Localizing external contact using proprioceptive sensors: The Contact Particle Filter

Localizing External Contact Using Proprioceptive Sensors: The Contact Particle Filter

Lucas Manuelli[1] and Russ Tedrake[1]

***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 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. 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 to track multiple external contacts on a simulated Atlas robot, and compare our work to existing approaches.**

I. INTRODUCTION

Currently 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.

As humans we have skin covering our entire body which allows us to easily sense external contacts. However, since high performance sensing skin is not yet commonplace on robots our algorithms must rely on proprioceptive sensors. The main contribution of this paper is an estimation algorithm that is capable of localizing multiple external contacts using only proprioceptive sensors.

The paper is organized as follows. Section II discusses related work, section III presents a brief introduction to the key results of previous work. In Section IV we describe our approach, the Contact Particle Filter. Section V presents experimental results, section VI considers limitations and extensions and section VII concludes.

II. RELATED WORK

One approach to solving the collision detection and localization problem is to use a sensitive skin [1], [10]. Unfortunately, most robots do not come equipped with such sensitive skins, and if they do they are usually only on a few key locations, such as the hands and feet. Thus we focus on the problem of collision detection and localization using only proprioceptive sensors. Initial approaches to this problem involved monitoring the measured currents in the robot's electrical motors and looking for fast transients that could be caused by a collision [11], [12]. More recently a collision detection method based on generalized momentum has been proposed in [3]. An advantage of the this method is that it doesn't require acceleration measurements, which are very noisy in practice. We use the generalized momentum observer as the starting point for our estimation algorithm, a brief overview of the method is given in III-A. [2] uses the method of [3] in a collision detection and safe reaction framework using a DLR-III lightweight manipulator arm. In particular, the momentum observer provides sufficient directional information to allow a manipulator arm to react safely after a collision. [9] uses the momentum observer of [3] together with time-varying collision detection thresholds to provide more accurate collision detection performance in the presence of model errors.

[7] uses the generalized momentum observer, together with an external depth camera to estimate the interaction force between a robot and an external contact. In this 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 method outlined in Section IV B of [4]. They show how to use the generalized momentum detector to determine the location of a single external contact. However, this method doesn't extend well to the case of multiple external contacts and is susceptible to measurement noise. We present a comparison of our method to that of [4] in section V-B.

III. 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

$$H(q)\dot{v} + C(q, v)v + g(q) = B\tau + \tau_{ext}. \tag{1}$$

$H(q) \in \mathbb{R}^{n_v \times n_v}$ 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

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

[1] CSAIL, Massachusetts Institute of Technology, Cambridge, MA, USA. {manuelli,russt}@csail.mit.edu

torques to the actuated joints, and τ_{ext} are the external joint torques acting on the robot. 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

$$\mathbf{J}_{r_c}(q)^T\Gamma_c = \mathbf{J}_{r_c}(q)^T \begin{bmatrix} F_c \\ M_c \end{bmatrix}. \tag{2}$$

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

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

A. Residual Obsever

In this section we provide an overview of the momentum observer method of [3] which provides and estimate of the external joint torque $\tau_e xt$. Following their treatment define the residual vector $\gamma(t) \in \mathbb{R}^{q_v}$ as

$$\gamma(t) = K_I \left(p - \int_0^t (B\tau + C^T(q,v)v + \gamma(s))ds \right), \tag{4}$$

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

$$\dot{\gamma}(t) = K_I(\tau_{ext} - \gamma). \tag{5}$$

If K_I is sufficiently large we can suppose that $\gamma \approx \tau_{ext}$, so 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

$$\begin{aligned} \gamma(t) =& K_I \left(p - \int_0^t (B\tau + \sum_{i=1}^{j} \mathbf{J}_{c_i}(q)^T\Gamma_{c_i} \right. \\ & \left. + C^T(q,v)v + \gamma(s))ds \right). \end{aligned} \tag{6}$$

Henceforth we let τ_{ext} denote the external joint torques produced by unmeasured external wrenches.

