Localizing External Contact Using Proprioceptive Sensors: The Contact Particle Filter

by

Lucas Manuelli

A.B., Princeton University (2012)
Masters of Science, Massachusetts Institute of Technology (2015)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2018

Author .....
Department of Electrical Engineering and Computer Science
January 31, 2018

Certified by .....
Russ Tedrake
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Theses

Localizing External Contact Using Proprioceptive Sensors: The Contact Particle Filter

by

Lucas Manuelli

Submitted to the Department of Electrical Engineering and Computer Science
on January 31, 2018, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

In order for robots to interact safely and intelligently with their environment they must be able to reliably estimate and localize external contacts. This paper introduces the CPF, the Contact Particle Filter, which is a general algorithm for detecting and localizing external contacts on rigid body robots without the need for external sensing. The CPF finds external contact points that best explain the observed external joint torque, and returns sensible estimates even when the external torque measurement is corrupted with noise. We demonstrate the capability of the CPF in multiple scenarios. We show how it can track multiple external contacts on a simulated Atlas robot, and also perform extensive simulation and hardware experiments on a Kuka iiwa robot arm.

Thesis Supervisor: Russ Tedrake
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

There are many people I want to thank for helping me to complete this workwork at MIT over the past couple of years.

Firstly I would like to thank my advisor, Russ Tedrake. Russ is the one who first challenged me to work on this problem, and has been a constant source of stimulating research discussions. He continues to challenge me to think bigger and tackle the hard problems.

I would also like to thank the members of the Robot Locomotion Group for helping me to learn almost everything I know about robotics. Thanks to the original members of the group during the final phase of the Darpa Robotics Challenge. In particular I learned so much from constantly asking questions to Pat Marion, Andres Valenzuela, Hongkai Dai, Robin Deits and others. A special thanks to Pat for helping me with all my Director related questions. The core algorithm and visualizations were all written as a Director python module. I also want to thank him for the many long conversations we had about all things robotics, as they really helped to shape my opinions and tastes in robotics. Thanks to Michael Posa for many technical discussions about this work and others. Pete Florence has been a great labmate and friend who I look forward to chatting with every day in lab. I want to particularly thank him for help in setting up our new Kuka robots that enabled the hardware experiments in this thesis. Thanks to Pang Tao and Twan Koolen for assistance in setting up the OptiTrack system. And finally thanks to all the other members of the group for constantly stimulating and exciting discussions.

Finally I would like to thank my family. Thanks to my Mom and Dad for supporting me always, even across a career change. Thanks to my sister Bianca for always being there to lend an ear. Katie thank you for everything.

Funding Acknowledgement

This work was supported by the Defense Advanced Research Projects Agency

via Air Force Research Laboratory award FA8750-12-1-0321 and by NSF Contract IIS-1427050.

Contents

Preface

This work is organized in five chapters. Chapter 1 introduces the work, states the contributions, discusses the motivation for external contact localization, and reviews related work. Chapter 2 describes the algorithmic formulation of the contact particle filter in detail. Chapter 3 presents experiments, both in simulation and hardware, that validate the algorithm. In addition this chapter provides an extensive discussion of limitations, failure modes and computational performance. Chapter 4 concludes.

An abridged version of this thesis without the hardware experiments was originally published as

> Lucas Manuelli, and Russ Tedrake. Localizing external contact using proprioceptive sensors: The Contact Particle Filter. In *International Conference on Intelligent Robots and Systems*, October 2016.

The intent is to submit an extended version that includes the hardware experiments of Chapter 3 as a journal paper.

Chapter 1

Introduction

One of the main challenges facing robotics is how to gracefully and safely handle contact with the environment. Most robots in the world today try to explicitly avoid contact with the environment. Quadrotors and autonomous vehicles explicitly try to avoid obstacles, and robot arms attempt to limit contact to specific locations on the end effector. There are some robots, however, that do have to make and break contact with the world. Walking robots need to push on the ground with their feet in order to locomote, and any robot wanting to manipulate an object must reach out and pick it up. However, in all these cases we try to control the contact interactions of the robot very carefully. If a walking robot makes contact with the world with something other than its feet, or a robot arm touches the environment with something other than its end effector, the results can be catastrophic. Fundamentally robots are not effective at handling unexpected contact events with their environment. This was exemplified by our experience as part of Team MIT at the DARPA Robotics Challenge Finals in June 2015. During the finals our robot experienced an unexpected contact with its environment which led to a fall.

Systems with changing contact states, such as walking robots, are fundamentally hybrid in nature. When an external contact occurs it causes a transition to a new hybrid mode. For control systems used on walking robots, having an accurate dynamic model is critical for effectively controlling the robot. If an unexpected contact event occurs which causes the system to switch hybrid modes, we need algorithms that

can detect this change and estimate the new hybrid mode so that we can update our dynamic model. Similarly for robot arms, if we are accidentally contacting the environment then the solution should be to detect this and handle it gracefully, rather than to let an integrator build up inside our controller and apply large forces on the environment.

As humans we don't even consider this problem as we have skin covering our entire body which allows us to easily sense external contacts. Although there has been some research in this direction, high performance sensing skin is not yet commonplace on robots. As such an algorithm is needed that can solve this problem using the sensors at hand, namely proprioceptive sensors. Proprioceptive sensors include joint encoders, torque sensors, IMUs etc.

1.1 Contribution

The main contribution of this thesis is an estimation algorithm, the Contact Particle Filter, which is capable of localizing multiple external contacts using only proprioceptive sensors. To the authors knowledge this is the first such algorithm that considers the problem from a sound probabilistic framework. In Chapter 3 we describe simulation experiments that validate the performance of our algorithm across a wide range of conditions, and show how our algorithm overcomes the main limitations of the method of [1]. We perform hardware experiments with a Kuka iiwa arm demonstrating the performance of our algorithm. To the authors knowledge these are the first such hardware experiments that include ground truth data for the contact locations using an instrumented force probe together with an external motion capture system.

1.2 Related Work

There is a large literature devoted to physical Human-Robot Interaction (pHRI) [2], [3], which is concerned with allowing robots to operate safely and collaboratively with humans in unstructured environments. One approach to interacting safely with

and around humans is to attempt to avoid collisions, rather than detect and react to them. Avoiding a collision typically requires the use of exteroceptive sensors such as externally mounted RGBD sensors [4] or onboard vision [5], [6].

Another approach to solving the collision detection and localization problem is to use a sensitive skin [7], [8], [9], [10]. These skins often use a capacitive material whose electric resistance changes as force is applied to the surface. One drawback of these skins is the complex wiring structure that must be constructed to collect the necessary signals. [11] uses model predictive control together with whole arm tactile sensing to perform reaching actions in clutter while controlling contact forces with the environment. Although their is potential in sensing skin, most robots do not come equipped with whole body sensitive skins. And if a robot does have a sensitive skin it is usually only at a few key locations, such as the hands and feet. Given this we focus on the problem of collision detection and localization using only proprioceptive sensors.

Initial approaches to this problem, mainly with applications to industrial robot arms, involved monitoring the measured currents in the robot's electrical motors and looking for fast transients that could be caused by a collision [12], [13], [14]. These methods are meant to detect abrupt collisions, rather than general contact events. In addition they don't provide any information about the location or direction of the contact. These collision detection strategies were used in conjuction with active compliance control [15], [16] to endow the robots with some safety in the presence of humans.

More recently a collision detection method based on generalized momentum [17], [18] has been proposed in [19], [20]. An advantage of this method is that it doesn't require acceleration measurements. Acceleration measurements are sensitive to noise since they require twice differentiating position measurements. Another advantage of using generalized momentum is that the collision detection strategy is independent of the control strategy used to generate the commanded motor torques. A variety of safe reaction strategies after a collision is detected are considered in [21], [22], [23].

We use the generalized momentum observer as the starting point for our estimation

algorithm. A brief overview of the generalized momentum observer is given in Section 2.1.1. [20] uses the method of [19] in a collision detection and safe reaction framework using a DLR-III lightweight manipulator arm. They are interested in the scenario of a collaborative robot arm working in a factory alongside human workers and their main motivation is to allow the robot to react safely in the event of a collision with a human. They show that the momentum observer contains sufficient information to allow the manipulator arm to react safely after a collision. However, since the implemented control strategies don't require precise information on the location of the contact point, they don't attempt to localize the contact point. [24] uses the momentum observer of [19] together with time-varying collision detection thresholds to provide more accurate collision detection performance in the presence of model errors.

[25] uses the generalized momentum observer, together with an external depth camera to estimate the interaction force between a robot and an external contact. As opposed to our work they use the depth camera to detect the location of the external contact point, whereas we localize the external contact without the use of external sensors.

The most related work to ours is the contact point localization algorithm outlined in section IV. B of [1]. As in our approach Haddadin et. al. [1] estimates the location of an external contact point using only proprioceptive information. A brief overview of this approach is presented in section 2.1.2. Although Haddadin et. al. is very computationally efficient, it doesn't use all the available information when estimating the contact location, and the algorithm isn't grounded in a probabilistic framework. This leads to several limitations, in particular their method isn't guaranteed to return a valid estimate, is susceptible to measurement noise and doesn't extend well to the case of multiple external contacts. We present a comparison of our method to [1] in Sections 3.2.3 and 3.3.3

[26] considers the problem of estimating the configuration of an object during manipulation using in-hand contact sensors. Although they focus on a different problem, the algorithmic approach is related. In particular they use a particle filter and

sample allowable object poses from the contact manifold consistent with current force sensor readings. Our approach is similar, in that we also use a particle filter, and sample allowable contact locations directly from the robot link surface, which is a manifold in $\mathbb{R}^3$.

Chapter 2

Localizing External Contact without Proprioceptive Sensors

This chapter we present the algorithmic foundations of the CPF. Section 2.1 explains the residual observer and presents the method of [1]. Section 2.2 explains the CPF in detail.

2.1 Preliminaries

We consider a robot with rigid links. Let $q \in \mathbb{R}^{n_q}$ describe the positions of the n_q joints. For a floating-base robot, the floating-base degrees of freedom also appear in q. Joint velocities are denoted by $v \in \mathbb{R}^{n_v}$. Note that a floating-base robot which uses quaternions to represent orientation we will have $n_q = n_v + 1$. The equations of motion are then

$$H(q)\dot{v} + C(q, v)v + g(q) = B\tau + \tau_{ext}. \tag{2.1}$$

$H(q) \in \mathbb{R}^{n_v \times n_v}$ is the inertia matrix, $C(q, \dot{q}) \in \mathbb{R}^{n_v \times n_v}$ are the Corioilis and centrifugal terms, and $g(q) \in \mathbb{R}^{n_v}$ is the gravity vector, τ are the motor torques, B maps the motor torques to the actuated joints. The external joint torques τ_{ext} arise from generalized contact forces acting on the robot. Suppose we have a contact on the surface of the i-th link whose position is given by r_c. Let $\mathbf{J}_{r_c}$ be the $6 \times n_v$ geometric Jacobian

corresponding to contact point r_c. An external force $F_c \in \mathbb{R}^3$ applied to the contact point r_c, and an external torque $M_c \in \mathbb{R}^3$ applied to the same point can be combined into a wrench $\Gamma_c = [F_c, M_c]^T \in \mathbb{R}^6$. Then the contribution of the generalized contact at point r_c to the external joint torque τ_{ext} is

$$\mathbf{J}_{r_c}(q)^T \Gamma_c = \mathbf{J}_{r_c}(q)^T \begin{bmatrix} F_c \\ M_c \end{bmatrix}. \tag{2.2}$$

If there are multiple contacts we have $\tau_{ext} = \sum_c \mathbf{J}_{r_c}(q)^T \Gamma_c$. Later we will make the simplifying assumption that $M_c = 0$, thus for notational convenience let $J_{r_c}(q)$ be the $3 \times n$ submatrix of $\mathbf{J}_{r_c}(q)$ corresponding to the linear velocity. Then

$$\mathbf{J}_{r_c}(q)^T \begin{bmatrix} F_c \\ 0_{3\times 1} \end{bmatrix} = J_{r_c}(q)^T F_c. \tag{2.3}$$

2.1.1 Residual Observer

In this section we provide an overview of the momentum observer method of [19] which provides an estimate of the external joint torque τ_{ext}. Following their treatment define the residual vector $\gamma(t) \in \mathbb{R}^{q_v}$ as

$$\gamma(t) = K_I \left(p - \int_0^t (B\tau + C^T(q, v)v + \gamma(s))ds \right), \tag{2.4}$$

where $p = H(q)v$ is the generalized momentum of the robot and $K_I > 0$ is a diagonal gain matrix. The residual has dynamics given by

$$\dot{\gamma}(t) = K_I(\tau_{ext} - \gamma). \tag{2.5}$$

Specifically γ is a first order observer of τ_{ext}. Hence the residual provides an estimate of the external joint torque that results from contact force/torques applied anywhere on the robot. If we have some known external torques, such as the feet of a walking robot, we may want to subtract these out when computing the residual so that γ estimates only the external torques resulting from unmeasured contacts. In particular

if we have contacts $c_1, \ldots, c_j$ for which we can measure the applied wrenches Γ_{c_i} (e.g. if our robot has 6-axis force-torque sensors at the ankles) we can subtract these out by defining

$$\begin{aligned}\gamma(t) =& K_I \left(p - \int_0^t (B\tau + \sum_{i=1}^{j} \mathbf{J}_{c_i}(q)^T \Gamma_{c_i} \right. \\ & \left. + C^T(q,v)v + \gamma(s))ds \right).\end{aligned} \tag{2.6}$$

Henceforth we let τ_{ext} denote the external joint torques produced by unmeasured external wrenches. [1] performs extensive experiments using a Kuka Light Weight Robot (LWR) characterizing the performance of the residual observer. As noted in [20], [1], there are situations where we can determine which link the contact is on simply by inspecting the residual γ. Suppose our robot is an n-link manipulator and there is a contact on the i-th link at location p_{c_i}. Then one can show that that the last $n-i$ columns of $\mathbf{J}_{c_i}$ will be zero. Hence only the first i components of τ_{ext} will be non-zero. In some situations this can allow us to isolate the link on which collision is occurring by searching for the largest index i such $\gamma_i \neq 0$. See [1] for a more detailed discussion.

