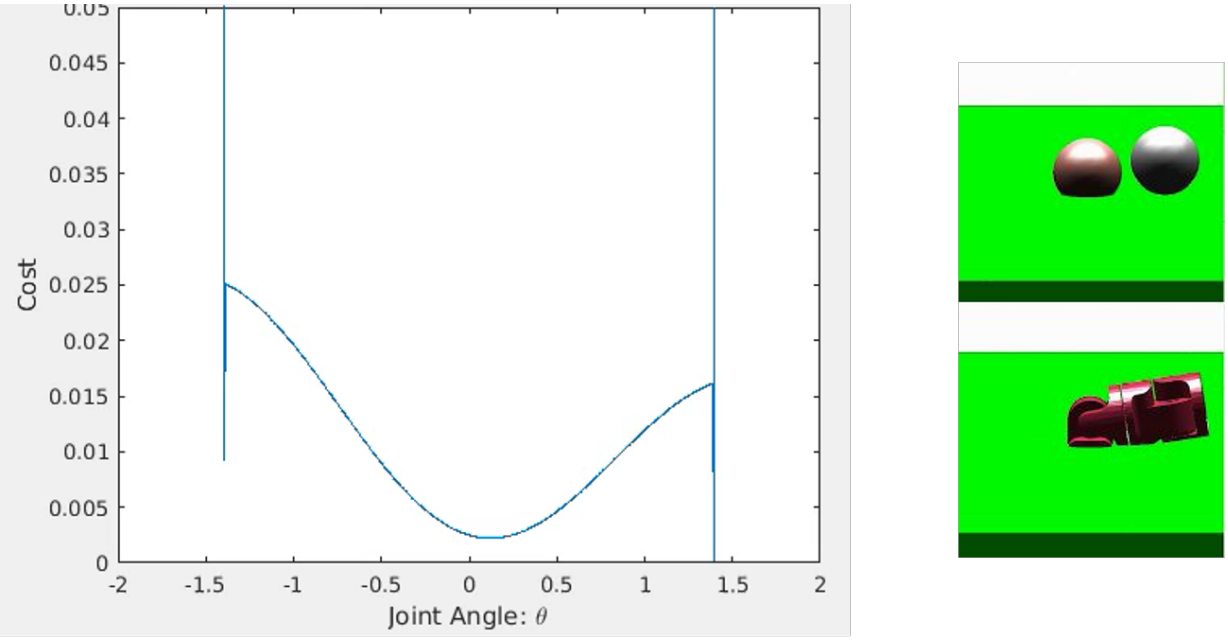


Figure 4.3: Naive cost function, with sharp changes around contact locations

Since the trajectory requires both maintaining low joint torques and reaching a goal point, it seems natural to define a cost function the penalizes joint torque and the distance from the goal:

$$L(s) = \sum ||\tau_{joint}||^2 + Cost_{task} \tag{4.2}$$

Figure 4.3 shows such a cost function for a single state for a simple robot arm with one movable base joint and one contact section on the end effector, with a goal position with the robot leaning to the right. For simplicity, in this figure the inertial forces are ignored and the only forces considered are gravity and contact. Although the minimum of this cost function uses contact to reduce joint torque while reaching the goal, this cost function is poorly conditioned for gradient descent optimization. Nearly all initializations will result in convergence to the local minimum with the wide basin of attraction at $\theta = 0.1$. To effectively use trajectoy optimization for this task, this cost function is augmented to be better suited for optimization.

4.2.1 Auxiliary Variables

The key approach used here is softening of contacts through the introduction of auxiliary variables, which drastically smooth the cost function at the expense of increasing the dimensionality of the state space. Appended to each state s_t is a set of auxiliary continuous variables c_t, one for each section of the robotic arm allowed to make contact with the environment. The variables c_t regulate the magnitude of the normal and frictional contact forces and will be described in detail in the next section. Without this augmentations, slight changes in these joint configurations result in slight motion of the robot which may result in huge changes in contact forces. Additionally without artificial terms in the cost function to model contact, the cost function is blind to potential contacts that may be close and useful.

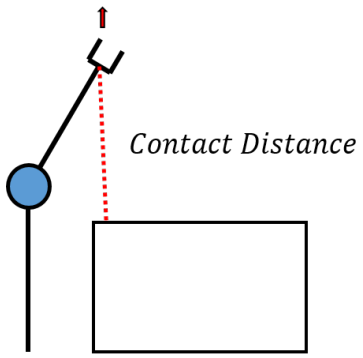


Figure 4.4: Contact distance for the end effector section

4.2.2 Smoothing the Cost Function

A local optimal trajectory s^* is computed by minimizing the cost function of the form

$$L(s) = L_{ContactViolation} + L_u \tag{4.3}$$
$$+ L_{ObjectPenetration} + L_{Goal} \tag{4.4}$$

$L_{ObjectPenetration}$ and L_{Goal} are straightforward, while the interplay between $L_{ContactViolation}$ and L_u provide the interesting structure allowing the optimization to find paths which use supporting contacts.

