Identifying External Contacts from Joint Torque Measurements on Serial Robotic Arms and Its Limitations

Tao Pang, Jack Umenberger and Russ Tedrake
{*pangtao*, *ju*, *russt*}@csail.mit.edu

***Abstract*— The ability to detect and estimate external contacts is essential for robot arms to operate in unstructured environments occupied by humans. However, most robot arms are not equipped with adequate sensors to detect contacts on their entire body. What many robot arms do have is torque sensors for individual joints. Through a quantitative analysis, we argue that it is fairly likely for two distinct contacts on the robot's surface to generate almost identical joint torque measurements. When this happens, the best contact estimate achievable is the set of possible contact positions, all of which would reproduce the measured joint torque. Searching for elements of this set is equivalent to solving to global optimality a nonlinear program.**

By combining rejection sampling with gradient descent, we propose a contact estimation method which in practice finds all local optima of the nonlinear program at real-time rates. In addition, we propose an active contact exploration method which falsifies spurious contact estimates in the set of local optima by making small motions around the robot's current configuration. The proposed methods highlight the caveats of contact estimation from only joint torque, which, coupled with known limitations of such estimators, suggest that a more capable sensor is probably needed for robust whole-body contact estimation.

I. Introduction

No longer confined to factory-floor workcells repeating painstakingly hand-coded trajectories, robot arms today have been tasked with increasingly open-ended assignments such as "put this shoe on the shelf" or "load the dish washer", where the robots need to operate in unknown, unstructured environments potentially populated by humans. Naturally, the safety of such operations hinges upon the robot's ability to reliably handle unplanned collisions between any part of itself and the environment.

The ultimate sensor for collision detection is perhaps a sensitive tactile skin covering the entire surface of the robot [1], [2]. However, such skins are rarely seen outside research labs, as they are usually expensive and prone to wear and tear. On the other hand, joint-level proprioceptive torque sensors are mature, robust and becoming more common in robot arms designed for human-robot interactions [3], [4].

Several techniques have been developed to estimate both the external contact force and its location on the whole robot using only proprioceptive torque sensors [5]–[8]. However, due to the sparse nature of proprioceptive measurements (one torque measurement per link), estimation of contact force and location from only joint torque has obvious limitations. For a typical serial robot arm with 7 links, when link i (numbered from the base) of the arm is in contact, there are i torque measurements available, and $i \leq 6$ if the contact is not on the end effector. The fact that the contact is pushing the robot gives two additional constraints: the force being in the friction cone and the position being on the robot's surface. In contrast, a contact position has 3 independent components to estimate, and a contact wrench has 6. As a result of this deficiency in available measurements, the literature on joint-torque-based contact estimation commonly assumes that (**i**) there exists at most one external contact, and (**ii**) the contact generates negligible moments at the contact point [5], [8].

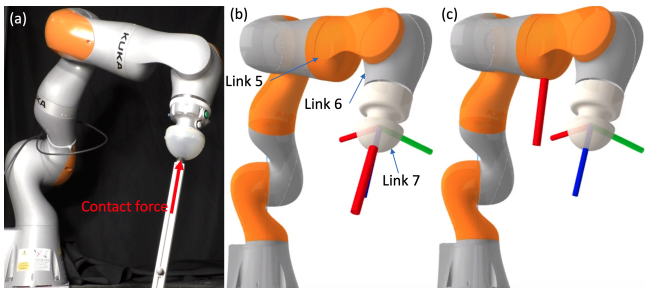


Fig. 1: For the true contact shown in (**a**), which generates little torque about joint 6 and 7, the two contacts in (**b**) and (**c**), represented by red cylinders, create almost identical torque measurements as (**a**).

However, even within the boundary set by these simplifying assumptions, joint-torque-based contact estimation is still limited by the loss of detectability of contacts from joint torque measurements, which we will discuss more formally in Section. IV. Intuitively, as shown in Fig. 1, it is likely that multiple contact positions and forces (Fig. 1b and 1c) create almost identical torque measurements, making them impossible to distinguish by looking at joint torque alone. Although this failure mode has been observed in existing work [7], a thorough analysis on how often joint-torque-based contact estimators fail and what can be done to mitigate such failures appears to be absent from the literature.

In this work, we show quantitatively that two distinct contacts generating almost identical joint torque measurements is far from a 0-probability event. Moreover, this probability can get alarmingly high for links whose geometry is "concentrated" around their joint axes, such as links of the IIWA arm. Therefore, we believe that solving for a single contact estimate from joint torque measurements is an inadequate problem formulation. Instead we propose to estimate the *set* of possible contact positions consistent with the measurements. Elements of this set can be cast as the global optimal solution of a nonlinear optimization problem, which is difficult to solve directly. Nevertheless, by combining rejection sampling and gradient descent on

This work is supported by Amazon and Lincoln Lab/Air Force Award PO# 7000470769.

manifolds, we propose an estimator that searches for local minima of the nonlinear optimization problem, and provide an efficient implementation capable of running at real-time rates. In practice, the proposed estimator usually finds all local minima on the links it searches. In addition, given the set of possible contact positions, we also propose an active contact discrimination strategy that falsifies spurious contact positions by slightly moving the robot.

II. Related work

Although raw measurements from joint torque sensors include gravitational and inertial effects, the torque generated by external contacts, also known as the residual torque, can be extracted from them using external torque observers [5]. Having become an integral part in many robot arms' firmware [3], [4], such observers can update residual torque estimates at hundreds of Hz, providing the foundation for all proprioceptive contact estimation methods.

The contact estimator by Haddadin *et al.* [5] first determines the link in contact as the last link with non-zero residual torque. It then solves for the line of action of the contact force from a system of linear equations relating torque measurements to the contact wrench. Finally, if the contact geometry of the link is convex, intersecting the line with the link will give two potential contact points, one for pulling and the other for pushing. As contact forces almost always push, the contact point can thus be uniquely determined. However, solving for the line of force action needs at least 6 torque measurements and a full-rank contact Jacobian, which implies this method does not produce any outcome on links more proximal to the base than link 6 or when the robot is close to singular. Moreover, when a link is not convex (e.g. link 5 in Fig. 1), the line of force action may intersect the link at more than two locations, making it impossible to uniquely determine the contact point.

