Tactile localization: dealing with uncertainty from the first touch

Eric Valls i Grünewald

Advisors:

Alberto Rodriguez (MIT)

Maria Alberich (UPC)

Bachelor's Degree in Informatics Engineering

Bachelor's Degree in Mathematics

May 2020

Abstract

In this thesis we present an approach to object tactile localization for robotic manipulation which explicitly deals with the uncertainty to overcome the locality of tactile sensing. To that purpose, we estimate full probability distributions of object pose. Moreover, given a 3D model of the object in question, our framework localizes from the first touch, meaning no physical exploration of the object is needed beforehand.

Given a signal from the tactile sensor, we divide the estimation of a probability distribution of object pose in two main steps. First, before touching the object, we sample a dense set of poses of the object with respect to the sensor, we simulate the signal the sensor would get when touching the object at these poses, and we train a similarity function between these signals. In the second part, while manipulating the object, we compare the signal coming from the sensor to the set of previously simulated ones, and the similarities between these give the discretized probability distribution over the possible poses of the object with respect to the sensor.

We extend this work by analyzing the scenario where multiple tactile sensors are touching the object at the same time, by fusing the probability distributions coming from the individual sensors to get a better distribution.

We present quantitative results for four objects. We also present the application of this approach in a larger system and an ongoing research direction towards *tactile active perception*.

Keywords list: *robotics, perception system, object localization, tactile sensing, uncertainty reasoning, multicontact, active perception.*

Mathematics Subject Classification American Mathematical Society code: 93C41 Control/observation systems with incomplete information

Resum

En aquesta tesi proposem un nou sistema per localitzar d'objectes amb sensors tàctils per a manipulació robòtica, que tracta, de forma explícita, la incertesa inherent al sentit del tacte. Amb aquesta fi, estimem la distribució de probabilitat completa de la posició de l'objecte. A més a més, donat el model 3D de l'objecte en qüestió, el nostre sistema no requereix una exploració prèvia de l'objecte amb el sensor, podent localizar des del primer contacte.

Donat un senyal provinent del sensor tàctil, dividim l'estimació de la distribució de probabilitat de la posició de l'objecte en dos passos. Primer, abans de tocar l'objecte, definim un conjunt dens de posicions de l'objecte respecte al sensor, simulem el senyal que esperaríem rebre del sensor si l'objecte fos tocat en aquestes posicions, i entrenem una funció de semblança entre aquests senyals. Segon, mentre l'objecte està sent manipulat, comparem el senyal provinent del sensor amb els senyals simulats prèviament, i les semblances entre aquests donen la distribució de probabilitat discreta a l'espai de posicions de l'objecte respecte al sensor.

Estenem aquesta feina analitzant l'escenari on múltiples sensors tàctils contacten l'objecte a la vegada. Fusionem les distribucions de probabilitat provinents dels diferents sensors per obtenir una distribució millorada.

Presentem resultats quantitatius per quatre objectes. També mostrem una aplicació d'aquest sistema en un sistema més ampli i presentem recerca en la qual estem treballant actualment en percepció activa.

Llista de paraules clau: *robòtica, sistema de percepció, localització d'objectes, sensors tàctils, raonament amb incertesa, contacte múltiple, percepció activa*

Codi de classificació de la American Mathematical Society: 93C41 Control/observació de sistemes amb informació incompleta

Resumen

En esta tesis proponemos un nuevos sistema para localizar objetos con sensores táctiles para manipulación robótica, que trata, de forma explícita, la incertidumbre inherente al sentido del tacto. Con este fin, estimamos la distribución de probabilidad completa de la posición del objeto. Además, dado el modelo 3D del objeto que cuestión, nuestro sistema no requiere una exploración previa del objeto con el sensor, pudiendo localizarlo desde el primer contacto.

Dada una señal proveniente del sensor táctil, dividimos la estimación de la distribución de probabilidad de la posición del objeto en dos pasos. Primero, antes de tocar el objeto, definimos un conjunto denso de posiciones del objeto respecto al sensor, simulamos la señal que esperaríamos recibir del sensor si el objeto fuese tocado en estas posiciones, y entrenamos una función de semejanza entre estas señales. Segundo, mientras el objeto está siendo manipulado, comparamos la señal proveniente del sensor con las señales simuladas previamente, y las semejanzas entre estas dan la distribución de probabilidad discreta en el espacio de posiciones del objeto.

Extendemos este trabajo analizando el escenario donde múltiples sensores táctiles tocan el objeto al mismo tiempo. Fusionamos las distribuciones de probabilidad que vienen de los diferentes sensores para obtener una distribución mejorada.

Presentamos resultados cuantitativos para cuatro objetos. También mostramos una aplicación de este sistema en un sistema más amplio y presentamos investigación en la que estamos trabajando actualmente en percepción activa.

Lista de palabras clave: *robótica, sistema de percepción, localización de objetos, sensores táctiles, razonar bajo incertidumbre, contacto múltiple, percepción activa*

Código de classificación de la American Mathematical Society: 93C41 Control/observación de sistemas con información incompleta

Contents

1 Acknowledgment

I would like to thank Prof. Alberto Rodriguez for placing trust in me and giving me this amazing opportunity of diving into the academic world. Thank you also to Maria Bauzà, with whom I have worked very closely in this research, and to all other MCube Lab members, who have taught me many things.

I also would like to thank MIT and CFIS, in particular Miguel Ángel Barja and Toni Pascual, for making this experience possible, and Maria Alberich, for all the advice.

Thank you to Fundació Cellex, for having placed trust in me during the past years and for the financial support which helped me focus on my academic development over all.

Finally, I am especially grateful to all the people who have made this experience super enjoyable and supported me in difficult moments: my Boston-Catalan family, my US family (Cheto, Jordi, Marc and Ferrando) and my parents, for their unconditional support.

2 Preamble

This thesis is the final project for my Bachelor's degrees in Mathematics and Informatics Engineering at Universitat Politècnica de Catalunya and it is the result of the research stay I have done at the Manipulation and Mechanisms Laboratory at MIT (MCube Lab) from September 2019 to May 2020, under the supervision of Prof. Alberto Rodriguez and in collaboration with Maria Bauzà.

Most of the work presented here has been part of a paper we called *Object Pose Estimation with Geometric Tactile Rendering and Tactile Image Matching*, that we submitted to the *Robotics: Science and Systems* conference and is now pending review. It has not been published yet and that's why it doesn't appear in the references. This work is also going to be part of a paper that is going to be presented to *Science: Robotics* in a few weeks, explaining the Kitting System, summarized in section *Complementary research and applications*, where tactile localization is an essential part.

This work finds its place in the two disciplines of my Bachelor's as most of the methods fall in the middle of both, requiring a strong mathematical background to develop the ideas and an algorithmic and programming mindset to implement them in an efficient and usable way. For example the ideas behind the probabilities reasoning in subsection *Multicontact* falls closer to Mathematics, while its implementation using parallelism, GPU and third party libraries so that it can work in real time, falls closer to Informatics Engineering.

I didn't have a direct implication in the development of the sensor signal preprocessing from section *Tactile sensor: GelSlim*, nor in the implementation of the registration algorithm from subsection *Point pose estimation and pose refinement*. I include the first for context and completeness, and the second one as a natural extension to refine point estimates in our methods. The main parts where I used third part libraries was with *pytorch* [1] and *pyrender* [2], used for implementing neural networks and graphics re-

spectively. I implemented the rest in collaboration with Maria Bauzà, and with help of Bryan Lim and Thodoros Sechopoulos. The code is not public because it is confidential at the moment.

The video and all the figures in this work have either been created by myself specifically for this thesis or come from the paper of which I am a coauthor, except Figure 1, which is of common use in the lab, for which I asked for authorization to use it.

3 List of abbreviations

For clearity and explanation fluidity, a few abbreviations and acronyms are used throughout this thesis. These are expanded here:

- w.r.t.: with respect to. Used when talking about local reference frames.
- LIDAR: Light Detection and Ranging sensor.
- a.k.a: also known as. Used to express another way to refer to the same thing.
- i.e.: *id est*. Used to introduce a rephrasing.

4 Introduction

4.1 Motivation

In robotics, a perception system gives the robot the ability to perceive, comprehend, and reason about the environment around it[3]. In the past years, cameras and LIDARs have dominated this field, fostering the development of new technologies for self-driving cars and warehouse automation, for example. However, other perception modalities, as tactile sensing, the one studied in this thesis, are key to overcome the present challenges in the robotics community.

While vision is a crucial sensor for humans, for some of the most simple tasks like using a screwdriver or fastening a button, vision alone is not enough. The first reason are occlusions. When we are manipulating an object, we cannot see it completely due to our hand standing between the object and our eyes. In the extreme case where the object is smaller than the end effector, this occlusion can be total, becoming a big problem. The second reason is that our eyes cannot observe contact or contact force. Even if we look at where the contact is occurring, we cannot describe this contact and thus, obtain the necessary information to perform many tasks. The sense of touch overcomes both by definition, and that's why it is essential for humans to perform a lot of tasks. These arguments are easily transferred to robots, making tactile sensing a key sensing modality in robotic manipulation.

The main challenges to provide robots of tactile sensing come from the hardware perspective – create robust, small and precise sensors that can be integrated in a robot end effector – and the software perspective – interpret the data coming from the sensors to extract useful information that can be used for planning and control –.