2.1.2 Method of Haddidin et. al.

In [1] the authors propose a method for estimating the location of a contact point Later we will use this as a benchmark for comparison, so we provide a brief overview here. As with the CPF, their method takes as input the residual estimate γ. This immediately imposes some restrictions on what types of external contacts which are identifiable. A key restriction is that the algorithms can only estimate point contacts. In particular contacts where both a force and torque are exerted are not identifiable. The reason for this is that if we consider contacts where a wrench Γ can be applied, then there exist situations where two different contact locations r_{c_1}, r_{c_2} with two different applied wrenches Γ_1, Γ_2 lead to exactly the same external torque τ_{ext}. Suppose both contacts

are on link k. The external torque is given by

$$\tau_{ext,i} = J_{r_{c_i}} \Gamma_i \tag{2.7}$$

and the contact Jacobian $J_{r_{c_i}}$ can be decomposed as

$$J_{r_{c_i}} = J_k \tilde{J}_{r_{c_i}} \tag{2.8}$$

where J_i is the Jacobian for a fixed frame on link i. For the contact forces to be indistinguishable it is sufficient that

$$\tilde{J}_{r_{c_i}} \Gamma_1 = \tilde{J}_{r_{c_2}} \Gamma_2 \tag{2.9}$$

which would imply that the induced external torques $J_k \tilde{J}_{r_{c_1}} \Gamma_1$ and $J_k \tilde{J}_{r_{c_2}} \Gamma_2$ are equal. Given that $\tilde{J}_{r_{c_i}} \in \mathbb{R}^6 \times \mathbb{R}^6$, and $\Gamma_i \in \mathbb{R}^6$ this is possible in many instances.

Since the only only information given to the contact estimation algorithm is γ, which is an estimate of τ_{ext}, there is no way to distinguish between the situation of wrench Γ_1 applied at contact point r_{c_1} and wrench Γ_2 applied at r_{c_2}. Henceforth we restrict ourselves to the situation of point contacts where only a force can be applied. Note that there is no restriction that the force be applied along the surface normal, thus the force can have non-zero frictional components.

Suppose that $\gamma \approx \tau_{ext}$, and that using the reasoning outline in Section 2.1.1, we have localized the contact to be on link i. This is an assumption that is often not satisfied in practice. A wrench Γ_c applied at some point r_c on link i can always be transformed to an equivalent wrench Γ_i applied at some known frame on link i. The idea of the two-step procedure is to break the problem into two parts, each of which can easily be solved. The first step is to estimate Γ_i. Then if the wrench $\Gamma_c = [F_c, M_c]$ consists of only a force, i.e. $M_c = 0$, we can attempt to recover F_c and r_c from Γ_i. Now we describe each step in detail.

Estimating the Link Wrench Γ_i

Let $r_{i,c}$ be the location of the contact point in a known frame attached to link i. Denote this frame by T_i. Let $J_i(q)$ be the geometric jacobian for frame T_i and let the wrench applied at the contact point be Γ_c when expressed in frame T_i. Let $\mathbf{A}$ be the force-moment transformation from $r_{i,c}$ to the origin of T_i. $\mathbf{A}$ transforms a wrench applied at $r_{i,c}$ to the equivalent wrench applied at the origin of T_i, and takes the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ S(r_{i,c}) & \mathbf{I} \end{bmatrix}, \tag{2.10}$$

where $S(r_{i,c})$ is the skew-symmetric matrix such that $S(r_{i,c}) \cdot x = r_{i,c} \times x$ for all $x \in \mathbb{R}^3$. The wrench $\Gamma_i = \mathbf{A}\Gamma_c$ applied at T_i is equivalent to Γ_c applied at $r_{i,c}$. Substituting this into the computation of the external torque gives

$$J_c(q)^T \Gamma_c = J_i(q)^T \Gamma_i, \tag{2.11}$$

Since $r_{i,c}$ is unknown, the geometric Jacobian $J_c(q)$ is also unknown. However, since T_i is a known frame on link i, the geometric Jacobian $J_i(q)$ is known. If there is a single external contact then $\tau_{ext} = J_c(q)^T \Gamma_c = J_i(q)^T \Gamma_i$. Thus we can find Γ_i that best explains the residual by minimizing

$$\min_{\Gamma_i \in \mathbb{R}^6} (\gamma - J_i(q)^T \Gamma_i)^T (\gamma - J_i(q)^T \Gamma_i), \tag{2.12}$$

One solution to the above minimization is given by the pseudo-inverse

$$\hat{\Gamma}_i = (J_i(q)^T)^{\#} \gamma \tag{2.13}$$

If $J_i(q)$ has full row rank, which requires $i \geq 6$, there is no loss of information in computing $\hat{\Gamma}_i$. Or equivalently the optimization problem (2.12) has a unique minimum.

Recovering the Contact Location From Γ_i

Now that we have solved for the link wrench Γ_i we want to recover r_c, the location of the contact point. As noted previously one cannot determine r_c if we allow generalized contact wrenches Γ_c. Thus we restrict ourselves to contact wrenches with zero moments as in Equation 2.14. A contact wrench with no moment is simply a point contact.

$$\Gamma_c = \begin{bmatrix} F_c \\ \mathbf{0} \end{bmatrix} \tag{2.14}$$

This is usually the case for most typical contact situations, so we don't view it as a major restriction. What remains is to solve for a force $\hat{F}_c$ and contact location r_c such that when $\hat{F}_c$ is applied at r_c it generates the wrench $\hat{\Gamma}_i$ at the origin of frame T_i. This requires

$$\hat{\Gamma}_i = \begin{bmatrix} \hat{F}_i \\ \hat{M}_i \end{bmatrix} = A \begin{bmatrix} \hat{F}_c \\ 0 \end{bmatrix} = \begin{bmatrix} \hat{F}_c \\ S(r_{i,c})\hat{F}_c \end{bmatrix} \tag{2.15}$$

Immediately we know that $\hat{F}_c = \hat{F}_i$. The remaining equation is

$$\hat{M}_c = S(r_{i,c})\hat{F}_c = r_{i,c} \times \hat{F}_c = -\hat{F}_c \times r_{i,c} = -S(\hat{F}_c)r_{i,c} \tag{2.16}$$

$S(\hat{F}_c)$ is a 3×3 skew-symmetric matrix that represents the operator $\hat{F}_c\times$.

$$S(\hat{F}_c) = \begin{bmatrix} 0 & -\hat{F}_{c,3} & \hat{F}_{c,2} \\ \hat{F}_{c,3} & 0 & -\hat{F}_{c,1} \\ -\hat{F}_{c,2} & \hat{F}_{c,1} & 0 \end{bmatrix} \tag{2.17}$$

Since $0 = \hat{F}_c \times \hat{F}_c = S(\hat{F}_c)\hat{F}_c$ we know that $\text{rank}(S(\hat{F}_c)) \leq 2$. In particular if $\hat{F}_c \neq 0$, which we will suppose from now on since otherwise there is no force to be estimated, then it is simple to check that $\text{rank}(S(\hat{F}_c)) = 2$. Define

$$r_{i,d} = (-S(\hat{F}_c))^{\#}\hat{M}_i \tag{2.18}$$

Since $\text{rank}(S(\hat{F}_c)) = 2$ there are a continuum of solutions to equation 2.16, and the solution space has dimension one. Since $S(\hat{F}_c)\hat{F}_c = 0$ the solutions to equation (2.16) are of the form $r_{i,c} = r_{i,d} + \lambda \hat{F}_c / ||\hat{F}_c||$ for scalar λ. We denote the set of solutions $r_{i,c}$ as the line of force action. If we know the surface $\mathcal{S}_i$ of link i, the contact point x_c can be found by intersecting the line of force action with the $\mathcal{S}_i$. In general this will yield two intersection points along the line, denoted by λ_A, λ_B. Then we simply choose the point where $\hat{F}_c$ is pointing into the link surface. This comes from the fact that external forces usually only push (and don't pull) on the surface of a link. The problem with this method arises when the line of force action fails to intersect the link surface. This issue will be discussed further in Sections 3.2.3 and 3.3.3.

Extension to multiple external contacts

The same procedure can be extended to attempt to simultaneously estimate multiple contact points. Suppose we knew that there was contact on links i, j. The the first step involves solving the analog of (2.12) for the case of multiple contacts. The only difference is that $J_i(q)^T \Gamma_i$ is replaced by $J_i(q)^T \Gamma_i + J_j(q)^T \Gamma_j$. Again this can be solved using the pseudoinverse.

$$\begin{bmatrix} \hat{\Gamma}_i \\ \hat{\Gamma}_j \end{bmatrix} = ([J_i(q)^T, J_j(q)^T])^{\#}\gamma = (\mathbf{J}^T)^{\#}\gamma, \tag{2.19}$$

where $\mathbf{J} = \begin{bmatrix} J_i(q) \\ J_j(q) \end{bmatrix}$. Again for there to be no information loss when going from γ to $[\hat{\Gamma}_i, \hat{\Gamma}_j]$, which is equivalent to the analog of (2.12) for the multi-contact case having a unique solution, the matrix $\mathbf{J}$ must be of rank 12. Once we have $\hat{\Gamma}_i, \hat{\Gamma}_j$ we follow the same procedure as in the single contact case to find the location of the contact on each link. In general this procedure can be used to localize any number of contacts. If there are k external contacts, then in order for there to be no information loss one needs that the stacked link Jacobian matrix $\mathbf{J} = [J_{i_1}(q)^T, \ldots, J_{i_k}(q)^T]$ is of rank at least $6k$.

2.2 Contact Detection and Localization

In this section we describe the details of the CPF algorithm. First we formulate the contact localization problem as a nonlinear optimization. Then we leverage some features of this optimization problem to approximate it using a tractable quadratic programming framework, and show how to use this framework as part of a particle filter.

For simplicity consider the case of a single external contact at location r_c on link i. Following [1] we make the assumption that only forces, and no torques, are applied at r_c. This is the case for most typical contact situations. Given a residual γ we want to find the contact location and contact force F_c which best explain γ. Let $\mathcal{S}_i \subset \mathbb{R}^3$ be the surface manifold of the i-th link. Since contact point r_c must lie on the surface of the robot the allowable contact locations are $\mathcal{S} = \bigcup_i \mathcal{S}_i$. Let $\mathcal{F}(r_c)$ denote the friction cone at contact point r_c. Note that the method of [1] doesn't consider the friction cone, and thus isn't taking advantage of all available information. Then solving for the contact location can be formulated as an optimization

$$\min_{r_c, F_c} (\gamma - J_{r_c}(q)^T F_c)^T (\gamma - J_{r_c}(q)^T F_c) \tag{2.20}$$

$$\text{subject to } r_c \in \mathcal{S}, F_c \in \mathcal{F}(r_c). \tag{2.21}$$

The optimization (2.20) is non-convex since r_c and F_c appear as a cross product in the term $J_{r_c}(q)^T F_c$. However, if we fix the contact location r_c then the optimization problem becomes convex.

$$\min_{F_c} (\gamma - J_{r_c}(q)^T F_c)^T (\gamma - J_{r_c}(q)^T F_c) \tag{2.22}$$

$$\text{subject to } F_c \in \mathcal{F}(r_c).$$

The problem is convex because once we fix r_c the Jacobian $J_{r_c}(q)$ is simply a known fixed matrix, and the friction cone $\mathcal{F}(r_c)$ is a convex set. A similar insight was

used in [27] in the context of grasp analysis. One way to approximate the solution to problem (2.20) is to sample contact locations $r_c \in \mathcal{S}$ and then solve the convex problem (2.22) for each contact location. By choosing the point r_c with the smallest objective value, we achieve an approximation to the solution of the full problem (2.20). In section 2.2.1 each particle in our particle filter will correspond to a contact location r_c and the measurement update will correspond to solving a version of (2.22). In this way we avoid the intractability of problem (2.20).

Section 2.2.1 describes the Contact Particle Filter for the case of a single external contact. In section 2.2.2 we extend the CPF to the general multi-contact case.

2.2.1 Single External Contact

Let $\gamma(t)$ be the residual observer from section 2.1. Each particle $r_t^{[m]} \in \mathbb{R}^3$ corresponds to a particular location of the external contact on the surface of the robot. A particle filter requires us to specify both a measurement model, described in section 2.2.1, and a motion model, detailed in section 2.2.1. [28] provides an extensive overview of particle filters and their associated convergence properties.