Methods based on the Markov Chain Monte Carlo (MCMC) methodology [6], [8] can theoretically work on any link and with rank-deficient contact Jacobians, although estimation accuracy typically degrades on links too close to the base, or when the robot is close to singular. The degradation is not a limitation of the methods themselves, but a result of the loss of detectability of contacts from joint torque measurements. Both [6] and [8] use random walk on the robot's surface as the proposal distribution, and evaluate the likelihood of samples using the L2 norm of the difference between the measured joint torque and the joint torque created by the sample. The difference is that [6] assumes frictional contacts whereas [8] assumes that contacts are frictionless. The biggest drawback of MCMC methods is that they typically converge to only one local minima of the likelihood function, and are oblivious of other local minima when they exist.

Proprioceptive contact estimator based on machine learning has also been explored. Zwiener *et al.* discretize a robot's surface into finitely many patches, and train a classification network to predict the patch in contact from joint torque [7]. Although the contact classifier works well on both the training and validation data sets, its limitations and failure modes are difficult to analyze; its ability to generalize beyond the data set it is trained on is also hard to gauge.

III. Problem Formulation

For given joint angles $\boldsymbol{q} \in \mathbb{R}^{n_q}$ and residual joint torque $\boldsymbol{\tau}_{\text{ext}} \in \mathbb{R}^{n_q}$ created by one external contact at point C, the problem is to find $\boldsymbol{p}_C \in \mathbb{R}^3$, the coordinate of contact point C, and the contact force $\boldsymbol{f}_C$. In other words, we would like to solve for $\boldsymbol{p}_C$ and $\boldsymbol{f}_C$ from the following equation:

$$\boldsymbol{\tau}_{\text{ext}} = \boldsymbol{J}_C(\boldsymbol{q}, \boldsymbol{p}_C)^\intercal \boldsymbol{f}_C, \tag{1}$$

where $\boldsymbol{J}_C(\boldsymbol{q}, \boldsymbol{p}_C) \in \mathbb{R}^{3 \times n_q}$ the contact Jacobian that maps joint velocity $\dot{\boldsymbol{q}}$ to the velocity of C.

As shown in Fig. 2a, C is confined to the robot's surface, and the contact force $\boldsymbol{f}_C$ needs to stay inside the friction cone at C: $\left\|\boldsymbol{f}_{C_f}\right\| \leq \mu \|\boldsymbol{f}_{C_n}\|$. The second-order cone can be approximated with the polyhedral cone in Fig. 2b, which is generated by a set of n_d extreme rays $\boldsymbol{v}_C = \{\boldsymbol{v}_{C_1}, \ldots, \boldsymbol{v}_{C_{n_d}}\}$, such that $\boldsymbol{f}_C = \sum_{i=1}^{n_d} \boldsymbol{v}_{C_i} \beta_i = \boldsymbol{v}_C \boldsymbol{\beta}$ with $\beta_i \geq 0$ [9].

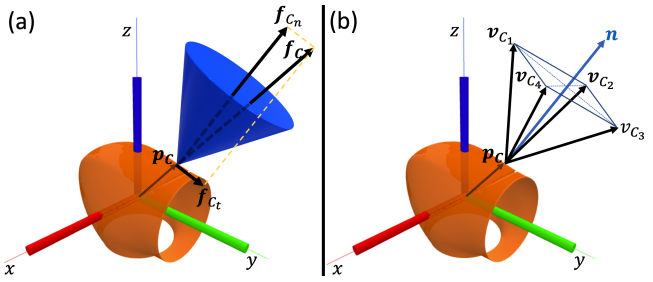


Fig. 2: A contact force on Link 6 of the KUKA IIWA robot. The orange mesh is the surface of the link. The triad represents the body frame of the link. (**a**) second-order friction cone. (**b**) Polyhedral approximation of the friction cone with $n_d = 4$.

Solving (1) with the friction cone constraint is equivalent to solving the following optimization problem:

$$\min_{\boldsymbol{\beta} \geq 0,\ \bar{\boldsymbol{p}}_C \in \mathcal{S}} \|\boldsymbol{J}(\boldsymbol{q}, \bar{\boldsymbol{p}}_C)^\intercal \boldsymbol{\beta} - \boldsymbol{\tau}_{\text{ext}}\|^2, \tag{2}$$

where $\boldsymbol{J}(\boldsymbol{q}, \bar{\boldsymbol{p}}_C) = \boldsymbol{v}_C^\intercal \boldsymbol{J}_C(\boldsymbol{q}, \bar{\boldsymbol{p}}_C) \in \mathbb{R}^{n_d \times n_q}$, and $\mathcal{S}$ is the manifold of the robot's surface. The overbar in $\bar{\boldsymbol{p}}_C$ emphasizes that the solution to (2) can be different from the actual $\boldsymbol{p}_C$. For fixed $\boldsymbol{q}$ and $\boldsymbol{\tau}_{\text{ext}}$, a contact position estimate $\bar{\boldsymbol{p}}_C$ has an associated cost defined as

$$l(\bar{\boldsymbol{p}}_C; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) \coloneqq \min_{\boldsymbol{\beta} \geq 0} \|\boldsymbol{J}(\boldsymbol{q}, \bar{\boldsymbol{p}}_C)^\intercal \boldsymbol{\beta} - \boldsymbol{\tau}_{\text{ext}}\|^2 \tag{3a}$$

$$= \min_{\boldsymbol{\beta} \geq 0} \left(\boldsymbol{\beta}^\intercal \underbrace{\boldsymbol{J}\boldsymbol{J}^\intercal}_{\boldsymbol{Q}} \boldsymbol{\beta} - 2 \left(\underbrace{\boldsymbol{J}\boldsymbol{\tau}_{\text{ext}}}_{-\boldsymbol{b}} \right)^\intercal \boldsymbol{\beta} + \boldsymbol{\tau}_{\text{ext}}^\intercal \boldsymbol{\tau}_{\text{ext}} \right). \tag{3b}$$