IV. CONTACT DETECTION AND LOCALIZATION

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 [4] 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. Then solving for the contact location can be formulated as an optimization

$$\min_{r_c, F_c} (\gamma - J_{r_c}(q)^T F_c)^T (\gamma - J_{r_c}(q)^T F_c) \tag{7}$$

$$\text{subject to } r_c \in \mathcal{S}, F_c \in \mathcal{F}(r_c). \tag{8}$$

The optimization (7) 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.

$$\min_{F_c} (\gamma - J_{r_c}(q)^T F_c)^T (\gamma - J_{r_c}(q)^T F_c) \tag{9}$$

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

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 [5] in the context of grasp analysis. One way to approximate the solution to problem (7) is to sample contact locations $r_c \in \mathcal{S}$ and then solve the convex problem (9) 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 (7). In section IV-A 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 (9). In this way we avoid the intractability of problem (7).

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

A. Single External Contact

For simplicity, we first describe a version of our contact particle filter in the case of a single external contact. Let $\gamma(t)$ be the residual observer from section III. Our goal is to estimate the location of the external contact point. In order to take advantage of the convex formulation (9) we use a particle filter. 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 IV-A.1, and a motion model, detailed in section IV-A.2.

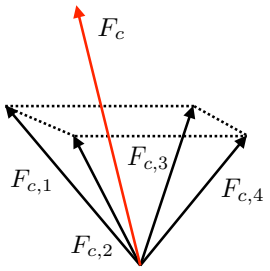


Fig. 1: Polyhedral approximation to the friction cone

1) 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,

$$\gamma = \tau_{ext} + \eta, \text{ where } \eta \sim \mathcal{N}(0, \Sigma_{meas}). \tag{10}$$

Now define

$$\begin{aligned} \varepsilon = \min_{F_c} &(\gamma - J_{r_t^{[m]}}^T F_c)^T \Sigma^{-1} (\gamma - J_{r_t^{[m]}}^T F_c) \\ &\text{subject to } F_c \in \mathcal{F}(r_t^{[m]}). \end{aligned} \tag{11}$$

Following the approach in [6] we replace the friction cone with a polyhedral approximation shown in Figure 1. This polyhedral approximation to the friction cone allows us to approximate (11) using a quadratic program.

$$\begin{aligned} QP(\gamma|r_t^{[m]}) = \min_{\alpha_i, F_c} &(\gamma - J_{r_t^{[m]}}^T F_c)^T \Sigma_{meas}^{-1} (\gamma - J_{r_t^{[m]}}^T F_c) \\ &\text{subject to } \alpha_i \geq 0, \quad F_c = \sum_{i=1}^{4} \alpha_i F_{c,i}. \end{aligned} \tag{12}$$

Then $\varepsilon \approx QP(\gamma|r_t^{[m]})$. We can recover a likelihood using the fact that $\gamma = \tau_{ext} + \eta$. Namely

$$p(\gamma|r_t^{[m]}) \propto \exp\left(-\frac{1}{2} QP(\gamma|r_t^{[m]})\right), \tag{13}$$

where we have ommitted 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.

2) 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

$$p(r_t|r_{t-1}) \propto \mathcal{N}(r_t; r_{t-1}, \Sigma_{motion}). \tag{14}$$

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 zero velocity 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.

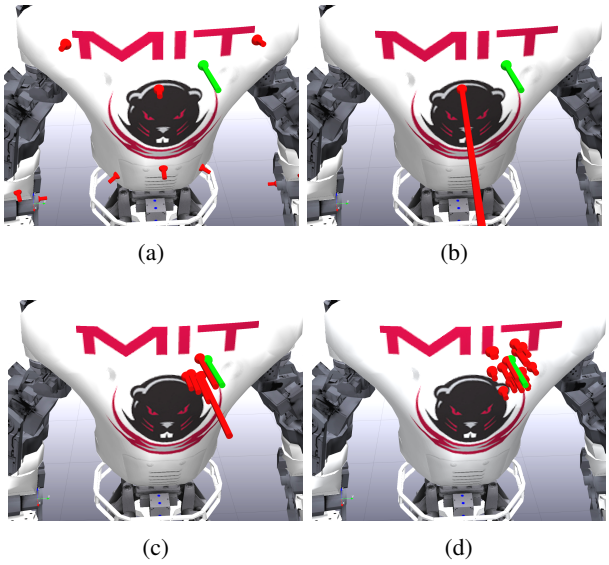
3) Contact Particle Filter: Now we combine the measurement and motion models to form the single contact particle filter (Single-CPF). As in [2] and [4] we must specify a threshold for determining when there is an external contact. Define $\epsilon(t) = \gamma(t)^T \Sigma_{meas}^{-1} \gamma(t)$. We say that there is an external contact if $\epsilon(t)$ is greater than some threshold $\bar{\epsilon}$. 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 $w_t^{[m]}$.