Cost $L_{ContactViolation}$

Each section of the robot able to make contact has an associated variable $c_t > 0$ at each time step. c intuitively represents the strength of the contact forces. Since if the robot is not physically in contact with the world there will not be a contact force the cost $L_{ContactViolation}$ is introduced to penalize non-zero values of c when the robot is not in contact.

$$L_{ContactViolation} = \sum_t \sum_i c_{t,i} d_{t,i}$$

Non-zero values for c are intentionally allowed when the robot is not in contact with the environment even though this is not physically realistic as this makes the cost function smooth. However when optimizing $c_{t,i}$ will tend towards 0 unless the robot is in contact.

Figure 4.4 shows the contact distance for a section of a robotic arm. Since this distance is substantial, the magnitude of the associated contact variable will determine the contribution to cost.

Figure 4.5 shows a robot arm in a variety of configurations and with several values of contact variables. The transparent configurations are physically infeasible and suffer a $L_{ContactViolation}$ cost. To ensure the final optimized trajectory is feasible, the contact violation cost is weighted highly towards the end of optimization, thus the optimization will not converge on any of the transparent configurations.

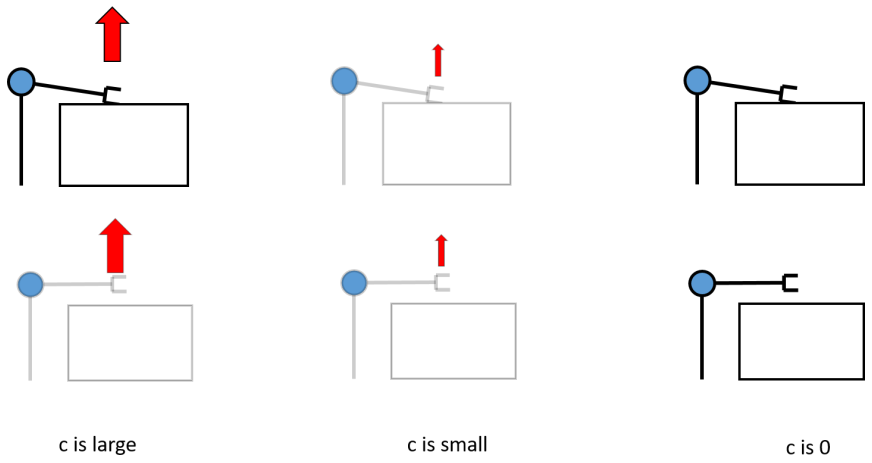


Figure 4.5: Arm configuration and simulated (not realistic) contact forces for a variety of arm configurations and continuous contact variables

Cost L_u

The cost L_u penalizes the input torque necessary to follow the trajectory specified. As discussed the physical contact forces are extremely sensitive to robot configuration, so to soften the forces we instead compute the contact forces that minimize the joint torque. When the trajectory is executed on the robot the joint controller will take responsibility for finding the slight adjustments in configuration to reach the desired contact forces. The path planner does not have to worry about the minute adjustments for a model that will not match reality to that detail anyway. Instead this path planner just needs to estimate the best contact forces possibly, according to the the following quadratic programming problem:

$$f, u = argmin_{\tilde{f},\tilde{u}} ||J^T \tilde{f} + \tilde{u} - \tau_{Free}|| \tag{4.5}$$

$$+ \tilde{f}^T W \tilde{f} + \tilde{u}^T R \tilde{u} \tag{4.6}$$

The input control regularization R is chosen based on the desired penalization of joint inputs. The contact force regularization W is dependent on the values of c, with

$$W_{j,j} = \frac{1}{c_{i,t}^2 + 1}$$

If c is large then the robot is in contact, the force regularization is small, so the contact force can be large. If c is small the robot is not near any contact location, the force regularization is large, thus the contact force is heavily penalized and will be small.

Figure 4.5 shows a robot arm in a variety of configurations and with several values of contact variables. The red arrows indicate the simulated contact force. If the contact variable is large, a large force can be applied to reduce joint torques even when there is no physical contact. This serves to reduce the cost due to joint torques. The competing cost $L_{ContactViolation}$ prevents this contact variable from growing too large for infeasible contacts.

