Object re-grasping with Dexterous Robotic Hands

Ana Carolina Grilo da Silva Januário
Instituto Superior Técnico, Lisboa, Portugal
Email: ana.januario@tecnico.ulisboa.pt

***Abstract*—This thesis addresses in-hand manipulation for dexterous robotic hands, in particular with the ability of changing the grasp type on an object while holding it. This is an important skill, since for the execution of many tasks it is important to change the way the object is grasped in the hand. For instance, when we grasp a pen to write, typically we have to manipulate it in the hand to place it in the right pose for writing. This is a very challenging problem because moving the fingers on the object may change its stability properties.**

We address this problem via a finger gaiting approach. Given an initial grasp, move the fingers in sequence, from one location to the other in the surface of the object, to achieve a desired final position. This is a challenging problem because removing one finger from the surface of the object may change its stability properties and make the object fall. Therefore, we develop a planning algorithm that chooses the best sequence of finger gaits to maintain stability during the re-grasping steps.

We have implemented the algorithm using the pipeline ROS-Gazebo-MoveIt. This pipeline already includes a generic robot simulation environment, (self-)collision detection, and motion planning. Thus, the developed algorithms could be adapted to other robots.

Results are presented in simulation for the iCub robot hand. We assume that an appropriate reaching method has put the robot hand close to an object in a convenient pose to grasp. Then, we do the motion planning, with collision avoidance, for the fingers to accomplish the first grasp. Next, we acquire and process the contact information from the tactile sensors so we can compute the stability properties of the grasp (force closure test). Then, we start testing which contacts can be removed without loosing stability. Finally, we proceed to the fingers repositioning, keeping a force closure grasp, until the final grasp is reached. We evaluate our approach in terms of the stability properties along the re-grasping sequence and robustness to object small shape variations.

***Index Terms*—re-grasp, finger gaiting, force closure, dexterous hand, ROS, Gazebo.**

I. Introduction

Robotics is an expanding area of knowledge, specially when it is about grasping with dexterous hands because they are such a complex part of the human body. When imagining our future, we would like robots to do somethings that we, humans, do not appreciate so much, like housework. The most objects we interact on a daily basis were built for humans so anthropomorphic hands are the natural manipulation devices for such objects. This kind of hands has many degrees of freedom, which gives them more ability than simple grippers. This complexity also makes the control of these devices harder.

For robot automation turn reality, the robot must be able to pick an object and change the pose of the fingers to grasp it in a way that can use the object in some task. The process associated with that change between grasps is the re-grasping, also known as finger gaiting.

The main motivation for the work presented in this paper is related with the finger gaiting while grasping an object without loosing a forced closure grasp.

The hand model used is based on the iCub hand model. However, our model has one controller for each joint while the iCub model control model has one controller for a coupled set of joints. It can be just one joint, two joint or more joints. The iCub humanoid robot is an open-source developed to be like a kid with about two years old so it has a similar height and a similar capability to learn. The model hand to be used has nineteen joints: each fingers has one joint for abduction (except middle finger), one for proximal, other for intermediate and another for distal.

As people grab objects, anthropomorphic robot hands should grab objects in the same way. In an almost automatic way, we pick an object with enough stability for the object do not drop if some external force is applied. This stability is verified using a test named Force Closure Test([9]). This technique is essential in this project since we need the grasp to be stable so the object is not thrown in some direction due to some uncompensated force that was applied.

A. *Objectives*

The main objective of this work is to develop a technique for in-hand object re-grasping through finger gaiting, by identifying the expendable fingers from grasps, so it becomes possible to move these fingers to the final positions without destabilising the grasp.

Our approach intends to be compatible with all robotic hands once it computes which fingers can be removable from the current grasp. With this information, the user only needs to choose which finger from the removable ones he needs to remove to achieve another grasp. With this work, it acquires more abstraction that does not exist in other approaches, since we just produce the information about which fingers can be removed from the current grasp.