This thesis is centered on tactile data interpretation, specifically on trying to localize the object in the robot end effector. The ability to localize an object w.r.t. the robot end

effector is essential for precise placing and tool use, for example. The main challenge is the locality of tactile sensing: the robot only has information from the regions of the object that are in direct contact with its tactile sensors, thus not having a global perspective of where the object lies.

4.2 Related work

For the reasons stated above, tactile localization has started to gain interest from the robotics community and different approaches have been pursued. Below, these are summarized and compared to ours.

Most initial works for tactile localization were developed with low resolution tactile sensors, mainly with sensors that could only detect contact or no-contact. For these reasons, the majority of algorithms were only applicable to simple planar objects or with very distinctive features [4, 5, 6, 7, 8, 9, 10, 11, 12].

To overcome the inherent locality of tactile sensing, some works combined tactile with vision, generally using vision as the major source of information because of its global perspective, and tactile only as refinement of the vision estimate [13, 14, 15, 16, 17, 18, 19]. These approaches still have problems dealing with occlusions, due to their high dependence on vision.

Other works, more aligned to ours, have opted for using high resolution tactile sensors. Most of them rely on a previous physical exploration of the object with the sensor [20, 21, 22]. In [23, 24] a point cloud is extracted from the imaged based tactile sensor, and by exploring the object physically, a global point cloud model of the object is created. Only then can the localization start working. The main drawbacks of these approaches are, first, that the pose estimation obtained is unimodal and, therefore, cannot easily manage the uncertainty of the estimate. Also, the necessity of the physical exploration, makes it hard to use with new objects.

In our system we use an image based high resolution tactile sensor called GelSlim [25]. We work with general objects and we predict 6 degrees of freedom poses, having tactile as the unique source of information. Instead of a predicting a single pose or mode, our approach deals with uncertainty by always keeping the full probability distribution over the possible poses, which explicitly deals with non-uniquenesses, a common problem occurring in tactile which means that an object has similar local geometries in different parts (for example, in a box, all the vertices are locally equal, so one cannot distinguish between them by only touching them). We also extend our approach to use multiple sensors at the same time, which is a more realistic scenario in robotic manipulation. Also, our approach doesn't require physical exploration of the object and, therefore, all our computations and neural network training are done in simulation. We only assume that a 3D model of the object is available, which is common as the design of manufactured objects is done in virtual environments.

From the computer vision community, there are also interesting approaches for localization that have intersection with ours. For the matching algorithm we converged to a similar solution as [26]. They do object pose estimation using vision, and they also precompute a set of views in simulation and train a similarity function to compare them. However, their challenges are very different from ours. They have to work with occlusions, clutter and different types of backgrounds, but they only do $SO(3)$ (i.e. orientation) estimation, as they rely on a bounding box algorithm to predict the translation of the object. We don't have these particular challenges but we cannot assume the translation is given, therefore we have to estimate full $SE(3)$ pose. Additionally, we have to deal with the locality of tactile, which is not a problem in vision, thus we have to put a lot of attention on dealing with pose distributions, which is central in our work.

4.3 Thesis overview

This work has two main contributions. First, we localize objects without the need of a physical exploration, as most of the computations are done using the 3D model of the object, which we assume is available. This is relevant because it enables to train our neural network models using only simulated data, which is useful in robotics because in general, real data from sensors is much more difficult to obtain. Second, our approach deals with the inherent uncertainty from tactile sensing by working with probability distributions over object pose. These distributions enable us to fuse information coming from multiple independent sensors in an intuitive and mathematically driven way.

In section *Tactile sensor: GelSlim* we describe the tactile sensor used and its output, a high resolution binary image that describes contact/no-contact throughout the surface of the sensor, and which we call *sensed local shape*.

In *Methods* we explain how we go from a *sensed local shape* coming from the sensor to a probability distribution over object pose space. Then we analyze the case where we have multiple sensors touching the object at the same time, where we fuse the distributions from the independent sensors to improve our estimation.

In *Results* we show quantitative and qualitative results of our algorithms for four different objects.

Finally, in *Complementary research and applications* and *Discussion* we talk about present and future lines of research and we conclude the thesis.

5 Tactile sensor: GelSlim

We use an image based tactile sensor called GelSlim [25] which is located in the robot palm or finger. It consists on a gel membrane, a source of light and a camera, as can be seen in Figure 1. When the contact between some object and the sensor happens, the membrane is deformed, changing the normal course of light. The camera is pointing towards the membrane and captures these changes in high-resolution images which we will call *tactile readings*. An example of a *tactile reading* can be seen in Figure 2.

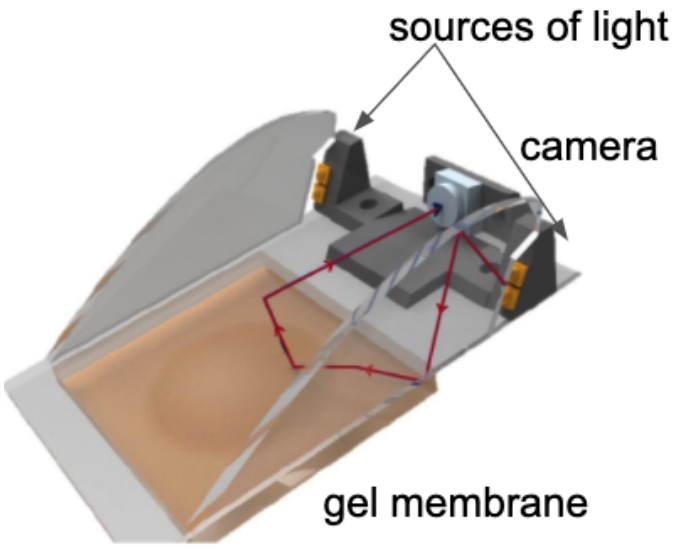

Figure 1: GelSlim diagram. In our particular version of GelSlim, the membrane is green, not brown, and that's why from now on we show it green.

The *tactile reading* coming from the camera, being an RGB image, has a lot of information, but we are only interested in the geometric information of the contact between the membrane and the object. Therefore, instead of directly giving the *tactile reading*, the sensor itself processes it to obtain what we call a *sensed local shape*, a binary image of the same size as the *tactile reading* where each pixel is 0 or 1 depending on whether the membrane is penetrated by the object at that point or not. We call it *local shape* because it depends on the geometry of the object at the place of contact and *sensed* to

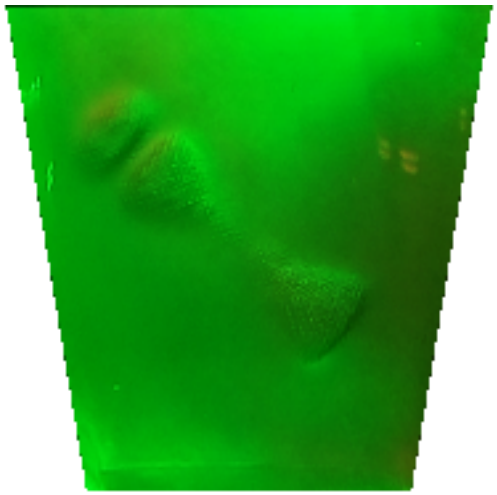

Figure 2: Example of a *tactile reading*

differentiate it from *simulated local shape* which we mention later.

The mapping from *tactile reading* to *sensed local shape* is a convolutional neural network with the architecture and training as described in [23].

The *sensed local shape* obtained from the *tactile reading* from Figure 2, can be seen in Figure 3.

Figure 3: *Sensed local shape* obtained after preprocessing the *tactile reading* from Figure 2

From now on, we consider the mapping from *tactile readings* to *sensed local shapes* to be part of the sensor itself. Therefore, we assume that the output of the sensor is directly a *sensed local shape*.

6 Methods

As mentioned previously, tactile is inherently local. In other words the function that goes from object poses to *sensed local shapes* is not one-to-one, which means that many different poses can result in the same *local shape*. We call these *local shapes* non-unique. Then, this function is non-invertible and by touching the object with a single tactile sensor, one can not rely on predicting a pose of the object w.r.t. the sensor and do it correctly in general. For this reason, we consider that any tactile perception system has to meet two conditions. First, it needs to be able to capture this inherent uncertainty. Second, and very related to the first, it has to to be able to incorporate information from other systems so as to decrease the uncertainty. For this, in section *Probability distribution over object contact pose space* we present how, given an isolated *sensed local shape*, we compute a probability distribution of the object pose. This solution deals with the uncertainty, as the distribution can capture the multimodality coming from non-uniquenesses, and can fuse information from other systems that also compute distributions.

In the same direction, in *Multicontact* we explain how we integrate information coming from multiple tactile sensors touching the object at the same time by exploiting the distributions coming from the independent sensors. This enables to reduce uncertainty and represents a more realistic scenario, as in general multiple fingers or palms of the robot are in contact with the object at the same time when manipulating it.

Finally, in *Point pose estimation and pose refinement* we explain how, given an object pose probability distribution, we can make point pose estimates when necessary and refine these estimates using a pointcloud registration algorithm.

6.1 Probability distribution over object *contact pose space*

In this section we explain how, given a *sensed local shape* of an object, we obtain a probability distribution over object pose w.r.t. the sensor.

