École polytechnique fédérale de Lausanne

Master Thesis

Proprioception for collision detection

Author:
Albert Climent Bigas

Professor: Aude Billard
Supervisor: Nicolas Sommer

A thesis submitted in fulfillment of the requirements for the degree of Master on Automatic Control and Robotics of the Universitat Politècnica de Catalunya

in the

Learning Algorithms and Systems Laboratory
Microengineering Institute
School of Engineering

August 2015

Dedicated to my family, who encourage me every day

Abstract

Nowadays a lot of research is done in the field of human-robot interaction to allow robots to work in human environments. Haptic perception is crucial for human and robot safety since it allows to detect contacts. It is also important for surface exploration be able to detect contact between a robot and its environment. This thesis presents a probabilistic representation of contacts in order to complement or in some cases replace computer vision systems. Torque measurements were used to determine probable collisions (i.e. positions and forces) on robot links. The suggested algorithm were tested in simulation and afterwards experiments were done with the robot arm Kuka LWR.

Contents

Chapter 1

Introduction

1.1 Motivation

Can robots be placed in human environments? They undoubtly can in most science fiction movies. Nonetheless, robots' capabilities are still limited and a lot of research on the human-robot interaction field needs to be done [1]. This is a wide field which includes the study of methods for motion planning and the perception of humans.

This project focuses on robot perception, a field which mostly uses computer vision-based algorithms. When dealing with collisions, this kind of algorithms are less useful and other means of sensing are required. Proprioceptive sensors (i.e. torque and force) can provide information in order to detect collisions (position and force). Knowing where collisions take place provides the robot information about the environment. This kind of information is essential in order to introduce robots to human environments.

1.2 Environment information for Human-Robot interaction

Factories frequently use robots in structured environments so they can execute tasks efficiently. The rooms where robots work are equipped with many sensors that allow to precisely monitor the state of every element around the robot. These sensors provide information about changes on their surroundings. Despite the fact that they collect environment information, most of the time they are not prepared for substantial changes. For instance, a robot that works within a known area, moving objects from one place to another, or welding pieces of metal together is programmed to do those tasks but it may not expect a change on the objects that it handles or maybe a variation in the light

conditions. For that reason, a robot's working area is restricted and physically isolated from many agents, including humans. But there are other arguments against the cooperation between humans and robots on the same place besides the limitations on the robot to substantial changes. Most of them are basically for human safety.

Human environment is different from the factory's controlled environment. It is constantly under modification and transformation. However, we are able to adapt thanks to a lot of sensors (e.g. eyes, ears, skin) that continuously give information to us. Since we are young we learn about our surroundings and cause-effect relationship, which allows us to foresee changes. Moreover, we are able to know which is the best or easiest way to solve a new problem.

Consequently, in order to introduce robots in human environments they need more tools to perceive the world like we do. Research has been focused on computer vision systems in order to equip robots with visual perception [2]. Although this is indeed essential, the study of other senses is also important. We focus on haptic perception because it can provide information about contact events which are also important when interacting with dynamic environments.

Two different kinds of haptic sensors exist: external and internal. External sensors are placed on the surface of the robot, emulating the human skin. Contacts could be directly determined by this artificial skin [3]. However, the technology is not developed enough to cover the whole surface of a robot. Internal sensors are able to measure torques or strain between parts of the body. Joint torque information can also be used to detect contacts on the robot's surface [4].

Chapter 2

Contact estimation

2.1 Problem statement

Robots should be able to work and interact with humans without any difficulties or risks. This would generate the possibility for having human-robots interaction and cooperation. For instance, they could help elderly people at home or take care of children. Moreover, as they have abilities we don't and vice versa, cooperation would allow to carry out some tasks humans and robots could not do separately. Consequently, we have to provide them with methods to perceive their surroundings and define safety standards.

However, we are still far from that reality and a lot of research needs to be done. The International Organization for Standardization (ISO) has standards for robot safety, however, they are limited for industry and personal care robots [5], [6]. For this reason, it is necessary to develop new technologies in order to enforce safety standards. One of these technologies is based on the detection of humans and environments. Despite the fact that computer vision algorithms are the most commonly used, other sensors could complement or even replace them in some cases. Just like humans, robots could use tactile sensors and contact detection algorithms to explore surfaces and detect collisions on the robot's surface.

The robot used in the experiments has no artificial skin that could provide direct information about contacts. However, it is equipped with a single-axis torque sensor at every joint, which could contribute some information towards the representation of the contacts depending on the configuration of the robot and the sensed torques. The issue of using a single-axis torque sensor is that we want to determine a contact location and force without complete information. For instance, if the applied force is parallel to some the measurement axis we won't be able to measure any torque from that contact on those sensors. Moreover, several contacts may have the same torque

response because the problem is not bijective. For instance, for a single sensor we could have the same measurement for a contact far away from the sensor with a low force and a contact close to the sensor and a high force. In order to overcome this problem we suggest a probabilistic representation of the contacts that allows multiple solutions given a the torque measurement of several sensors. We are going to sample a set of possible contacts over the surface of the robot and then we will compute for each one the probability to be the actual contact. We are going to see that for the accuracy depends on the contact location and also the robot's configuration.

2.2 Related work

As mentioned before, vision systems have been studied in depth [2], [7], [8] even though the first studies on tactile perception algorithms date back to the 80s. A first contact location estimation method was presented by Salisbury [4], who used a six-axis force-torque sensor placed at the wrist of the arm and the knowledge of the geometry of the sensing tool to estimate the contact position between two rigid bodies. Takeshi et al. [9] found a more general solution that can be applied to any kind of sensing tool. Later, other authors [10] estimated external forces at the end-effector of a robot manipulator using joint torque measurements.

When dealing with human-robot interaction, complete information is required because collisions anywhere on the surface have to be detected. As a consequence, the methods mentioned previously are not satisfactory, as they only consider collisions at the end-effector. Research in several fields suggested different solutions in order to overcome this problem.

Some authors [11], [12] were focused on the design of a haptic non-convex-shaped end-effector that offers full-body coverage, with which they can directly compute the position and force of a single contact point without using sensor arrays. However, this method is not suitable for our purpose because the end-effector tool shape has to be specific for each task and we are focusing the study of robot arms, which have to interact actively with the environment, performing different tasks.

In machine learning, Lu et al. [13] suggested a neural network approach and a model based method. Using a base and wrist force-torque sensor they were able to detect collision forces and disturbance torques on the manipulator joints. Another way to predict external torques and detect collisions was developed recently on [14] where fuzzy identification and time series modeling were used. Contrary to other methods previously mentioned, this last one enables simultaneous detection of multiple contacts.

Another multiple-contact detection method is introduced in [15] merging visual information obtained from a depth camera and a model-based technique.

Probabilistic estimation approaches have also been studied. A first work [16] presents a way to estimate contacts by means of a probabilistic approach using force control. This research was

improved later [17] with a Bayesian approach called Scaling Series in order to perform global object localization.

Finally, the work done in [18] uses the same information we used (i.e. only torque measurements) to estimate contacts. The approach suggested is based on nonlinear constrained optimization. The drawback of this method is the complexity of the optimization. For this reason, they considered some assumptions: cylindric links and no friction on the robot's surface. With those assumptions they were able to run the algorithm in real time.

2.3 Our solution

Our proposed solution is based on a probabilistic approach. We used the measurement of external torques as inputs for our method, which represents a distribution of the most probable collisions. The algorithm samples a set of N particles and determines the probability of being the actual contact for each one. Particles are defined by a collision and the link where it takes place. The collision is defined in the frame of a joint by a contact position $\vec{r} \in \mathbb{R}^3$ and a contact force $\vec{F} \in \mathbb{R}^3$. The external torque is defined by $\vec{\tau} \in \mathbb{R}^n$, n being the number of torque sensors. The probability of having an external torque for a collision (F, r) is defined as:

$$P(\vec{\tau}, \vec{F}, \vec{r}) \tag{2.1}$$

This probability can be decomposed as:

$$P(\vec{\tau}, \vec{F}, \vec{r}) = P(\vec{F}, \vec{r}|\vec{\tau})P(\vec{\tau}) \tag{2.2}$$

Using the Bayes rule one can obtain[1]:

$$P(F, r|\tau) = \frac{P(\tau|F, r)P(F, r)}{P(\tau)} \tag{2.3}$$

As τ_i are conditional independent we can rewrite 2.3 as:

$$P(F, r|\tau) = \frac{\left(\prod_{i=1}^{n} P(\tau_i|F, r)\right) P(F, r)}{P(\tau)} \tag{2.4}$$

[1]From now on it is assumed that the contact force, the contact position and the torque are vectors.

At this point, the prior $P(F, r)$ needs to be discussed because it could have a significant effect on the performance of the algorithm. Clearly, the collision must happen on the robot's surface $S \subset \mathbb{R}^3$. This implies that the prior has to be null for all $r \notin S$. We think that execution time is an important factor to take into account when developing algorithms for robots. In our case, it depends on several factors, one of which is the number of particles. If we increase it, then the probability to find a particle close to the actual contact increase as well as the execution time. In order to reduce the number of particles used we add two more constrains on the contact. The first one is to consider no friction on the contact point so all forces are normal to it. The second one was to consider only pushing forces. The forces and positions where the prior is not null are less now. Function f is defined to rewrite this prior. The output of this function is the normal vector of the surface S at the position $\vec{r}$ pointing inside the surface:

$$\begin{aligned} f\colon\ \mathbb{R}^3 &\to \ \ \mathbb{R}^3 \\ \vec{r}\ \ &\to \text{-}\hat{n}(\vec{r}) \end{aligned}$$

So the prior can be written as:

$$P(F, r) = \begin{cases} 1, & \text{if } r \in S \text{ and } \frac{F}{||F||} = f(r) \\ 0, & \text{otherwise} \end{cases} \tag{2.5}$$

We need to discuss briefly the shape of the likelihood $P(\tau_i|F, r)$. One can consider that measurements of external torques have white noise so they would follow a normal distribution $\mathcal{N}(\mu, \sigma^2)$. Consequently it is not a strong assumption to consider the likelihood as a normal distribution too:

$$P(\tau_i|F, r) = \frac{1}{\sigma_i^2\sqrt{2\pi}} e^{\frac{-(\tau_i-\mu_i)^2}{2\sigma_i^2}} \tag{2.6}$$

The value of σ_i^2 may depend somehow on the amplitude of the sensor's noise and μ_i is equal to the torque produced by the collision (F, r) at joint i.

2.4 Practical procedure

Considering the robot arm as open kinematic chains of rigid bodies with n rotational joints and $q \in \mathbb{R}^n$, the robot dynamic model is:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = \vec{\tau} \tag{2.7}$$

Where $M(q) > 0$ is the symmetric inertia matrix, $C(q, \dot{q})$ is the Coriolis and centrifugal vector, $g(q)$ is the gravity vector and τ_t corresponds to the generalized torques vector which can be decomposed into friction torque (τ_f), motor torque (τ_m) and external torque (τ_e):

$$\tau_t = \tau_f + \tau_m + \tau_e \tag{2.8}$$

It is assumed that there is no friction between joints. In addition, in simulations and in the experiments with the real robot neither gravity was considered nor motor torques were applied. Consequently the dynamic model is reduced to:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} = \tau_e \tag{2.9}$$

In the case of the simulations done with Matlab, this external torque was computed manually knowing the collision force and location and it was introduced as a parameter to the algorithm.