Figure 4.6 shows this augmented cost function for the arm in Figure 4.3. The dimensionallity of this cost function has increased, yet now there is a strictly descending path from the local

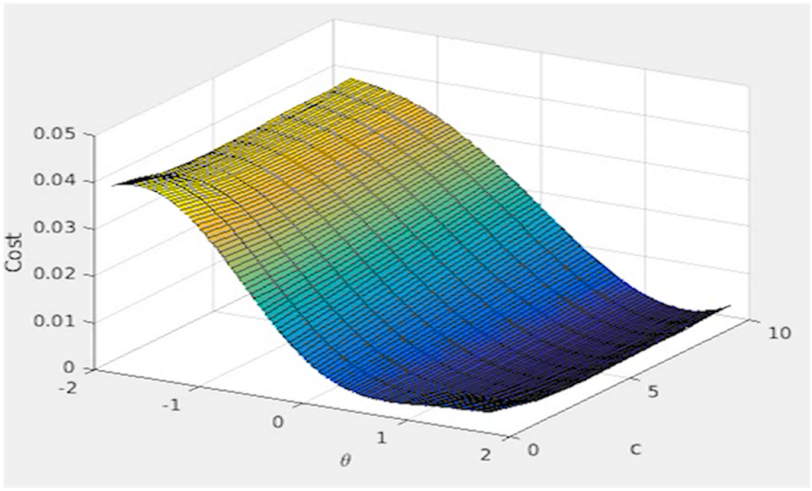


Figure 4.6: The cost function for the arm in Figure 4.3, augmented with the auxiallary contact variable

minima of the naive cost function, $\theta = 0.1$, to the global minima. The basin of attraction for the global minima has increased dramatically.

Other Cost Terms

$L_{Obstacle}$ penalizes penetration of the robot into the environment which is calculated using the robot forward kinematics. For simplicity, the same simplified spherical model is resued here. This addition is necessary since the artificial contact model created by L_u and $L_{ContactViolation}$ no longer ensures obstacle penetration produces high joint torques.

$$L_{Obstacle} = \sum d_{ObstaclePenetration} \tag{4.7}$$

L_{Goal} is a penalty on the distance from the robot end effector at the last state s_T to the goal location.

$$L_{Goal} = ||EndEffector_T - goal|| \tag{4.8}$$

4.2.3 Initialization Challenges

Although the cost function augmentation described above greatly improves the conditioning of the cost function with respect to contacts, in practical problems this cost function is still plagued with local minima. As a simple example, consider the situation in Figure 4.7, where most initial trajectories will result in the arm never reaching the goal "X".

Straightfoward initialization techniques include random initialization with random restarts, and linear joint angle interpolation between starting configuration and a configuration that achieves the goal. These initializations work well in open environments, but poorly in confined spaces. Thus a sample based planner is employed to provide a diversity of good initializations.

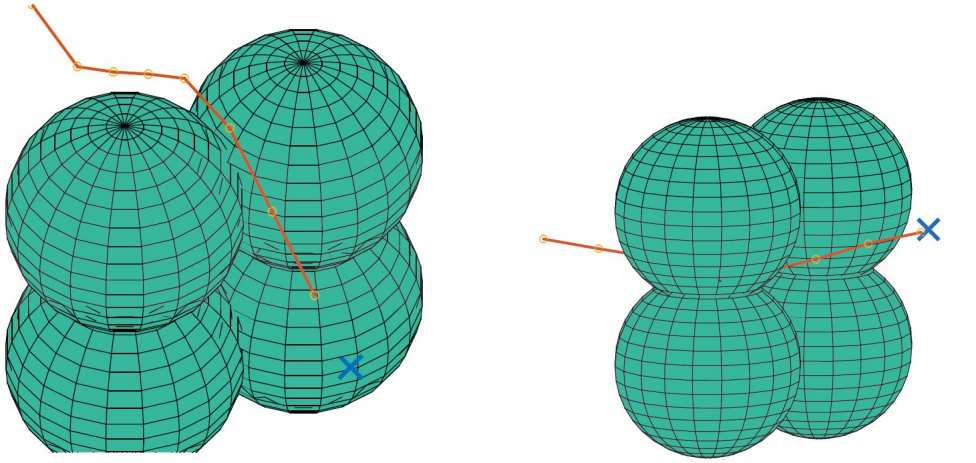


Figure 4.7: Trajectory optimization finding a local minimum (left) and the global minimum (right) for an arm attempting to reach the blue "X"

4.3 Sample Based Planning

It is possible to solve planning problems using a sample based planner, but once again the thin contact manifold causes challenges. Sample based planners probabilistically sample configurations to explore the feasible space, and through sufficent exploration seek to find the trajectories to a goal while obeying constraints. Specifically, this section employs the RRT, described in Algorithm 1. The basic RRT algorithm maintains a tree of trajectories (edges) to configurations (nodes), and grows this tree by ending to randomly sampled configuration. As discussed in Section 2.2.2, the basic linear extension fails when the only allowable space for the trajectory to grow is a thin manifold, as in the case of a supporting contact.

Figure 4.8 illustrates this contact manifold. In this example, in some regions of configuration space a contact is not needed to support the arm, and trajectories may progress in any direction. The thin ribbon shows the manifold in configuration space where the arm makes contact with the environment. In the left region the contact is not needed and a trajectory is free to break the contact, however, the arm is able to reach the right region only by using the support of a contact force. In cases where the configuration must remain on this manifold, linearly extending towards a randomly sampled point will quickly cause the trajectory to leave the manifold, and make little progress in extending the tree.