The function $l(\ \cdot\ ; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) : \mathbb{R}^3 \to \mathbb{R}$ is also called the *residual*, and can be computed by solving (3b), which is a convex QP. Using $l(\cdot)$, the set of solutions of (2) can be described by the set

$$P_0(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) \coloneqq \{\bar{\boldsymbol{p}}_C \in \mathcal{S} : l(\bar{\boldsymbol{p}}_C; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) = 0\}. \tag{4}$$

The true contact location $\boldsymbol{p}_C$ is clearly in $P_0(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$. However, it is possible that $|P_0| > 1$, i.e. different contact forces at different positions can generate the same joint

torque $\boldsymbol{\tau}_{\text{ext}}$, as shown in Fig. 3. In general, it is not possible to distinguish $\boldsymbol{p}_C$ from other elements of $P_0(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$ using only $\boldsymbol{q}$ and $\boldsymbol{\tau}_{\text{ext}}$.

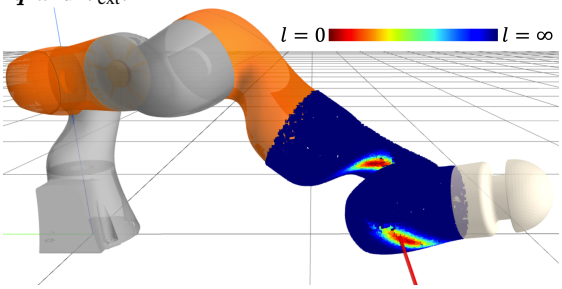


Fig. 3: The residual $l(\cdot)$ for 20000 sampled points on link 5 and 6 of the IIWA arm. The true contact position and direction is indicated by the red line. Multiple global minima of $l(\cdot)$ can be found on the robot's surface $\mathcal{S}$.

IV. Detectability of Contacts

The joint torque sensor of a link is only able to measure the torque about the axis of the link's revolute joint. It is therefore possible for the torque sensor to register zero or small torque measurement, even if the link is in contact with significant contact force. In this section, we give a quantitative analysis on how likely a contact force creates little or no measurable torque by studying the simplest case of a single link.

A. Detectability

We assume that a link's joint axis is aligned with the z-axis of the link's body frame. Let $\boldsymbol{p}_C \in \mathcal{S}$ denote a generic point on the robot's surface, $\mathcal{K}_C$ the friction cone at $\boldsymbol{p}_C$, and $\tau_z(\cdot) : \mathbb{R}^3 \to \mathbb{R}$ the function that returns the z-component of the torque generated by a force. For illustrative purposes, the friction coefficient μ is set to 1. We call the contact at $\boldsymbol{p}_C$ *fully detectable*[1] if $\forall \boldsymbol{f}_C \in \mathcal{K}_C$, $\|\boldsymbol{f}_C\| \neq 0 \Rightarrow \tau_z(\boldsymbol{f}_C) \neq 0$. Similarly, the contact is *partially detectable* if $\exists \boldsymbol{f}_C \in \mathcal{K}_C$, $\|\boldsymbol{f}_C\| \neq 0 \Rightarrow \tau_z(\boldsymbol{f}_C) \neq 0$, and *undetectable* if $\forall \boldsymbol{f}_C \in \mathcal{K}_C$, $\tau_z(\boldsymbol{f}_C) = 0$.

We first look at the condition under which a contact at $\boldsymbol{p}_C$ is fully detectable. A non-zero force at $\boldsymbol{f}_C$ satisfies $\tau_z(\boldsymbol{f}_C) = 0$ if and only if its line of action intersects with the z-axis. The set of such $\boldsymbol{f}_C$'s belong to the plane which passes through both $\boldsymbol{p}_C$ and the z-axis. We denote this plane by $\mathcal{P}_C$. The contact at $\boldsymbol{p}_C$ is fully detectable if and only if $\mathcal{K}_C \cap \mathcal{P}_C = \{\boldsymbol{0}\}$, which is equivalent to the following linear program being infeasible:

$$\text{Find } \boldsymbol{\beta}, \text{ subject to} \tag{5a}$$
$$(\boldsymbol{p}_C \times \boldsymbol{e}_z)^\intercal (\boldsymbol{v}_C \boldsymbol{\beta}) = 0, \tag{5b}$$
$$\boldsymbol{\beta} \geq 1, \tag{5c}$$

where $\boldsymbol{e}_z$ is the unit vector along the z-axis; $(\boldsymbol{p}_C \times \boldsymbol{e}_z)$ is normal to $\mathcal{P}_C$. (5b) constrains the contact force $\boldsymbol{f}_C = \boldsymbol{v}_C \boldsymbol{\beta}$ to $\mathcal{P}_C$. (5c) makes sure that $\boldsymbol{f}_C$ is non-zero.

To find out which part of a link is fully detectable, we can sample uniformly on the link's surface and solve (5) for all samples. The results for two links with distinct geometries are shown in Fig.4. As the IIWA link's surface is "concentrated" around its joint axis, only a tiny fraction of the link's surface is fully detectable. In contrast, the "elongated" link of UR5 has significantly more fully detectable surface, which is located further away from the joint axis. On both links, the majority of the samples are not fully detectable.