External torques were computed using 2.9 for simulations with Gazebo. A world plug-in reads the robot's joint positions, velocities and accelerations[2]. With those information and the robot chain data stored in a URDF file, the KDL [19] library computed the inertia matrix and the Coriolis and centrifugal vector required for solving the equation. Finally, we used ROS to communicate the plug-in and the program with the algorithm. The use of ROS nodes for this project allowed us to change

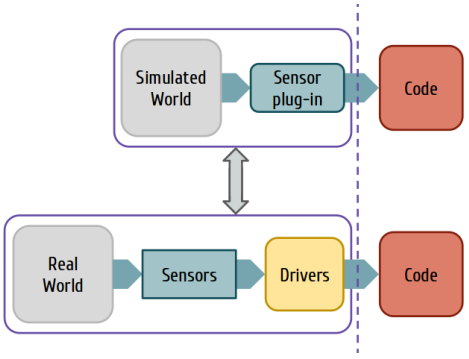


FIGURE 2.1: Coding structure

from the simulated world in Gazebo to the real robot easily. It also allowed us to display graphical results using the ROS tool RViz.

In order to carry out the experiments with Kuka LWR IV, a fixed position was set for the robot. Estimated external torques were read directly from the robot using the Fast Research Interface (FRI) library. Joint positions were read using the same procedure. Although the same code could run in simulation and with the real robot, we modified the code so that the robot experiments would have a better performance. First, we reduced the execution time by dividing the algorithm in three phases

[2]Those measurements were filtered to have less noisy values.

A, B and C with different priors. Consequently, possible contacts on each phase are different. We call $\mathbb{A}$, $\mathbb{B}$ and $\mathbb{C}$ the sets that group those contacts on each phase. Contacts $c_1, ..., c_m \in \mathbb{C}$ are obtained directly from the original mesh. Contacts $b_1, ..., b_n \in \mathbb{B} \subset \mathbb{C}$ is obtained using the K-means procedure described in algorithm 1. The same algorithm is used to get contacts $a_1, ..., a_k \in \mathbb{A} \subset \mathbb{B}$, but in this case the input set is $\mathbb{B}$ and the output set is $\mathbb{A}$. Notice that in line 16 we select as a centroid of cluster i the element of $\mathbb{C}$ closer to the mean of the cluster. This force centroids to lie on the surface of the robot as it would make little sense to have the centroids off the surface. The number of possible positions for phase A was 200, for phase B was 1000 and for phase C was 4605. Figure 2.2 shows contact positions for each phase.

Algorithm 1 K-means procedure

1: **procedure** CLUSTERING($\mathbb{C} = \{c_1, c_2, ..., c_m\}$) ▷ It clusters contacts in $\mathbb{C}$
2: n := number of clusters
3: $\mathbb{B} = \{b_1, b_2, ..., b_n\}$ (set of cluster centroids)
4: L = $\{l(c)|c = 1, 2, ..., m\}$ (set of cluster labels of $\mathbb{C}$)
5: **for each** $b_i \in \mathbb{B}$ **do** ▷ Select n random centroids
6: $b_i \leftarrow c_j \in \mathbb{C}$ (random selection)
7: **end for**
8: **for each** $c_i \in \mathbb{C}$ **do** ▷ Assign a cluster for each element in $\mathbb{C}$
9: $l(c_i) \leftarrow \underset{j \in \{1,...,n\}}{\text{argmin}} ||c_i - b_j||^2$
10: **end for**
11: *change* $\leftarrow$ *true*
12: **while** *change* **do**
13: *change* $\leftarrow$ *false*
14: **for each** $b_i \in \mathbb{B}$ **do** ▷ Update centroids
15: $x := \{c_j | l(c_j) = i\}$
16: $b_i \leftarrow \underset{\{c_j | l(c_j) = i\}}{\text{argmin}} ||\bar{x} - c_j||^2$
17: **end for**
18: **for each** $c_i \in \mathbb{C}$ **do**
19: newLabel := $\underset{j \in \{1,...,n\}}{\text{argmin}} ||c_i - b_j||^2$
20: **if** $l(c_i) \neq$ newLabel **then** ▷ Update cluster labels if required
21: $l(c_i) \leftarrow$ newLabel
22: *change* $\leftarrow$ *true*
23: **end if**
24: **end for**
25: **end while**
26: **return** $\mathbb{B}$, L
27: **end procedure**

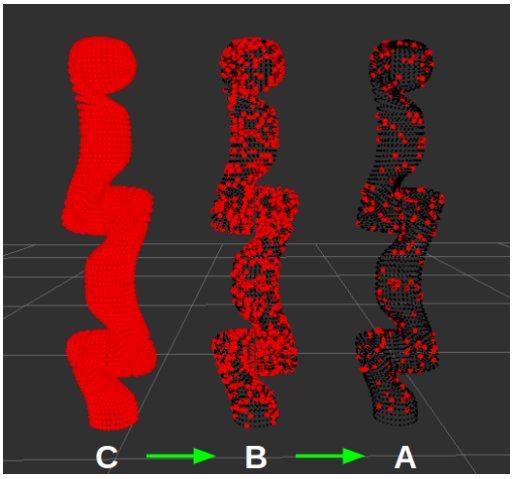


FIGURE 2.2: Possible contact positions for sets $\mathbb{C}$ (left), $\mathbb{B}$ (middle) and $\mathbb{A}$ (right) appear in red and in black all the points of the surface.

A second modification was done regarding the contact distribution. Random sampling was used in Matlab and Gazebo to select contacts following the prior distribution. This is, for each position we sampled several particles with a random force intensity within a range $[F_{min}, F_{max}]$. This sampling method could provide a smooth probability representation for a large number of samples. The problem was that large number of particles means large computational time. In order to had a smooth probability representation and low computational time we suggested to sample particles in a deterministic way. This sampling method was used for the experiments with the real robot. The method consists on sampling for each position m particles with different contact force (F_{i_k}) within a range $[F_{min}, F_{max}]$.

$$F_{i_k} = \left[F_{min} + \frac{k \cdot (F_{max} - F_{min})}{m} \right] \cdot \hat{n}_i, \;\; k = 1, ..., m \tag{2.10}$$

Where $\hat{n}_i$ is the normal to the surface at the position i.

In short, the algorithm compute the probability defined by equation 2.6 for all particles on phase A. Consider a_i the collision with higher probability and a_j a collision with a probability $P(a_j) \geqslant \alpha \cdot P(a_i)$, with $\alpha \in (0, 1)$. When phase B is executed, only contacts that belong the centroids a_i and a_j will be explored. The same process is done to perform phase C. Figure 2.3 shows the result of this process. Positions explored at each phase appear with a green-red scale, where green means high probability and red low probability. Small black balls show all the positions of the surface of the robot. At the end, the algorithm can be faster because it has to compute the probability of less particles and it may has a better performance because in phase B and C only particles close to the solution are explored.

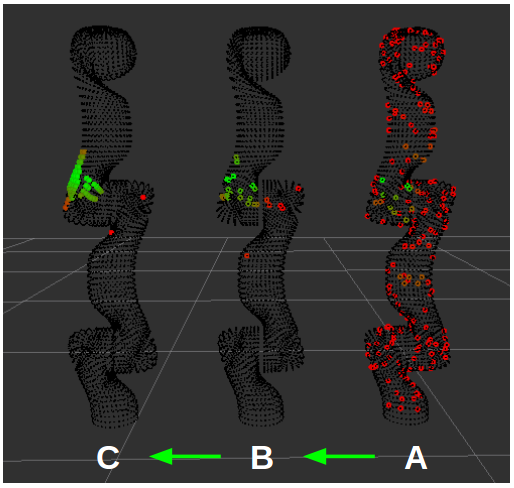


FIGURE 2.3: High probable contacts in green, low probable contacts in red and robot surface in black for phases A, B and C.

2.5 Robot Kuka LWR

The robot used for the experiments was Kuka LWR 4+. This robot arm has 7 degrees of freedom (DoF) and it has one axis torque sensor integrated per joint which will make it suitable for this project. Its outer structure is made of aluminum and its weight is about 14 kg so it is easy to move and its shape is rounded, which makes it is perfect to be introduced in human environments.

FIGURE 2.4: Kuka LWR 4+

Chapter 3

Experiments

In chapter 2 we defined the problem we want to solve and a solution based on a probabilistic representation of contacts on the robot's surface using external torques. This chapter shows results obtained with the suggested algorithm. It is divided into two main sections. In the first one, the method is implemented in Matlab and Gazebo to test it on simulations. In the second one, experiments are done with the real robot.

3.1 In simulation

We present results obtained with Matlab and Gazebo in this section. In each subsection we first explain how the experiments are carried out and the way results are displayed. Afterwards, we show and discuss them. We analyzed the performance of the algorithm for several collisions in each experiment.

3.1.1 Matlab: Simple robot

A first version of the algorithm was tested with a simple robot shown in figure 3.1. It has 6 rotational joints with a single-axis torque sensor in the direction of the red arrows. Each link has a length of 1 meter and they are unidimensional. The range of forces we consider in this first simulation was from 0 to 1.5 newtons. The algorithm computes the probability for 100000 particles. The way it samples each particle is as follows:

1. Sample randomly the collision link.

2. Sample randomly the collision position within the range $[0, 1]$ along the link's direction.

3. Sample randomly the collision force norm within the range $[0, 1.5]$.

4. Sample randomly the collision force direction within the plane perpendicular to the link.

Once a particle is sampled, the probability to be the actual collision is computed using equation 2.6 with $\sigma_i = 0.1, \forall i = 1...6$.

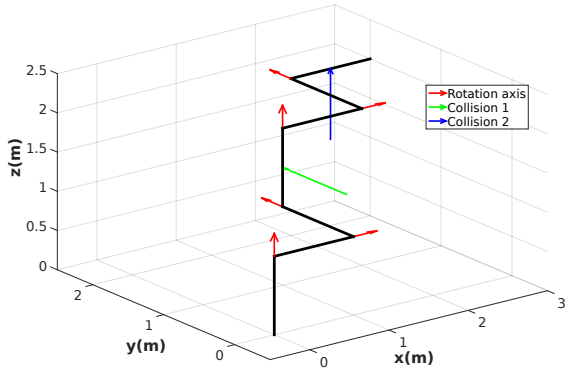


FIGURE 3.1: Simple robot structure in black, collision applied in green and blue arrows and rotation axis in red arrows.

Figure 3.1 shows in blue and green arrows the two collisions applied to this robot. Table 3.1 has the numerical values of the force ($\vec{F}$) and position ($\vec{r}$) for each collision by components (x, y and z), as well as the link. The measured external torques of each collision are presented in table 3.2.