4.3.1 Adaptation to Encourage Contacts

This section describes the policyRRT algorithm, which by using a policy to guide the path extension is able to follow the contact manifold.

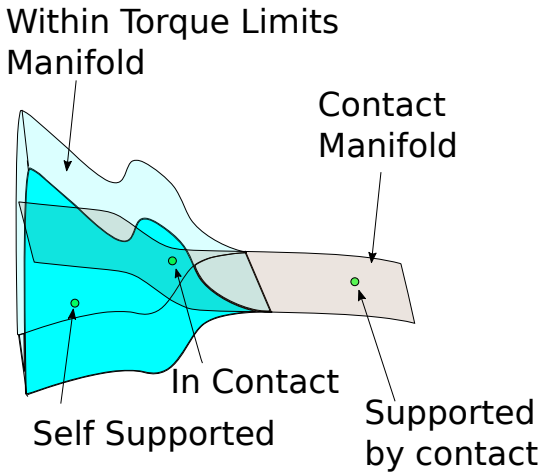


Figure 4.8: Illustration of valid configuration space for an arm potentially supported by contacts

Given:

$X \in \mathbb{R}^d$:	d-dimensional configuration space	(4.9)
$X_{obs} \subset X$:	obstacles in the configuration space	(4.10)
$X_{free} = X \setminus X_{obs}$:	free space	(4.11)
$x_{start} \in X_{free}$:	starting configuration	(4.12)
$X_{goal} \subset X_{free}$:	set of goal configurations	(4.13)
$policy : X \times X \to TX$	Map from and initial and goal state to a motion	(4.14)

The algorithm aims to find a path in X_{free} from x_{start} to $x_{goal} \in X_{goal}$.

Algorithm 3 $T = (V, E) \leftarrow$ policyRRT(x_{start})

```
T ← InitTree(x_start)
while GoalNotReached(T, X_goal) do
    x_rand ∈ X ← Sample()
    x_nearest ← Nearest(T, x_rand)
    T ← followPolicy(x_nearest, x_rand, T, extensionLimit)
    if ExtensionSuccessful then
        T ← followPolicy(x_new, x_goal ∈ X_goal, T, ∞)
    end if
end while
```

Algorithm 4 $T = (V, E) \leftarrow$ followPolicy(x_{begin}, x_{end}, T, iterLimit)

```
T_new ← InitTree(x_begin)
x_prev ← x_begin
x_new ← policy(x_prev, x_end)
i ← 0
while MovingTowardsGoal(x_new, T_new) and [2] Nearest(T, x_new) == x_begin and [2] i < iterLimit do do
    T_new ← AddNode(x_new, x_prev, T_new)
    x_prev ← x_new
    x_new ← policy(x_prev, x_end)
    i ← i + 1
end while
T ← AddTreeToTree(T_new, T, x_begin)
```

As motivation for this approach, in many environments it is relatively easy to compute a reactive policy that can make forwards progress towards a goal while avoiding collision with obstacles, such as a potential field approach, but such policies are prone to getting stuck in local minima. PolicyRRT is a variant of RRT that seeks to combine the efficient, goal directed trajectory generation of reactive policies with the ability of RRT to find paths around local minima/dead ends. PolicyRRT fundamentally behaves like a standard RRT by growing a tree by iteratively extending towards a randomly sampled configuration from the nearest point in a search

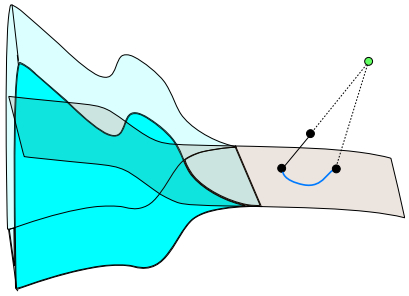


Figure 4.9: Illustration of Policy RRT extension on the contact manifold

tree T. PolicyRRT uses the reactive policy to move towards the random sample, potentially enabling the planner to avoid collisions with obstacles during the extension. When PolicyRRT is successful in extending towards the sample (does not collide with an obstacle and does not get stuck in a local minima) it then follows the policy to extend towards the goal. To avoid oversampling regions associated with local minima, when PolicyRRT extends towards a point from some vertex $v \in T$ PolicyRRT halts extension when the trajectory leaves the Voronoi cell of v. Extension is also halted once the policy ceases to make progress (reaches a local minima) or exceeds a maximum number of iterations.

The policy used in this algorithm should incorperate knowledge of the value of regions of configuration space to guide the extension function. For the long arm planning with contact problem discussed in this chapter, the policy must direct the trajectory to maintain contact when necessary so the extension will be productive, but must allow the arm to break current and make new contacts for sufficiently exploration. Fortunately, the gradient of the cost function use in the previous section for trajectory optimization (Eq. 4.4) has exactly these properties. Figure 4.10 illustrates both the linear extension and this policy extension towards a randomly sampled goal configuration. Leaving the contact manifold around these configurations causes large torques as the contact forces are removed, however because of the augmentation done in section 4.2, the gradient remains smooth so the policy pushes the arm back toward the support configurations.