As we are doing tactile localization, the object has to be in contact with the membrane of the sensor when we are localizing it, meaning that not all pose of the object w.r.t. to the sensor make sense. Therefore, first we define the object *contact pose space*, which is the space where the probability distribution will live. Then, we explain how we discretize this space by taking structured samples of poses that densely represent it.

Given a *sensed local shape* LS and a pose G from the discretization, we want to compute $P(G|LS)$, the probability of the object being at pose G when LS is observed, which will give us the discretized probability distribution we are looking for.

To compute these probabilities, our approach has two steps. First, for every pose G of the discretization we simulate the *local shape* we would expect to get if the object was at G. We call them *simulated local shapes*. Then, we train a similarity function so that, given a *sensed local shape* and a *simulated local shape* associated to a pose G, returns a real number between 0 and 1. After normalization, we interpret this as $P(G|LS)$.

In other words, given a 3D model of the object and previous to any touch of the object, we discretize object *contact pose space* with a sample of poses, we simulate their respective *simulated local shapes* and we train a *local shape* similarity function. Then, already in real life, when we touch the object and we get a *sensed local shape*, we use the similarity function to assign a probability to all of the sampled poses, which results in the probability distribution we wanted. A diagram of this pipeline can be seen in Figure 4.

Note that if several poses G_1, G_2, ..., G_n of the object have similar *local shapes* when touching any of them, the *sensed local shape* will be relatively similar to all of the *simulated local shapes* from G_i. Therefore, all poses G_i will have a relatively high probability,

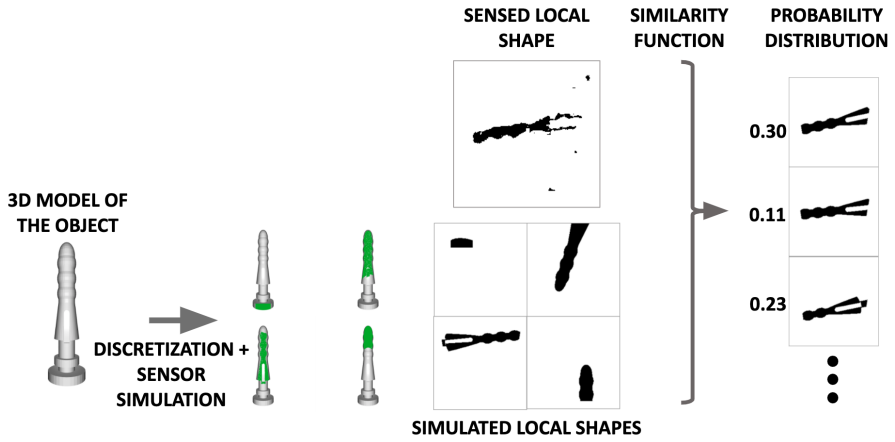

Figure 4: Obtention of the probability distribution over object *contact pose space*. Below, we discretize *contact pose space* and simulate the sensor to obtain the *simulated local shapes* associated to the poses from the discretization. Above, a *sensed local shape* comes from the sensor. We compare it to the *simulated* ones, and, after normalization, we get the probability distribution.

thus capturing the uncertainty. This can seem a problem because we are not able to predict the correct G_i, but as said before, in case of non-uniqueness, capturing uncertainty is the best one can do, with the hope that information coming from other sensors or other perceptions systems will help disambiguate the possible poses or that the algorithms using this probability distribution will also be able to handle this uncertainty in some way.

This process is general for any object, but the discretization, the simulation of *local shapes* and the similarity function training have to be done for every object independently before wanting to localize it. While this can seem like an initial burden, it is important to see that these steps don't need anything else than the 3D model of the object. Therefore, they can be totally automated and computed in just a few hours, enabling from then on to localize the object in real time and avoiding the need of a physical exploration of the object with the sensor, which is relevant given the fact that obtaining real datasets from the sensor is hard and not scalable.

6.1.1 Object *contact pose space*

Given a fixed, reference frame for the object and the sensor, the space of possible object poses w.r.t. the sensor is, by definition, $SE(3)$. However, our probability distribution only makes sense in the space of poses where there is contact between the sensor and the object, because outside this space there is no tactile sensing. As for an arbitrary pose of $SE(3)$ there is not necessarily contact, the space of interest is a subset of $SE(3)$ and we call it *contact pose space*.

The reader can get the intuition behind this *contact pose space* following this three points. First, given an arbitrary $SO(3)$ rotation of the object, there is always a translation (or many) that results in the object being in contact with the sensor. Therefore, orientation is not restricted by itself. Second, the translation inside the plane of the membrane is bounded. In other words, with a fixed orientation, if we translate the object inside that plane, the object will lose contact with the sensor at some point, as can be visualized in Figure 5. Third, given a rotation and a translation inside the plane of the membrane, there is only one translation perpendicular to the membrane that results in contact[1], as can be visualized in Figure 6.

With these three points, we see that the *contact pose space* is bounded and has 5 degrees of freedom, the 6 of SE(3) minus the translation perpendicular to the membrane, which is fixed.

[1]This third point needs two clarifications:

1. It doesn't hold for objects which have a concavity big enough so as to fit the sensor, but as we work with objects of the same magnitude as the sensor, it isn't a problem.
2. The object can penetrate the membrane in different levels. Therefore, it is not strictly true that there is only one translation in the axis perpendicular to the membrane. However, as the maximum penetration is around 2mm, all of the translations resulting in contact are inside a 2mm interval, approximately. We decided to consider only one of them for simplicity.

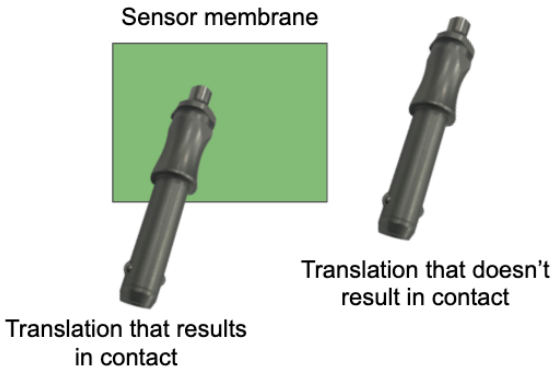

Figure 5: Given an orientation of the object, the translation in the plane of the sensor membrane is bounded.

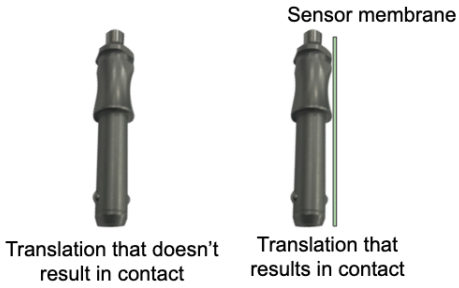

Figure 6: In here we can see a different perspective of the sensor membrane and the contact. Given an orientation and a translation in the plane of the membrane, only one perpendicular translation results in contact. The rest, result in no contact or in a non-feasible penetration of the sensor.

6.1.2 Discretization of *contact pose space*

We discretize *contact pose space* using the algorithm from [27], obtaining, for objects of about 100 mm, a total of 150.000 samples, more or less. The discretization results in a type of lattice structure, which can be visualized in 3D as a grid similar to a rummikub. Further discussion about the discretization falls beyond the scope of this thesis.

6.1.3 Simulation of the sensor

Once we have a discretization of the *contact pose space*, we associate to every sampled pose its *simulated local shape*, i.e. the *local shape* we expect to get from the sensor if the object is touched there.

To do this, given an object pose w.r.t. the sensor, we need an an algorithm that simulates the sensor and returns a binary image with 1 where the object would touch the membrane and 0 where not: the *simulated local shape*.

We use a graphics Python library called *pyrender* [2] which basically lets you put objects and cameras and get their images in a simulated environment.

Given a pose P which determines the relative pose between the object and the sensor, the pose P' of the object w.r.t. the camera from the sensor is also fixed, independently on where we put the reference frames. In the *pyrender* environment we place a camera, with the same intrinsics as the real camera, and the object in the same relative pose P'. We call d the distance, in the real sensor, between the camera and the membrane. Then, in the virtual environment we take a depth image with the camera. Those pixels whose orthogonal depth is smaller than d are set to 1 and the rest to 0, representing where the object would be penetrating the membrane. In other words, we are placing camera and object with relative pose P' and taking a picture with $zfar = d$ [28] and putting to 0 the pixels whose depth is d. A simplified visualization can be seen in Figure 6.

This outputs a binary image that, as said before, we call *simulated local shape* because it simulates the *sensed local shape* we would get from the sensor. We compute this *local shape* for every pose of our discretization of the *contact pose space* and store image and pose together.

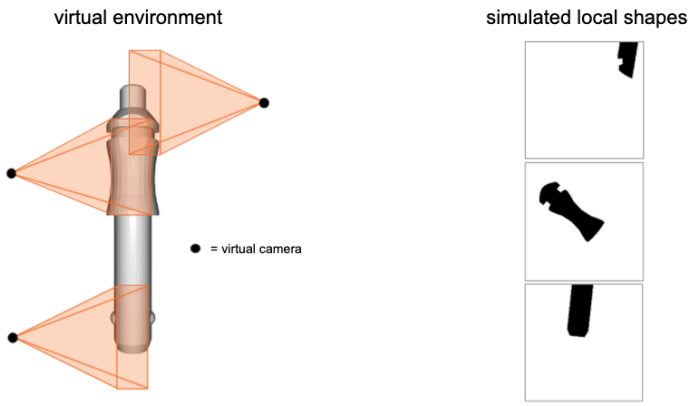

Figure 7: Sensor simulation diagram

6.1.4 Similarity function

Once we have the discretization poses and the associated *simulated local shapes*, we want a similarity function that given a *sensed local shape* and a *simulated* one, outputs a similarity between 0 and 1 that, after normalization, we interprete as the probability of both coming from the same object pose.