Collision	F_x	F_y	F_z	r_x	r_y	r_z	Link
1	0.0	1.0	0.0	1.0	1.0	1.5	3
2	0.0	0.0	1.0	2.5	2.0	2.0	5

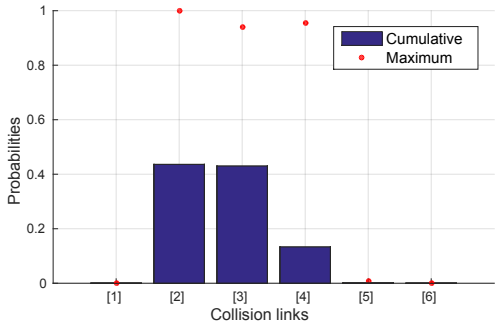
TABLE 3.1: Contact forces (Newtons) and locations (meters) for each experiment

Collision	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6
1	1.0	-0.5	0.0	0.0	0.0	0.0
2	0.0	2.0	-1.5	0.0	1.0	-0.5

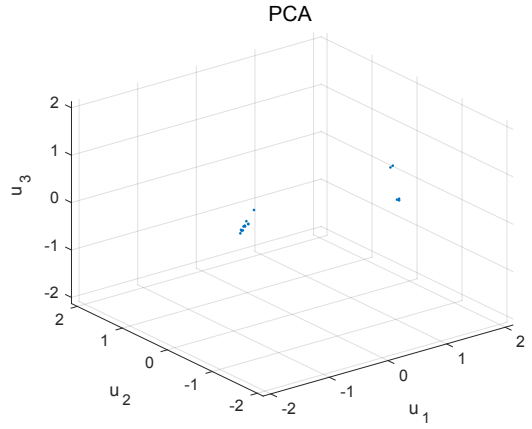
TABLE 3.2: Measured external torques for each experiment expressed in $N \cdot m$.

Results are displayed on figure 3.2 and 3.3. Red dots in subplots 3.2(a) and 3.2(c) represent the maximum probability of a particle on each link and blue bars represent sum of the probabilities for all particles divided per link and then normalized. Subplots 3.2(b) and 3.2(d) show the Principal

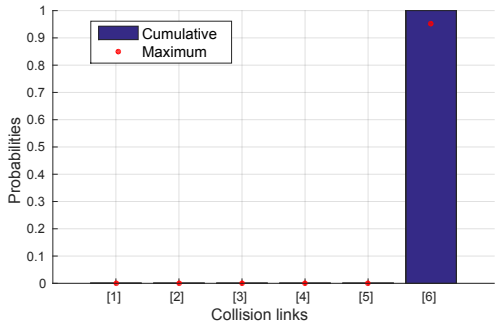
Component Analysis (PCA) applied over particles with a probability $P_i \geqslant 0.9 \max_j (P_j)$, where P_i and P_j are particle probabilities. This allows us to plot these particles on 3D projections.



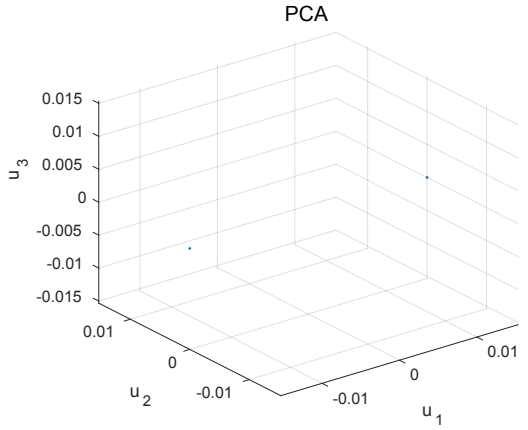
(A) Cumulative probability of collision and maximum probability for each link.



(B) 3D projection of the principal components of the most probable particles.



(C) Cumulative probability of collision and maximum probability for each link.



(D) 3D projection of the principal components of the most probable particles.

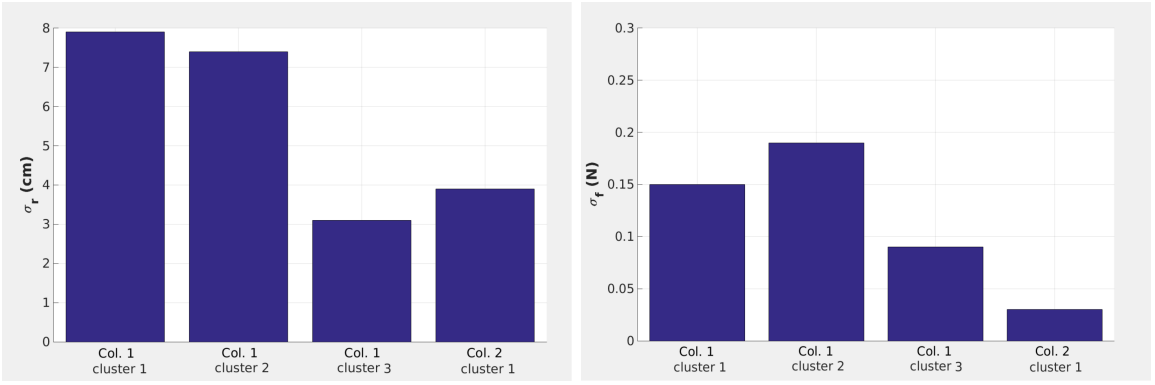
FIGURE 3.2: Results obtained with the simple robot using 100000 particles.

We clustered the most probable particles according to their link. Probable particles for collision 1 are labeled in Figure 3.3 as Col. 1 cluster 1, Col. 1 cluster 2, Col. 1 cluster 3 for links 2, 3 and 4 respectively. Probable particles for collision 2 are labeled as Col. 2 cluster 1. We define dispersion on position (σ_r) on equation 3.1 and dispersion on force (σ_F) on equation 3.2. As large this values are, as spread the result is so a less concrete solution is found. Figure 3.3 shows these dispersion on position (σ_r) and on force (σ_F) of the particles of each cluster.

$$\sigma_r = \sqrt{\frac{\sum_i P_i \cdot ||\vec{r}_i - \vec{\mu}_r||^2}{\sum_i P_i}} \tag{3.1}$$

$$\sigma_F = \sqrt{\frac{\sum_i P_i \cdot \left|\left|\vec{F}_i - \vec{\mu}_F\right|\right|^2}{\sum_i P_i}} \tag{3.2}$$

Where $\vec{\mu}_r$ and $\vec{\mu}_F$ are the mean on position and force of each cluster respectively. P_i is the probability, $\vec{r}_i$ is the contact location and $\vec{F}_i$ is then contact force of the particle i.



(A) Dispersion on position σ_r for each collision and each cluster.

(B) Dispersion on force σ_F for each collision and each cluster.

FIGURE 3.3: Dispersion on position and force for all clusters and collisions applied on the simple robot using 100000 particles.

3.1.2 Matlab: Kuka LWR

In order to use the algorithm with the Kuka LWR robot, a description of the surface is required. There is no mathematical expression to describe it. However, a first approximation could be obtained using a mesh description. We have a description of the robot with 4605 points on the surface of the robot and their respective normal vectors[1]. For the simulation described in this section, the robot in the configuration with all angles equal to zero is shown in Figure 3.4.

Using the mesh description, the number of possible positions is finite and consequently less particles are used. In this simulation just 20000 particles were used. We consider this number large enough because since we have 4605 points on the surface, it allows to have about 5 different forces per position on the surface of the robot. This fact is important so that the speed of the algorithm can be improved. Moreover, particles are well spread on the robot's surface so no area is omitted.

[1]Normal vectors are pointing inside the body of the robot because only pushing forces were considered.

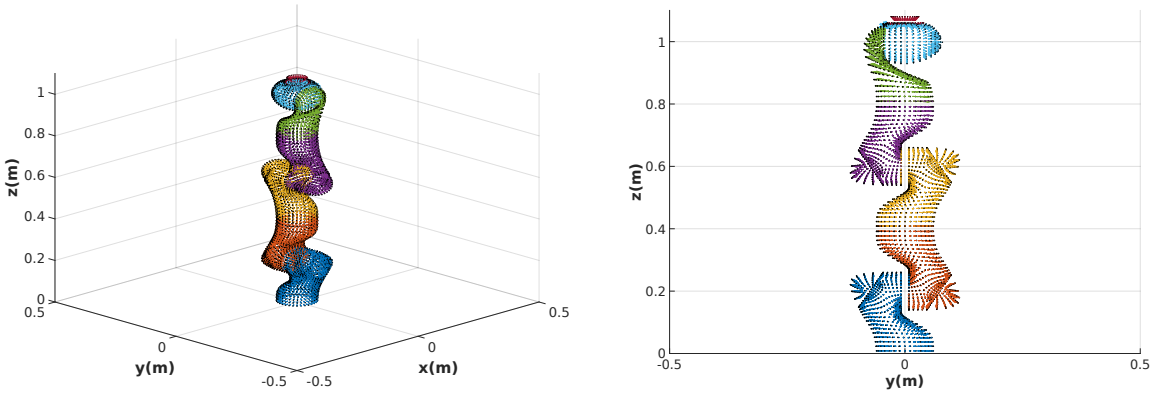


FIGURE 3.4: Surface mesh points and normal vectors for Kuka LWR 4+. Each link has a different color.

In addition, they are distributed homogeneously, so by giving each particle the same probability, we represent a uniform prior distribution. The experiments were done with the robot with the configuration mentioned before with $\sigma_i = 0.03, \forall i = 1...6$. Notice that this value is lower than the one used in the previous subsection. The reason why we choose a lower value is because the robot is smaller with a singular shape and we need more precision. The module is included in the [0.5, 3.0] newtons range. The minimum force is greater than zero because a collision with force zero has no sense. The maximum force norm is 3N in order to see the performance of the algorithm for a larger range of forces. Three different collisions were applied to the robot to compare the performance of the algorithm using the robot of the previous section and the one used here. The values of the force and position are shown by components (x, y and z) in table 3.3 as well as the link where the collision takes place. The measured external torque for each joint and collision are presented in table 3.4.