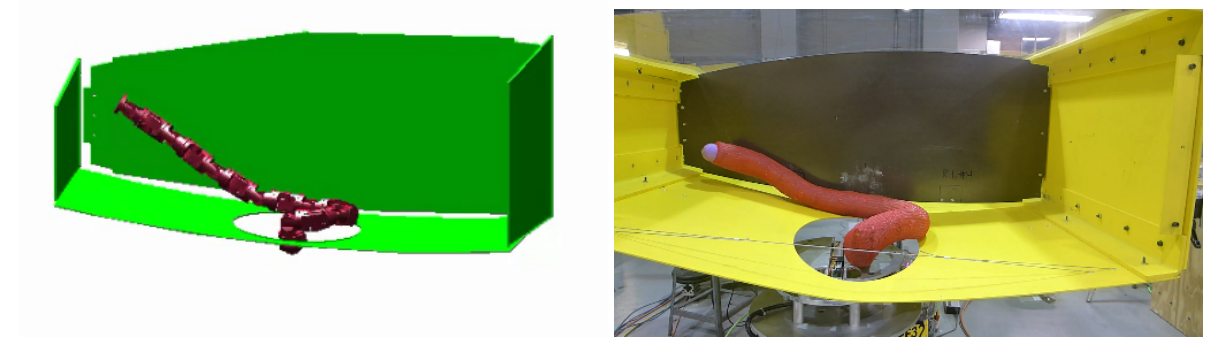


Figure 4.10: Simulated (left) and actual (right) snake arm robots executing motions planned in a model of an airplane wing section

4.4 Experiments

Trajectories were generated for a 16-DOF snake robot arm in a model of a wing section by combining the approaches shown above. The initial configuration and goal position for the end effector were chosen manual and contained goal positions at the corners of the available volume so that the arm must maintain contact for support.

First, the policyRRT of section 4.3 was used to generate a trajectory a for the arm to a goal position while avoiding collisions between the robot and the wing, however ignoring the torque limits of the joints. This trajectory was used as an initialization for the trajectory optimization approach of section 4.2

Chapter 5

Conclusion and Future Work

This thesis has explored the challenges and benefits of contacts in two robotics problems. Contacts can provide accurate sensory information and aid in robot motion, but the local nature of contact forces adds difficulties. Specifically considered were a localization problem using a touch probe, and a planning problem using contacts for support. Chapter 2 reviewed existing methods for localization and planning using contacts.

In localization, touch probing yields accurate measurements about the location of the surface of a part. However, only a thin manifold of configurations are consisent with the measurement, which presents challenges when using this measurement to update the belief of the object. Chapter 3 first reviewed work done in collaboration with Shiyuan Chen [47] on designing a particle filter capable of updating using contact measurement. It then extends this particle filter to parts with internal degrees of freedom, where the goal is to localize a specific section defined by datums. This chapter describes how to use the particles to define an information gain metric that can be calculated efficiently, and this metric is used to select informative probing actions.

In motion planning, contact forces can reduce joint torques in a cantilevered robotic arm. However, utilizing contacts involves discovering and traversing a thin manifold through configuration space, which presents difficulties for common methods of planning. Chapter 4 describes approaches for trajectory optimization and samples based planners for handling the effects of contacts. The heart of the approach is a dynamics model that can provide unrealistically optimistic contact forces at a distance, and thereby providing useful gradient information to a cost function. The degree to which forces at a distance can be used is controllable, enabling the enforcement of realistic final trajectories.

There is a plethora of future directions for this work. Perhaps most obviously, these two approaches can be combined on the same robot. The same contacts that can be used for planning can also be used for planning, augmenting the planning cost function further to encourage informative contact locations. Progress can still be made independently.

This thesis demonstrated the localization approach on only a single part in practice, and only a limited set in simulation. The datum based particle filter can be applied to a wide variety of environments consisting of objects with bayesian linked poses. It's power is the ability to focus on only the information gathering actions that aid a task. In addition, through collaborators we have identified a practical application in need of the approach described involving a robot in a manufacturing setting.

Obvious extensions exist to the planning methods describes. The sample based planner will yield only feasible trajectories, not optimal trajectories. The methods of RRT* can be used to produce better trajectories for initializing the optimization. In addition, when the trajectory optimization fails the current solution is a random restart. However, this new optimization result in convergence to the same local minima. The same approaches employed in section 4.3 can be used to avoid repeated exploration of the same region. Finally, the dynamics model and cost function used were constructed specifically for contacts, but the methods used may be extended to other challenges in robotics.