One of the requirements to use this approach is that the robotic hand needs to have tactile sensors so the contact points and forces can be measured to do the calculations.

To perform the finger gaiting, we need to move the fingers from one position to another one avoiding undesired collision situations. So we approach the fingers from the object avoiding collisions and then, they touch the object creating the desired contacts for the grasp. We propose to combine techniques already developed in similar areas to accomplish the re-grasping method such as some present in ROS (Robot Operating System) like the package MoveIt which contains some libraries collision detection, like FCL(Flexible Collision

Library)[6]. More specifically, the manipulation method we present is holding the object and change some fingers to other poses to reach the final grasp. That final grasp has the purpose of the robot to do a task by holding the object with that configuration.

B. Assumptions

We have to take into account object's characteristics required for this problem. Objects need to be:

- Rigid, so they can be simulated using standard tools;
- Light, so the robotic hand is strong enough to compensate any external forces such gravity and inertia;
- High friction coefficient, so they do not slip over during the manipulation.

We assume that the robotic hand controller is able to compensate external disturbances and keep the object static in the hand. The control for rejecting disturbances will not be developed in this work.

C. Contributions

With this work, we aim to contribute developing a safe framework for finger gaiting with an anthropomorphic robotic hand in 3 dimensions. As previously referred, we will use a hand model similar to the iCub robotic hand enhancing the re-grasping applications of the robot.

We will use collision detection features from MoveIt to avoid and generate collision while removing the current contact and creating the desired contact, respectively. The contact model to be used and implemented is the soft contact model since it is a more realistic model to represent contacts.

To develop the method, we will use the tools available in ROS, Robot Operating System, and as simulation tool, we will use Gazebo Simulator.

D. *Document Structure*

This article is divided as follow:

In sections II and III we present some background criteria, like force closure and some related work about techniques already developed in grasp and manipulation.

Next, in VI, VII, VIII sections we will document the work developed, the model hand used and how to compute contacts between the fingers and the object, force closure computation and how to do grasp and re-grasp objects.

In the section IX there are present the results of applying the methods by explain the problem description, the baseline solution and the enhanced solution.

Finally, in XI concludes our work and discuss some applications and point some proposals for future work.

II. Background

While doing finger gait to perform re-grasp, we need to make sure if the current grasp is a force closure grasp. This requirement ensures that the object will not fall during the manipulation.

With soft finger contact model, we can apply forces in a cone. This cone is named as friction cone [7], [9] and it is built aligned with the normal to the surface of contact.

The model chosen, soft finger, is a more realistic model and allows the forces and the torques to be applied in the friction cone. The friction model to be used is Coulomb's friction model, [7]. The formula for Coulomb's friction model when we have soft finger contact model is:

$$FC_{c_i} = \mathbf{f} \in \mathbb{R}^4 : \begin{cases} \sqrt{f_1^2 + f_2^2} \leq \mu_f f_3, \\ f_3 \geq 0, \\ |f_4| \geq \gamma f_3, \end{cases} \tag{1}$$

where γ ($\gamma \geq 0$) is the torsional friction coefficient, f_4 is the torque magnitude along the contact normal direction, $f_1,\ f_2\ and\ f_3$ represent the force along $x, y\ and\ z$ respectively, μ_f is the static coefficient of friction.

To analyze a grasp from the set of wrenches that represents it, we compute the friction cone sampled over its outer limits and we obtain equation (2), with the normal force being unit of magnitude, $f_3 = 1$. Hereupon, equation (2) provides us the base framework for the grasp formulation.

$$w_{ci}^k = \begin{bmatrix} \mu_f \sin(\theta_k) & \mu_f \sin(\theta_k) \\ \mu_f \cos(\theta_k) & \mu_f \cos(\theta_k) \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \\ \gamma & -\gamma \end{bmatrix}, \theta_k \in [0, 2\pi] \tag{2}$$

All contacts are defined as sets of wrenches (forces and torques), positions and normal. We just need to transform all coordinates into a common reference frame so we can treat data correctly.