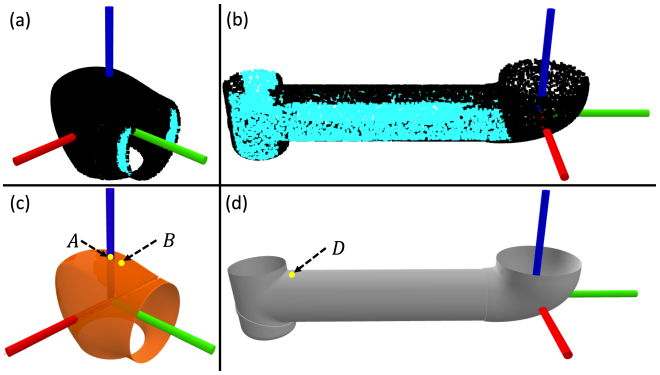


Fig. 4: Full detectability of (**a**) link 6 of the IIWA arm and (**b**) "forearm" link of the UR5 arm. Out of the 20000 samples generated on each link, 2.19% and 32.58% of the samples on (**a**) and (**b**) are fully detectable, respectively. Fully detectable samples are shown in cyan, others in black. The meshes of the links are shown in (**c**) and (**d**).

B. Quantifying "undetectability"

A contact at C is fully detectable if any contact force in $\mathcal{K}_C$ generates *any* non-zero torque measurement. However, as real torque sensor measurements are usually corrupted by noise, it is useful to know when a contact force generates a torque measurement no less than a threshold ϵ.

Moreover, for contacts that are not fully detectable, it is worth noting that the extent of their "undetectability" varies. For example, point A in Fig. 4c is undetectable, as it is the intersection of the z-axis and the link's surface. Although both point D (Fig. 4d) and B (Fig. 4c) are partially detectable, it generally takes a smaller force to generate $\tau_z > \epsilon$ at D than at B, as D is further away from its joint axis.

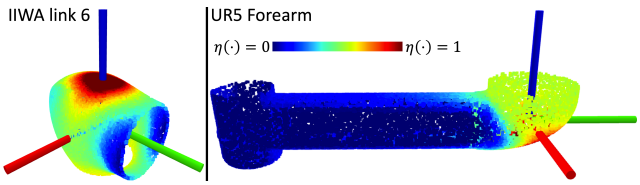
The degree of a contact's "undetectability" with respect to the threshold ϵ can be quantified by:

$$\eta_\epsilon(\boldsymbol{p}_C) \coloneqq \frac{\text{Area}(\mathcal{D}_c^\epsilon)}{\text{Area}(\mathcal{D}_c)} \in [0, 1], \tag{6a}$$
$$\mathcal{D}_C \coloneqq \{\boldsymbol{f}_C \in \mathbb{R}^2 : \|\boldsymbol{f}_C\| = 1, \boldsymbol{f}_C \in \mathcal{K}_C\}, \tag{6b}$$
$$\mathcal{D}_C^\epsilon \coloneqq \{\boldsymbol{f}_C \in \mathcal{D}_C : \tau_z(\boldsymbol{f}_C) \leq \epsilon\}, \tag{6c}$$

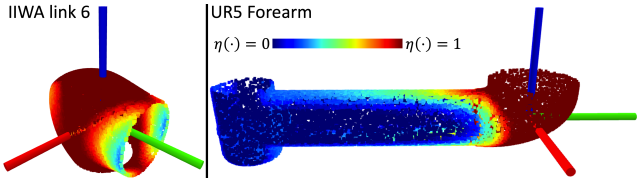
where $\mathcal{D}_C$ is a "dome" in the friction cone consisting of unit-norm forces, and $\mathcal{D}_C^\epsilon$ is the subset of $\mathcal{D}_C$ consisting of forces whose torque measurement $\tau_z(\boldsymbol{f}_C)$ is less than ϵ. Intuitively, $\eta_\epsilon(\boldsymbol{p}_C)$ is the proportion of contact forces in $\mathcal{K}_C$ that create small ($\leq \epsilon$) torque measurements. The closer $\eta(\boldsymbol{p}_C)$ is to 1, the less detectable $\boldsymbol{p}_C$ becomes. Using Fig. 4c and 4d again as examples, we expect $\eta_\epsilon(\boldsymbol{p}_A) = 1$, $\eta_\epsilon(\boldsymbol{p}_B)$ be close to 1, and $\eta_\epsilon(\boldsymbol{p}_D) < \eta_\epsilon(\boldsymbol{p}_B)$.

Computing $\eta_\epsilon(\boldsymbol{p}_C)$ can be done analytically, but a sampling based approach is much easier to implement. Uniform samples on $\mathcal{D}_C$ can be easily generated, and checking membership of $\mathcal{D}_C^\epsilon$ is trivial. We can then approximate $\eta_\epsilon(\boldsymbol{p}_C)$ by the ratio of samples in $\mathcal{D}_C^\epsilon$ to the total number of samples.

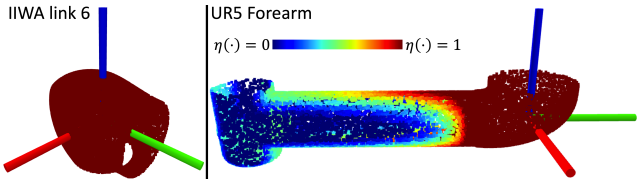
[1]The term "detectable", which is rigorously-defined in control theory, is abused in this section in order to facilitate exposition.



(a) ϵ=0.02. IIWA: $\eta_{\text{min}} = 0.011, \eta_{\text{max}} = 1$. UR5: $\eta_{\text{min}} = 0, \eta_{\text{max}} = 1$.



(b) ϵ=0.05. IIWA: $\eta_{\text{min}} = 0.249, \eta_{\text{max}} = 1$. UR5: $\eta_{\text{min}} = 0, \eta_{\text{max}} = 1$.



(c) ϵ=0.1. IIWA: $\eta_{\text{min}} = 1, \eta_{\text{max}} = 1$. UR5: $\eta_{\text{min}} = 0, \eta_{\text{max}} = 1$.

Fig. 5: $\eta_\epsilon(\cdot)$ of IIWA link 6 and UR5 forearm.