Bibliography

[1] Seyed Reza Ahmadzadeh and Postal Code Tel. Modeling of Hyper-Redundant Manipulators Dynamics and Design of Fuzzy Controller for the System. pages 248–253, 2005. doi: 10.1109/KIMAS.2005.1427089.

[2] Tomas Akenine-Möllser. Fast 3D Triangle-Box Overlap Testing. *Journal of Graphics Tools*, 6(1):29–33, 2001. ISSN 1086-7651. doi: 10.1080/10867651.2001.10487535. 3.1.4

[3] Chris Atkeson and et. al. What Happened at the DARPA Robotics Challenge, and Why? 2015. 1, 2.2.1

[4] Dmitry Berenson, Siddhartha S. Srinivasa, Dave Ferguson, and James J. Kuffner. Manipulation planning on constraint manifolds. *2009 IEEE International Conference on Robotics and Automation*, i:625–632, 2009. ISSN 1050-4729. doi: 10.1109/ROBOT.2009.5152399. 2.2.2

[5] W.J. Book. Bracing micro/macro manipulators control. *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 2362–2368, 1994. ISSN 10504729. doi: 10.1109/ROBOT.1994.350933. URL `http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=350933$\delimiter"026E30F$nhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=350933`. 2.2.1

[6] T. Bretl. Motion Planning of Multi-Limbed Robots Subject to Equilibrium Constraints: The Free-Climbing Robot Problem. *The International Journal of Robotics Research*, 25 (4):317–342, 2006. ISSN 0278-3649. doi: 10.1177/0278364906063979. URL `http://rms.ae.illinois.edu/papers/Bretl2006.pdf`. 2.2.1

[7] Joel W. Burdick. Robotic endoscopy, aug 1994. URL `https://www.google.com/patents/US5337732`.

[8] Gregory S. Chirikjian and Joel W. Burdick. Hyper-redundant manipulator. *IEEE Robotics and Automation Magazine*, 1(4):22–29, 1994. ISSN 10709932. doi: 10.1109/100.388263.

[9] Gregory S. Chirikjian and Joel W. Burdick. Modal approach to hyper-redundant manipulator kinematics. *IEEE Transactions on Robotics and Automation*, 10(3):343–354, 1994. ISSN 1042296X. doi: 10.1109/70.294209.

[10] D Choukroun, Y Oshman, Senior Member, Control Conference, Los Angeles, and Y Oshman. Novel Quaternion Kalman Filter. 42(1), 2006. 2.1.2

[11] Robin Deits and Russ Tedrake. Footstep planning on uneven terrain with mixed-integer

convex optimization. *IEEE-RAS International Conference on Humanoid Robots*, 2015-February:279–286, 2015. ISSN 21640580. doi: 10.1109/HUMANOIDS.2014.7041373. 2.2.1, 2.2.3

[12] Andrea Del Prete, Francesco Nori, Giorgio Metta, and Lorenzo Natale. Control of contact forces: The role of tactile feedback for contact localization. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 4048–4053. IEEE, 2012. 2.1.1

[13] Florian Enner, David Rollinson, and Howie Choset. Motion estimation of snake robots in straight pipes. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 5168–5173, 2013. ISSN 10504729. doi: 10.1109/ICRA.2013.6631316.

[14] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Distance transforms of sampled functions. *Cornell Computing and Information Science Technical Report TR20041963*, 4:1–15, 2004. ISSN 1557-2862. doi: 10.4086/toc.2012.v008a019. URL `http://ecommons.library.cornell.edu/handle/1813/5663`. 3.1.4, 3.1.4

[15] Jeremy A Fishel and Gerald E Loeb. Sensing tactile microvibrations with the biotaccomparison with human sensitivity. In *Biomedical Robotics and Biomechatronics (BioRob), 2012 4th IEEE RAS & EMBS International Conference on*, pages 1122–1127. IEEE, 2012. 2.1.1

[16] Dieter Fox. Adapting the sample size in particle filters through KLD Sampling. *Intl Jour of Robotics Research*, 22(12):985–1004, 2003. 3.1.4

[17] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Active Markov localization for mobile robots. *Robotics and Autonomous Systems*, 25(3-4):195–207, 1998. ISSN 09218890. doi: 10.1016/S0921-8890(98)00049-9.

[18] Marco Gabiccini, Alessio Artoni, Gabriele Pannocchia, and Joris Gillis. A Computational Framework for Environment-Aware Robotic Manipulation Planning. *International Symposium on Robotics Research (ISRR)*, 2015.

[19] Chaohui Gong, Matthew Tesch, David Rollinson, and Howie Choset. Snakes on an Inclined Plane: Learning an Adaptive Sidewinding Motion for Changing Slopes. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (Iros):1114–1119, 2014. ISSN 21530866. doi: 10.1109/IROS.2014.6942697.