Measurement Model

Our measurement will be the residual $\gamma(t)$, also abbreviated as γ. The measurement update $p(\gamma|r_t^{[m]})$ captures how well a force applied at point $r_t^{[m]}$ can explain the residual $\gamma(t)$. To find a probability for $p(\gamma|r_t^{[m]})$ we suppose that the residual is the true external joint torque plus noise,

$$\gamma = \tau_{ext} + \eta, \text{ where } \eta \sim \mathcal{N}(0, \Sigma_{meas}). \tag{2.23}$$

Now define

$$\begin{aligned} \varepsilon = \min_{F_c} & (\gamma - J_{r_t^{[m]}}^T F_c)^T \Sigma^{-1} (\gamma - J_{r_t^{[m]}}^T F_c) \\ & \text{subject to } F_c \in \mathcal{F}(r_t^{[m]}). \end{aligned} \tag{2.24}$$

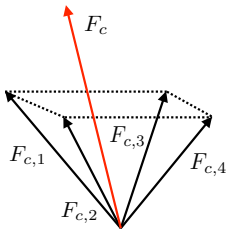


Figure 2-1: Polyhedral approximation to the friction cone

Following the approach in [29] we replace the friction cone with a polyhedral approximation shown in Figure 2-1. This polyhedral approximation to the friction cone allows us to approximate (2.24) using a quadratic program.

$$QP(\gamma|r_t^{[m]}) = \min_{\alpha_i, F_c} (\gamma - J_{r_t^{[m]}}^T F_c)^T \Sigma_{meas}^{-1} (\gamma - J_{r_t^{[m]}}^T F_c) \tag{2.25}$$
$$\text{subject to } \alpha_i \geq 0, \quad F_c = \sum_{i=1}^{K} \alpha_i F_{c,i}.$$

where K is the dimension of the approximation to the friction cone. In our case we use $K = 4$ as shown in Figure 2-1. Then $\varepsilon \approx QP(\gamma|r_t^{[m]})$. We can recover a likelihood using the fact that $\gamma = \tau_{ext} + \eta$. Namely

$$p(\gamma|r_t^{[m]}) \propto \exp\left(-\frac{1}{2} QP(\gamma|r_t^{[m]})\right), \tag{2.26}$$

where we have omitted the normalizing constant. The key insight is that once we specify a contact location, the measurement update can be formulated as a quadratic program, abbreviated as QP.

Motion Model

The other half of a particle filter is the motion model. In particular we must specify $p(r_t|r_{t-1}, u_t)$ where u_t are the control inputs at time t, in this case the torques applied to the robot. Our motion model won't depend on the control inputs so we define

$$p(r_t|r_{t-1}) \propto \mathcal{N}(r_t; r_{t-1}, \Sigma_{motion}). \tag{2.27}$$

Particles must correspond to contact locations on the surface of the link, so in order to sample from this distribution we first generate $\tilde{r} \sim \mathcal{N}(r_{t-1}, \Sigma_{motion})$ and then project $\tilde{r}$ back to the closest point r_t on the robot's surface. Given a set of particles $\mathcal{X}$ let Motion-Model($\mathcal{X}$) be the result of applying the motion model to each particle. This motion model corresponds to a mean-zero random walking assumption of the contact location in the link frame. In other words the contact point moves around randomly on the surface of the link. Other motion models, such as zero velocity of the contact point in the world frame are also possible. We believe that the specific choice of motion model does not have a large impact on filter performance, as the measurement update step provides strong information for resampling particles near the true contact location.

Contact Particle Filter

Now we combine the measurement and motion models to form the single contact particle filter (Single-CPF). As in [20] and [1] we must specify a threshold for determining when there is an external contact. Define $\epsilon(t) = \gamma(t)^T \Sigma_{meas}^{-1} \gamma(t)$. We say that there is an external contact if $\epsilon(t)$ is greater than some threshold $\bar{\epsilon}$. Imposing a threshold serves to reduce false positives, but it can also delay detection for contacts with small forces. If the thresholding approach is not sufficient, we can apply model-based adaptive thresholding [30] or learning based methods [31]. Let $\mathcal{X}_t$ denote the current set of particles $\{r_t^{[1]}, \ldots, r_t^{[m]}\}$, and $\mathcal{X}_{init}$ be fixed set of particles which are evenly sampled from the surface of the robot. The Single-CPF is described in Algorithm 1. The Importance-Resample function simply performs the standard particle filter importance resampling using the importance weights $w_t^{[m]}$.

The final step is to recover the most likely contact location given a particle set $\mathcal{X}_t$. This is done by computing an average contact location for the particle set and projecting this point back to the surface of the robot. Label this procedure Get-Contact-Location($\mathcal{X}_t$).

Figure 2-2 shows 4 iterations of the Single-CPF algorithm while localizing a contact on the torso. The particles are drawn in red just after importance resampling,

Algorithm 1 Single-CPF($\mathcal{X}_{t-1}, \gamma(t)$)

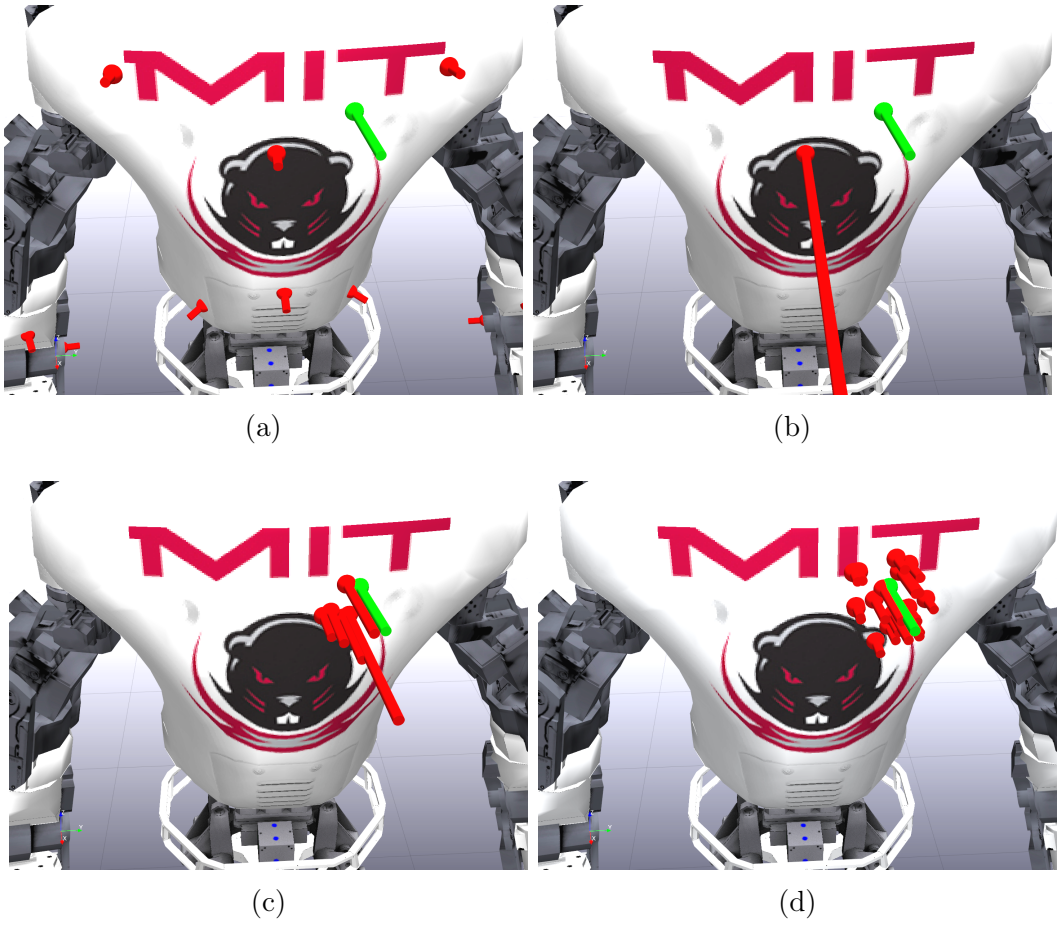
1: **if** $\epsilon(t) = \gamma(t)^T \Sigma_{meas}^{-1} \gamma(t) < \bar{\epsilon}$ **then**
2: $\quad \mathcal{X}_t = \emptyset$
3: $\quad$ **return** $\mathcal{X}_t$
4: **if** $\mathcal{X}_{t-1} = \emptyset$ **then**
5: $\quad \mathcal{X}'_t = \mathcal{X}_{init}$
6: **else**
7: $\quad \mathcal{X}'_t =$ Motion-Model($\mathcal{X}_{t-1}$)
8: $\overline{\mathcal{X}}_t = \emptyset$
9: **for** $r_t^{[m]}$ in $\mathcal{X}'_t$ **do**
10: $\quad w_t^{[m]} = p(\gamma(t)|r_t^{[m]})$
11: $\quad \overline{\mathcal{X}}_t = \overline{\mathcal{X}}_t + \langle r_t^{[m]}, w_t^{[m]} \rangle$
12: $\mathcal{X}_t =$ Importance-Resample($\overline{\mathcal{X}}_t$)
13: **return** $\mathcal{X}_t$

line 12 of Algorithm 1. The true contact location is shown in green. Initially there is no external force and the filter has $\mathcal{X}_t = \emptyset$. Then an external force of 10 newtons is continuously applied at a location on the torso, shown in green. The filter detects this and enters the `if` statement at line 5 and sets $\mathcal{X}'_t = \mathcal{X}_{init}$. This is visualized in Figure 2-2a. Subsequent filter steps shown in Figure 2-2b-2-2d show the particles converging to the true contact location.

2.2.2 Multiple External Contacts

In this section we extend the Single-CPF algorithm to handle multiple external contacts. First we consider a naive generalization and show why it is not computationally tractable. Then we propose a computationally tractable alternative.

Suppose we have l external contacts. Now the state space is the location of all l contact points, thus a particle $\mathbf{r}_t$ in our filter encodes the locations of all l contact points, $\mathbf{r}_t = (r_{t,1}, \ldots, r_{t,l})$, where $r_{t,j}$ is the location of the j-th contact point. There is a simple extension of the measurement update from section 2.2.2 to the multi-contact case. Let $F_{c,1}^{[k]}, \ldots, F_{c,4}^{[k]}$ be the polyhedral approximation to the friction cone of the



(a) (b)

(c) (d)

Figure 2-2: Four iterations of the Single-CPF algorithm. Particles are shown in red just after importance resampling. The length of the arrow is proportional to the number of particles at that location. The green arrow is the true contact location.

k-th contact point $r_{t,k}$. Then define

$$QP(\gamma|(r_{t,1},\ldots,r_{t,l})) = \min_{\alpha_{i,k},\hat{\tau}_{ext}} (\gamma-\hat{\tau}_{ext})^T\Sigma_{meas}^{-1}(\gamma-\hat{\tau}_{ext}) \tag{2.28}$$

$$\text{s.t. } \alpha_{i,k}\geq 0,\quad F_c^{[k]}=\sum_{i=1}^{4}\alpha_{i,k}F_{c,i}^{[k]},\quad \hat{\tau}_{ext}=\sum_{k=1}^{l}J_{r_{t,k}}^T F_c^{[k]}.$$

This a quadratic program with $4*l$ decision variables. As in the single contact case the likelihood is

$$p(\gamma|(r_{t,1},\ldots,r_{t,l}))\propto \exp\left(-\frac{1}{2}QP(\gamma|(r_{t,1},\ldots,r_{t,l}))\right). \tag{2.29}$$

Thus our measurement model extends naturally to multiple contact points. However the complexity of this algorithm will grow exponentially in the number of contact points. If we have l contact points, then the particle representing all the contact locations belongs to a space of dimension l. The number of particles needed in a particle filter grows exponentially with the dimension of the state space, and so as the number of contact points increases we would need to increase the number of particles exponentially, which would require solving an exponentially increasing number of quadratic programs. Clearly this is not tractable so we propose an approximate scheme.

Instead of having a single particle encode the location of all the external contacts, each particle will encode the location of a single external contact, as in the single-CPF algorithm. If there are l actual external contacts, labeled $c_1,\ldots,c_l$, then we will have l particle sets $\mathcal{X}_{t,1},\ldots,\mathcal{X}_{t,l}$, one for each contact point. Let $\mathscr{X}_t=\{\mathcal{X}_{t,1},\ldots,\mathcal{X}_{t,l}\}$ denote the set of particle sets. The particles in particle set $\mathcal{X}_{t,k}$ will estimate the location of the k-th contact. As in section 2.2.1 we must define the measurement model and the motion model.

Measurement Model

To get around the computational problems mentioned in the previous section we make an independence assumption when computing the measurement update for particle

$x_{t,j} \in \mathcal{X}_{t,j}$. Specifically we take the location of the other contacts as given. The Get-Contact-Location($\mathcal{X}_{t,k}$) method described in section 2.2.1 naturally provides an estimate of the location of contact c_k, given by $r^*_{c,k}$ = Get-Contact-Location($\mathcal{X}_{t,k}$). If we define $\mathbf{r} = \{r^*_{c,k}\}_{k \neq j}$ then the measurement update for a particle $r^{[m]}_{t,j} \in \mathcal{X}_{t,j}$ is defined as

$$p(\gamma | r^{[m]}_{t,j}, \mathscr{X}) \propto \exp\left(-\frac{1}{2} QP(\gamma | (r^{[m]}_{t,j}, \mathbf{r}))\right). \tag{2.30}$$

where $QP(\gamma|(r^{[m]}_{t,j}, \mathbf{r}))$ refers to (2.28). The full measurement update is detailed in Algorithm 2.