To evaluate the "undetectability" of an entire link, we can once again compute $\eta_\epsilon(\cdot)$ for samples generated uniformly from the link's surface. The results for two different links using different ϵ's are summarized in Fig. 5. As ϵ increases, the surface close to the z-axis quickly becomes fully undetectable. If we treat ϵ as a detection threshold, the result becomes particularly alarming for links with a more "concentrated" shape: even for $\epsilon = 0.02$ (Fig. 5a): $\eta_\epsilon \geq 0.4$ for 80% of the samples. In other words, if a force of 1N is applied at a random point C along a random direction in $\mathcal{K}_C$, the probability of not detecting the contact force is at least 0.32.

C. Implications

The analysis in this section reveals a fundamental limitation of existing joint-torque-based contact estimation methods: when a contact force generates small torque about the axis of a link, the observed $\boldsymbol{\tau}_{\text{ext}}$ could be equally well explained by a different contact point on a different link.

Haddadin's method [5] determines the link in contact as the most distal link with $\tau_z(\cdot) \geq \epsilon$. As there is a significant chance that $\tau_z(\boldsymbol{p}_C) \leq \epsilon$ at the true contact position C, it is likely that their strategy believes the contact to be on a wrong link. MCMC methods [6], [8] evaluate the likelihood of points being the true contact point C using the residual (3). If $\eta_\epsilon(\boldsymbol{p}_C)$ is large, it is likely for other points to have a residual that is only larger by $\epsilon^2 \left\|\boldsymbol{f}_C\right\|^2$, thereby trapping MCMC methods in a wrong local minimum of $l(\cdot)$.

V. Contact Estimation from External Torque Measurements

We have demonstrated that the best contact estimate achievable from $\boldsymbol{q}$ and $\boldsymbol{\tau}_{\text{ext}}$ is $P_0(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$ (Section III), which is likely to have more than one element (Section IV). Elements of $P_0(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$ are global optimizers of (2), which are also global minima of the residual $l(\cdot; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$ on $\mathcal{S}$. However, due to the dependence of $\boldsymbol{J}$ on $\bar{\boldsymbol{p}}_C$ and the manifold constraint $\bar{\boldsymbol{p}}_C \in \mathcal{S}$, (2) is nonlinear and difficult to solve directly.

In this section, we present a contact estimation strategy called RSGD, which is named after the combination of Rejection Sampling and Gradient Descent. RSGD is able to find every local minima of $l(\cdot; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$. Although not as ideal as finding $P_0(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$, RSGD is stronger than MCMC methods: it locates every point on $\mathcal{S}$ to which MCMC methods may converge.

A. Rejection sampling

Starting with $P \coloneqq \{\bar{\boldsymbol{p}}_C \in \mathcal{S}\}$, a set of uniform samples drawn from $\mathcal{S}$, we can calculate the residual $l(\cdot)$ for all samples in P, and keep the samples which satisfy $l \leq \epsilon$:

$$P_\epsilon(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) \coloneqq \{\bar{\boldsymbol{p}}_C \in \mathcal{S} : l(\bar{\boldsymbol{p}}_C; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) \leq \epsilon\}. \tag{7}$$

When using a dense P (Fig. 6a), $|P_\epsilon|$ is usually sufficiently large so that the local minima of $l(\cdot)$ can be estimated, for instance, by clustering the points in P_ϵ, finding the cluster centers, and then projecting the centers back onto the robot surface $\mathcal{S}$. In contrast, when P is sparse (Fig. 6b), the few samples in P_ϵ are typically too noisy to make a reasonable estimate.

The biggest drawback of rejection sampling is the high rejection rate, which, for example, can get to approximately 98% for the robot and contact configuration in Fig. 6. Calculating residuals for a very dense P is therefore needed for an accurate contact estimate, which will incur prohibitively high computational cost.

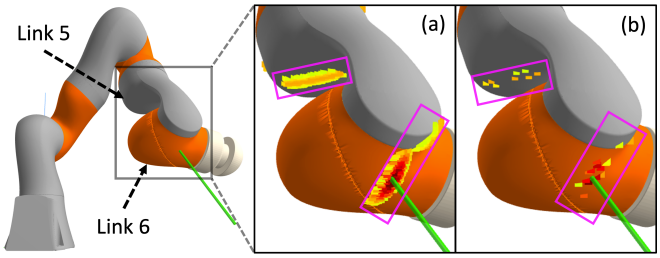


Fig. 6: Finding elements of P_ϵ on link 5 and link 6 with $\epsilon = 0.005$ using rejection sampling. A force of 10N is applied to the robot along the green line. Enclosed by magenta boxes, the colored squares represent accepted samples. (**a**) Dense P: 491 out of 20000 samples are accepted. (**b**) Sparse P: 24 out of 1000 samples are accepted.

B. Rejection sampling + gradient descent (RSGD)

The high computational cost of vanilla rejection sampling, due to the high rejection rate for small ϵ, can be reduced significantly by

1) Generating a set of potential contact positions by rejection sampling using a sparse sample set P and a large threshold δ: $P_\delta(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) \coloneqq \{\bar{\boldsymbol{p}}_C \in \mathcal{S} : l(\bar{\boldsymbol{p}}_C; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) \leq \delta\}$.
2) Running gradient descent (Algorithm 1) for every $\bar{\boldsymbol{p}}_C \in P_\delta(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$, collecting the converged points $\bar{\boldsymbol{p}}_C^*$ into a set of locally optimal contact position estimates: $P_\delta^\star(\boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}}) \coloneqq \{\bar{\boldsymbol{p}}_C^* \in \mathcal{S}\}$.

As shown in Fig. 7, a large δ in Step 1 increases the acceptance rate, ensuring that P_δ has enough samples even if the initial sample set P is sparse. Although samples in P_δ are spread out at the beginning, Step 2 runs them through Algorithm 1, making most of them converge to local minima of $l(\cdot)$.