Our first approach was to use L1 distance, a.k.a. pixel distance, as an explicit distance metric for image pairs. This metric does the pixelwise difference between both images and outputs the L1 norm of the resulting image. This didn't give good results, as this metric cannot doesn't capture details from the images correctly. It cannot manage to differentiate *local shapes* that have different contact patches located in a similar position in the binary image and it can neither detect similarity when two contact patches look exactly the same but have a slightly different position in the image. An example of this metric failing can be seen in Figure 8.

Some tweaks like sliding windows or registration techniques could be tried here to overcome these limitations, but we feel that this is an inherent problem related to the distance itself.

For these reasons we decided to use a convolutional neural network trained in sim-

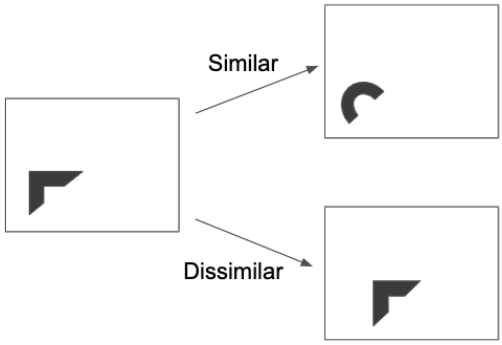

Figure 8: Pixel distance failure case. The image above is more similar to the one on the left than the one below, which is opposite of what we want. This is caused because pixel distance cannot capture shape details, only general contact regions.

ulation with *simulated local shapes* to determine the similarities. The basic architecture of our similarity function can be seen in Figure 9. The idea is that we pass both *local shapes* through the neural network, obtaining a 1000-dimensional vector output for each of the images. Both vectors are compared using cosine similarity, which outputs a real number between -1 and 1, which is taken to 0 if negative and left the same otherwise, obtaining the similarity of the two images. The idea is that the 1000-dimensional vectors are interpreted as *local shape* encodings in a 1000-dimensional space, and the neural network has to learn that similar *local shapes*, i.e. with similar contact patches in similar regions of the image, need to be encoded to similar vectors, so that the cosine similarity is close to 1. Also, it has to learn to put different *local shapes* to relatively different encodings.

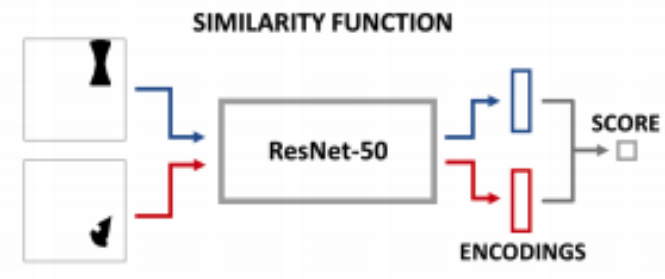

Figure 9: Similarity function

We use a convolutional neural network called Resnet50 [29] without the last two layers and with last activation function *Identity* instead of *ReLU*.

To train the neural network, we create a data set of 100.000 pairs of *simulated local shapes* from the object labeled with a similarity value between 0 and 1. We divide the data in two groups: negative and positive pairs. Negative pairs are two *local shapes* which come from touching the object in two different poses, meaning that the pose distance between both is more than 20 mm, see section *Results* for a detailed explanation of this metric. Negative pairs are labeled with a 0. Positive pairs consist of two *local shapes* that come from very close parts of the object, in particular, less than 5 mm pose distance, and are labeled between 0.9 and 1, inversely proportional to the pose distance between both.

For each pair, both *local shapes* go through the neural network, i.e. get encoded, the cosine similarity of the encodings is computed and compared to the label of the pair. The loss function used is mean square error but, when the label is 0, a similarity of up to 0.3 (and negative values) is accepted with 0 loss. This is done because to ask for complete perpendicularity of encodings is a very strong and difficult to learn condition. This loss can be seen in Equation 1:

$$\text{loss(prediction, label)} = \begin{cases} \text{(label - prediction)}^2 & \text{if label} > 0 \\ \text{(label - prediction)}^2 & \text{if label} = 0 \text{ and prediction} > 0.3 \\ 0 & \text{if label} = 0 \text{ and prediction} \leq 0.3 \end{cases} \tag{1}$$

The model is build, trained and used using *pytorch* [1], optimized by stochastic gradient descent with batch size of 32, learning rate of 0.005, momentum of 0.3 and weight decay of 0.001.

An important thing to note is that, while we train the neural network to encode only

simulated local shapes, when working with the real sensor, it has to encode the *sensed local shapes* as well. We are assuming that both *simulated* and *sensed local shapes* come from a similar distribution and, therefore, the network can generalize enough so that it still encodes the ones coming from the sensor in a correct way. This is one of the main assumptions of this work, and depends on the level of noise of the sensor, but, as we show in section *Results*, this assumption holds true, enabling good localization without the need of previous real data from the sensor.

Once we have the trained neural network, we can already start working with the object. When we touch it with the sensor and get a *sensed local shape* we can compare it to all the *simulated local shapes* from the discretization, normalize the probabilities and get the probability distribution of poses of the object pose w.r.t. the sensor.

6.2 Multicontact

Getting isolated *sensed local shapes* is not a very realistic scenario, as, whenever a robot is manipulating a robot, either with a hand or a simpler gripper, for stability, at least two fingers or palms are touching the object at the same time. That is why in this section we fuse the probability distributions coming from independent sensors to obtain a better distribution.

In this section we show how, having a probability distribution over *contact pose space* enables us to integrate information coming from multiple sensors (multicontat) in a very intuitive and mathematically driven way. One could also want to integrate information coming from vision, for example. We don't analyze this case here but analogous arguments can be done.

As an important note, as we will use probability reasoning in the next sections, we need to formalize the vocabulary around the probability distribution in pose space. If we call *LS* to the *sensed local shape* coming from the sensor, this distribution gives us

$P(G|LS)$ for every pose G of the discretization, i.e. the probability of the object being at pose G when the sensor gives the *sensed local shape LS*.

Below, for simplicity, we describe how we do pose estimation when 2 sensors are used at the same time, but the approach extends directly to any number of sensors. Proof of that can be found in the section *Appendix: Multicontact for N sensors*.

Given two sensors for which we know their poses in some global reference frame, our goal is to estimate the pose of an object also in that frame. We refer to the sensors as sensors 1 and 2, and assume that both sensors are in contact with object, and at the end we comment on what changes when one or both of them don't touch the object. We name the *local shapes* coming from each sensor LS_1 and LS_2.

Note that, as the object is touching both sensors, it has to fall in the intersection of the *contact pose space* of both of them. With this in mind, we are going to compute the probability $P(G|LS_1, LS_2)$ for every G in the discretization w.r.t. sensor 1. This will give us a probability distribution of the object pose with respect to sensor 1, but with a simple change of reference it can be moved to sensor 2.

We use Bayes Theorem [30] to expand this probability:

$$P(G|LS_1, LS_2) = \frac{P(G, LS_1, LS_2)}{P(LS_1, LS_2)}. \tag{2}$$

And again:

$$P(G|LS_1, LS_2) = \frac{P(LS_2|LS_1, G)P(G, LS_1)}{P(LS_1, LS_2)}. \tag{3}$$

And one last time:

$$P(G|LS_1, LS_2) = \frac{P(LS_2|LS_1, G)P(LS_1|G)P(G)}{P(LS_1, LS_2)}. \tag{4}$$

Given the object pose G, we can assume that LS_1 doesn't affect the likelihood of LS_2 and thus $P(LS_2|LS_1, G) = P(LS_2|G)$. Making this assumption is equivalent to saying that $P(LS_1, LS_2|G) = P(LS_1|G) \cdot P(LS_2|G)$ which holds if you don't take into account that noise can correlate in both measurements due, for example, to illumination, but obviously we can't take this into account. Therefore, we can rewrite the expression as:

$$P(G|LS_1, LS_2) = \frac{P(LS_2|G)P(LS_1|G)P(G)}{P(LS_1, LS_2)}. \tag{5}$$

Because $P(LS_1, LS_2)$ is independent of G and, therefore, a constant for all G of the discretization we can obviate this term and normalize at the end. We then have that:

$$P(G|LS_1, LS_2) \propto P(LS_2|G)P(LS_1|G)P(G). \tag{6}$$

Where the $\propto$ symbols means that both terms are proportional (equal except for a multiplicative constant independent on G).