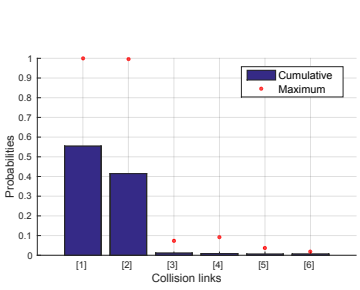
Collision	F_x(N)	F_y(N)	F_z(N)	r_x(cm)	r_y(cm)	r_z(cm)	Link
1	-2.11	-0.09	-0.58	6.0	4.8	21.3	1
2	1.50	0.00	0.00	-6.0	0.0	39.2	2
3	-1.37	-0.85	0.33	4.6	9.8	58.2	4

TABLE 3.3: Collisions for each experiment

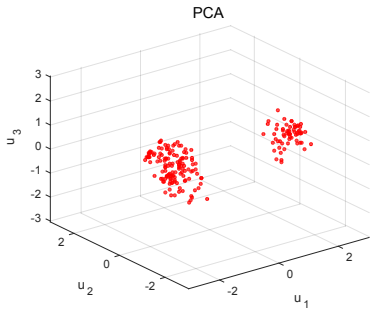
Collision	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6
1	9.6	0.0	0.0	0.0	0.0	0.0
2	0.0	28.7	0.0	0.0	0.0	0.0
3	9.5	-53.6	9.5	1.0	0.0	0.0

TABLE 3.4: Measured external torques for each experiment ($N \cdot cm$)

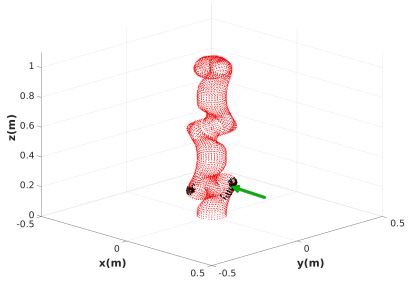
Results for collisions 1, 2 and 3 are displayed on figures 3.5, 3.6 and 3.7 respectively. Subplots 3.5(a), 3.6(a) and 3.7(a) show in red the probability of the most probable particle on each link and blue bars represent sum of the probabilities for all particles divided per link and then normalized. Subplots 3.5(b), 3.6(b) and 3.7(b) display particles with $P_i > 0.9 \max_j (P_j)$ projected on their three principal components. Finally, subplots 3.5(c), 3.6(c) and 3.7(c) show with red dots all positions with a probability $P_i \leq 0.9 \max_j (P_j)$ and black dots those with $P_i > 0.9 \max_j (P_j)$. In addition, a green arrow is pointing to the actual contact location.



(A) Cumulative probability of collision and maximum probability for each link.

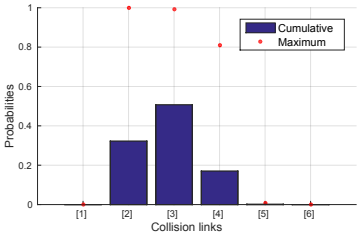


(B) 3D projection of the principal components of the most probable particles.

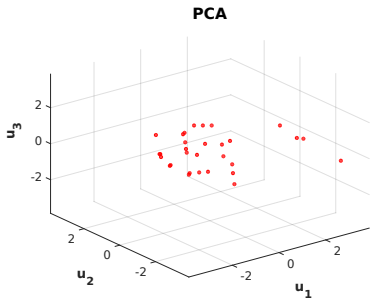


(C) Surface grid. In red positions with a probability lower than 0.9 times the maximum probability, in black the other positions.

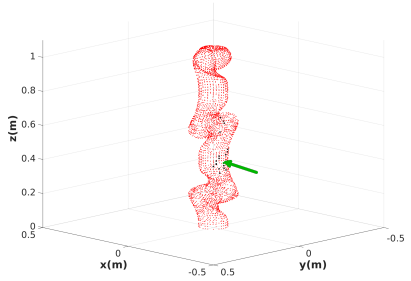
FIGURE 3.5: Results for **collision 1** using 20000 particles.



(A) Cumulative probability of collision and maximum probability for each link.

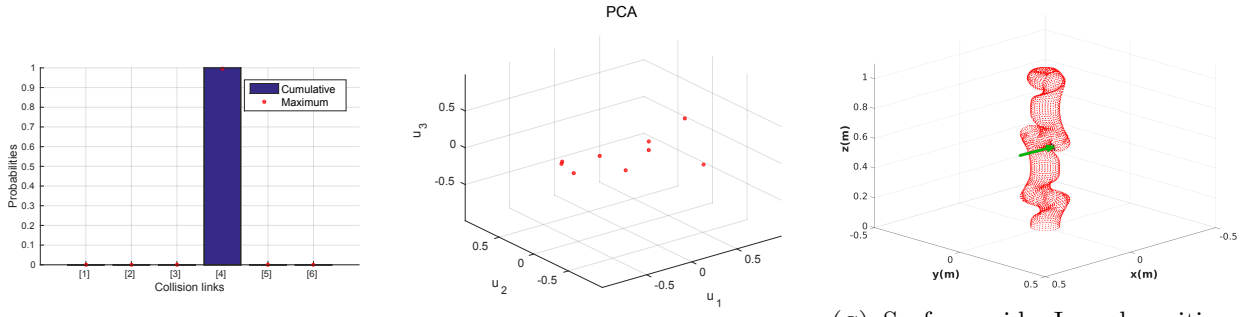


(B) 3D projection of the principal components of the most probable particles.



(C) Surface grid. In red positions with a probability lower than 0.9 times the maximum probability, in black the other positions.

FIGURE 3.6: Results for **collision 2** using 20000 particles.



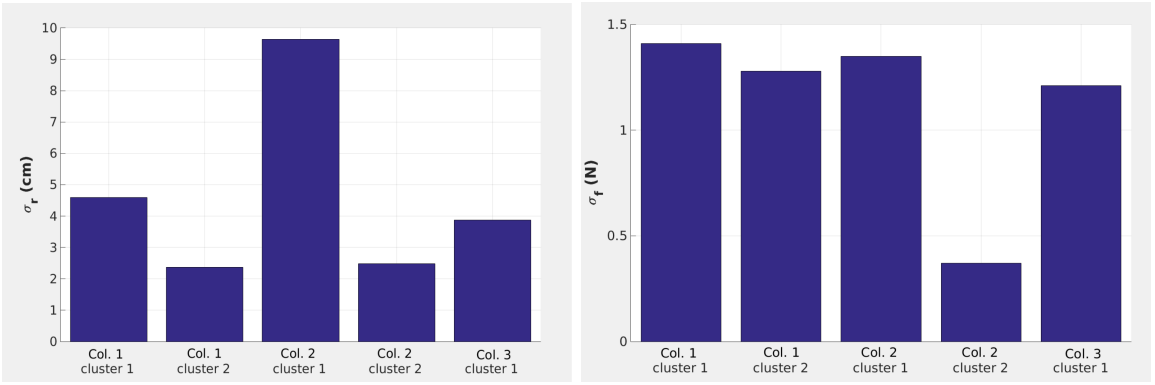
(A) Cumulative probability of collision and maximum probability for each link.

(B) 3D projection of the principal components of the most probable particles.

(C) Surface grid. In red positions with a probability lower than 0.9 times the maximum probability, in black the other positions.

FIGURE 3.7: Results for **collision 3** using 20000 particles.

Similarly as in Subsection 3.1.1, we cluster the most probable particles using K-means. Figure 3.8 shows the dispersion on position (σ_r) and on force (σ_F) defined in equations 3.1 and 3.2 respectively for each cluster.



(A) Dispersion on position σ_r for each collision and cluster.

(B) Dispersion on force σ_F for each collision and cluster.

FIGURE 3.8: Dispersion on position and force for all collisions applied on the Kuka LWR robot.

We explained in chapter 2 that our approach allows us to determine multiple contacts that produce the same torque sensor readings. In subfigure 3.2(a) one can see that there are contacts on links 2, 3 and 4 are equally probable. As predicted, we observed in figures 3.3 and 3.8 that in general contacts closer to the base of the robot are difficult to determine meaning that probable contacts have large dispersion no matter the robot used. When contacts are applied closer to the end of the robot, more torque information is available and this leads to better results for the estimation of the contact's position and force. Moreover, in figure 3.3 we can see that for more of the results there

is a relation between force dispersion σ_F and position dispersion σ_r in a way that as lower is one, lower is the other. In figure 3.8 we see that this rule does not work for all cases because the shape of the robot is odd. For instance, cluster 1 of the second collision has a high dispersion on position but more or less the same dispersion on force as the other contacts.

Thanks to Kuka LWR mesh information we could discretize the surface of the robot and build a set of particles that represent a uniform distribution over the robot's surface, for which the prior probabilities are non-zero. This allowed us to find probable collisions with less particles, and thus at a higher frequency. In addition, the shape of the surface also contributes to that improvement. However, notice in figure 3.8 that we still have the problem that we are not able to find one unique solution for contacts close to the base (i.e. collision 1 and 2) even though the robot's shape helps to reduce those cases. The first contact applied on Kuka LWR showed how two contact regions with opposite forces are equally probable. Subplot 3.5(b) allowed us to distinguish graphically those regions.

3.1.3 Gazebo

The engine used to solve the dynamics of the system was the Open Dynamics Engine (ODE). The contact locations used in this section were the same ones as in Matlab (recall table 3.3) although forces were 4 times bigger in order to be closer to the human range[2]. We used a personal computer with 2.3 GHz Intel Xeon processor to do the experiments. In the simulations done with Gazebo, we studied how different values of σ (standard deviation of equation 2.6) and N (number of particles) affect the performance of the algorithm for each collision. Four simulations with the following parameters were done for each collision:

- Simulation 1: $N = 18000$ and $\sigma = 0.1$
- Simulation 2: $N = 30000$ and $\sigma = 0.1$
- Simulation 3: $N = 50000$ and $\sigma = 0.1$
- Simulation 4: $N = 30000$ and $\sigma = 0.2$

Figures 3.9 to 3.16 are used for a qualitative appreciation of the algorithm. They show Rviz displays for collision 1, 2 and 3. At each time-step the display is updated. In each figure you can see spheres at each possible collision location. They are red-green scale colored where red means zero probability and green maximum probability. We display probabilities according to positions because

[2]In the case of collision 1, probable particles can be grouped into two clusters. Results on figures 3.12, 3.13, 3.18 and 3.19 consider only the most probable particle of the cluster the actual collision belongs to.

it is easy to see where the collision can take place. The probability assigned to each position is the maximum probability of all particles on that position. This is because at each position there can be several forces with same direction and different norm. Besides the display in Rviz, we also did a quantitative evaluation of the performance of the algorithm. Each time the algorithm is executed, we save in a file the contact force and location that have the highest probability. Afterwards, we compute a weighted dispersion on position (σ_r) and force (σ_F) of the contacts saved. Equations 3.3 and 3.4 show the way these dispersions are computed:

$$\sigma_r = \sqrt{\frac{\sum_i P_i \cdot ||\vec{r}_i - \vec{\mu}_r||^2}{\sum_i P_i}} \tag{3.3}$$

$$\sigma_F = \sqrt{\frac{\sum_i P_i \cdot \left|\left|\vec{F}_i - \vec{\mu}_F\right|\right|^2}{\sum_i P_i}} \tag{3.4}$$