Algorithm 1 Gradient descent on manifold $\mathcal{S}$

1: **Input**: $\boldsymbol{q}$, $\boldsymbol{\tau}_{\text{ext}}$, $\bar{\boldsymbol{p}}_C$; **Output**: $\bar{\boldsymbol{p}}_C^*$
2: **while** $\|\nabla_{\bar{\boldsymbol{p}}_C} l(\bar{\boldsymbol{p}}_C; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})\| > \varepsilon_G$ **do**
3: $\quad \boldsymbol{t} \leftarrow (\boldsymbol{I}_3 - \boldsymbol{n}_C \boldsymbol{n}_C^\intercal) \nabla_{\bar{\boldsymbol{p}}_C} l(\bar{\boldsymbol{p}}_C; \boldsymbol{q}, \boldsymbol{\tau}_{\text{ext}})$
4: $\quad a \leftarrow \text{LineSearch}(\bar{\boldsymbol{p}}_C, \boldsymbol{t})$
5: $\quad \bar{\boldsymbol{p}}_C \leftarrow \bar{\boldsymbol{p}}_C + a\boldsymbol{t}$
6: $\quad \bar{\boldsymbol{p}}_C \leftarrow \text{Retract}(\bar{\boldsymbol{p}}_C, \mathcal{S})$
7: **end while**
8: $\bar{\boldsymbol{p}}_C^* \leftarrow \bar{\boldsymbol{p}}_C$

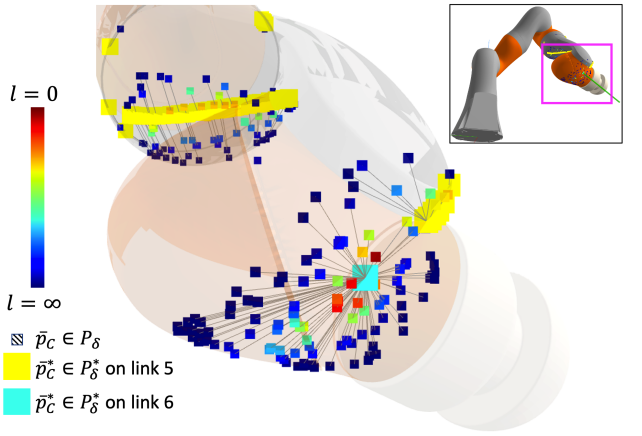


Fig. 7: Running RSGD on the same robot and contact configuration as Fig. 6. Step 1: 175 out of 1000 samples are accepted with $\delta = 0.1$, which are shown as small squares color-coded by their residual values l. Step 2: 155 of the 175 samples in P_δ converge after running Algorithm 1, which are shown as large squares. Every $\bar{\boldsymbol{p}}_C \in P_\delta$ is connected by a line to the corresponding element in P_δ^* to which it converges.

Algorithm 1 is the standard Gauss-Newton method for Riemannian optimization. In Line 3, the gradient $\nabla_{\bar{\boldsymbol{p}}_C} l$ is projected to the local tangent plane, which has normal $\boldsymbol{n}_C$ and passes through $\bar{\boldsymbol{p}}_C$. In Line 4, a standard line search method is used to ensure that l is decreasing after taking the gradient step $a\boldsymbol{t}$ [10]. In Line 6, the new point in the local tangent plane is projected back onto $\mathcal{S}$.

Using (3b), the gradient $\nabla_{\bar{\boldsymbol{p}}_C} l$ can be written as

$$(\nabla_{\bar{\boldsymbol{p}}_C} l)^\intercal = \frac{\partial l}{\partial \bar{\boldsymbol{p}}_C} = \frac{\partial l}{\partial \boldsymbol{Q}} \frac{\partial \boldsymbol{Q}}{\partial \bar{\boldsymbol{p}}_C} + \frac{\partial l}{\partial \boldsymbol{b}} \frac{\partial \boldsymbol{b}}{\partial \bar{\boldsymbol{p}}_C} \tag{8}$$

where $\frac{\partial \boldsymbol{Q}}{\partial \bar{\boldsymbol{p}}_C}$ and $\frac{\partial \boldsymbol{b}}{\partial \bar{\boldsymbol{p}}_C}$ can be obtained using automatic differentiation [11]; $\frac{\partial l}{\partial \boldsymbol{Q}}$ and $\frac{\partial l}{\partial \boldsymbol{b}}$ can be obtained from differentiating the implicit function defined by the optimality condition of QP (3b) [12].

As shown in Fig. 8a, Algorithm 1 is effective at reducing the residual $l(\cdot)$ of samples in P_δ. Note that about 60 samples converge to positions on $\mathcal{S}$ with $l = 0.003$, which is a local minimum on link 5, but not the global minimum on link 6. An example gradient descent run is shown in Fig. 8b.

The ability of an approach such as RSGD to find all local minima depends on two factors: the sampling strategy

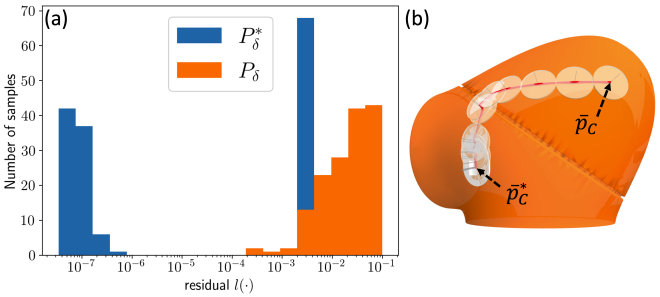


Fig. 8: (**a**) Distribution of residual $l(\cdot)$ of the P_δ and P_δ^* from Fig. 7. (**b**) A gradient descent run on link 6 of IIWA. Algorithm 1 starts at $\bar{\boldsymbol{p}}_C \in \mathcal{S}$ and converges to $\bar{\boldsymbol{p}}_C^*$. Red lines represent the path taken by gradient descent. White translucent disks represent local tangent planes.

and the convergence properties of gradient descent. By construction the entire surface of the robot is contained with the support of the distribution of the sampling procedure. As such, for every local minimum that has a non-measure-zero region of attraction, the probability that we draw a sample in that region and converge to the minimum in non-zero. Characterizing these regions of attraction, however, is more challenging, and one cannot rule out, e.g. saddle points and limit cycles. Nevertheless, empirically, we observe that the algorithm succeeds in finding all local minima for sufficiently dense sampling.