We observe that $P(G)$ is our prior knowledge on the likelihood of this particular pose G of the object that could come from other perception systems or from a tactile tracker for example. However, in this case we assume all object configurations are equally likely and $P(G)$ is a constant, but one could simply add this term in a case where this isn't true. We end up with:

$$P(G|LS_1, LS_2) \propto P(LS_1|G)P(LS_2|G). \tag{7}$$

Now, for the term $P(LS_1|G)$, we see that, applying Bayes Theorem:

$$P(LS_1|G) = \frac{P(G|LS_1) \cdot P(G)}{P(LS_1)}. \tag{8}$$

Using equivalent arguments as before:

$$P(LS_1|G) \propto P(G|LS_1). \tag{9}$$

And the same can be done with the term $P(LS_2|G)$ from Equation 7 and we get

$$P(G|LS_1, LS_2) \propto P(G|LS_1)P(G|LS_2). \tag{10}$$

Now, as G is a pose from the discretization with respect to sensor 1, $P(G|LS_1)$ is what we calculated in the previous section, where we computed the probability distribution and, thus, this term can be computed directly comparing LS_1 to the *simulated local shape* associated to G with the neural network.

The second term from the right-hand side in Equation 7 is trickier. G is an object pose w.r.t. to sensor 1 and belongs to its discretization. However, when changing G to the reference frame of sensor 2, which we will cal *G'*, two things happen. First, *G'* can fall outside the *contact pose space* of sensor 2. This means that we wouldn't expect contact between the object and sensor 2, which is wrong as we are assuming both sensors touch the object. Therefore, if this happens $P(G|LS_2)$ is 0. Second, even if *G'* falls inside *contact pose space* of sensor 2, *G'* isn't necessarily an element of the discretization, and, thus, we don't have a *simulated local shape* precomputed for it. We could simulate this *local shape* but this would be very inefficient, as we would have to do this for every G. A more clever way to do it is that, given that *G'* falls in the *contact pose space* of sensor 2, given that we have a dense discretization of this space, there must be a pose *G"* of the discretization that is very close to *G'*. This *G"* is easy to find due to the lattice structure of our sampling. Therefore, we can take the *simulated local shape* associated to *G"*, compare it to LS_2 and get an approximation of $P(G|LS_2)$.

We can then compute all these terms for every pose G of the discretization with respect to sensor 1, compute $P(G|LS_1, LS_2)$ and finally obtain the distribution we were looking for.

Now we analyze what happens when one or both sensors are not touching the object, which can be detected by comparing LS_i to an empty *local shape*.

If both sensors are not touching the object, we can only say that the object pose is in SE(3) but not in the *contact pose space* of any of the sensors, which, of course is not a very useful information.

When it is only one of the two sensors that is not touching the object, we can rename the sensors so that it is sensor 2 that is not in contact, while 1 is. Then, we can follow an analogous argument to the one above but, when computing $P(G|LS_2)$, if *G'* falls inside the *contact pose space* of sensor 2, this term is 0, and 1 otherwise, because the only information we have is that the object is not in this space, as it is not touching the sensor.

6.3 Point pose estimation and pose refinement

We have showed how we obtain probability distributions over *contact pose space* using tactile information. It is probable that at some point in our robotic system we will want to have a point estimation instead of a distribution, i.e. get an estimated pose instead of the full set with their associated probabilities. In that case we return the mode of the probability distribution, the pose of the discretization with a higher probability.

Related to this, it is important to note that, as we always work over a discretized space, the accuracy of our estimation is bounded by the granularity of our discretization. To overcome this, we refine the point estimation with *FilterReg* [31], a pointcloud registration algorithm. This algorithm takes the *sensed local shape* from one of the sensors and the *simulated local shape* associated to our point estimate, which should look relatively similar, and tries to find a local transformation that transforms the *simulated* into the *sensed*. Then we can compose the pose of our point estimate and this local transformation and get a new pose, that doesn't belong to the discretization, and is

more accurate. We call this process pose refinement, and helps us get more continuous estimation, despite working with a discretization of the pose space.

7 Results

In this section we show the results of the methods from the previous section. As mentioned throughout the thesis, one of the main goals of this work is to compute and manage probability distributions coming from tactile. However, there isn't a ground truth of how these distributions should look like and we cannot compare our whole distributions to anything and define a metric of how good they are. To overcome this, and get a feel of whether the uncertainty is captured correctly or not, we use two different metrics. The first metric consists on taking the mode of the distribution, compare it to the true pose of the object and get the error of the estimation (we explain later in this section how we obtain the true pose and how we compute the pose error). From now on we call this metric *best* error. The second metric, which we call *best10*, consists on taking the 10 more probable poses from the distribution, compare them to the true pose and take the one with less error. Obviously, this is "cheating", as in real life we don't know the true pose, which is exactly what we want to find. The goal when using it is to show that, even when the *best* error is high due to non-uniquenesses, the uncertainty is captured by also giving high probabilities to other modes of the distribution, which contain the true pose.

As mentioned above, to obtain quantitative errors of the localization, the true pose of the object pose w.r.t. the sensor, or ground truth, is necessary. For that purpose we use a robotic platform that moves in the three axes and rotates perpendicular to the vertical axis. With the sensor fixed, and a 3D printed copy of the object attached to this platform, we can perform touches of the object at known poses and therefore obtain the ground truth we need. It is important to note that, due to the limitations of the collection of ground truth data, all quantitative results are obtained considering general translations but only rotations around the axis perpendicular to the membrane of the sensor.

When predicting a pose of the object and comparing it to the true pose, the error comes from orientation and translation errors. To account for both, we use a distance metric between different poses called Average Distance of Model Points (ADD) [32] which we will call pose error or pose distance from now on. This distance consists on, given the two object poses, compute the average distance of corresponding points when the object is at each of both poses. This metric depends on the object size, making it difficult to compare localization errors across different objects. Therefore, for better comparison between objects, we use a normalization of this metric we call normalized pose distance, which consists on dividing the pose distance by the expected error when sampling two random object poses from *contact pose space*. A visualization of this error metric for the *damping pin* can be seen in Figure 10.

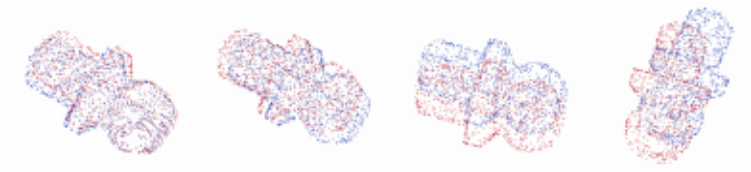

Figure 10: Examples of normalized pose distances in object *damping pin*. The first one is 0.05, corresponding to the average distance between neighbour poses in the discretization. The others are, 0.10, 0.16 and 0.32 respectively.

The objects we used for the quantitative results are *pin*, *damping pin*, *head* and *elbow pipe* and can be seen in Figure 11.

Figure 11: Objects used for quantitative results. In this order: *pin*, *damping pin*, *head* and *elbow pipe*. Their size is between 100 mm and 150 mm.

7.1 Probability distribution evaluation

When getting a single *sensed local shape*, we compute both the *best* and *best10* metric. Also, we use registration to refine the *best* estimate. For each object we show the results in Table 1 and Figure 12.

	Pin	Damping pin	Head	Elbow pipe
best	4.9 mm / 0.11	5.6 mm / 0.18	11.8 mm / 0.28	38.5 mm / 0.59
refined *best*	4 mm/ 0.09	2.8 mm / 0.09	7.4 mm / 0.18	36.4 mm / 0.56
best10	2.3 mm / 0.05	3.1 mm / 0.10	6.5 mm / 0.15	8.8 mm / 0.14

Table 1: Median pose error (mm) and median normalized pose error (unitless) for *best* before and after doing refinement and *best10*. We use median because it is more informative than the mean, as can be seen in Figure 12.

We can observe that for all objects and metrics, the error distributions have the main mode centered a little above the red line (granularity of the discretization) and that there is another mode which is around a normalized error of 1. This is due to the fact that, as tactile is local, when localizing an object, our system either localizes it correctly (producing very little error) or not (producing a random error). Therefore the error is almost binary, with little gradient. That's why we use the median to present the results, as the mean is very affected by this random error, being less meaningful and more unstable.

In Table 1 can see that the *best* is low for *pin* and *damping pin*, but it is much higher for *elbow pipe*. We found this to be caused by the fact that this object has multiple non-uniquenesses and because it has many edgy small features which are difficult to capture by the sensor, adding a lot of noise to the *sensed local shape*. This is more or less what we expected, due to the locality of tactile sensing. However, we can see in the plots and the table that *best10* is much better. This means that, even when the mode of the pose probability distribution has a lot of error, there are other secondary modes that contain the true pose of the object, thus capturing the uncertainty. We can also see that when refining, using registration, we improve the accuracy of the localization in