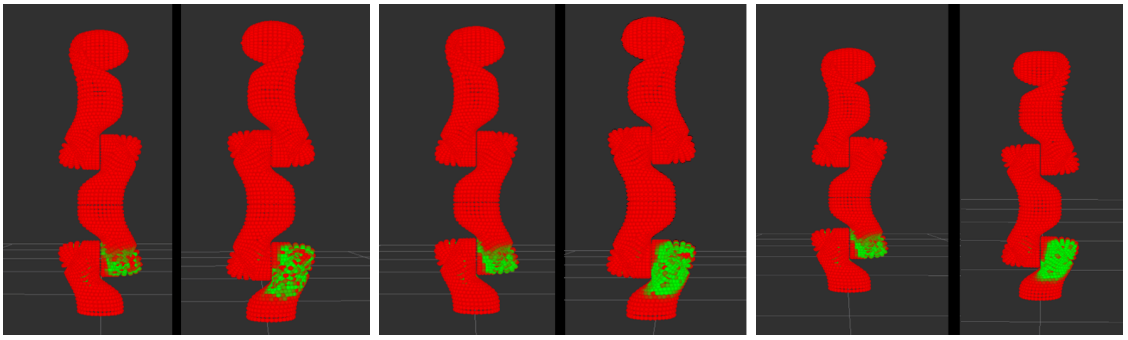
Where $\vec{\mu}_r$ and $\vec{\mu}_F$ are the mean on position and force of each simulation. P_i is the probability, $\vec{r}_i$ is the contact location and $\vec{F}_i$ is then contact force of the time-step i. As we know the collision location ($\vec{r}_i$) and force ($\vec{F}_i$) applied, we computed the error on position (ϵ_r) and force (ϵ_F) as:

$$\epsilon_r = \frac{1}{M} \cdot \sqrt{\sum_i \left|\left|\vec{r}_i - \vec{R}\right|\right|^2} \tag{3.5}$$

$$\epsilon_r = \frac{1}{M} \cdot \sqrt{\sum_i \left|\left|\vec{F}_i - \vec{F}\right|\right|^2} \tag{3.6}$$

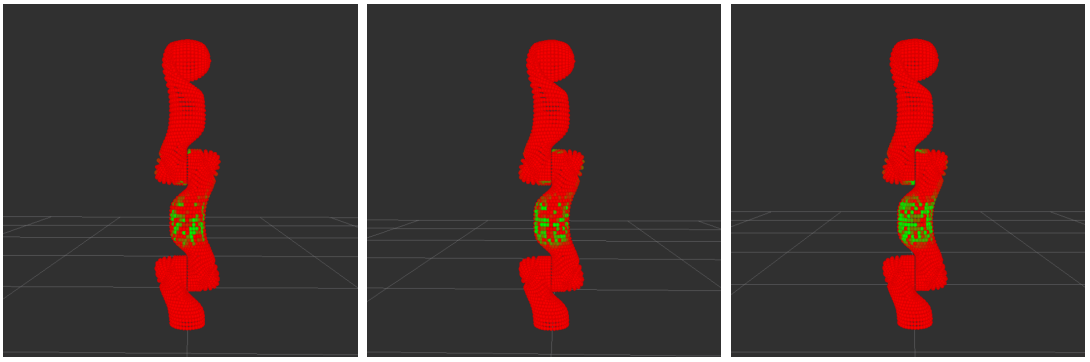
Where M is the number of time-steps, $\vec{r}_i$ and $\vec{F}_i$ are the contact location and force of the time-step i. Figures 3.12 to 3.19 show these dispersions and errors for all collisions.

The execution time changes only when we change the number of particles and it does not affect the results presented in this section. For simulations with $N = 18000$, it was about 0.22 seconds, the one for $N = 30000$ was about 0.35 seconds and the one for $N = 50000$ was about 0.67 seconds.



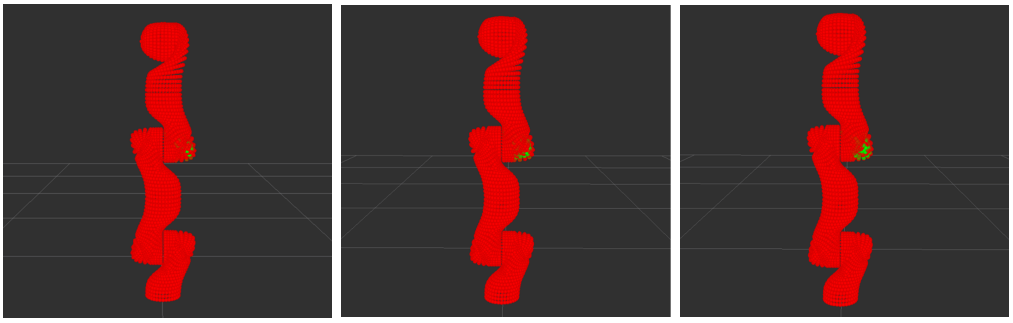
(A) $N = 18000$ (B) $N = 30000$ (C) $N = 50000$

FIGURE 3.9: Rviz display for **collision 1** with $\sigma = 0.1$. The left view of the robot is represented on the left side of each subplot and the right view on the right. Red spheres represent zero probability contact locations and green spheres high probability locations.



(A) $N = 18000$ (B) $N = 30000$ (C) $N = 50000$

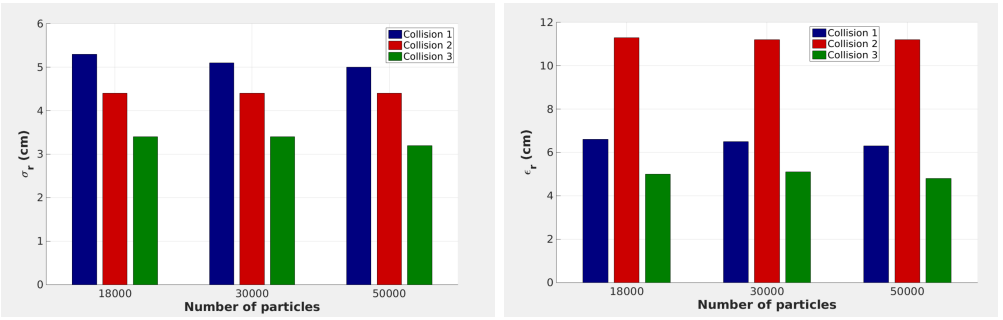
FIGURE 3.10: Rviz display for **collision 2** with $\sigma = 0.1$. Red spheres represent zero probability contact locations and green spheres high probability locations.



(A) $N = 18000$ (B) $N = 30000$ (C) $N = 50000$

FIGURE 3.11: Rviz display for **collision 3** with $\sigma = 0.1$. Red spheres represent zero probability contact locations and green spheres high probability locations.

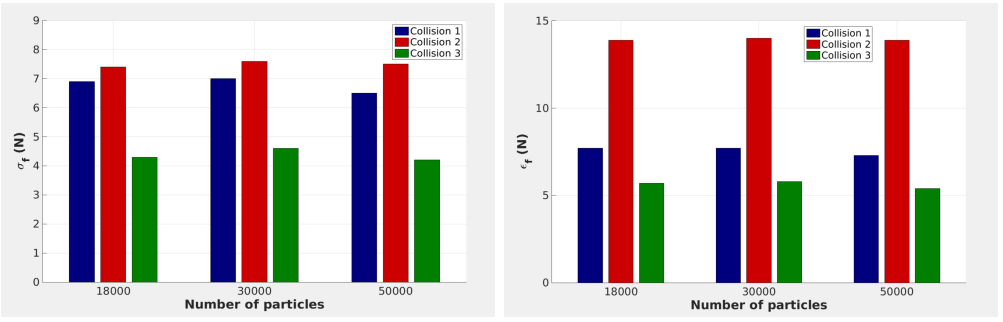
In figures 3.9 to 3.11 we can observe how the distribution of probable positions changes when changing the number of particles N. For all these three cases, the larger the number of particles, the more positions appear with high probability and the more uniform is the distribution. The reason of this effect is that increasing N, more forces per position are sampled so more uniform is the sampling. However, in figures 3.12 and 3.13 we can not observe an important change on the numerical results that consider only the most probable particle at each time-step. The variation on dispersion and error of each collision is not significant although it is remarkable comparing different collisions. In general, as the collision takes places closer to the end effector, the solution is more precise. This results agree with the ones obtained in Subsection 3.1.2.



(A) **Dispersion on position** σ_r in function of the number of particles.

(B) **Error on position** ϵ_r in function of the number of particles.

FIGURE 3.12: σ_r and ϵ_r in function of the number of particles for collisions 1, 2 and 3.

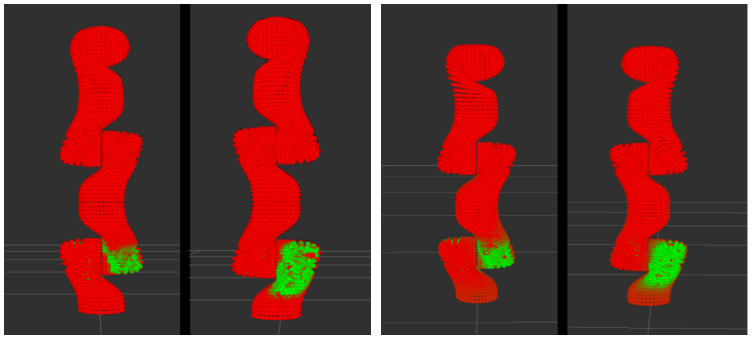


(A) **Dispersion on force** σ_F in function of the number of particles.

(B) **Error on force** ϵ_F in function of the number of particles.

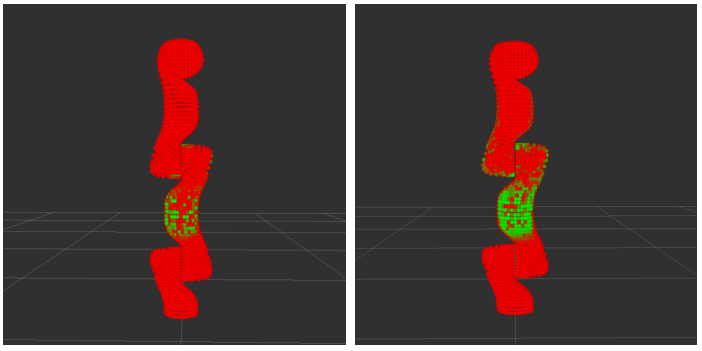
FIGURE 3.13: σ_F and ϵ_F in function of the number of particles for collisions 1, 2 and 3.

In figures 3.14 to 3.16 we can observe how the distribution of probable positions changes when changing the value of σ. Like the simulations modifying N, simulations with larger σ have more positions with high probability. Nevertheless, the reason for that effect in this case is different than before. In here, we do not have a more uniform sampling but we consider a noisier distribution. Consequently, for $\sigma = 0.2$ we can see that points far away from the solution have a remarkable



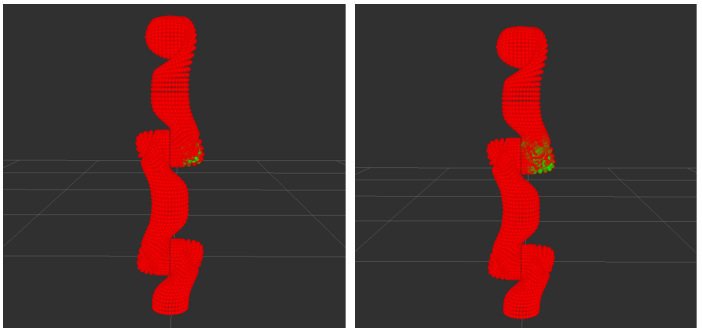
(A) $\sigma = 0.1$ (B) $\sigma = 0.2$

FIGURE 3.14: Rviz display for **collision 1** with $N = 30000$. Left view of the robot is shown on the left side of each subfigure and right view on the right one. Red spheres represent zero probability contact locations and green spheres high probability locations.