Algorithm 2 Multi-Measurement-Update(γ, $\mathscr{X}_t$)

$\overline{\mathscr{X}}_t = \emptyset$
for $\mathcal{X}_{t,j} \in \mathscr{X}$ **do**
  $\overline{\mathcal{X}}_{t,j} = \emptyset$
  $\mathbf{r} = \{r^*_{t,k}\}_{k \neq j}$
  for $r^{[m]}_{t,j}$ in $\mathcal{X}_{t,j}$ **do**
    $w^{[m]}_{t,j} = \exp\left(-\frac{1}{2} QP(\gamma | (r^{[m]}_{t,j}, \mathbf{r}))\right)$
    $\overline{\mathcal{X}}_{t,j} = \overline{\mathcal{X}}_{t,j} + \langle r^{[m]}_{t,j}, w^{[m]}_{t,j} \rangle$
return $\overline{\mathscr{X}}_t$

Motion Model and Importance Resampling

The motion model for a single particle is the same as in the single contact case with the sampling density $p(r_{t,j}|r_{t-1,j})$ defined as in section 2.2.1. Then we simply apply the motion model to each particle set independently.

Importance resampling just consists of independently resampling each particle set using the importance weights $w^{[m]}_{t,j}$ computed in the measurement update step.

Adding and Removing Particle Sets

Since each particle set $\mathcal{X}$ represents a single external contact, we keep track of the number of external contacts and update the number of particle sets accordingly. Given $\mathscr{X} = \{\mathcal{X}_{t,1}, \ldots, \mathcal{X}_{t,l}\}$ let $\mathbf{r}^*(\mathscr{X}) = \{r^*_{t,1}, \ldots, r^*_{t,l}\}$, where $r^*_{t,k}$ is the most likely contact

location for $\mathcal{X}_{t,k}$. Define

$$\epsilon(\mathscr{X}_t, \gamma) = QP(\gamma|\mathbf{r}^*(\mathscr{X}_t)). \tag{2.31}$$

If $\mathscr{X}_t$ is empty then let

$$QP(\gamma|\mathbf{r}^*(\mathscr{X}_t)) = \gamma^T \Sigma_{meas}^{-1} \gamma. \tag{2.32}$$

If $\epsilon(\mathscr{X}_t, \gamma) > \bar{\epsilon}$ then it means that with our current estimate of the locations of the external contacts we are not able to explain the residual γ. This means that there is likely another contact point that is not accounted for by one of the current particle sets, so we add a new particle set to $\mathscr{X}_t$ to represent this new external contact point. Since we don't know where this new contact point is we initialize the new particle set to $\mathcal{X}_{init}$.

When should we remove a particle set? If the residual is well explained without using a force at a particular contact location then it is likely that there is no force at this contact location. In this situation we remove the particle set corresponding to that contact point. Formally if $\mathcal{X}$ is such that $\epsilon(\gamma, \mathscr{X}_t \backslash \{\mathcal{X}_{t,k}\}) < \bar{\epsilon}$, then we should eliminate the particle set $\mathcal{X}_{t,k}$ from $\mathscr{X}_t$. The full procedure is outlined in Algorithm 3

Algorithm 3 Manage-Particle-Sets(γ, $\mathscr{X}_{t-1}$)

```
X_t = X_{t-1}
if ε(γ, X_{t-1}) > ε̄ then
    add X_init to X_t
    return X_t
for X_{t,k} in X_t do
    if ε(γ, X_t \ {X_{t,k}}) < ε̄ then
        remove X_{t,k} from X_t
        return X_t
return X_t
```

Multi-Contact-Particle-Filter

For the multi-contact particle filter we simply combine the motion model, the Multi-Contact-Measurement-Update, and the Manage-Particle-Sets algorithms. The details are given in Algorithm 4.

Algorithm 4 Multi-CPF(γ, $\mathscr{X}_{t-1}$)

$\mathscr{X}_{motion}$ = Motion-Model($\mathscr{X}_{t-1}$)
$\mathscr{X}_{meas}$ = Multi-Measurement-Update(γ, $\mathscr{X}_{motion}$)
$\mathscr{X}_{resample}$ = Importance-Resample($\mathscr{X}_{meas}$)
$\mathscr{X}_t$ = Manage-Particle-Sets(γ, $\mathscr{X}_{resample}$)
return $\mathscr{X}_t$

To recover the best estimate of the contact locations we simply apply Get-Contact-Location to each particle set $\mathcal{X}_{t,k} \in \mathscr{X}_t$.

Chapter 3

Experimental Validation

This chapter analyzes a variety of experiments that demonstrate the performance of the CPF. Section 3.1 describes implementation details of the CPF algorithm. Section 3.2 describes experiments using a simulated 36 DOF robot, where we show the ability of the CPF to localize up to 3 simultaneous external contacts. Section 3.3 describes simulation experiments performed on a Kuka iiwa 7 DOF robot arm. In particular we analyze the performance of the CPF across a variety of contact locations, robot poses and noise levels in the residual. We also compare our approach to the method of [1]. Section 3.4 shows the results of hardware experiments on the Kuka iiwa that match the simulation experiments from Section 3.3.

3.1 CPF Implementation Details

To speed up development the CPF was implemented in a single thread Python process using the Director software package [32]. We use the FORCES Pro [33] software package to generate efficient solvers for the quadratic programs in the measurement update step. This is possible since the size of the quadratic programs is known ahead of time. The CPF must be passed a complete surface mesh of the robot at initialization, but the only required real-time information are the joint positions q and the residual γ. In all the experiments the CPF was run on an Intel Core i7-5820K X6 3.3GHz with 16GB of RAM.

3.2 Simulation Experiments with Atlas Humanoid Robot

Section 3.2.1 describes the experiment setup, 3.2.2 gives quantitative results on the localization performance of the CPF. Section 3.2.3 provides a comparison of the performance of the CPF and the method of [1]. A video of the CPF is available at http://youtu.be/ckvsMK0QhB0.

3.2.1 Experimental Setup

We perform our first experiments using a simulated model of the Atlas robot. In particular we ran the simulations using the Drake robotics toolkit [34]. The Atlas robot has 36 degrees of freedom and 30 actuated joints. To properly formulate the residual detector we need joint position and velocity measurements q, v, in addition to torque measurements τ for the actuated joints, i.e. excluding the floating base. Although the Atlas hardware only has 3-axis force-torque sensors at the feet we simulate full 6-axis force-torque sensors. As discussed in Section 2.1.1 these 6-axis force-torque measurements are necessary in order to properly "subtract out" the known contact wrenches at the feet when computing the residual γ. Other humanoids such as NASA's Valkyrie [35] have these 6-axis force-torque sensors. To test our algorithm we also augment the simulator allow application of arbitrary contact forces along the surface of the robot. This allows us to simulate many different potential contact situations without constructing complex environments that the robot can collide with.

3.2.2 Filter Performance

In all the experiments the parameters were held constant, specifically we used particle sets of size 50. We ran three classes of experiments, distinguished by the number of simultaneous external contacts. The first set of experiments applied a single contact force to the robot. There were 7 distinct locations where this contact force was

applied. The second set of experiments considered pairs of external contacts. In particular we had four pairs of contacts. Since the filter cannot handle multiple contacts that arise simultaneously[1] we add the contacts sequentially 1 second apart. The final set of experiments considers 3 simultaneous contacts. In this case we considered three triples of contact locations. As in the situation of two contacts, the contact forces were applied sequentially. In all experiments a 20 Newton force was applied at each contact location. The robot's pose can also affect filter performance since the contact Jacobians J_{r_c} in equation (2.2) are a function of the joint angles. To incorporate this effect we moved the robot through a series of poses for each set of contact locations. The poses involved moving the upper body and adjusting the center of mass height, but did not include walking. Finally, in practice the residual γ may not be perfectly accurate due to inaccuracies in the dynamic model of the robot, and errors in the joint position and torque sensor. To capture this effect we added Gaussian noise with standard deviation $\{0, 0.1, 0.2\}$ to the residual. The results are summarized in Figure 3-1. The CPF was able to localize contacts to within 3 centimeters in most cases. In general filter performance deteriorates as the amount of noise increases, however the localization accuracy remained fairly constant as the number of external contacts increased. Since the experiments were performed in simulation the residual observer (2.4) had the correct inertial parameters of the robot. Since the simulation doesn't include friction in the robot's joints, this ultimately implies that the residual observer was using a very accurate dynamic model of the robot, an assumption that may or may not be satisfied in real world operation. However, the addition of noise in the experiments shows that the filter performs fairly well even if the residual contains errors.

3.2.3 Comparison to Haddadin et. al.

In this section we compare the performance of the CPF to the contact localization method of [1], described in Section 2.1.2. A typical situation is shown in Figure 3-2. As we can see the CPF precisely estimates the location of both contact points, while

[1] see Section 3.5.6 for a more extensive discussion

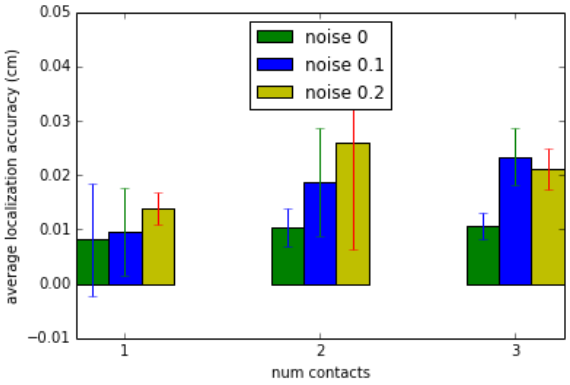


Figure 3-1: This chart shows the localization performance of the CPF across the experiments described in Section 3.2.2. The experiments are classified according to the number of external contacts, either one, two or three. Gaussian noise with standard deviation $\{0, 0.1, 0.2\}$ was added to the residual. For each noise level, we report the average localization performance across the different contact locations in that set.

the method of [1] doesn't provide accurate estimates [2]. In particular the line of force action for the arm fails to intersect the robot surface. The reason that the method of [1] does poorly in this scenario is because when solving for the link wrenches Γ_i (in this case one for the arm and one for the torso) in the first stage, it allows arbitrary wrenches $\Gamma_i \in \mathbb{R}^6$. If $A(r_c)$ denotes the force moment transformation which converts a force applied at r_c to a wrench at the known link frame then the set of wrenches that can be generated by point forces applied to the robot is given by

$$\mathcal{W}_i = \{\Gamma : \Gamma = A(r_c)F_c \text{ for } r_c \in \mathcal{S}_i, F_c \in \mathcal{F}(r_c)\}. \tag{3.1}$$

By allowing $\Gamma_i \in \mathbb{R}^6$ rather than restricting $\Gamma_i \in \mathcal{W}_i$ in the first stage, the method of [1] doesn't take advantage of all the information in the residual γ. In the situation of Figure 3-2 the external joint torque is given by

[2]For the purposes of this experiment we gave the method of [1] the identity of the two links where contact was occurring. In general this is something that would need to be deduced from the residual. This is relatively easy in the case of a single contact, or multiple contacts on distinct kinematic chains, but it is not generally possible for multiple contacts on the same kinematic chain. However, for the sake of comparing the two methods we allow the method of [1] this additional information.

$$\gamma \approx \tau_{ext} = \mathbf{J}_1(q)^T \Gamma_1 + \mathbf{J}_2(q)^T \Gamma_2, \tag{3.2}$$

where the $i = 1, 2$ subscripts denote quantities for the arm and torso, respectively. The method of [1] fails because if we allow $\Gamma_1, \Gamma_2 \in \mathbb{R}^6$, then equation (3.2) admits multiple solutions. [1] uses the pseudo-inverse to choose a particular solution $\hat{\Gamma}_1, \hat{\Gamma}_2$, but if $\hat{\Gamma}_1 \notin \mathcal{W}_1$ then there does not exist a contact location r on the arm and force F which generate this link wrench $\hat{\Gamma}_1$. This is manifested in Figure 3-2 as the line of force action for the arm failing to intersect the robot surface. Ultimately not imposing the restriction that $\Gamma_i \in \mathcal{W}_i$ in the first stage causes the failure of the method of [1] to localize the contact points. On the other hand the CPF does impose that $\Gamma_i \in \mathcal{W}_i$ as can be seen in (2.20). This additional restriction eliminates the multiplicity of solutions to Equation (3.2) that plagued the method of [1] and allows the CPF to accurately localize the external contacts.

As illustrated in Figure 3-2 one of the failure modes of the method of [1] is that the line of force action fails to intersect the link surface. This is a result of the first stage estimate $\hat{\Gamma}_i$ not being in $\mathcal{W}_i$. If the residual γ is a sufficiently noisy estimate of τ_{ext} then the first stage of the method of [1], which attempts to solve $\gamma = J_i(q)^T \hat{\Gamma}_i$ can return $\hat{\Gamma}_i \notin \mathcal{W}_i$. Since the resulting line of force action fails to intersect the link, the method of [1] doesn't return an estimate. On the other hand since the CPF samples particles on the link surface it never suffers this problem. The estimates may be degraded by a noisy γ, as seen in Figure 3-1, but by construction they always lie on the link surface.