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

The final step is to recover the most likely contact location given a particle set $\mathcal{X}_t$. This is done by averaging the contact locations $r_t^{[m]}$ for $r_t^{[m]} \in \mathcal{X}_t$ and projecting this point back to the surface of the robot. Label this procedure Get-Contact-Location($\mathcal{X}_t$).

Figure 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, line 15 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 6 and sets $\mathcal{X}'_t = \mathcal{X}_{init}$. This is visualized in Figure 2a. Subsequent filter steps shown in Figure 2b-2d show the particles converging to the true contact location.



(a) (b) (c) (d)

Fig. 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.

B. 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 IV-B.1 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 k-th contact point $r_{t,k}$. Then define

$$QP(\gamma|(r_{t,1}, \ldots, r_{t,l})) = \min_{\alpha_{i,k}, \hat{\tau}_{ext}} (\gamma - \hat{\tau}_{ext})^T \Sigma_{meas}^{-1} (\gamma - \hat{\tau}_{ext}) \tag{15}$$

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

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

$$p(\gamma|(r_{t,1}, \ldots, r_{t,l})) \propto \exp\left(-\frac{1}{2} QP(\gamma|(r_{t,1}, \ldots, r_{t,l}))\right). \tag{16}$$

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. 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 IV-A we must define the measurement model and the motion model.

1) 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 IV-A.3 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 $r_{t,j}^{[m]} \in \mathcal{X}_{t,j}$ is defined as

$$p(\gamma|r_{t,j}^{[m]}, \mathscr{X}) \propto \exp\left(-\frac{1}{2} QP(\gamma|(r_{t,j}^{[m]}, \mathbf{r}))\right). \tag{17}$$

where $QP(\gamma|(r_{t,j}^{[m]}, \mathbf{r}))$ refers to (15). The full measurement update is detailed in Algorithm 2.

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

```
\overline{\mathscr{X}}_t = ∅
for \mathcal{X}_{t,j} ∈ \mathscr{X} do
    \overline{\mathcal{X}}_{t,j} = ∅
    r = {r*_{t,k}}_{k≠j}
    for r^{[m]}_{t,j} in \mathcal{X}_{t,j} do
        w^{[m]}_{t,j} = exp(-½ QP(γ|(r^{[m]}_{t,j}, r)))
        \overline{\mathcal{X}}_{t,j} = \overline{\mathcal{X}}_{t,j} + ⟨r^{[m]}_{t,j}, w^{[m]}_{t,j}⟩
    end for
end for
return \overline{\mathscr{X}}_t
```

2) 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 IV-A.2. 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 $w_{t,j}^{[m]}$ computed in the measurement update step.

3) 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

$$\epsilon(\mathscr{X}_t, \gamma) = QP(\gamma|\mathbf{r}^*(\mathscr{X}_t)). \tag{18}$$

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

$$QP(\gamma|\mathbf{r}^*(\mathscr{X}_t)) = \gamma^T \Sigma_{meas}^{-1} \gamma. \tag{19}$$

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($\gamma, \mathscr{X}_{t-1}$)

```
X_t = X_{t-1}
if ε(γ, X_{t-1}) > ε̄ then
    add X_init to X_t
    return X_t
end if
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
    end if
end for
return X_t
```

4) 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($\gamma, \mathscr{X}_{t-1}$)

```
X_motion = Motion-Model(X_{t-1})
X_meas = Multi-Measurement-Update(γ, X_motion)
X_resample = Importance-Resample(X_meas)
X_t = Manage-Particle-Sets(γ, X_resample)
return X_t
```

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

V. SIMULATION RESULTS

We perform experiments using a simulated model of the Atlas robot. 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 III-A 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 [8] have these 6-axis force-torque sensors. To test our algorithm we also augment the simulator in order to be able to apply 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.