VI. Active Contact Discrimination

In the event that RSGD returns multiple (possible) contacts, some form of active exploration may be desirable to discriminate the true contact from the spurious. Assuming contact with a static object, that will remain (approximately) in place irrespective of the robot's motion, a simple strategy to falsify a spurious contact is to move the robot so as to break contact (pull away) at that location; if residual torque remains, then this cannot have been the true contact (assuming no additional contacts were introduced during the robot's motion). Similarly, the robot motion may preserve (push into) a possible contact; if the residual torque vanishes, then this (spurious) contact is falsified. Given N possible contacts $\{\boldsymbol{p}^i\}_{i=1}^N$, rather than test each (possible) contact individually, it is more efficient to "pull away from" $\lfloor N/2 \rfloor$ such contacts, and "push into" the other $\lceil N/2 \rceil$, thereby falsifying half of the contacts with each change of robot pose. The following program searches for such a change in pose, $\delta_{\boldsymbol{q}} \in \mathbb{R}^{n_q}$.

$$\min_{\delta_{\boldsymbol{q}},\ \boldsymbol{b} \in \mathbb{B}^N} \ \Big|\sum_{i=1}^N \boldsymbol{b}_i - \lfloor N/2 \rfloor\Big| \tag{9a}$$

$$\boldsymbol{n}_i^\intercal \boldsymbol{J}_i \delta_{\boldsymbol{q}} \le \epsilon_{\text{push}}^{\max} - (\epsilon_{\text{push}}^{\max} + \epsilon_{\text{push}}^{\min})\boldsymbol{b}_i, \ i = 1, \dots, N \tag{9b}$$

$$\boldsymbol{n}_i^\intercal \boldsymbol{J}_i \delta_{\boldsymbol{q}} \ge \epsilon_{\text{push}}^{\min} - (\epsilon_{\text{push}}^{\min} + \epsilon_{\text{pull}}^{\max})\boldsymbol{b}_i, \ i = 1, \dots, N \tag{9c}$$

$$|(I - \boldsymbol{n}_i \boldsymbol{n}_i^\intercal)\boldsymbol{J}_i \delta_{\boldsymbol{q}}| \le r_{\text{orth}}, \ i = 1, \dots, N \tag{9d}$$

$$|\delta_{\boldsymbol{q}}| \le \delta_{\boldsymbol{q}}^{\max} \mathbf{1}. \tag{9e}$$

Here, $\boldsymbol{b} \in \mathbb{B}^N$ represents the decision(s) to pull away ($\boldsymbol{b}_i = 1$) or push into ($\boldsymbol{b}_i = 0$) the ith contact. The objective (9a) attempts to push into as close to half ($\lfloor N/2 \rfloor$) of the contacts as possible. The constraints (9b) and (9c) require

that, e.g., a "pull away" moves the ith possible contact point (on the robot) at least $\epsilon_{\text{pull}}^{\text{min}}$, but at most $\epsilon_{\text{pull}}^{\text{max}}$, in the opposite direction to the outward facing surface normal $\boldsymbol{n}_i$, assuming a linearized relationship between the change in position and change in pose, $\delta \boldsymbol{p} \approx \boldsymbol{J}_i \delta_{\boldsymbol{q}}$. Constraint (9d) restricts the motion (of each contact point) close to the corresponding surface normal, to minimize the chance of introducing new contacts after the change in pose. Constraint (9e) restricts the change in each joint angle. An example of this method in action is shown in Fig. 9.

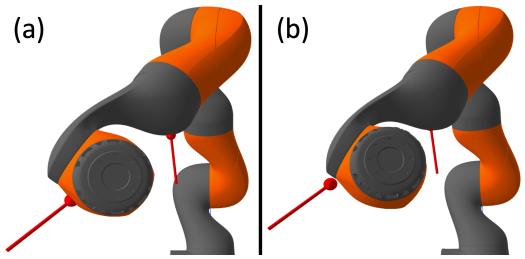


Fig. 9: (**a**) Two possible contact positions to disambiguate. The centers of the small red spheres are coincident with the candidate contact positions. (**b**) Solving (9) finds a motion that pulls away from the contact on link 6 and pushes into the contact on link 5.

There is no guarantee that this simple discrimination strategy will falsify all spurious contacts; success depends on problem specifics, e.g. robot pose, robot geometry, and the contact locations. In particular, (9) may return `infeasible`, or $\boldsymbol{b} \equiv \mathbf{1}$ ($\boldsymbol{b} \equiv \mathbf{0}$) (i.e. pull/push on all contacts, which gathers no useful information). However, problem (9) is a (convex) mixed-integer linear program (MILP) that can be efficiently solved to global optimality by commercial solvers. This means that, in the event of failure, we have a certificate that no such (sequence of) discriminating actions $\delta_{\boldsymbol{q}}$ exist, at least not without relaxing the constraints (9b)-(9e), or abandoning the linearized model and resorting to nonlinear motion planning.

VII. Implementation

RSGD requires a lot more computation than existing methods. Nevertheless, by leveraging efficient open-source libraries, our implementation can run at real-time rates on a single CPU thread.

Fig. 10 shows the run-time breakdown of a typical iteration of RSGD, collected on a Mac mini with Intel i7-8700B CPU and 64GB of RAM. In Step 1, the residual $l(\cdot)$ is computed for 1000 points drawn uniformly from S, of which 184 points satisfy $l < \delta$. In Step 2, Algorithm 1 is run on each of the 184 points until convergence, or until the limit on gradient steps is reached.