[20] Aaron Greenfield, Alfred A. Rizzi, and Howie Choset. Dynamic ambiguities in frictional rigid-body systems with application to climbing via bracing. *Proceedings - IEEE International Conference on Robotics and Automation*, 2005(April):1947–1952, 2005. ISSN 10504729. doi: 10.1109/ROBOT.2005.1570398. 2.2.1

[21] Frank L Hammond, Rebecca K Kramer, Qian Wan, Robert D Howe, and Robert J Wood. Soft tactile sensor arrays for micromanipulation. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 25–32. IEEE, 2012. 2.1.1

[22] Paul Hebert, Thomas Howard, Nicolas Hudson, Jeremy Ma, and Joel W. Burdick. The next best touch for model-based localization. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 99–106, 2013. ISSN 10504729. doi: 10.1109/ICRA.2013.6630562. 2.1.1

[23] R.L. Hollis and R. Hammer. Real and virtual coarse-fine robot bracing strategies for precision assembly. *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 767–774, 1992. doi: 10.1109/ROBOT.1992.220276. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=220276. 2.2.1

[24] Kaijen Hsiao and Leslie Pack Kaelbling. Task-Driven Tactile Exploration. *Proceedings of Robotics Science and Systems*, 2010. ISSN 2330765X. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.165.9204{&}rep=rep1{&}type=pdf.

[25] Shervin Javdani, Matthew Klingensmith, J. Andrew Bagnell, Nancy S. Pollard, and Siddhartha S. Srinivasa. Efficient touch based localization through submodularity. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1828–1835, 2013. ISSN 10504729. doi: 10.1109/ICRA.2013.6630818. 2.1.1, 3

[26] Shervin Javdani, Andreas Krause, Yuxin Chen, and J Andrew Bagnell. Near Optimal Bayesian Active Learning for Decision Making. 33, 2014.

[27] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics, 1997.

[28] Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal. STOMP: Stochastic trajectory optimization for motion planning. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 4569–4574, 2011. ISSN 10504729. doi: 10.1109/ICRA.2011.5980280.

[29] Rudolph E Kalman and Richard S Bucy. New results in linear filtering and prediction theory. *Journal of basic engineering*, 83(3):95–108, 1961.

[30] Rudolph Emil Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[31] Nikos C. Karavas, Nikos G. Tsagarakis, and Darwin G. Caldwell. Design, modeling and control of a series elastic actuator for an assistive knee exoskeleton. *Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 1813–1819, 2012. ISSN 21551774. doi: 10.1109/BioRob.2012.6290757.

[32] Michael C Koval, Nancy S Pollard, and Siddhartha S Srinivasa. Pre- and Post-Contact Policy Decomposition for Planar Contact Manipulation Under Uncertainty. pages 1–27, 2011. 2.1.1

[33] Michael C. Koval, Mehmet R. Dogar, Nancy S. Pollard, and Siddhartha S. Srinivasa. Pose estimation for contact manipulation with manifold particle filters. *IEEE International Conference on Intelligent Robots and Systems*, (Section 3):4541–4548, 2013. ISSN 21530858. doi: 10.1109/IROS.2013.6697009. 1.2.1, 2.1.1, 3.1.2

[34] S M LaValle. Rapidly-Exploring Random Trees: A New Tool for Path Planning. *In*, 129:98–11, 1998. doi: 10.1.1.35.1853. URL http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Rapidly-exploring+

random+trees:+A+new+tool+for+path+planning{#}0. 2.2.2, 2.2.2

[35] JY Lew and W.J. Book. Hybrid control of flexible manipulators with multiple contact. *[1993] Proceedings IEEE International Conference on Robotics and Automation*, pages 242–247, 1993. doi: 10.1109/ROBOT.1993.292153. URL http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=292153$\delimiter"026E30F$nhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=292153. 2.2.1

[36] Ivanescu Mircea and Cojocaru Dorian. Hyper Redundant Manipulators. *Advanced Strategies for Robot Manipulators*, pages 27–60, 2008.

[37] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics*, 31(4):1–8, 2012. ISSN 07300301. doi: 10.1145/2185520.2335394. 2.2.3, 4

[38] Richard M Murray, Zexiang Li, S Shankar Sastry, and S Shankara Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994. 1.2.2, 4.1

[39] Heui Jae Pahk and Woo Jung Ahn. Advanced manufacturing Technology. 3:442–449, 1996.

[40] Federico Parietti and H. Harry Asada. Supernumerary Robotic Limbs for aircraft fuselage assembly: Body stabilization and guidance by bracing. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1176–1183, 2014. ISSN 10504729. doi: 10.1109/ICRA.2014.6907002.

[41] Anna Petrovskaya and Oussama Khatib. Global Localization of Objects via Touch. *IEEE Transactions on Robotics*, 27(3):569–585, 2011. ISSN 1552-3098. doi: 10.1109/TRO.2011.2138450. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5784199. 2.1.1

[42] Anna Petrovskaya and Andrew Y. Ng. Probabilistic mobile manipulation in dynamic environments, with application to opening doors. *IJCAI International Joint Conference on Artificial Intelligence*, pages 2178–2184, 2007. ISSN 10450823.