To speed up development the CPF was implemented in a single thread Python process. The quadratic programs in the measurement update step were solved using FORCES Pro. The CPF must be passed a complete surface mesh of the robot at initialization, but the only required realtime information are the joint positions q and the residual γ.

Section V-A gives quantitative results on the localization performance of the CPF. Section V-B provides a comparison of the performance of the CPF and the method of [4]. Section V-C discusses the computational speed of the CPF. A video of the CPF is available at http://youtu.be/ckvsMK0QhB0.

A. Localization Performance

In all the experiments the filter used particle sets of size 50. During the simulations the robot was moving but not walking around. We considered situations with either 1,2 or 3 external contacts with a 10 Newton force applied at each contact point. We tested 7, 4 and 3 different contact locations for the 1,2 and 3 contact scenarios, respectively. In addition we artificially injected noise into the residual γ to test the performance of the filter under non-ideal conditions. The results are summarized in Figure 3.

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 (4) had the correct inertial parameters of the robot. In addition 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.

B. Comparison to Two-Step Method

In this section we compare the performance of the CPF to the contact localization method of [4]. The method of [4] breaks the problem into two parts. First they note that a contact force F_c applied at a point r_c on link i can be transformed to an equivalent wrench $\Gamma_i = [F_i, M_i]^T$ at a

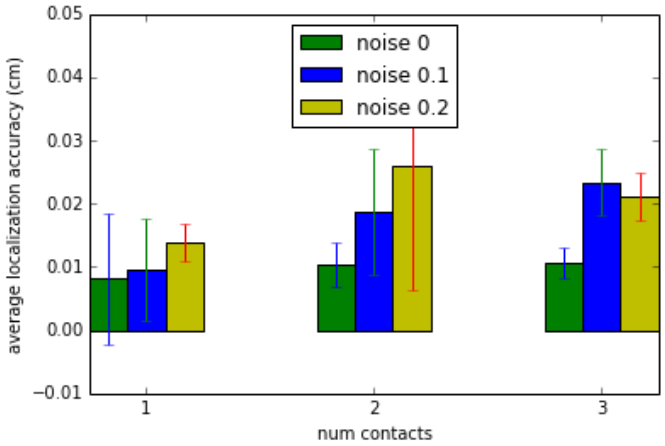


Fig. 3: This chart shows the localization performance of the CPF across experiments with different numbers of external contacts and different noise properties. We added noise with the standard deviations listed above to the residual before it was used in CPF. For each noise setting we tested 7, 4 and 3 different contact locations for the 1,2 and 3 contact scenarios, respectively. We report the average localization performance across the multiple trials.

known frame attached to link i. The method in [4] first solves for Γ_i, and then attempts to back out the contact location r_c and force F_c that generated this link wrench. They achieve this by finding a line $\mathcal{L}_i$, called the line of force action, on which r_c must lie. Namely the force F_c applied at any point $r_c \in \mathcal{L}_i$ would generate the wrench Γ_i. The point where $\mathcal{L}_i$ intersects the link surface then gives the contact location r_c.

A typical situation is shown in Figure 4. As we can see the CPF does a good job estimating the location of both contact points, while the method of [4] doesn't provide accurate estimates. [1] In particular the line of force action for the arm faisl to intersect the robot surface. The reason that the method of [4] 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

$$\mathcal{W}_i = \{\Gamma : \Gamma = A(r_c)F_c \text{ for } r_c \in \mathcal{S}_i, F_c \in \mathcal{F}(r_c)\}. \quad (20)$$

By allowing $\Gamma_i \in \mathbb{R}^6$ rather than restricting $\Gamma_i \in \mathcal{W}_i$ in the first stage, the method of [4] doesn't take advantage of all the information in the residual γ. In the situation of Figure 4 the external joint torque is given by

[1]For the purposes of this experiment we gave the method of [4] 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 mutliple contacts on the same kinematic chain. However, for the sake of comparing the two methods we allow the method of [4] this additional information.