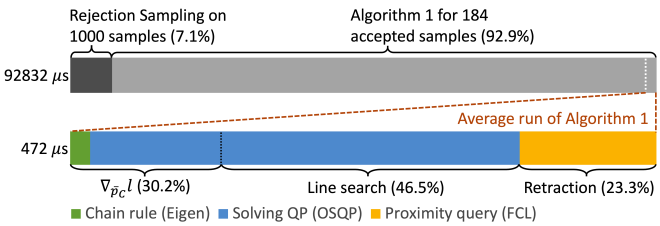


Fig. 10: Run-time breakdown of a typical iteration of RSGD.

As most of the time is spent on running Algorithm 1 for accepted samples in P_δ, how long one iteration of RSGD takes is almost linearly proportional to $|P_\delta|$. In this case, $|P_\delta| = 184$ leaves RSGD running at roughly 10Hz. This can be improved, for instance, by using a sparser P, putting an upper bound on $|P_\delta|$, using more CPU threads, or a combination of these strategies.

An average run of Algorithm 1 takes $472\mu s$. The most frequently-used atomic operation is computing the residual $l(\cdot)$, which involves solving QP (3b). In every gradient step, $l(\cdot)$ needs to be computed once to evaluate the gradient $\nabla_{\bar{\boldsymbol{p}}_C} l(\cdot)$, and a couple more times by line search. Moreover, $l(\cdot)$ is also computed for every sample in P in the earlier rejection sampling step. The lightweight QP solver OSQP [13] allows us to compute $l(\cdot)$ quickly: it takes $6\mu s$ on average to solve QP (3b).

Retracting points back onto the robot surface $\mathcal{S}$ is the second most time-consuming operation in Algorithm 1. With the robot surface $\mathcal{S}$ represented by triangle meshes, retraction can be done efficiently by a mature proximity query routine implemented in the Flexible Collision Library (FCL) [14].

The chain rule for computing $\nabla_{\bar{\boldsymbol{p}}_C} l(\cdot)$ in (8) is implemented with Eigen [15], which takes only 2% of the total time needed for Algorithm 1 to converge.

Algorithm 1 may fail to converge if gradient descent passes through a region of $\mathcal{S}$ with almost discontinuous surface normal, e.g. a groove or an engraved letter. Proximity queries also occasionally return a point off the mesh, throwing gradient descent off its track. Nonetheless, such failures are relatively rare and easy to detect and reject when they do occur.

Concerning the active contact discrimination, although complexity of the MILP is exponential in the number of contact locations to be falsified, moderate-size problems can be solved efficiently with SOTA solvers, such as GUROBI [16]; e.g., a problem with $N = 10$ contacts can be solved in approximately 5ms.

VIII. Conclusions

With a detailed analysis on two notions of contact detectability, we have demonstrated that a contact estimate from joint torque measurements typically consists of more than one possible contact positions, which are the global minima of the residual function $l(\cdot)$. Finding all global minima of $l(\cdot)$ is generally hard, but the proposed RSGD estimator empirically locates all local minima of $l(\cdot)$. Considering that joint torque measurements are inherently noisy, being able to find contact points with a small but positive residual could actually be beneficial. We have also provided a strategy to search for small robot motions which falsify as many spurious contact positions found by RSGD as possible. Moreover, when this strategy fails, it provides a certificate that no other small motion can do better.

On a robot that streams joint angle and residual torque ($\boldsymbol{\tau}_{\text{ext}}$) signals, such as the KUKA IIWA, deploying RSGD is expected to be straightforward. Nevertheless, pre-processing of the raw $\boldsymbol{\tau}_{\text{ext}}$ signal provided by the robot's driver, which filters out noise and ensures that the signal is unbiased, will probably be necessary.

REFERENCES

[1] G. Cannata, M. Maggiali, G. Metta, and G. Sandini, "An embedded artificial skin for humanoid robots," in *2008 IEEE International conference on multisensor fusion and integration for intelligent systems*. IEEE, 2008, pp. 434–438.

[2] A. Jain, M. D. Killpack, A. Edsinger, and C. C. Kemp, "Reaching in clutter with whole-arm tactile sensing," *The International Journal of Robotics Research*, vol. 32, no. 4, pp. 458–482, 2013.

[3] C. Loughlin, A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, and G. Hirzinger, "The dlr lightweight robot: design and control concepts for robots in human environments," *Industrial Robot: an international journal*, 2007.

[4] Franka Emika GmbH, "Franka control interface documentation," 2019. [Online]. Available: https://frankaemika.github.io/docs/index.html

[5] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1292–1312, 2017.

[6] L. Manuelli and R. Tedrake, "Localizing external contact using proprioceptive sensors: The contact particle filter," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 5062–5069.

[7] A. Zwiener, C. Geckeler, and A. Zell, "Contact point localization for articulated manipulators with proprioceptive sensors and machine learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 323–329.

[8] A. Zwiener, R. Hanten, C. Schulz, and A. Zell, "Armcl: Arm contact point localization via monte carlo localization." in *IROS*, 2019, pp. 7105–7111.

[9] D. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with coulomb friction," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1. IEEE, 2000, pp. 162–169.

[10] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[11] A. Griewank *et al.*, "On automatic differentiation," *Mathematical Programming: recent developments and applications*, vol. 6, no. 6, pp. 83–107, 1989.

[12] J. C. Boot, "On sensitivity analysis in convex quadratic programming problems," *Operations Research*, vol. 11, no. 5, pp. 771–786, 1963.

[13] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "Osqp: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, pp. 1–36, 2020.

[14] J. Pan, S. Chitta, and D. Manocha, "Fcl: A general purpose library for collision and proximity queries," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 3859–3866.

[15] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," http://eigen.tuxfamily.org, 2010.

[16] Gurobi Optimization, Inc., "Gurobi optimizer reference manual," 2017. [Online]. Available: http://www.gurobi.com/