3.3 Simulation Experiments with Kuka iiwa Arm

3.3.1 Experimental Setup

We use an analogous setup to the experiments in Section 3.2. The only difference is that we use the 7 degree of freedom Kuka iiwa robot model instead of Atlas. To test out the benefit of adding additional sensors, Section 3.3.4 explores the effect of

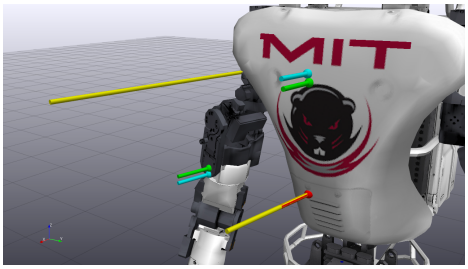


Figure 3-2: There are two external contacts on the robot, shown in green. Each contact is applying a 10 Newton force. The estimated contact locations from the CPF are shown in cyan. The lines of force-action for the method of [1] are the long yellow rays.

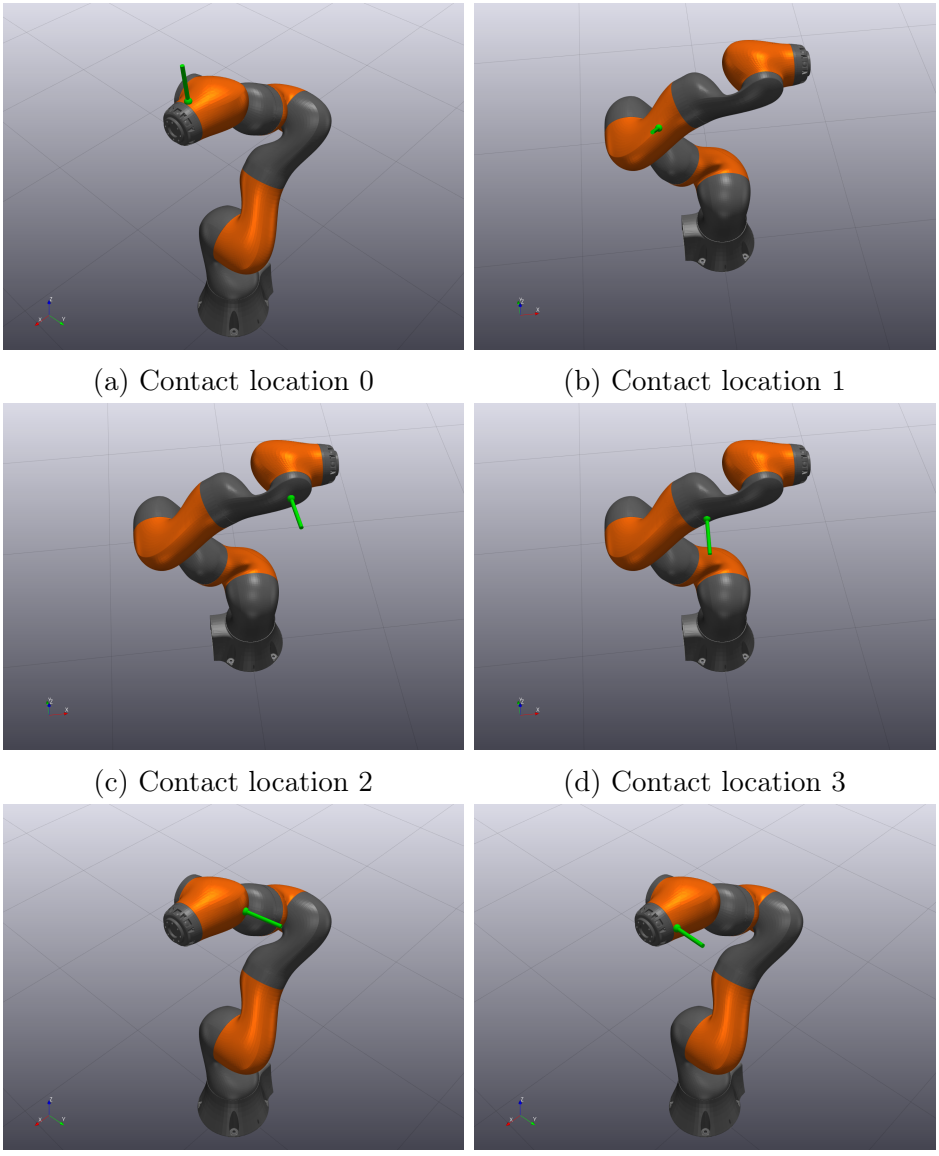
adding a 6-axis force torque sensor to the base of the Kuka robot.

3.3.2 Filter Performance

Since the Kuka only has 7 degrees of freedom the residual $\gamma \in \mathbb{R}^7$, there isn't sufficient information to localize multiple external contacts, thus all the experiments with the 7 degree of freedom Kuka will be with a single external contact force. This is in keeping with the method of [1] which was primarily developed and tested for industrial robot arms such as the Kuka. The experiments consist of evaluating the CPF across a variety of contact locations, robot poses, and noise levels in the residual torque γ.

The parameters of the filter were held constant across all experiments. We sampled six contact locations on the surface of links 4,5,6,7 of the Kuka robot, shown in Figure 3-3. Each experiment applied a single point force to one of the locations in Figure 3-3. Since the contact Jacobians J_{r_c} in equation (2.2) are a function of the joint angles, the robot pose can affect filter performance. To study this effect we ran experiments for the four poses shown in Figure 3-4. Finally, we added noise to the residual estimate γ. This served as a proxy for inaccuracies in the robot model used when computing the residual. We used Gaussian noise with standard deviations $\{0, 0.1, 0.5\}$. For each combination of contact location, robot pose and noise level we ran 5 experiments and computed the average performance. The results are show in Figure 3-5.

Figure 3-5a shows accuracy of the contact location estimate, across different robot poses, contact locations and noise levels. In particular, even with noise the contact



(a) Contact location 0

(b) Contact location 1

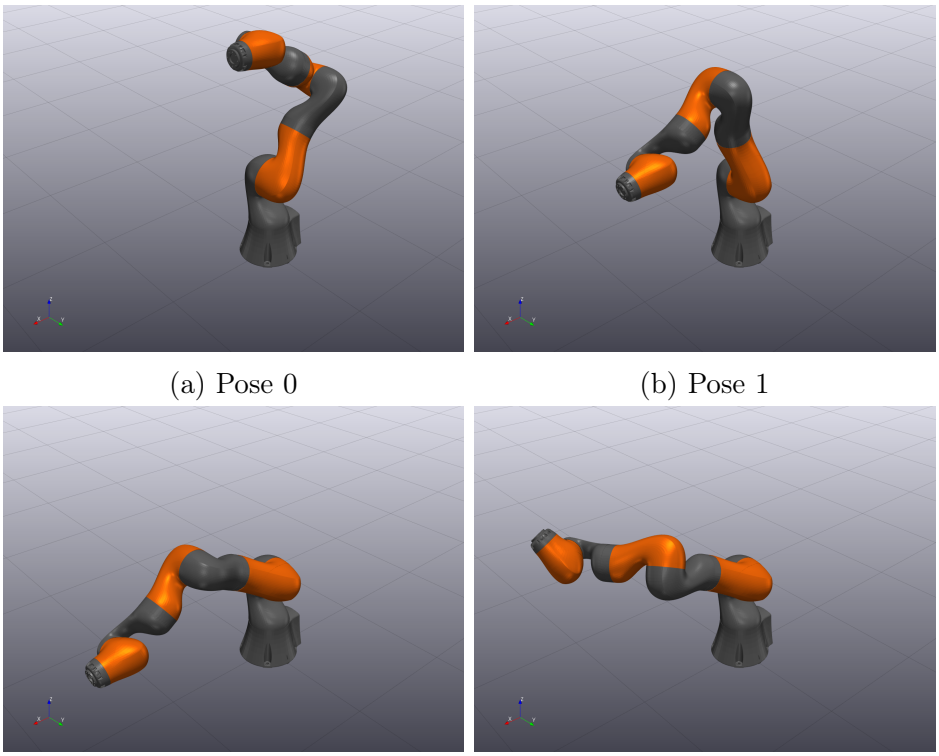
(c) Contact location 2

(d) Contact location 3

(e) Contact location 4

(f) Contact location 5

Figure 3-3: The six contact locations used in the Kuka iiwa simulation and hardware experiments. Each contact location is indicated with a green line segment along the normal direction.

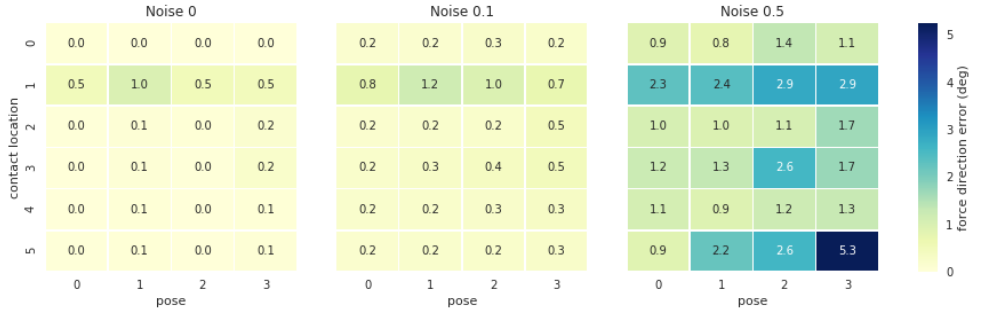


(a) Pose 0 (b) Pose 1

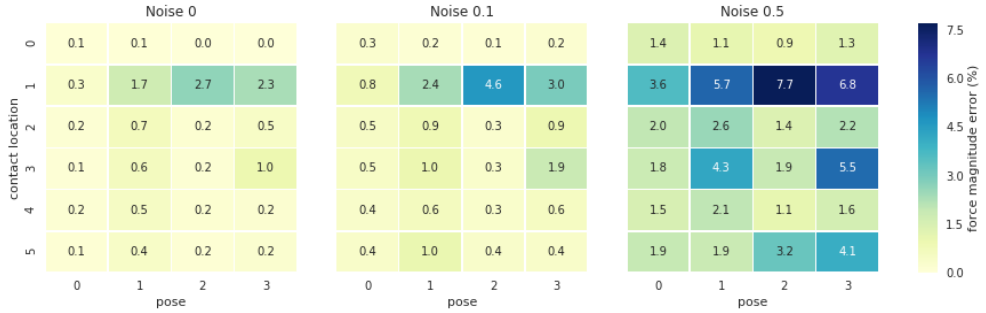
(c) Pose 2 (d) Pose 3

Figure 3-4: The four robot poses used in the Kuka iiwa simulation and hardware experiments.

(a)



(b)



(c)

Figure 3-5: Results of simulation experiments run on the Kuka robot. A simulation experiment was run for each combination of contact location, robot pose and noise added to the residual. (3-5a) shows the average error, measured in centimeters, of the estimated contact location as compared to the true contact location. (3-5b) shows the average angle, measured in degrees, between the estimated force direction and the measured force direction. (3-5c) shows the estimated force magnitude error as a percentage of the actual force.

location estimate was always within 2 cm, and in the case of no noise within 1 cm. This shows that the CPF is able to accurately localize external contact across a range of poses and contact locations. Since the input to the algorithm is the residual estimate, it is expected that filter performance will suffer as the quality of the residual degrades. This can be seen by comparing the three plots in Figure 3-5a. The leftmost plot, which corresponds to no noise in the residual, has the best performance, and localization accuracy decreases as more noise is added to the residual.

Figure 3-5b shows the average angle error between the estimated and actual contact force directions. Again we see that force direction accuracy is excellent, within 4 degrees for all scenarios. Similarly to the contact location estimation, performance deteriorates as more noise is added to the residual.

Figure 3-5c shows the accuracy of the force magnitude estimate, where the estimated force magnitude error is given as a percentage of the actual force. Across all trials we were able to estimate the force to within 8% of the true magnitude. As mentioned above the general trend is that the force accuracy deteriorates as noise is added to the residual.

The plots also show us that both contact location and robot pose can affect accuracy. In particular performance of the CPF for contact location 1 is significantly worse than for the other contact locations. As shown in Figure 3-3 this contact is on the fourth link of the iiwa. This implies that the last three columns of the contact Jacobian J_{r_c} are zero. Hence the residual $\tau_{ext} \in \mathbb{R}^7$ defined in section 2.1 has only four non-zero components. Effectively this means that there is less information for the CPF to use in estimating the contact location. This is reflected Figure 3-5 with contact location 1 having the worst performance, among the different contact locations, in all three metrics, location, direction and magnitude.

3.3.3 Comparison to Haddadin et. al.

In this section we compare the performance of the CPF to the method of [1] outlined in section 2.1.2. We analyze the same set of experiments as in Section 3.3.2, focusing on the scenario with no added noise. Figure 3-6 shows the results. The plots are

analogoous to those in Figure 3-5 with one key difference. As mentioned in Section 3.2.3 the method of [1] does not always return valid estimates for the contact location and force. As analyzed in detail in Section 3.2.3 this is due to the fact that the method of [1] doesn't impose the restrictions implied the fact that the contact must lie on the robot surface and the applied force must be within the friction cone. This situation is exacerbated on the Kuka arm where the Jacobian $J_{r_c} \in \mathbb{R}^{6\times 7}$. In particular if the contact is on the k^{th} link then $\text{rank}(J_{r_c}) \leq k$. Since in our experiments we have contacts with $k = 4, 5, 6, 7$ it is often the case that J_{r_c} is not of full rank. In some of these situations the line of force action, described in Section 2.1.2, will fail to intersect the link surface. In this case the method of [1] simply fails to provide an estimate for the contact location and force. In Figure 3-6 we denote these instances by gray squares.