(A) $\sigma = 0.1$ (B) $\sigma = 0.2$

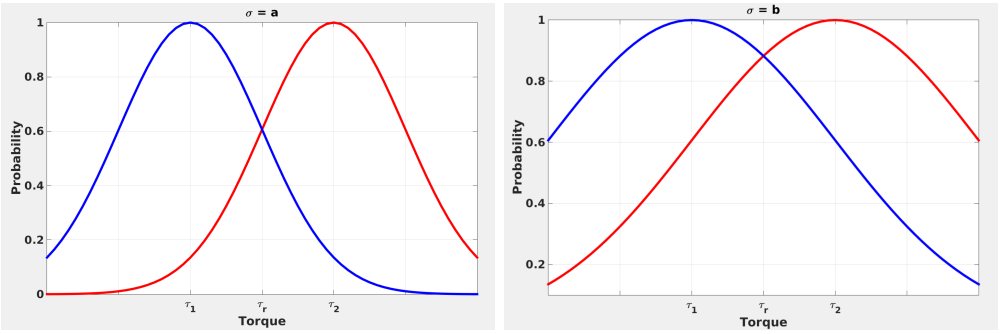
FIGURE 3.15: Rviz display for **collision 2** with $N = 30000$. Red spheres represent zero probability contact locations and green spheres high probability locations.



(A) $\sigma = 0.1$ (B) $\sigma = 0.2$

FIGURE 3.16: Rviz display for **collision 3** with $N = 30000$. Red spheres represent zero probability contact locations and green spheres high probability locations.

probability, for example subfigure 3.15(b). For this reason, no simulations were done with σ>0.2. The reason why we did not consider σ<0.1 either is a bit more complex. Recall that torque is the cross product between $\vec{r}$ and $\vec{F}$. Therefore, as the possible positions and forces are not continuous because of the prior (2.5), the possible torques are not continuous either. In order not to have a really low probability for torques different from the possible ones the Gaussian has to be wide enough. Figure 3.17 represents that property in 1D. If the real torque (τ_r) is between two possible torques (τ_1 and τ_2), then, the maximum probability for τ_1 and τ_2 to be the real torque is lower with $\sigma = a$ than $\sigma = b$ if b>a. With the mesh used for simulations, a standard deviation greater or equal than 0.1 is enough to have an homogeneous probability over the torque space. In other words, for a standard deviation lower than 0.1 you may not find any solution, or you may find a solution with very low probability.

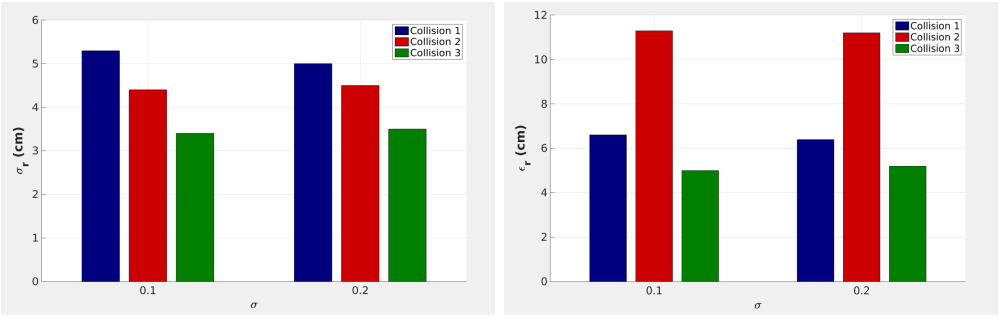


(A) Gaussian distribution with mean τ_1 for τ_2 and standard deviation $\sigma = a$.

(B) Gaussian distribution with mean τ_1 for τ_2 and standard deviation $\sigma = b$.

FIGURE 3.17: Gaussian distributions with different standard deviations.

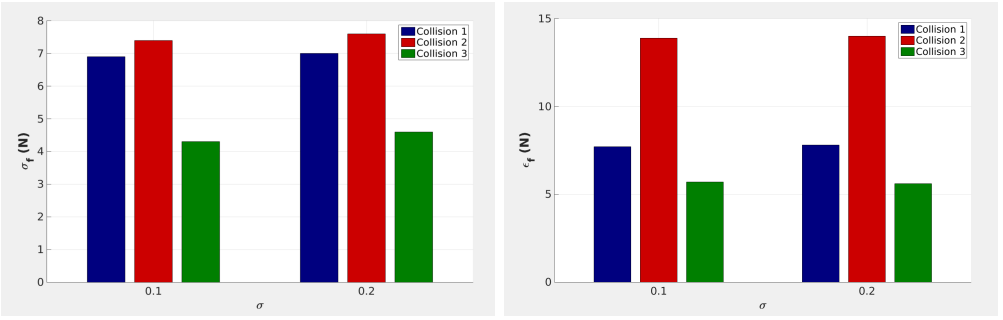
Figures 3.18 and 3.19 show results similar to the ones displayed on figures 3.12 and 3.13. The conclusion is then the same: in general, the collision force and position errors (ϵ_F and ϵ_r) and dispersions (σ_F and σ_r) decrease when the collision takes place closer to the end of the robot.



(A) Weighted dispersion on position σ_r in function of σ.

(B) Error on position ϵ_r in function of σ.

FIGURE 3.18: σ_r and ϵ_r in function of σ for collisions 1, 2 and 3.



(A) Weighted dispersion on force σ_F in function of σ.

(B) Error on force ϵ_F in function of σ.

FIGURE 3.19: σ_F and ϵ_F in function of σ for collisions 1, 2 and 3.

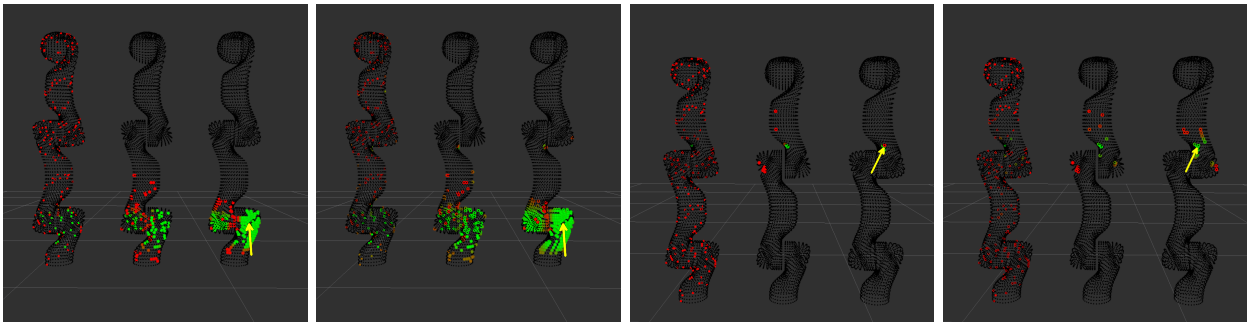
3.2 On the robot

In this section, we present the results for the experiments done with the real robot. We studied the performance of the algorithm for several contact positions and forces in different links and also in some robot configurations.

We changed the value of σ to see how that affects the results. In this case, the values used were 0.2 and 0.5. The reason why values lower than 0.2 were not considered is the same one described with figure 3.17 although this value is larger than the one used in section 3.1.3 because the number of possible positions is lower and then distance between two possible torques larger. Values greater than 0.5 were not considered either because they increased the probability of positions far away from the solution. Due to the fact that the algorithm was running on three different phases, only the most probable contact on phase C was considered because is the one with highest density of contacts. We compared the measured force norm with the estimated force norm because we assumed that there is no friction on the contact so measured force components and estimated force components should be the same but expressed in different frames. The range of forces tested was between 3N and 30N with a step of 0.7N. One difference between the simulations and the real robot was the way to know the applied contact force intensity and direction. In simulations we just define it, but on the real robot we measure the position with a ruler and the force with a six axis torque-force sensor.

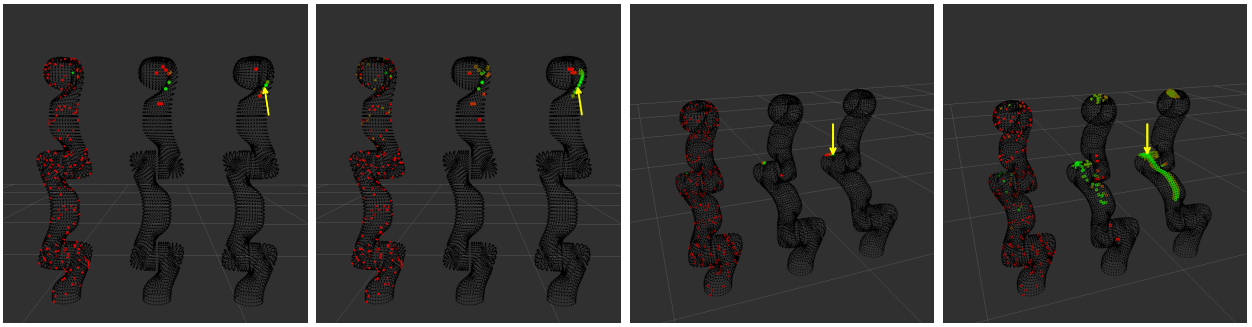
Figures 3.20, 3.21 and 3.22 show all locations considered at each phase. These positions are represented with red-green scale colored balls. Red correspond to low probability and green to high probability. There are also small black balls to represent the surface of the robot. On figure 3.20 and subplots 3.21(a) and 3.21(b) the robot is on the same configuration and we can clearly see that the collisions close to the end of the robot have less probable particles. Collisions on subfigures 3.21(a), 3.21(a), 3.22(a) and 3.22(b) are applied on the same position on the surface but with different robot configurations. One can observe that the probable particles in both cases have more or less the same

distribution. In all the cases, when we increased σ the probability of the particles increased and, consequently, more particles are displayed on these figures. Subfigures 3.21(c) and 3.21(d) are the one where this effect is more apparent.



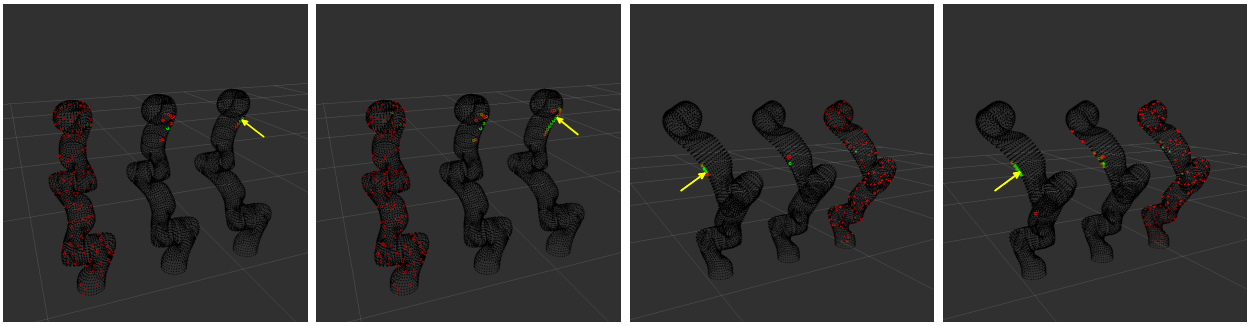
(A) $\sigma = 0.2$ (B) $\sigma = 0.5$ (C) $\sigma = 0.2$ (D) $\sigma = 0.5$

FIGURE 3.20: Contact location display on Rviz for the first collision (a) and (b) applied on the first link and the second collision (c) and (d) applied on the fourth link.