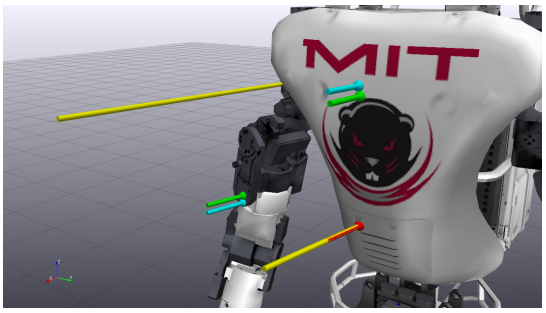


Fig. 4: There are two external contacts on the robot, shown in green. Each contact is applying a 10 Newton force. The estimated contact locations from the CPF are shown in cyan. The lines of force-action for the method of [4] are the long yellow rays.

$$\gamma \approx \tau_{ext} = \mathbf{J}_1(q)^T\Gamma_1 + \mathbf{J}_2(q)^T\Gamma_2, \quad (21)$$

where the $i = 1, 2$ subscripts denote quantities for the arm and torso, respectively. The method of [4] fails because if we allow $\Gamma_1, \Gamma_2 \in \mathbb{R}^6$, then equation (21) admits multiple solutions. [4] 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 4 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 [4] 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 (7). This additional restriction eliminates the multiplicity of solutions to (21) that plagued the method of [4] and allows the CPF to accurately localize the external contacts.

As illustrated in Figure 4 one of the failure modes of the method of [4] 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 [4], 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 [4] 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 but by construction they always lie on the link surface.

C. Computational Complexity

Table I 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. We use the Python interface of FORCES Pro to solve the QP's. Interestingly the QP 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

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

TABLE I: 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

the robot's surface. This portion of the code is not optimised, but with some care a substantial speedup could be achieved.

VI. Limitations and Extensions

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

A. 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

$$H(q)\dot{v} + C(q,v)v + g(q) - B\tau. \tag{22}$$

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, correspding 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.

B. Identifiability

In section V-B we showed an example situation where the method of [4] 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

$$\sum_{j=1}^{k} J_{c_j}^T F_{c_j} = \tau_{ext} = \sum_{j=1}^{k} \tilde{J}_{c_j}^T \tilde{F}_{c_j} \tag{23}$$

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.

C. 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 5. 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.

D. 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 $r_{t,j}^{[m]} \in \mathcal{X}_{t,j}$ the force F_c applied at $r_{t,j}^{[m]}$ 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.

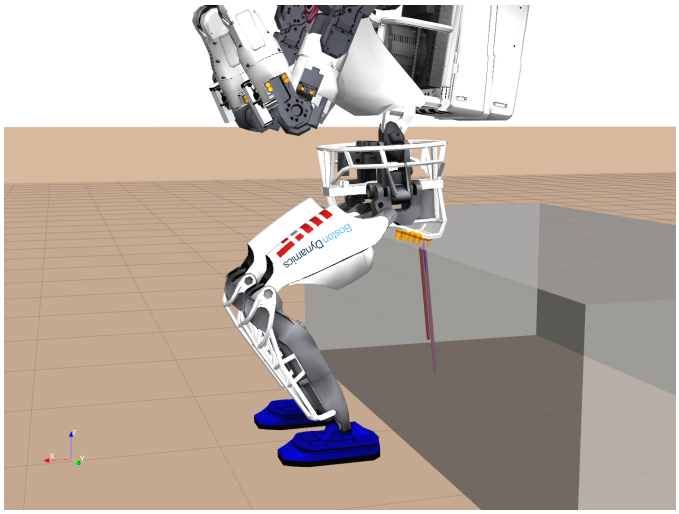


Fig. 5: The CPF particles are shown in orange as Atlas sits on a box

However the filter never diverged during the 42 simulation runs used in Figure 3.

E. 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.

F. 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.

VII. Conclusion

This paper introduces CPF, the Contact Particle Filter, a general algorithm for detection and localization of external contacts on rigid body robots using only proprioceptive sensing. CPF finds external contact points that best explain the estimated external joint torque. It takes advantage of the fact that once we specify a set of potential contact locations, 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. 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 the 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.

References

[1] 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.

[2] 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.

[3] 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.

[4] Sami Haddadin and Alessandro De Luca. Robot Collisions : Detection, Isolation, and Identification. *IEEE Transactions on Robotics*, (Submitted), 2015.

[5] 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.

[6] 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.

[7] 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.

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

[9] 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.

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

[11] 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.

[12] 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.