The results of Figure 3-6 show that the method of [1] doesn't return a valid estimate for 6 of the 24 test cases, while the CPF by construction always returns a valid estimate. In addition, even though an estimate is provided for contact location 4 the accuracy is very poor, with errors of more than 12 cm in the estimated contact location. Comparing the left column and right column of Figure 3-6 we see that even in the small number of scenarios where their method returns valid estimates, the CPF almost always provides more accurate estimates. We can conclude that although the method of [1] is very computationally efficient, the fact that it doesn't impose the restriction that the contact force lie on the link surface and satisfy the friction cone restrictions, as detailed in Section Section 3.2.3, severely limits its usefulness in real world scenarios.

The method of [1] is also more sensitive to noise in the residual than the CPF. Figure 3-7 plots the results of exactly the same experiments as 3-6 but with Gaussian noise with standard deviation 0.5 added the residual. The squares containing red lettering indicate that the method of [1] failed to return a valid estimate at some point during the experiment. In this case we see that the number of experiments in which an invalid estimate was returned jumps up to 13 from 6. In addition to simply not returning an estimate, the localization accuracy of the method of [1] suffers more

than that of the CPF with the addition of noise. This can be seen across all the dimensions, contact location, contact force magnitude and contact force direction.

3.3.4 Additional Sensors

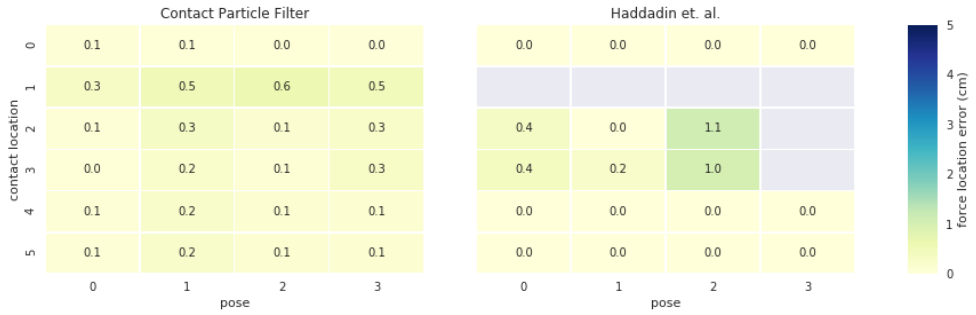
As mentioned previously the dimension of the residual vector γ captures the difference between the modeled and the sensed dynamics. In particular as more sensors are added the amount of information contained in the residual increases. In this section we consider the effect of adding a 6-axis force torque sensor to the robot base. This raises the dimension of the residual γ from $\mathbb{R}^7$ to $\mathbb{R}^{13}$ so there is much more information contained in the residual. In particular we are able to reliably and accurately localize forces that occur on the first 3 links of the Kuka robot, something that wasn't possible without the force-torque sensor. Figure 3-8 shows the contact locations used in the experiments. Results of the experiments are given in Figure 3-9. In particular we note that in the situation where we have the 6-axis force-torque sensor the localization accuracy is almost perfect. However, when we don't have the 6-axis force-torque sensor localization accuracy is significantly diminished. This is because when the contact is on the j^{th} link only the first j rows of the contact Jacobian are non-zero. Thus in the case of a contact on link 2, there are only two non-zero components of the residual γ. Section (3.5.3) provides a more detailed discussion.

3.4 Hardware Experiments with Kuka iiwa Arm

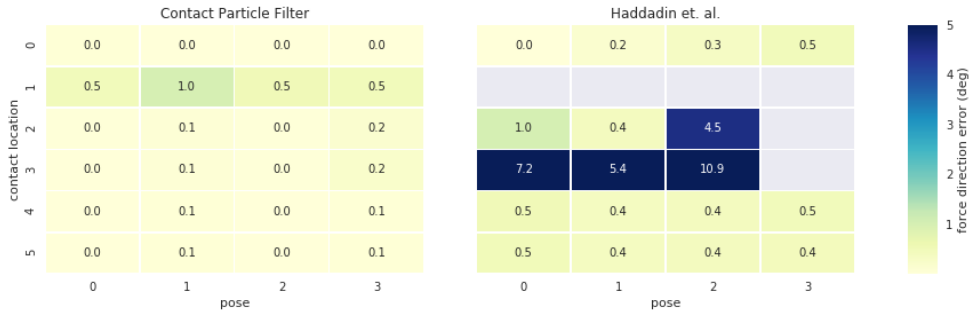
3.4.1 Experimental Setup

Robot

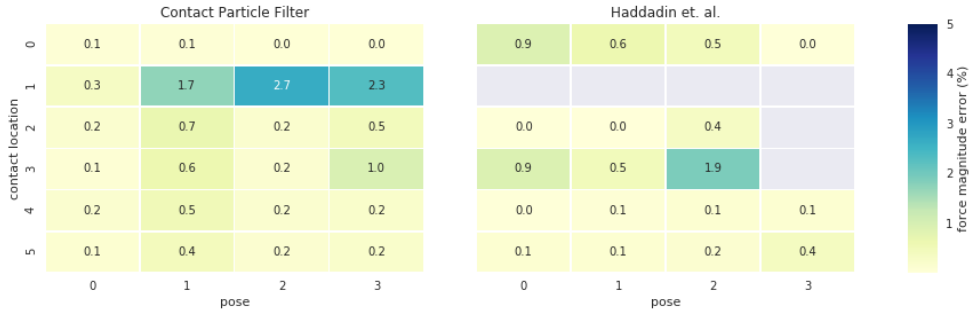
All the hardware experiments outlined below used the same experimental setup. The robot we used was a Kuka iiwa robot arm, see Figure 3-10a. This robot arm has seven degrees of freedom and is equipped with high resolution encoders and torque sensors at each joint. These torque sensors are essential in providing the information needed to compute the residual torque. Kuka uses a high fidelity internal model to compute



(a)

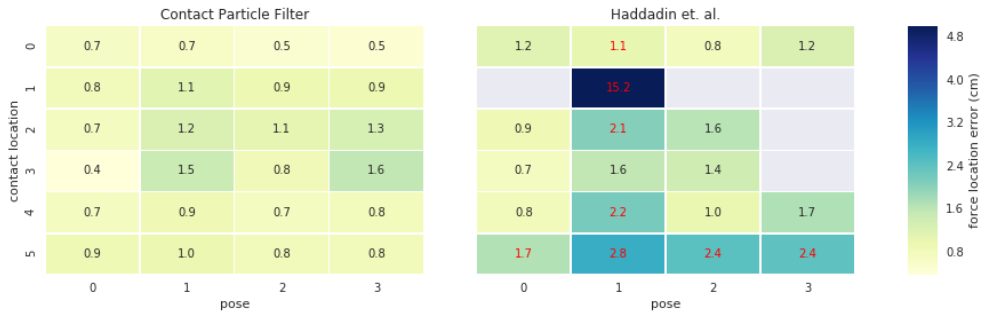


(b)

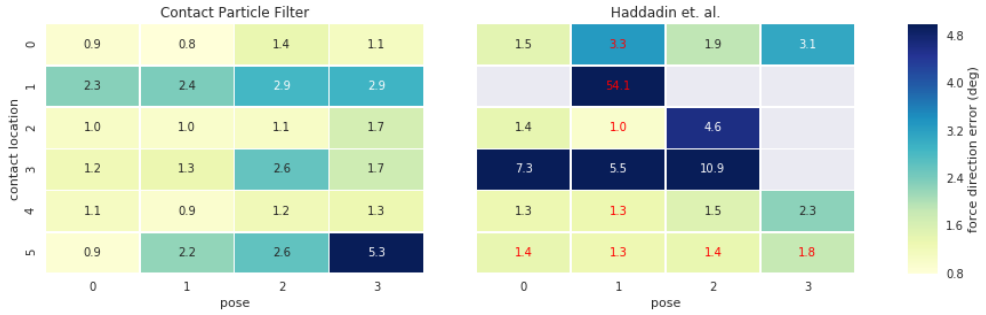


(c)

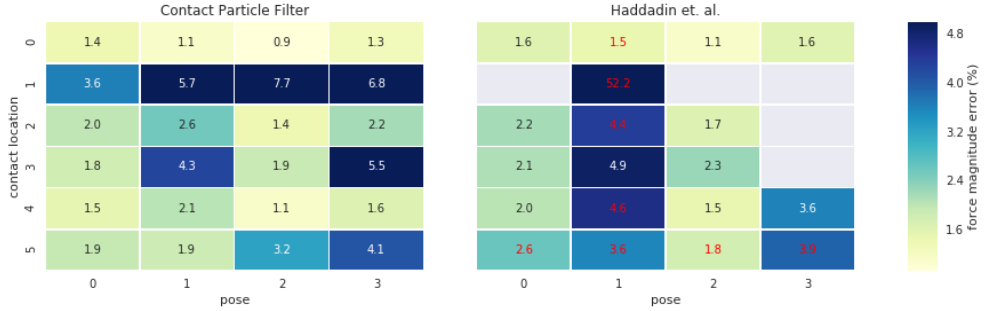
Figure 3-6: Results of simulation experiments run on the Kuka robot. A simulation experiment was run for each combination of contact location and robot pose. For purposes of comparison to the two step estimator no noise was added to the residual. The left hand plots show the performance of the CPF. The right hand plots show the performance of the method of [1] described in Section 2.1.2. This method can fail to return an estimate for the contact location, force and direction, when the line of force action doesn't intersect the robot surface. This is indicated by a grey square in the above. (3-6a) shows the average error, measured in centimeters, of the estimated contact location as compared to the true contact location. (3-6b) shows the average angle, measured in degrees, between the estimated force direction and the measured force direction. (3-6c) shows the estimated force magnitude error as a percentage of the actual force.



(a)

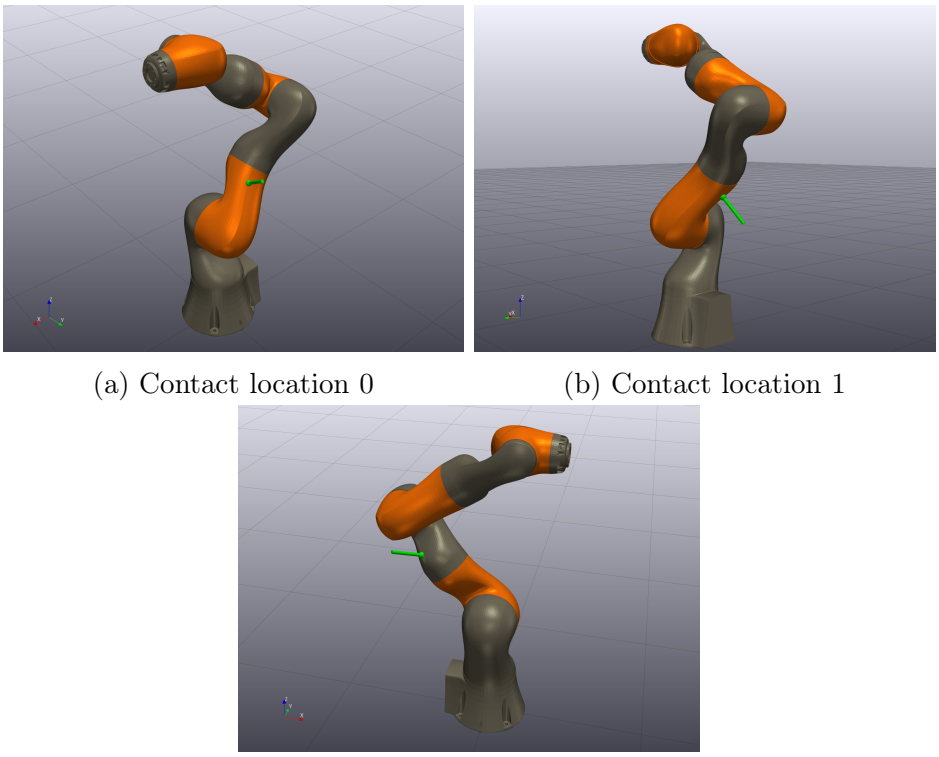


(b)



(c)

Figure 3-7: Results of simulation experiments run on the Kuka robot. A simulation experiment was run for each combination of contact location and robot pose. In these experiments the Gaussian noise with standard deviation 0.5 was added the residual. The left hand plots show the performance of the CPF. The right hand plots show the performance of the method of [1] described in Section 2.1.2. This method can fail to return an estimate for the contact location, force and direction, when the line of force action doesn't intersect the robot surface. This is indicated by a grey square in the above. If the method of [1] only returned a valid estimate some of the time, the average performance when an estimate was returned is plotted and the value is shown in red. (3-7a) shows the average error, measured in centimeters, of the estimated contact location as compared to the true contact location. (3-7b) shows the average angle, measured in degrees, between the estimated force direction and the measured force direction. (3-7c) shows the estimated force magnitude error as a percentage of the actual force.