(A) $\sigma = 0.2$ (B) $\sigma = 0.5$ (C) $\sigma = 0.2$ (D) $\sigma = 0.5$

FIGURE 3.21: Contact location display on Rviz for the third collision (a) and (b) applied on the fifth link and the fourth collision (c) and (d) applied on the third link.



(A) $\sigma = 0.2$ (B) $\sigma = 0.5$ (C) $\sigma = 0.2$ (D) $\sigma = 0.5$

FIGURE 3.22: Contact location display on Rviz for the fifth collision (a) and (b) applied on the fifth link and the sixth collision (c) and (d) applied on the fourth link.

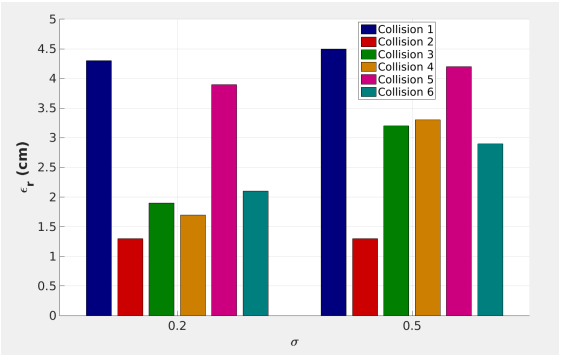
Like in section 3.1.3, numerical results were used to compare the performance of the algorithm. Errors on collision force (ϵ_F) and position (ϵ_r) are computed similarly to equations 3.6 and 3.5 and dispersion on position (σ_r) is computed similarly to equation 3.3. The first two values help us to see if the solution is close to the actual collision and the third one helps us to know if the solution is disperse or not.

$$\epsilon_F = \sqrt{\frac{\sum_i P^i_{max} \cdot \left(\left|\left|\vec{F}_{ap_i}\right|\right| - \left|\left|\vec{F}_{es_i}\right|\right|\right)^2}{\sum_i P^i_{max}}} \tag{3.7}$$

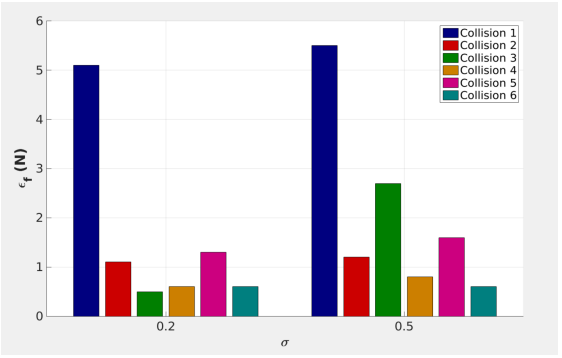
$$\epsilon_r = \sqrt{\frac{\sum_i P^i_{max} \cdot ||\vec{r}_{ap} - \vec{r}_{es_i}||^2}{\sum_i P^i_{max}}} \tag{3.8}$$

$$\sigma_r = \sqrt{\frac{\sum_i P^i_{max} \cdot ||\vec{r}_{es_i} - \vec{\mu}_r||^2}{\sum_i P^i_{max}}} \tag{3.9}$$

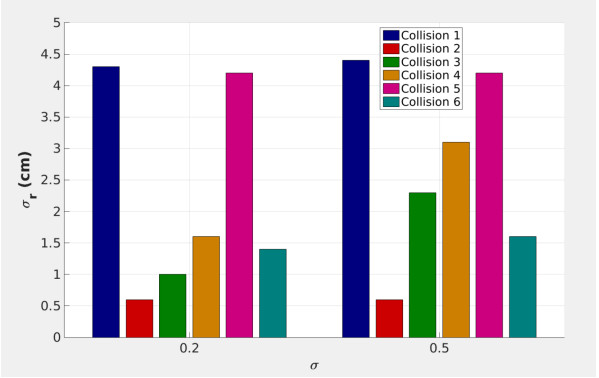
Where P^i_{max} is the probability of the most probable particle, $\vec{F}_{ap_i}$ is the applied force, $\vec{F}_{es_i}$ is the estimated force and $\vec{r}_{es_i}$ is the estimated position at time-step i. $\vec{r}_{ap}$ is the position where the force is applied and $\vec{\mu}_r$ is the mean of all $\vec{r}_{es_i}$. This errors are shown in Figure 3.23. In this figure we can see that the error on position (ϵ_r) and the position dispersion (σ_r) for collisions 1 and 5 are larger than the rest. This result is coherent with the graphical representation on figure 3.20(a), 3.20(b), 3.21(c) and 3.21(d), where there are a lot of particles with high probability. We can also see in figure 3.23 that results obtained for $\sigma = 0.2$ are better because the errors and dispersion are lower. However, for most of the cases these results vary more when comparing different collisions than when comparing different values of σ. This means that there is a strong relation between the probability to find a good solution and the torque information we read on the sensors.



(A) Collision position error for all 6 collisions applied to the robot for $\sigma = 0.2$ and $\sigma = 0.5$.



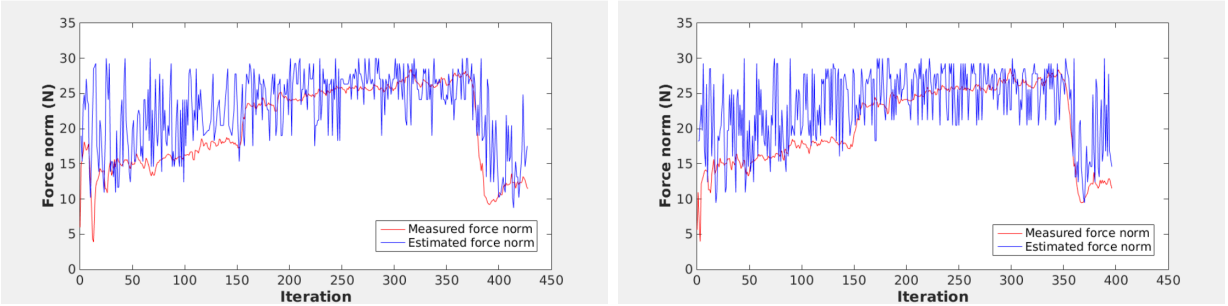
(B) Collision force norm error for all 6 collisions applied to the robot for $\sigma = 0.2$ and $\sigma = 0.5$.



(C) Dispersion on position for all 6 collisions applied to the robot for $\sigma = 0.2$ and $\sigma = 0.5$.

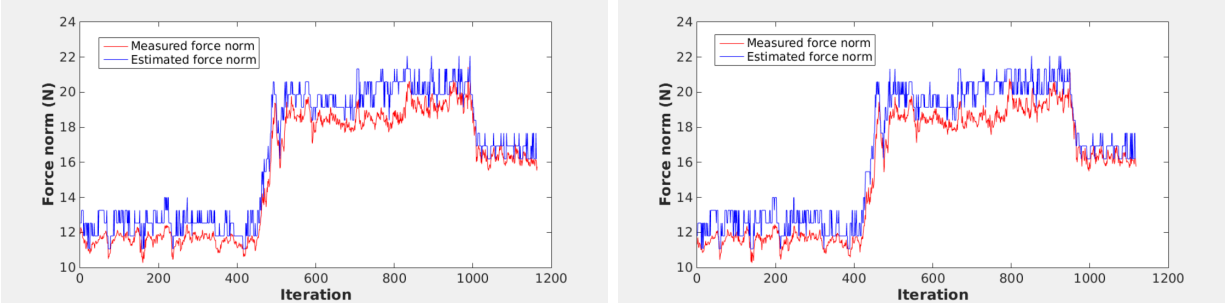
FIGURE 3.23: Errors on the estimation of collisions and dispersion on the estimation of the collision position.

The results we present below belong to the measurements done with the force-torque sensor. In each graph we plot in blue the estimated force norm and in red the force norm measured with the sensor. Figure 3.24 show the results for $\sigma = 0.2$ and $\sigma = 0.5$ for the first collision. This collision was applied on the first contact so only one measurement on torques was different from zero. For this reason the estimated force norm is oscillating close to the real value. Figure 3.25 show the result for the second collision. It was applied on the fourth link then more torque measurements were different from zero. For this reason we can see that the oscillations are smaller. The same happens in figure 3.26 because the collision was applied on the fifth link.



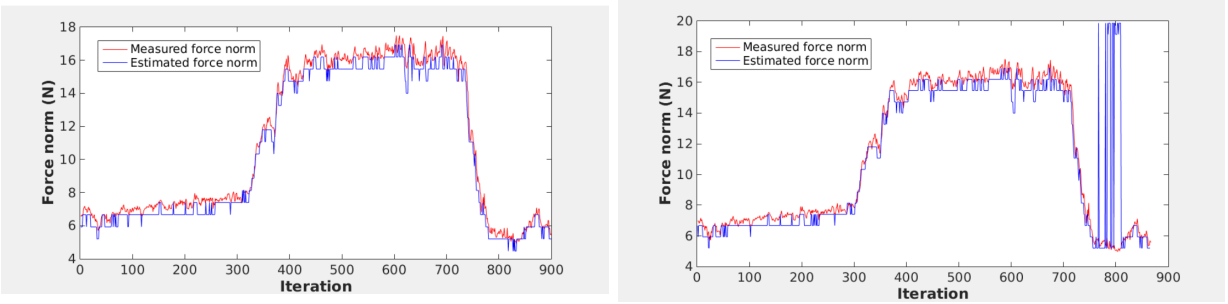
(A) $\sigma = 0.2$ (B) $\sigma = 0.5$

FIGURE 3.24: Force norms for **collision 1**.



(A) $\sigma = 0.2$ (B) $\sigma = 0.5$

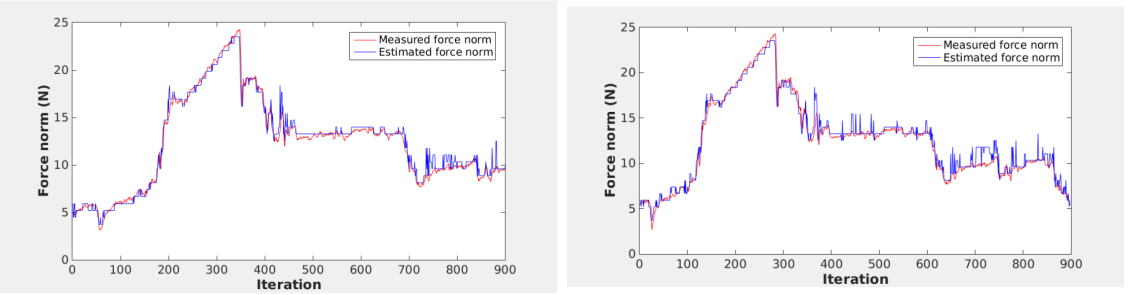
FIGURE 3.25: Force norms for **collision 2**.