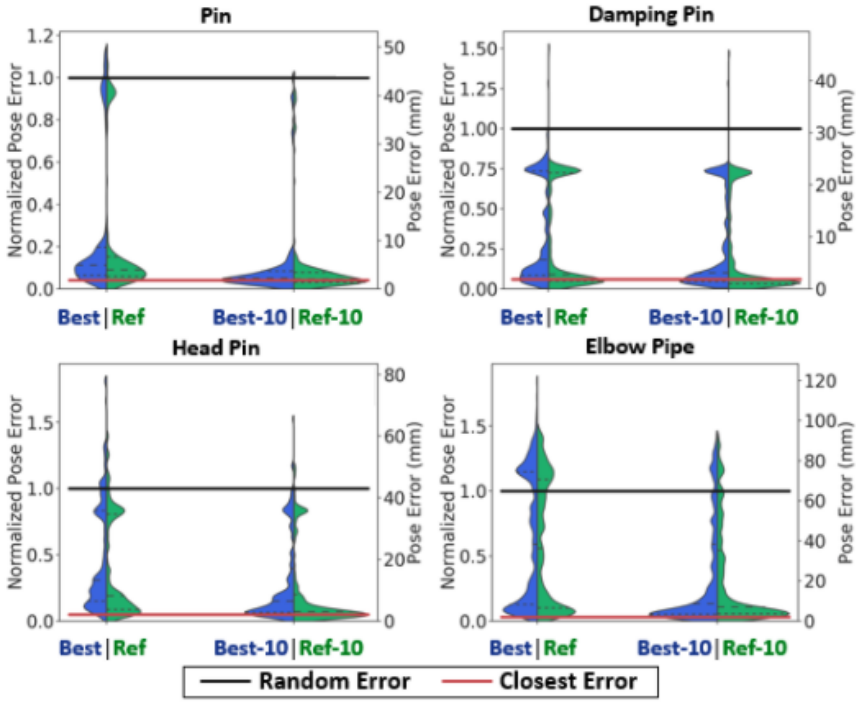

Figure 12: Distribution of the pose error for the 4 objects with the same metrics as in Table 1, where *ref* means refinement has been made, and with the addition of a fourth one, ref-10, which is *best-10* with refinement. In the vertical axis we have the pose error, and in the horizontal, the number of examples that resulted in that error. In other words, it is a continuous histogram of the errors when we run our system on a set of examples. The horizontal black line is the expected error when selecting two random poses from the *contact pose space*. The red line represents the granularity of the discretization (it is the mean distance between neighbour poses in the discretization).

all cases.

7.2 Multicontact

In this section we show results for multicontact. Because of the need of ground truth and the difficulty to use multiple sensors in our set up with the robotic platform, instead of touching directly the object with multiple sensors, we perform sequential controlled touches to different parts of the object and compute our estimates as if all the contacts had happened at the same time. This enables us to use an arbitrary number of sensors without the complication of changing all the setup and acquiring more sensors.

In Figure 13 we show these results.

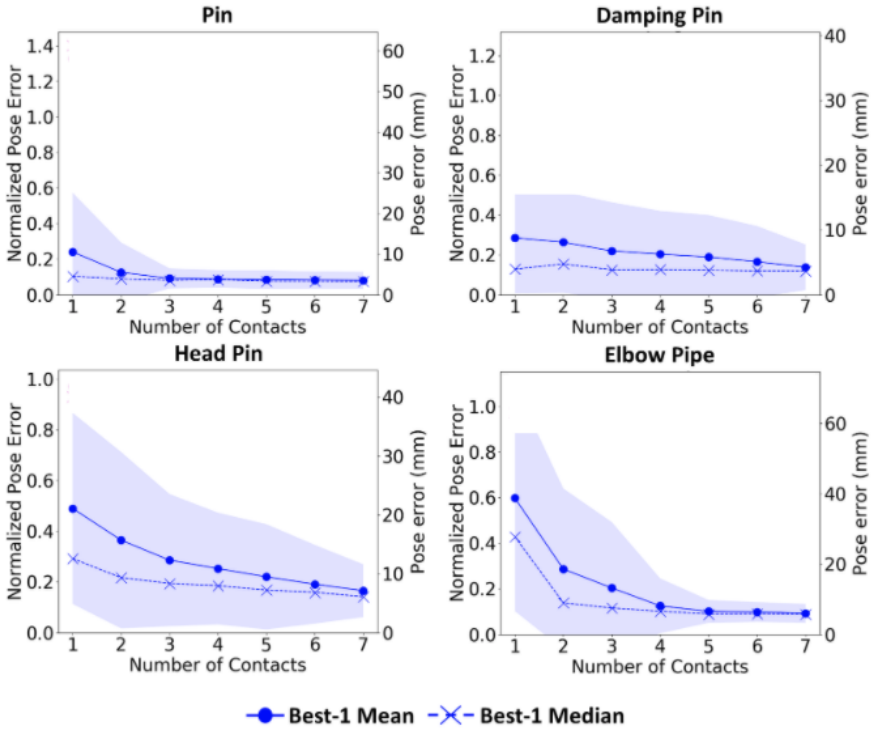

Figure 13: *Best* error of the multicontact localization depending on the number of contacts. Dotted, we see the median error, solid the mean and the area is the standard deviation.

We can see that for all objects, all metrics go down when the number of sensors touching the object increases, with the more obvious improvement being in *elbow pipe*, the more complicated object as explained before. This is what expected, but also goes in the direction of showing that our distributions are well conditioned.

7.3 Qualitative results

Attached to this thesis presentation is a video showing some qualitative results.

In the first part of the video, we show results for localization when there is only one sensor touching the sensor. In Figure 14 we explain what is seen in this first part of the video.

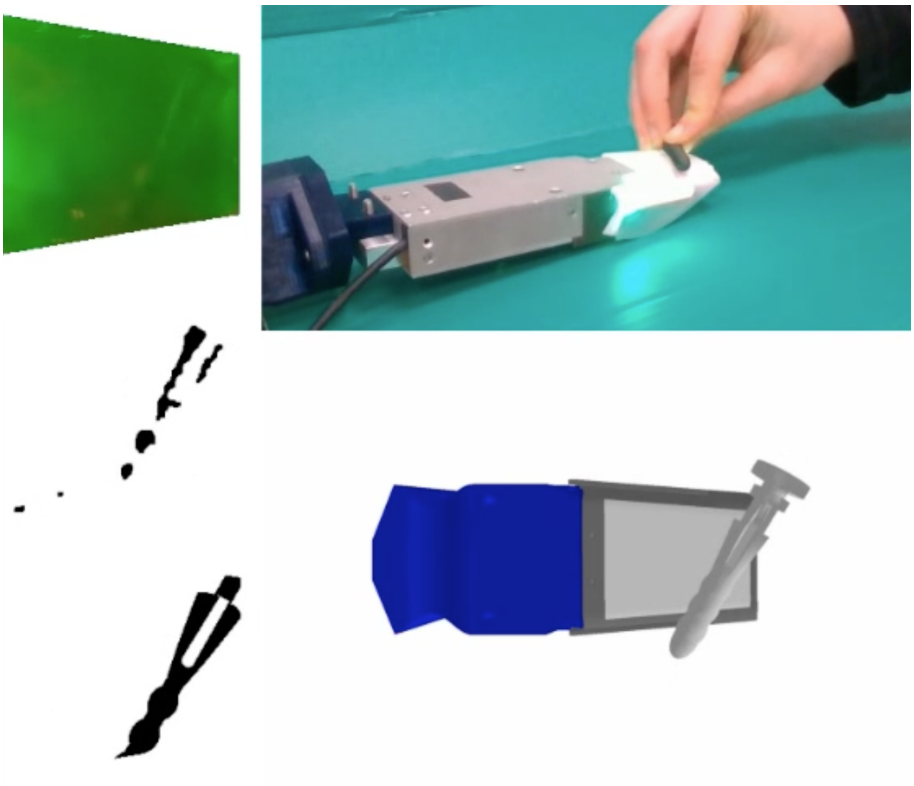

Figure 14: Screenshot of the first part of the video attached. At the top, on the right, we see the object making contact with the membrane of the sensor. On the left, the *tactile reading*. Below, the *sensed local shape*. At the bottom, on the left, the *simulated local shape* associated to the most probable pose. Finally, on the right, a representation of our pose estimation in a simulator called *pybullet* [33]. It represents the sensor and the object in the relative pose specified by the most probable pose of the object.

In the second part, we show results for multicontact, as explained in Figure 15.

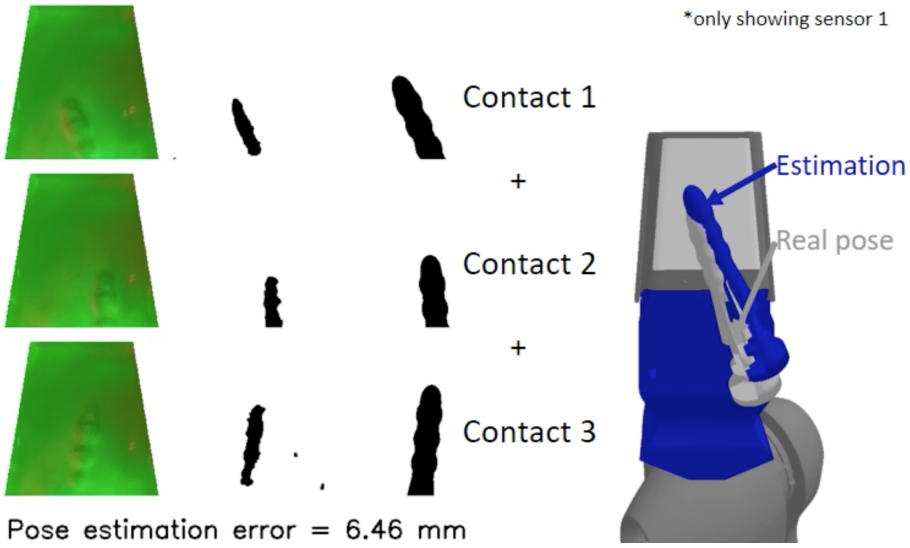

Figure 15: Screenshot of the second part of the video attached, where results of multicontact are shown. The contacts are added one by one, to show how, in general, localization improves. On the left we have the contacts, with *tactile reading*, *sensed local shape* and *simulated local shape* associated to the most probable pose, in this order. The first contact is at the top, and the last one at the bottom. On the right, we represent in *pybullet* the poses of the ground truth and the estimation w.r.t the sensor.