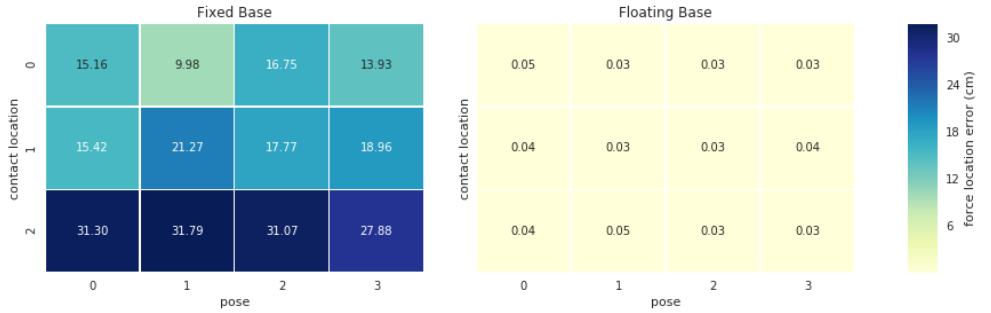


(a) Contact location 0

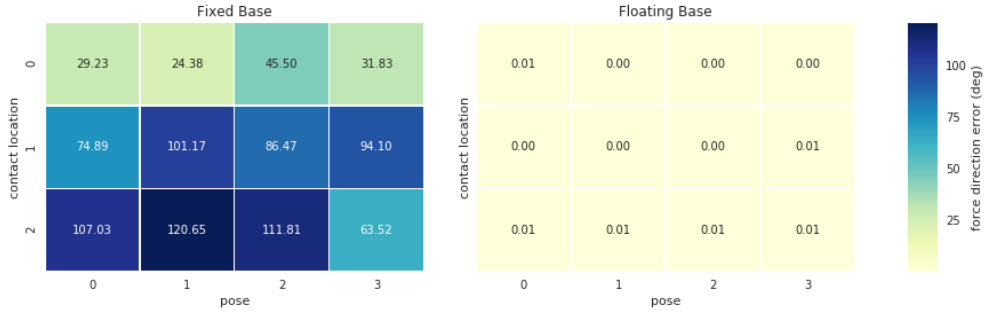
(b) Contact location 1

(c) Contact location 2

Figure 3-8: The three contact locations used in the Kuka iiwa simulation experiments of Section 3.3.4. Each contact location is indicated with a green line segment along the normal direction.



(a)



(b)

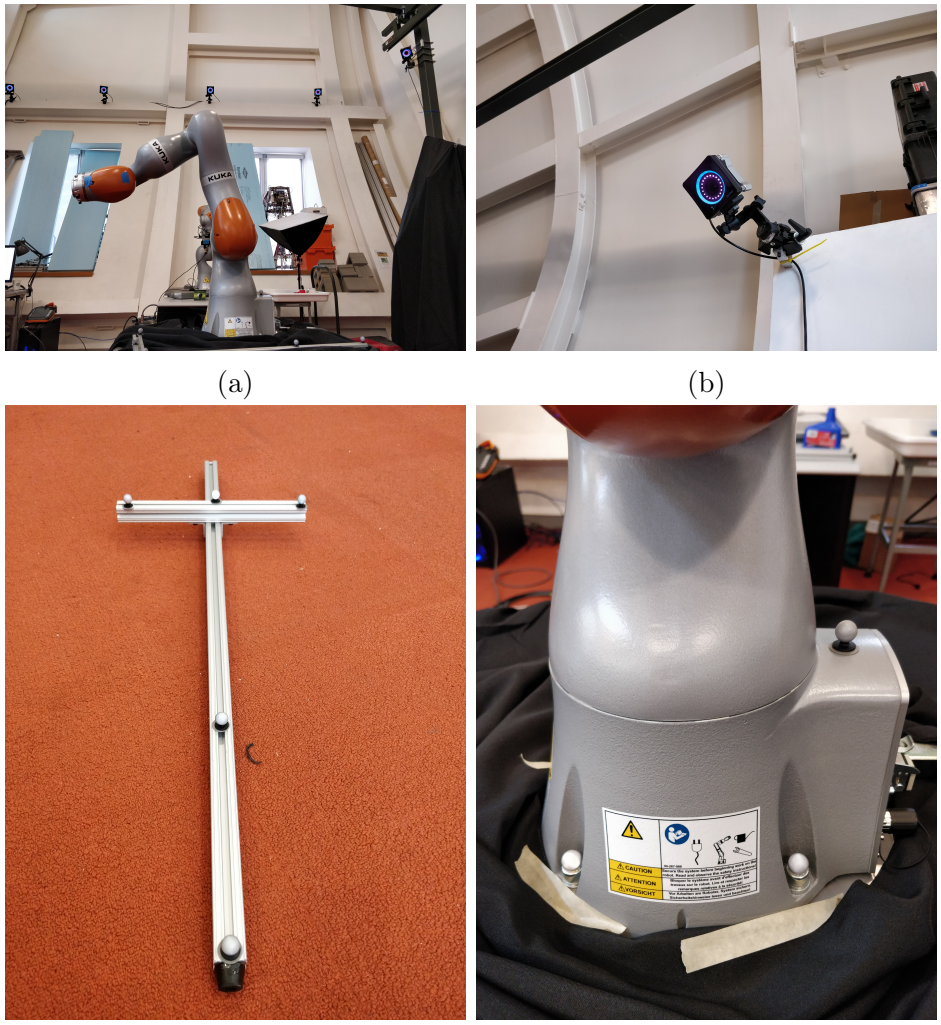
Figure 3-9: Results of simulation experiments run on the Kuka robot. A simulation experiment was run for each combination of contact location and robot pose. No noise was added the residual in these experiments. The left hand plots show the performance of the CPF without the additional force-torque sensor. The right hand plots show the performance of the CPF with the 6-axis F/T sensor. (3-9a) shows the average error, measured in centimeters, of the estimated contact location as compared to the true contact location. (3-9b) shows the average angle, measured in degrees, between the estimated force direction and the measured force direction.

an estimate of the residual γ (defined in Equation 2.4) which is exposed via an API call. The provided estimate is quite accurate. If no external forces are acting on the robot then the residual torque should be the zero vector. In practice, when no external forces are acting on the Kuka the magnitude of each component of the residual torque is less than 0.5Nm for all robot poses. Although the residual torque estimate exhibits a small amount of bias, the signal is constant for different poses and doesn't exhibit significant noise characteristics apart from the bias. Performance while the robot is in motion is worse. By the same logic as before moving the robot while not applying any external forces should result in the residual vector being identically zero. In practice individual components of the residual exhibit magnitudes up to 10Nm depending on the specific joint and the speed at which it is moving.

OptiTrack and Force Probe

In order to estimate the ground truth contact location we built a custom force probe, depicted in Figure 3-10c. It consists of two pieces of 80-20 shaped into a cross with a rubber cap at one end. The force probe is outfitted with reflective OptiTrack markers for pose tracking. The lab space has a 12 camera OptiTrack system, see Figure 3-10, which provides 200Hz 6 DOF pose estimation for any rigid body outfitted with the markers. By having OptiTrack markers on the force probe we know the location of the force probe in the world frame. The rubber cap has a diameter of 1.5 cm so the force location measurement is only accurate up to this resolution. We can also estimate the force direction using the 6 DOF pose estimate of the force probe. However since the force probe is not equipped with a 3-axis force sensor we estimate the ground truth force direction by assuming that the applied force is in the direction of the long axis of the probe. Effectively this supposes that the frictional components of the applied force are zero. During experiments we attempt to apply a purely normal force, but some frictional forces are likely being applied. Thus the force direction estimate is subject to some errors due the possibility of frictional forces. These errors are bounded by the angle of the friction cone.

In addition the base of the Kuka iiwa is also outfitted with OptiTrack markers,



(a) (b)

(c) (d)

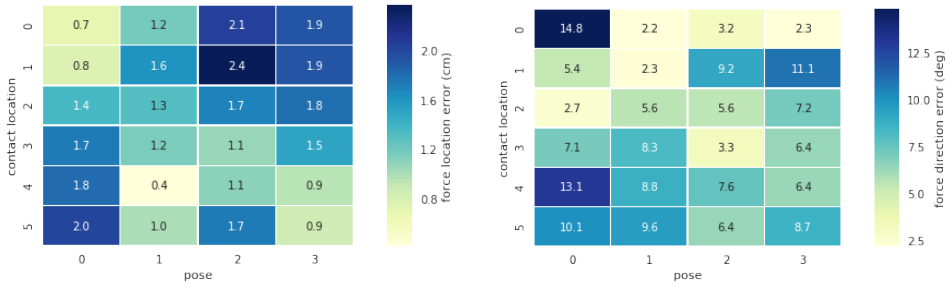
Figure 3-10: Setup for the hardware experiments. (3-10a) The Kuka iiwa robot used in all the hardware experiments. Additionally five OptiTrack cameras can be seen mounted above and behind the robot. In total we had 12 cameras covering the workspace. (3-10b) A close up view of a single OptiTrack camera. (3-10c) The force probe used to apply external forces to the robot. It consists of two pieces of 80-20 in a cross shape. At the bottom is black rubber stopper which is where we contact the robot. Additionally there are a total of five OptiTrack markers in order to be able to accurately track the pose of the force probe. (3-10d) OptiTrack markers placed on the base of the Kuka robot so that it can be localized with respect to the force probe.

shown in Figure 3-10d, which gives us the pose of the robot in the world frame. By combining this information with the pose of the force probe we can infer the location at which the probe is making contact with the robot, and also the direction of the force being applied. Combining the diameter of the rubber cap along with calibration errors in aligning the OptiTrack markers with the robot base, we believe that the estimated contact locations from force probe are within 2 cm of the actual values.

Filter Performance

The experiments we ran on the Kuka hardware are designed to match the simulation experiments performed in section 3.3. In particular we choose the same set of force locations shown in Figure 3-3, and the same set of poses as in Figure 3-4. Since there are six contact locations and four robot poses this resulted in 24 unique data points. Results are shown in Figure 3-11. In particular Figure 3-11a shows that for all the experiments the maximum error was less than 2.4 cm, with the majority being within 2 cm of the ground truth location. Given that there is up to 2cm of calibration error in our system (see Section 3.4.1) we believe it is possible that the results are actually more accurate, and that the reported numbers are slightly conservative. To the author's knowledge these are the first such hardware experiments to be conducted. Overall the CPF delivers excellent performance, localizing the contact point to within 2.4 cm consistently across all tested contact locations and poses.

Figure 3-11b shows the angle between the estimated contact direction and the measured contact direction from the force probe. Overall the results are quite good, with a maximum of 14.8 degrees of error. As discussed in Section 3.4.1 the force direction estimates using the force probe are subject to small errors. Hence the results in Figure 3-11b should be considered an approximation. However, since our force probe cannot measure these frictional forces, we believe that the results given are an upper bound on the angle error. So it is likely that the true results are somewhat more accurate. Even with these caveats, the CPF performs well in estimating the contact direction, as well as the contact location.



(a) (b)

Figure 3-11: Results of physical experiments run on the Kuka robot. A physical experiment was run for each combination of contact location and robot pose. (3-11a) shows the average error, measured in centimeters, of the estimated contact location as compared to the true contact location. (3-11b) shows the average angle, measured in degrees, between the estimated force direction and the measured force direction. The measured force direction is only accurate up to approximately a 15 degree angle due to the fact that the force probe is not frictionless where it makes contact with the robot. Hence the applied force need not be in the direction of the axis of the force probe.

3.5 Discussion

3.5.1 Computational Complexity

Table 3.1 shows the runtime performance of the CPF for different numbers of external contacts. Currently the code is not optimized for performance and is implemented in a single threaded Python process. As discussed in Section 3.1 we used FORCES Pro [33] to solve the quadratic programs. Interestingly the quadratic program solves account for a relatively small portion of the total time, less than 13%. The majority of the time is spent in the motion model sampling points and projecting them back to the robot's surface. This portion of the code is not optimized, but with some care a substantial speedup could be achieved.

In this section we discuss some of the main limitations of the CPF and highlight areas where the algorithm can be extended.

# Contacts	Total (ms)	QP Solves (ms)	Num QP Solves
1	161	18	51
2	244	25	102
3	395	50	153

Table 3.1: Total is the total time for a single step of the filter. QP Solves is the total time spent solving quadratic programs. Num QP Solves is how many distinct QP's were solved. There were 50 particles in each particle set for this example. All computations were run on a single thread Intel Core i7 @ 3.30 GHz

3.5.2 Model Error

Probably the most important limitation of CPF is that it relies on having an accurate dynamic model of the robot. Model inaccuracies can result from unmodeled actuator dynamics, friction, link compliance, incorrect link masses and/or inertias, etc. Having an accurate model is essential since the residual detector is effectively estimating the model error given by

$$H(q)\dot{v} + C(q,v)v + g(q) - B\tau. \tag{3.3}$$

Model error affects the first three terms in the above equation, but the last term $B\tau$ depends on joint torque measurements. So the accuracy of our joint torque measurements also affects the quality of the residual estimate. When the observed dynamics don't match the model dynamics, this difference is captured by the residual γ. If our model and torque measurements are accurate then these discrepancies correspond exactly to the external torques $\tau_{ext} = \sum_c J_c^T F_c$ we want to measure. If our model or torque measurements aren't accurate then our residual will be contaminated by these errors. In short, the fidelity of our model affects the accuracy of the relation $\gamma \approx \tau_{ext}$. Since the CPF takes γ as an input, the worse the approximation $\gamma \approx \tau_{ext}$ the worse the performance of the CPF will be. On the other hand this limitation is not specific to the CPF, but rather is a fundamental limitation of any approach that relies on proprioceptive sensors. As an example of this limitation consider a pendulum with a single degree of freedom $q \in \mathbb{R}$. Suppose a control torque τ is applied which according to our model should cause the robot to move. If the robot doesn't move then there are at least two possibilities. One is that there is unmodeled friction in

the joint, corresponding to the case of an inaccurate model. The second possibility is that the pendulum is pushing against a wall and the wall is applying a contact force which exactly opposes the commanded torque. If all we have is proprioceptive sensors, in this case joint position and joint torque sensors, then there is no way to tell the difference between these two possibilities.