[43] M. Posa, C. Cantu, and R. Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2013. ISSN 0278-3649. doi: 10.1177/0278364913506757. URL http://ijr.sagepub.com/content/33/1/69.short. 2.2.3

[44] Mabaran Rajaraman, Michael Dawson-Haggerty, Kenji Shimada, and David Bourne. Automated workpiece localization for robotic welding. In *Automation Science and Engineering (CASE), 2013 IEEE International Conference on*, pages 681–686. IEEE, 2013. 2.1.1

[45] Nathan Ratliff, Matt Zucker, and J Andrew Bagnell. CHOMP : Gradient Optimization Techniques for Efficient Motion Planning. pages 489–494, 2009.

[46] David Rollinson, Steven Ford, Ben Brown, and Howie Choset. Design and Modeling of a Series Elastic Element for Snake Robots. *Volume 1: Aerial Vehicles; Aerospace Control; Alternative Energy; Automotive Control Systems; Battery Systems; Beams and Flexible Structures; Biologically-Inspired Control and its Applications; Bio-Medical and Bio-*

Mechanical Systems; Biomedical Robots and, page V001T08A002, 2013. doi: 10.1115/DSCC2013-3875. URL http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?doi=10.1115/DSCC2013-3875.

[47] Brad Saund, Shiyuan Chen, and Reid Simmons. Touch Based Localization of Parts for High Precision Manufacturing [PREPRINT]. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, in press. 1.2.1, 3, 3.1, 3.4.2, 3.4.2, 5

[48] Bradley Saund and Russell DeVlieg. High accuracy articulated robots with CNC control systems. *SAE International Journal of Aerospace*, 6(2):1–6, 2013. ISSN 19463855. doi: 10.4271/2013-01-2292. URL http://www.sae.org/technical/papers/2013-01-2292.

[49] Simon J Sheather et al. Density estimation. *Statistical Science*, 19(4):588–597, 2004.

[50] B. W. Silverman. Density Estimation for Statistics and Data Analysis. *Biometrical Journal*, 30(7), 1986. ISSN 03233847. doi: 10.1002/bimj.4710300745.

[51] Rangaprasad Arun Srivatsan, Gillian T Rosen, D Feroze Naina Mohamed, and Howie Choset. Estimating SE (3) elements using a dual quaternion based linear Kalman filter. (3). 2.1.2

[52] Cyrill Stachniss, G Grisetti, and W Burgard. Information Gain-based Exploration Using Rao-Blackwellized Particle Filters. *Robotics: Science and Systems*, pages 65–72, 2005. ISSN 2330765X. URL http://pdf.aminer.org/000/685/672/information{_}gain{_}based{_}exploration{_}using{_}rao{_}blackwellized{
pdf.

[53] Matthew Tesch, Kevin Lipkin, Isaac Brown, Ross Hatton, Aaron Peck, Justine Rembisz, and Howie Choset. Parameterized and Scripted Gaits for Modular Snake Robots. *Advanced Robotics*, 23(9):1131–1158, 2009. ISSN 0169-1864. doi: 10.1163/156855309X452566.

[54] Sebastian Thrun, D Fox, and W Burgard. Monte carlo localization with mixture proposal distribution. *Proceedings of the National Conference on*, pages 859–865, 2000. URL http://www.aaai.org/Papers/AAAI/2000/AAAI00-132.pdf. 2.1.1, 3.1.2

[55] Sebastian Thrun, Burgard Wolfram, and Fox Dieter. Probabilistic Robotics. pages 1999–2000, 2000. ISSN 00010782. doi: 10.1145/504729.504754. 1.2.1, 2.1.1, 3

[56] S. et al. Tonneau. An efficient acyclic contact planner for multiped robots. *The International Journal of Robotics Research*, pages 1–11, 2016. 2.2.1

[57] Zhenhua Xiong, Michael Yu Wang, Senior Member, and Zexiang Li. A Near-Optimal Probing Strategy for Workpiece Localization. 20(4):668–676, 2004.

[58] Hiroya Yamada, Makoto Mori, and Shigeo Hirose. Stabilization of the head of an undulating snake-like robot. *IEEE International Conference on Intelligent Robots and Systems*, pages 3566–3571, 2007. doi: 10.1109/IROS.2007.4399390.

[59] Hong-Tzong Yau and Chia-Hsiang Menq. An automated dimensional inspection environment for manufactured parts using coordinate measuring machines. *International Journal of Production Research*, 30(7):1517–1536, 1992. ISSN 0020-7543. doi: 10.1080/00207549208948105. URL http://www.informaworld.com/openurl?

genre=article{&}doi=10.1080/00207549208948105{&}magic=
crossref||D404A21C5BB053405B1A640AFFD44AE3.