8 Complementary research and applications

In this section we present other lines of research that have been pursued or are being pursued at the moment, including a filtering algorithm for tactile, an approach to tactile active perception and an application of the work explained in this thesis in a larger system.

8.1 Filtering

In many real scenarios, we don't get isolated *sensed local shapes*, but a sequence of them coming from the sensor over time, where the object is in continuous contact with the sensor. The object can both be fixed with respect to the sensor or move due to external forces, but the information coming from this sequence of *local shapes* can help us filter out noise, and shrink the probability distribution by eliminating incorrect modes. We can exploit the fact of working with probability distributions to implement a straight forward filter Bayes filter[30].

When we get a sequence of *sensed local shapes* LS_1, LS_2, ... , LS_t over time, instead of computing $P(G|LS_t)$ for every time t and every pose G of the discretization, we can use all the information available and compute $P(G|LS_1, LS_2, ..., LS_t)$, i.e. the probability of the object being at G at time t given all the *local shapes* that have come from the sensor. As a comment, note than in general we don't have any explicit information of the movement of the object w.r.t. to the sensor, therefore we cannot use anything other than the *local shapes* coming from the sensor. We also have to assume that the movement of the object is small between consecutive local shapes, thus having some continuity in our observations.

To compute this new distribution $P(G|LS_1, LS_2, ..., LS_t)$, which lives in the same object *contact pose space*, we use a Bayes filter [30]. As a motion model we use a

uniform distribution centered in 0 with range 8 mm in pose distance. As observation model at time t we use the probability distribution $P(G|LS_t)$ computed as explained in section *Methods*.

Our filter operates over the discretization of object *contact pose space*, meaning that, as before, the distribution we get from the filter at each time step, is a discretized distribution. To make the problem more tractable we further simplify the representation of this distribution by representing it only with the 50 more probable poses, meaning that all the rest are set to 0. This reduces the capacity of the discretization to represent arbitrary distributions. However, it only has a significant negative impact if there is a lot of multimodality in the distribution needed to represent. This is one of the main limitations of this tracking approach, but it is necessary to make it work in real time.

In conclusion, at each time step, the tracker:

1. Takes the 50 most likely poses from the previous iteration. Only in the first iteration all poses from the discretization are considered.

2. Applies the motion model to these poses. This re-assigns probabilities along the discretized poses by giving higher probabilities to poses that are close to poses which have a high probability.

3. Applies the observation model to all poses. This means every probability associated to a pose G is multiplied by $P(G|LS_t)$ at time t.

We compute quantitative results by evaluating our filtering approach in 4 randomly generated trajectories for the *pin* object. We collected the data for these trajectories using the same robotic setup. In 3 of them the tracker outperforms the estimation when considering the *local shapes* independent. In the fourth trajectory, the filter was much worse. When looking into the reason we saw that this trajectory had a long sequence of non-unique *local shapes*. The filter was keeping track of the two modes coming from

the non-uniqueness but gave higher probability to the incorrect one. Longer trajectories that can break the non-uniqueness or extra constrains coming from other contacts or perception systems would help the filter disambiguate these situations.

8.2 Active tactile perception

Throughout this thesis we have assumed that the perception system explained is a passive actor inside the whole robotic system performing some task. In other words, some planner and control algorithms decide for robotic actions, and the tactile localization only waits for data to come from the sensors and, when it comes, it tries to do the best localization with the data available. In this section we change this assumption and we try to give a hint on how we are trying to make this perception system and active agent which can decide to do actions with the sensors, with the goal of improving the localization. This idea of deciding to take actions so as to maximize the information from a perception system is usually known as active perception [34]. This is research we are doing at the present, therefore we still have many open questions and no meaningful quantitative results.

Imagine a parallel jaw gripper, with tactile sensors at both fingers, which has a grasp of the object. In our example, due to the locality of tactile the probability distribution obtained from the *sensed local shapes* from the sensors, is bimodal, meaning that two different regions of *contact pose space* have a high probability. Imagine also the robot has a third independent tactile sensor. Knowing that the localization is bimodal, and that for this reason maybe the robot can not manipulate the object correctly, the robot could perform random touches of the object with the third sensor, and with the added information, try to disambiguate between the two possibilities. This is inefficient because the touches performed on the object don't necessarily disambiguate the two modes, as they could be on regions that don't add any useful constraints. On the other hand, the perception system is fully aware of the two possibilities of where the object could be

and, therefore, could decide to perform some actions with the third sensor that obtain useful constraints which will enable the localization to be unimodal.

For visualization, imagine the grasp is done on a pencil, as can be seen in Figure 16.

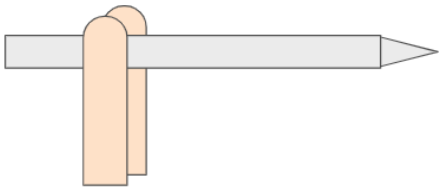

Figure 16: Parallel jaw gripper grasping a pencil in a non-unique region

Due to the non-unique grasp that is performed, the system cannot know the pose of the object. For example, the object could also be in the red pose seen in Figure 17.

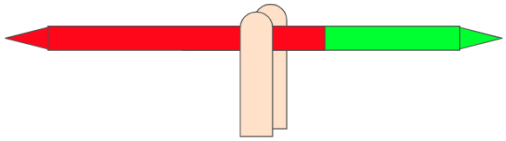

Figure 17: Two possible poses coming from the non-unique grasp

If a touch is performed in some region of the intersection between the red and the green poses, it will not give any more information. However, if the touch is performed with the goal of localizing the tip of the pencil, by approaching the pencil from the left or from the right, it will be easy to disambiguate between the two options. Our goal is to define a way to, given a probability distribution over object pose space, optimize the action that will give more information, in expected value.

To optimize for actions, we need a function that, given an action and the probability distribution, returns a real number which represents the expected amount of information the system will get from performing the action. We call it information function.

We are in the present working on how to define this function. For now we have

defined it for the setup when, instead of using a third sensor, we use a finger to only add kinematic constraints. In other words, instead of optimizing for actions which will give us a *local shape* from a third sensor, we optimize for actions which tell us if the object is occupying certain region of space or not. For example, by doing the strategy of approaching from the left or right from the pencil, we would be able to tell where the pencil finishes, but not where the tip is. We have found that, even if the constraints are much softer than if done with a tactile sensor, this enables us to improve localization in most of the objects and most of the cases.

Our basic approach to this information function is that, given an action, we compute the expected number of poses from the discretization that are going to be discarded (or set to probability 0) if that action is performed, given the probability distribution that we have from the actual touch from the gripper.

8.3 Kitting

In the present, in most automated industry scenarios where robots have to manipulate objects, both the initial and final state of the object are well defined. For example, the robot takes an object from a known pose and puts it somewhere else, in a predetermined way. However, robots have a hard time getting from an unstructured environment to a structured one, for example, from a bin full of pencils to small ordered boxes of pencils. This is due to many reasons, one of them, object localization.

We have been working in a project we call Kitting, where a bin of objects of the same type is given, and the robot has to take them one by one and place them precisely somewhere else. This system has many challenges. First, the objects have to be grasped in a stable manner. Also, sometimes, regrasping the object using a second hand is needed so as to be able to place the object. Finally, as two grippers (four fingers) intervene, there are a lot of occlusions, which prevents vision of localizing the object when it is being manipulated. Therefore, we have integrated our tactile localization

system, showing a very clear application of the work implemented in this thesis. For now, the full system has only been tested in simulation.

9 Discussion

In this thesis we have shown a tactile localization system which is able to capture the uncertainty inherent to tactile sensing, and doesn't need a previous physical exploration of the object with the sensor, assuming a model of the object is available.

Given a *sensed local shape* we are able to obtain a probability distribution over object pose space which captures this uncertainty. We extend the approach to a setup where multiple sensors are touching the object at the same time.

We have presented quantitative results for both setups. The conclusion we extract from these results is that, while point estimates work well, they fall short because of noise and non-uniquenesses. However, thanks to working with probability distributions, the approach deals correctly with uncertainty, which is shown with the *best10* metric. The multicontact also validates our approach by showing that, as we are not making point estimates but computing distributions from every sensor, we are able to fuse all this information to get a better distribution which then results in even better point estimates. It also shows that our distributions are well conditioned for this problem.

The main assumption, besides having a model of the object, has been that the similarity function, trained in simulation, correctly generalizes to *sensed local shapes*. This only has been a problem in one of the objects, *elbow pipe*, due to the noise in the sensor, but even in this case, the probability distribution has been able to capture this, which is shown with the *best10* metric and the multicontact. To decrease the gap between *simulated* and *sensed local shapes* one could explore the idea of adding noise to our simulator, in a similar way as in the real sensor.

The good results of this assumption are one of the key contributions of this work. We don't need any real data from the sensor to train the neural networks or create the discretizations, which makes this system very scalable, and enables it to easily change

parts of it, for example, the sensor simulator, only requiring a new training of the neural networks, which only takes a few hours.

Other future lines of research include active tactile perception, filtering and kitting.