3.5.3 Identifiability

In section 3.2.3 we showed an example situation where the method of [1] couldn't accurately estimate the contact locations but the CPF could. Identifiability of the CPF depends on there being a unique set of contact points that can generate the current residual. In general suppose there are contact points $c_1, \ldots, c_k$ on the surface of the robot with associated contact forces $F_{c_1}, \ldots, F_{c_k}$. Then we say the contact situation is "identifiable" if there does not exist another set of contact points $\tilde{c}_1, \ldots, \tilde{c}_k$ with associated contact forces $\tilde{F}_{c_1}, \ldots, \tilde{F}_{c_k}$ which obey the friction cone and have

$$\sum_{j=1}^{k} J_{c_j}^T F_{c_j} = \tau_{ext} = \sum_{j=1}^{k} \tilde{J}_{c_j}^T \tilde{F}_{c_j} \tag{3.4}$$

If we are in a contact situation that is not identifiable, then there are multiple sets of contacts that could all produce the observed residual. In this case there two sets of contact locations are equally likely and thus the CPF could converge to $\tilde{c}_1, \ldots, \tilde{c}_k$ rather than the true contact locations $c_1, \ldots, c_k$. In practice if contact c_k is the last contact to become active, and the filter was correctly estimating the location of contacts $c_1, \ldots, c_{k-1}$ then it is likely that when contact c_k is added the filter will converge to the correct set of contact locations $c_1, \ldots, c_k$. In several of the experiments with two and three contact points the contact locations were not identifiable, however, the filter was still able to converge.

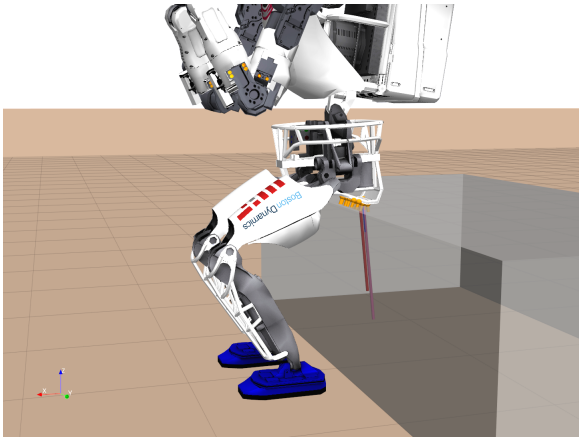


Figure 3-12: The CPF particles are shown in orange as Atlas sits on a box

3.5.4 Point Contacts

Another restriction of the CPF is that it only considers point contacts. For many real world contact situations this is a reasonable assumption for a rigid robot. There are however situations in which we have multiple or continuous contact. For example if the robot is sitting then it is possible to have many multiple contacts. Ultimately if the wrench exerted on link i by these contacts can be well approximated by the wrench exerted by a point force then the CPF will return a reasonable estimate. The estimated contact point will likely be a weighted average of the true contact points. An example of Atlas sitting on a box is shown in Figure 3-12. In this simulation there are two contacts on the pelvis, but the CPF particles still return reasonable estimates located on the bottom of the pelvis.

3.5.5 Filter Divergence

For computational efficiency we perform the measurement update for particle set $\mathcal{X}_{t,j} \in \mathscr{X}$ taking the locations of the other contacts as given. Consider a situation where the estimate $r^*_{t,k}$ of the location of the k-th contact point given by $\mathcal{X}_{t,k}$ is not accurate. Then when performing the measurement update for a particle $r^{[m]}_{t,j} \in \mathcal{X}_{t,j}$ the force F_c applied at $r^{[m]}_{t,j}$ will have to match the contribution of the j-th contact point c_j to τ_{ext} but also the contribution of contact point c_k that cannot be explained by the poor estimate $r^*_{t,k}$. This can lead to incorrect importance weights for particles

in X_t. If $r^*_{t,k}$ is a sufficiently bad estimate of c_k then the filter can diverge. However the filter never diverged during the 42 simulation runs used in Figure 3-1.

3.5.6 Sequential Arrival of External Contacts

The CPF maintains as many particle sets as there are external contacts. Algorithm 3, Manage-Particle-Sets, adds a particle set if the current residual is not well explained by the existing particles. This relies on the assumption that new contacts arrive sequentially, not simultaneously. We think that this is not an unreasonable assumption in practice. If two point contacts arrived at once the the filter would add a single particle set to try to localize one new contact. Since there are two new contacts, the particles in this new particle set are would move towards a location that could best explain the two new contact forces with a single contact force. This can lead to filter divergence as described in the previous section.

3.5.7 Additional Proprioceptive Sensors

A nice feature of the CPF is that additional proprioceptive sensors can easily be incorporated into the algorithm without increasing the complexity. If we had an additional force-torque sensor somewhere on the robot, for example at the wrist or the shoulder, then we could augment the state of the robot to incorporate this as a fixed joint. All this does is increase the dimension of q, say from n to $n+6$. Denote quantities that use this augmented state with tildes, e.g. $\tilde{q}$. The effect of this is that external forces are projected into a higher dimensional external torque space. Namely $\tilde{\tau}_{ext} = \tilde{J}_c(\tilde{q})^T F_c \in \mathbb{R}^{n+6}$ as opposed to $\tau_{ext} = J_c(q)^T F_c \in \mathbb{R}^n$. This means that there is effectively more information encoded in $\tilde{\tau}_{ext}$, which increases the likelihood that the quadratic program in the measurement update step has a unique optimum. There is almost no additional computation cost to this, as the only changes are computing Jacobians in the new space $\tilde{q}$ instead of q. Thus the CPF can easily incorporate additional proprioceptive information.

Chapter 4

Conclusion

This thesis introduces the CPF, the Contact Particle Filter, a general algorithm for detection and localization of external contacts on rigid body robots using only proprioceptive sensing. The algorithm uses the measurements of joint position, velocity and effort to compute an estimate of the difference between expected dynamics from the model and direct measurements of torques. A non-zero residual must be due to external forces. The CPF runs a particle filter to find external contact points that can best explain the estimated external joint torque. In particular we take advantage of the fact that once a set of potential contact locations is specified, computing how well they explain the observed residual can be formulated as quadratic program. The CPF leverages this insight to tractably formulate the problem in the framework of a particle filter.

We demonstrate successfull localization of up to 3 external contacts in a simulated environment on a complex humanoid robot with 36 degrees of freedom. In addition we perform extensive simulation analysis on a Kuka iiwa robot, and show that performance carries over to the real robot hardware. Each application of CPF requires only a dynamic model of the robot and a description of the surface manifold of each link, no underlying algorithmic changes are necessary.

We believe that being able to reliably estimate and localize external contacts, both expected and unexpected, is a necessary first step in developing robots that can interact safely and intelligently with their environment. The CPF presents an

approach to solving this problem.

Bibliography

[1] S. Haddadin, A. De Luca, and A. Albu-SchÃďffer. Robot collisions: A survey on detection, isolation, and identification. *IEEE Transactions on Robotics*, 33(6):1292–1312, Dec 2017.

[2] Antonio Bicchi, Michael A. Peshkin, and J. Edward Colgate. *Safety for Physical Human–Robot Interaction*, pages 1335–1348. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[3] Agostino De Santis, Bruno Siciliano, Alessandro De Luca, and Antonio Bicchi. An atlas of physical humanâĂŞrobot interaction. *Mechanism and Machine Theory*, 43(3):253 – 270, 2008.

[4] Fabrizio Flacco, Torsten Kröger, Alessandro De Luca, and Oussama Khatib. A depth space approach to human-robot collision avoidance. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 338–345, 2012.

[5] D. M. Ebert and D. D. Henrich. Safe human-robot-cooperation: image-based collision detection for industrial robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1826–1831 vol.2, 2002.

[6] S. Kuhn and D. Henrich. Fast vision-based minimum distance determination between known and unkown objects. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2186–2191, Oct 2007.

[7] Ravinder S Dahiya, Philipp Mittendorfer, Maurizio Valle, Gordon Cheng, and Vladimir J Lumelsky. Directions Toward Effective Utilization of Tactile Skin : A Review. *IEEE Sensors*, 13(11):4121–4138, 2013.

[8] Michael Strohmayr. *Artificial Skin in Robotics*. PhD thesis, KIT, 2012.

[9] G De Maria, C Natale, and S Pirozzi. Sensors and Actuators A : Physical Force / tactile sensor for robotic applications. *Sensors & Actuators: A. Physical*, 175:60–72, 2012.

[10] Vladimir J Lumelsky and Edward Cheung. Real-Time Collision Avoidance in Teleoperated Whole-Sensitive Robot Arm Manipulators. 23(1):194–203, 1993.

[11] M. D. Killpack and C. C. Kemp. Fast reaching in clutter while regulating forces using model predictive control. In *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 146–153, Oct 2013.

[12] K. Suita, Y. Yamada, N. Tsuchida, K. Imai, H. Ikeda, and N. Sugimoto. A failure-to-safety "Kyozon" system with simple contact detection and stop capabilities for safe human-autonomous robot coexistence. *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, 3:3089–3096, 1995.

[13] Shinji Takakura, Toshiyuki Murakami, and Kouhei Ohnishi. An Approach to Collision Detection and Recovery Motion in Industrial Robot. *15th Annual Conference of IEEE Industrial Electronics Society*, pages 421–426, 1989.

[14] Yoji Yamada, Yasuhiro Hirasawa, Shengyang Huang, Yoji Umetani, and Kazutsugu Suita. Human-robot contact in the safeguarding space. *IEEE/ASME Transactions on Mechatronics*, 2(4):230–236, 1997.

[15] S. Morinaga and K. Kosuge. Collision detection system for manipulator based on adaptive impedance control law. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 1, pages 1080–1085 vol.1, Sept 2003.

[16] T. Matsumoto and K. Kosuge. Collision detection of manipulator based on adaptive control law. In *2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. Proceedings (Cat. No.01TH8556)*, volume 1, pages 177–182 vol.1, 2001.

[17] A. De Luca and R. Mattone. Actuator failure detection and isolation using generalized momenta. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 1, pages 634–639 vol.1, Sept 2003.

[18] H.-B Kuntze, Christian Frey, K Giesen, Giulio Milighetti, Fraunhofer Iitb, and Page . Fault tolerant supervisory control of human interactive robots. 01 2003.

[19] Alessandro De Luca and Raffaella Mattone. Sensorless robot collision detection and hybrid force/motion control. *Proceedings - IEEE International Conference on Robotics and Automation*, 2005(April):999–1004, 2005.

[20] Alessandro De Luca, Alin Albu-Schäffer, Sami Haddadin, and Gerd Hirzinger. Collision detection and safe reaction with the DLR-III lightweight manipulator arm. *IEEE International Conference on Intelligent Robots and Systems*, pages 1623–1630, 2006.

[21] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger. Collision detection and reaction: A contribution to safe physical human-robot interaction. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3356–3363, Sept 2008.

[22] A. De Luca and L. Ferrajoli. Exploiting robot redundancy in collision detection and reaction. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3299–3305, Sept 2008.

[23] S. Parusel, S. Haddadin, and A. Albu-SchÃďffer. Modular state-based behavior control for safe human-robot interaction: A lightweight control architecture for a lightweight robot. In *2011 IEEE International Conference on Robotics and Automation*, pages 4298–4305, May 2011.

[24] Vahid Sotoudehnejad, Amir Takhmar, Mehrdad R Kermani, and Ilia G Polushin. Counteracting Modeling Errors for Sensitive Observer-Based Manipulator Collision Detection. In *IEEE International Conference on Intelligent Robots and Systems*, pages 4315–4320, 2012.

[25] Emanuele Magrini and Alessandro De Luca. Estimation of Contact Forces using a Virtual Force Sensor. Number IEEE International Conference on Intelligent Robots and Systems, pages 2126–2133, 2014.

[26] M. C. Koval, M. R. Dogar, N. S. Pollard, and S. S. Srinivasa. Pose estimation for contact manipulation with manifold particle filters. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4541–4548, Nov 2013.

[27] Li Han, Jeff C Trinkle, and Zexiang X Li. Grasp Analysis as Linear Matrix Inequality Problems. *IEEE Transactions on Robotics*, 16(6):663–674, 2000.

[28] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[29] Scott Kuindersma, Frank Permenter, and Russ Tedrake. An Efficiently Solvable Quadratic Program for Stabilizing Dynamic Locomotion. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1–6, 2014.

[30] V. Sotoudehnejad, A. Takhmar, M. R. Kermani, and I. G. Polushin. Counteracting modeling errors for sensitive observer-based manipulator collision detection. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4315–4320, Oct 2012.

[31] R. Isermann. *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Springer Berlin Heidelberg, 2005.

[32] Pat Marion. Director: A robotics interface and visualization framework, 2015.

[33] Alexander Domahidi and Juan Jerez. FORCES Professional. embotech GmbH (`http://embotech.com/FORCES-Pro`), July 2014.

[34] Russ Tedrake and the Drake Development Team. Drake: A planning, control, and analysis toolbox for nonlinear dynamical systems, 2016.

[35] Nicolaus Radford et. al. Valkyrie : NASA's First Bipedal Humanoid Robot. *Journal of Field Robotics*, 32(3):397–419, 2015.