(A) $\sigma = 0.2$ (B) $\sigma = 0.5$

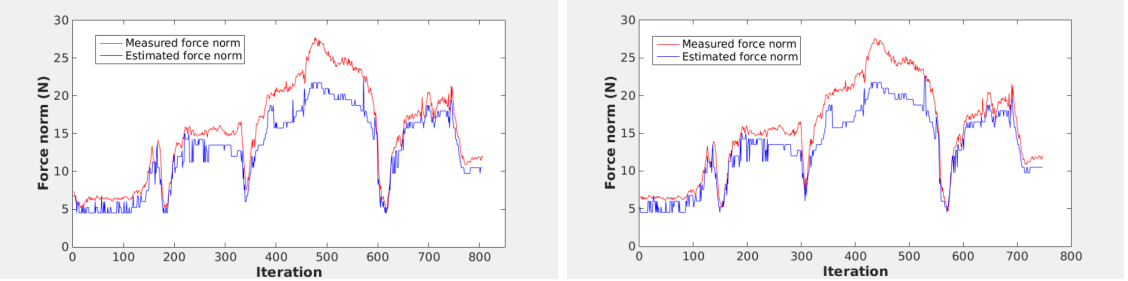
FIGURE 3.26: Force norms for **collision 3**.

In figures 3.24 and 3.25 we can not appreciate an improvement on the estimation of the force norm changing σ. However, in figure 3.26 we can see that for really low forces and $\sigma = 0.5$ the estimation of the force norm does not agree with the measured one. Low applied forces means low measured torques. If we have a value of σ large enough, then particles with positions on the surface of the robot that can not create torques increase their probability. Eventually, this probability can be equal or higher than the probability of particles close to the actual collision.



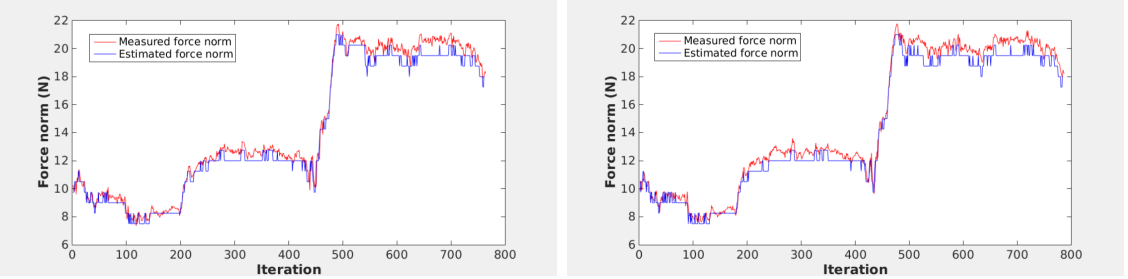
(A) $\sigma = 0.2$ (B) $\sigma = 0.5$

FIGURE 3.27: Force norms for **collision 4**.



(A) $\sigma = 0.2$ (B) $\sigma = 0.5$

FIGURE 3.28: Force norms for **collision 5**.



(A) $\sigma = 0.2$ (B) $\sigma = 0.5$

FIGURE 3.29: Force norms for **collision 6**.

Figure 3.27 show the fourth collision applied on the third link. Subfigure 3.27(b) has more fluctuations on the results than subfigure 3.27(a). As we explained before, a large value of σ may increase the probability of particles far away from the solution. Consequently, the force norm of the most probable particle may fluctuate. Actually, we can see in subfigure 3.21(d) that there are a lot of probable particles distributed over a large surface. Comparing figure 3.28 and 3.29 we can say again that it is more important where the collision takes place than the value of σ to have good results.

Chapter 4

Conclusions and future work

In conclusion, the developed method is able to obtain contacts with high probability in several conditions although the problem is not bijective and the information used to determine collisions is often insufficient. Moreover, the algorithm can find multiple regions were contacts are probable. Different approaches can be used to reduce the number of regions and have more precise collisions. One could increase the information used. The use of three axis torque sensor per joint may be the simplest solution in that case, although it may be expensive and difficult to implement in practice. The implementation of a control strategy may be also useful. In this way, robot could move and distinguish between several contact areas. Furthermore, it could be used to explore environment surfaces. The only problem would be the frequency at which the algorithm runs. One solution is to split the computation of the particle probabilities on several processors running in parallel and increase the speed of the algorithm.

We observed that the way of sampling particles is important when finding a solution. Random sampling give a non homogeneous solutions while using the deterministic sampling suggested results are more uniform. In addition, deterministic sampling allowed us to have an homogeneous representation of the probable contact area. We used the most probable particle at each time-step to know where the contact was, although a weighted mean over the particles on the contact area could also be good to know.

Finally, we expose some conclusions regarding σ, which was modified in the experiments to see the performance of the algorithm. We can say that smaller values are better than larger values in general because only contacts close to the actual one are probable. However, due to the fact that possible contacts and possible torques are discrete there is a lower bound on σ. In addition, we have observed on the experiments with the real robot that although the accuracy increases when decreasing sigma, it depends more on where the contact takes place and the configuration of the robot. Consequently, we can not determine a priory the accuracy with the parameter σ.

Appendix A

Tables of results

TABLE A.1: Contact location dispersion (σ_r) and contact force dispersion (σ_F) for each experiment and cluster tested on the simple robot with Matlab.

Collision		σ_r (cm)	σ_F (N)
1	cluster 1	7.9	0.15
	cluster 2	7.4	0.18
	cluster 3	3.1	0.09
2	cluster 1	3.9	0.03

TABLE A.2: Contact location dispersion (σ_r) and contact force dispersion (σ_F) for each experiment and cluster tested on Kuka LWR with Matlab.

Collision		σ_r(cm)	σ_F(N)
1	cluster 1	4.6	1.4
	cluster 2	2.4	1.3
2	cluster 1	9.6	1.4
	cluster 2	2.5	0.4
3	cluster 1	3.9	1.2

TABLE A.3: Contact location dispersion (σ_r), contact force dispersion (σ_F), contact location estimation error (ϵ_r) and contact force estimation error (ϵ_r) for each collision, number of particles (N) and σ tested on Gazebo simulations.

Collision	N	σ	σ_r (cm)	σ_F(N)	ϵ_r (cm)	ϵ_F(N)
1	18000	0.1	5.3	6.9	6.6	7.7
	30000	0.1	5.1	7.0	6.5	7.7
	50000	0.1	5.0	6.9	6.3	7.3
	30000	0.2	5.0	7.0	6.4	7.8
2	18000	0.1	4.4	7.4	11.3	13.9
	30000	0.1	4.4	7.6	11.2	14.0
	50000	0.1	4.4	7.5	11.2	13.9
	30000	0.2	4.5	7.6	11.2	14.0
3	18000	0.1	3.4	4.3	5.0	5.7
	30000	0.1	3.4	4.6	5.1	5.8
	50000	0.1	3.2	4.2	4.8	5.4
	30000	0.2	3.5	4.6	5.2	5.6

TABLE A.4: Contact force estimation error (ϵ_F), contact location estimation error (ϵ_r), contact location dispersion (σ_r) and algorithm frequency for each collision and each value of σ tested on the real robot.

Collision	σ	ϵ_F(N)	ϵr(cm)	σ_r(cm)	Frequency(Hz)
1	0.2	5.1	4.3	4.3	5
	0.5	5.5	4.5	4.4	3
2	0.2	1.1	1.3	0.6	8
	0.5	1.2	1.3	0.6	8
3	0.2	0.5	1.9	1.0	10
	0.5	2.7	3.2	2.3	7
4	0.2	0.6	1.7	1.6	8
	0.5	0.8	3.3	3.1	5
5	0.2	1.3	3.9	4.2	9
	0.5	1.6	4.2	4.2	7
6	0.2	0.6	2.1	1.4	10
	0.5	0.6	2.9	1.6	8

Bibliography

[1] C. Kemp, A. Edsinger, and E. Torres-Jara, "Challenges for robot manipulation in human environments [grand challenges of robotics]," *IEEE Robotics and Automation Magazine*, vol. 14, pp. 20–29, March 2007.

[2] D. Kragic and H. I. Christensen, "Survey on visual servoring for manipulation," tech. rep., Royal Institute of Technology, Stocolm, Sweden, 2002.

[3] O. Kerpa, K. Weiss, and H. Worn, "Development of a flexible tactile sensor system for a humanoid robot," in *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 1–6, Oct. 2003.

[4] J. K. Salisbury, "Interpelation of contact geometries from force measurements," in *Proc. 1st Int. Symp. Robot. Res.*, pp. 565–577, 1984.

[5] "ISO 10218:2011. Robots and robotic devices – Safety requirements for industrial robots," 2011.

[6] "ISO 13482:2014. Robots and robotic devices – Safety requirements for personal care robots," 2014.

[7] N. M. Oliver, B. Rosario, and A. P. Pentland, "A bayesian computer vision system for modeling human interactions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, August 2000.

[8] L. Xia, C.-C. Chen, and J. Aggarwal, "Human detection using depth information by kinect," in *Proc. Int. Workshop HAU3D*, pp. 15–22, June 2011.

[9] T. Tsujumura and T. Yabuta, "Object detection by tactile sensing method employing force/-torque information," *IEEE Transactions on Robotis and Automation*, vol. 5, August 1989.

[10] L. D. Phong, J. Choi, and S. Kang, "External force estimation using joint torque sensors for a robot manipulator," in *Proc. IEEE International Conference on Robotics and Automation*, pp. 4507–4512, 2012.

[11] T. Tsuji, Y. Kaneko, and S. Abe, "Whole-body force sensation by force sensor with shell-shaped end-effector," *IEEE Transactions on industrial electronics*, vol. 56, pp. 1357–1382, May 2009.

[12] N. Kurita, S. Sakaino, and T. Tsuji, "Whole-body force sensation by force sensor with end-effector of arbitrary shape," in *Proc. IEEE International Conference on Intelligent Robots and Systems*, October 2012.

[13] S. Lu, J. H. Chung, and S. A. Velinsky, "Human-robot interaction detection: a wrist and base force/torque sensors approach," in *Proc. IEEE International Conference on Robotics and Automation*, pp. 3796–3801, April 2005.

[14] F. Dimeas, J. D. Avendaño-Valencia, and N. Aspragathos, "Human-robot collision detection and identification based on fuzzy and time series modelling," *Robotica*, pp. 1–13, April 2014.

[15] E. Magrini, F. Flacco, and A. D. Luca, "Estimation of contact forces using a virtual force sensor," in *Proc. IEEE International conference on intelligent Robots and Systems*, September 2014.

[16] A. Petrovskaya, J. Park, and O. Khatib, "Probabilistic estimation of whole body contacts for multi-contact robot control," in *Proc. IEEE International conference on robotics and automation*, April 2007.

[17] A. Petrovskaya and O. Khatib, "Global location of objects via touch," *IEEE Transactions on robotics*, vol. 27, pp. 569–585, June 2011.

[18] N. Likar and L. Žlajpah, "Estimation of contact information using nonlinear optimization," in *Proc. of the RAAD*, September 2014.

[19] R. Smits, "KDL: Kinematics and Dynamics Library." http://www.orocos.org/kdl.