Our conclusion is that we have accomplished the goal of developing a tactile perception system which is able to deal with uncertainty and add constraints coming from other sensors or perceptions systems, which was the main goal of this work. We also have been able to better understand the problems inherent to tactile, specially its locality, but also its noise and how to simulate it. The usage of a convolutional neural network has proven of great value due to its ability to generalize the idea of similarity that we were imposing, but it also has been one of the parts where we have had to put more effort so as to obtain the desired results. The mathematical derivation in section *Multicontact* has shown that if our perception distributions are well conditioned, an intuitive and efficient way of integrating data can be used.

We also think that, while tactile still has a long way until being robust and easily integrated into large systems in industry, for example, this work sets a first step on a framework on how to extract useful information from tactile data.

10 Appendix: Multicontact for N sensors

In this section we extend the proof in section *Multicontact* to an arbitrary number of sensors N. We are going to follow a similar argument to the one explained, but we require mathematical induction.

In the same way as for two sensors, we want to prove that, for every pose G of the discretization of *contact pose space* w.r.t. sensor 1:

$$P(G|LS_1, LS_2, ..., LS_n) \propto P(G|LS_1)P(G|LS_2)...P(G|LS_n). \tag{11}$$

We will prove it by mathematical induction.

For n $= 1$ the statement is true because we have the same term at both sides.

We assume the statement is true for n and we want to show that this implies that it is true for $n+1$

Applying Bayes Theorem one time we get that:

$$P(G|LS_1, ..., LS_n, LS_{n+1}) = \frac{P(LS_1, ..., LS_n, LS_{n+1}, G)}{P(LS_1, ..., LS_n, LS_{n+1})}. \tag{12}$$

And applying it again:

$$P(G|LS_1, ..., LS_n, LS_{n+1}) = \frac{P(LS_{n+1}|LS_1, ..., LS_n, G)P(LS_1, ..., LS_n, G)}{P(LS_1, ..., LS_n, LS_{n+1})}. \tag{13}$$

Where:

$$P(LS_{n+1}|LS_1, ..., LS_n, G) = P(LS_{n+1}|G), \tag{14}$$

because LS_i are independent given G. Also, as $P(LS_1, LS_2, ..., LS_n, LS_{n+1})$ is a con-

stant independent of G:

$$P(G|LS_1, LS_2, ..., LS_n, LS_{n+1}) \propto P(LS_{n+1}|G)P(LS_1, LS_2, ..., LS_n, G). \tag{15}$$

And applying again the Bayes Theorem:

$$P(G|LS_1, LS_2, ..., LS_n, LS_{n+1}) \propto P(LS_{n+1}|G)P(G|LS_1, LS_2, ..., LS_n)P(LS_1, LS_2, ..., LS_n). \tag{16}$$

As explained in section *Multicontact*:

$$P(LS_{n+1}|G) \propto P(G|LS_{n+1}). \tag{17}$$

Therefore:

$$P(G|LS_1, LS_2, ..., LS_n, LS_{n+1}) \propto P(G|LS_{n+1})P(G|LS_1, LS_2, ..., LS_n)P(LS_1, LS_2, ..., LS_n). \tag{18}$$

And, applying the induction hypothesis:

$$P(G_g|LS_1, LS_2, ..., LS_n, LS_{n+1}) \propto P(G|LS_{n+1})P(G|LS_n)...P(G_|LS_1). \tag{19}$$

And by the principle of mathematical induction, the initial statement is proven.

To compute the terms $P(LS_i|G_g)$ for $i > 1$ we do the same as explained in the paper for sensor 2.

References

[1] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[2] Pyrender. `https://github.com/mmatl/pyrender`.

[3] Cristiano Premebida, Rares Ambrus, and Zoltan Marton. *Intelligent Robotic Perception Systems*. IntechOpen, 2018.

[4] Monika Schaeffer and Allison Okamura. Methods for intelligent localization and mapping during haptic exploration. In *2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance*, volume 4, pages 3438–3445, 2003.

[5] Anna Petrovskaya, Oussama Khatib, Sebastian Thrun, and Andrew Y Ng. Bayesian estimation for autonomous object manipulation based on tactile sensors. In *Proceedings 2006 IEEE International Conference on Robotics and Automation*, pages 707–714, 2006.

[6] Craig Corcoran. Tracking object pose and shape during robot manipulation based on tactile information. In *2010 IEEE International Conference on Robotics and Automation*, volume 2, 2010.

[7] Anna Petrovskaya and Oussama Khatib. Global localization of objects via touch. In *2011 IEEE Transactions on Robotics*, volume 27, pages 569–585, 2011.

[8] Maxime Chalon, Jens Reinecke, and Martin Pfanne. Online in-hand object localization. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2977–2984, 2013.

[9] Joao Bimbo, Shan Luo, Kaspar Althoefe, and Hongbin Liu. In-hand object pose estimation using covariance-based tactile to geometry matching. In *2016 IEEE Robotics and Automation Letters*, volume 1, pages 570–577, 2016.

[10] Brad Saund, Shiyuan Chen, and Reid Simmons. Touch based localization of parts for high precision manufacturing. In *2017 IEEE International Conference on Robotics and Automation*, pages 378–385, 2017.

[11] Shervin Javdani, Matthew Klingensmith, J Andrew Bagnell, Nancy S Pollard, and Siddhartha S Srinivasa. Efficient touch based localization through submodularity. In *2013 IEEE International Conference on Robotics and Automation*, pages 1828–1835, 2013.

[12] Yevgen Chebotar, Oliver Kroemer, and Jan Peters. Learning robot tactile sensing for object manipulation. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3368–3375, 2014.

[13] Joao Bimbo, Lakmal D Seneviratne, Kaspar Althoefer, and Hongbin Liu. Combining touch and vision for the estimation of an object's pose during manipulation. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4021–4026, 2013.

[14] Joao Bimbo, Petar Kormushev, Kaspar Althoefer, and Hongbin Liu. Global estimation of an object's pose using tactile sensing. In *Advanced Robotics*, volume 29, pages 363–374, 2015.

[15] Marten Björkman, Yasemin Bekiroglu, Virigile Högman, and Danica Kragic. Enhancing visual perception of shape through tactile glances. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3180–3186, 2013.

[16] Peter Allen, Andrew Miller, Pau Oh, and Brian Leibowitz. Integration of vision, force and tactile sensing for grasping. In *International Journal of Intelligent Machines and Robotics*, volume 4, pages 129–149, 1999.

[17] Jarmo Ilonen, Jeannette Bohg, and Ville Kyrki. Three-dimensional object reconstruction of symmetric objects by fusing visual and tactile sensing. In *The International Journal of Robotics Research*, volume 33, pages 321–341, 2014.

[18] Pietro Falco, Shuang Lu, Andrea Cirillo, Ciro Natale, Salvatore Pirozzi, and Dongheui Lee. Cross-modal visuo-tactile object recognition using robotic active exploration. In *2017 IEEE International Conference on Robotics and Automation*, pages 5273–5280, 2017.

[19] Kuan-Ting Yu and Alberto Rodriguez. Realtime state estimation with tactile and visual sensing. application to planar manipulation. In *2018 IEEE International Conference on Robotics and Automation*, pages 7778–7785, 2018.

[20] Robert Platt, Frank Permenter, and Joseph Pfeiffer. Using bayesian filtering to localize flexible materials during manipulation. In *IEEE Transactions on Robotics*, volume 27, pages 586–598, 2011.

[21] Zachary Pezzementi, Caitlin Reyda, and Gregory D Hager. Object mapping, recognition, and localization from tactile geometry. In *2011 IEEE International Conference on Robotics and Automation*, pages 5942–5948, 2011.

[22] Shan Luo, Wenxuan Mou, Kaspar Althoefer, and Hongbin Liu. Localizing the object contact through matching tactile features with visual map. In *2015 IEEE International Conference on Robotics and Automation*, 2017.

[23] Maria Bauza, Oleguer Canal, and Alberto Rodriguez. Tactile mapping and localization from high-resolution tactile imprints. *arXiv preprint arXiv:1904.10944*, 2019.

[24] Rui Li, Robert Platt, Wenzhen Yuan, Andreas ten Pas, Nathan Roscup, Mandayam A. Srinivasan, and Edward Adelson. Localization and manipulation of small parts using gelsight tactile sensing. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3988–3993, 2014.

[25] Elliott Donlon, Siyuan Dong, Melody Liu, Jianhua Li, Edward Adelson, and Alberto Rodriguez. Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1927–1934, 2018.

[26] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *Proceedings of the European Conference on Computer Vision*, pages 699–715, 2018.

[27] Julie Mitchell. Sampling rotation groups by successive orthognal images. In *SIAM Journal on Scientific Computing*, volume 30, pages 525–547, 2008.

[28] Opengl: Viewing and transformations. `https://www.khronos.org/opengl/wiki/Viewing_and_Transformations`. Accessed 15-04-2020.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

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

[31] Wei Gao and Russ Tedrake. Filterreg: Robust and efficient probabilistic point-set registration using gaussian filter and twist parameterization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11095–11104, 2019.

[32] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *IEEE International Conference on Computer Vision*, pages 858–865, 2011.

[33] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. *GitHub repository*, 2016.

[34] R. Bajcsy. Active perception. In *1988 Proceedings of the IEEE*, volume 76, pages 966–1005, 1988.