The set of all transformed wrenches defines a grasp and it is designated as grasp map.

The grasp map is represented by the matrix G [9]:

$$G_i = \begin{bmatrix} R_{c_i} & 0 \\ [p_{c_i}]_x R_{c_i} & R_{c_i} \end{bmatrix} \omega_{c_i} \quad i \in [1, ..., n]. \tag{3}$$

where R_{c_i} is the rotation matrix from one reference frame to another reference frame and $[p_{c_i}]_x$ is the skew-symmetric matrix of p_{c_i} which is defined by:

$$p_x = \begin{bmatrix} 0 & -p_3 & p_2 \\ p_3 & 0 & -p_1 \\ -p_2 & p1 & 0 \end{bmatrix}, \tag{4}$$

and n is the total number of contacts.

One important note is that, the choice of the reference frame will not influence the final result which allows us to use any framework to transform the coordinates. Although, we will use as global reference frame, the palm of the hand.

The resulting grasp map is:

$$G = \left[G_1, ..., G_n\right]. \tag{5}$$

With the grasp map defined, we have the tools to compute force closure test to validade the stability of a grasp.

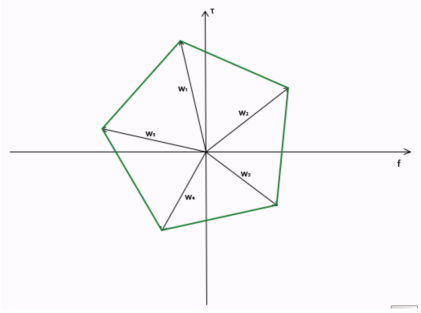

Fig. 1: A set of five wrenches represented in the wrench space (abscissa- forces; ordinate-torques). Image source [9].

A. Force Closure Criteria

Force closure is one of the most used criteria in robotic grasp evaluation. By definition, force closure is a binary test to evaluate the grasp stability. A grasp is stable, and so a *force closure* grasp, if the grasp can resist any applied wrench. Mathematically speaking, a grasp is force closure if there is a combination of contact forces, f_c such that, when applying an external wrench to the object,w_e:

$$Gf_c = -w_e, \tag{6}$$

where f_c is a vector of the forces of each contact in friction cone:

$$f_c = \begin{bmatrix} f_{c_1} \\ f_{c_2} \\ . \\ . \\ . \\ f_{c_n} \end{bmatrix}, \mathbf{f}_{c_i} \in FC \tag{7}$$

and n is the number of contacts that compose the grasp.

We can easily evaluate if a grasp is a force closure grasp by analyzing the convex hull of the grasp map, $ConvexHull(G)$. The convex hull of G is the minimum convex region spanned by G on the wrench space. In figure 1 it is represented the wrench space (force, torque). It has only 2 dimensions since, if we discretize all force components, with the torque it would be a 4D space. So, with the wrench vectors, corresponding to each contact, in it, we have the minimum convex region that contains all wrenches that represent the grasp, in green, as known as convex hull of the set of wrenches. In this example, we can say that the grasp is force closure since the origin is inside the convex hull.

Since each contact between one finger and the object has many points of contact, the convex hull of all points generates a patch that will be the surface of contact. We will use all points of the surface contact to obtain the force closure result.

III. Related Work

In [5] there are defined some grasps performed with an anthropomorphic hand using synergies. In figure 2 we can see eight different grasps from that work. Furthermore, in figure 2, we have the eight separated grasps and we want the robot write something and to do so, the selected grasp may not be the writing tripod. Let's admit that writing tripod is the only one able to writing. If the pen or pencil is on some surface, the robot need to pick up the pen and the robotic hand cannot pick the pen with writing tripod at first, so it need to pick the pen with, for example, palmar pinch and then change some fingers positions to achieve the writing tripod grasp.

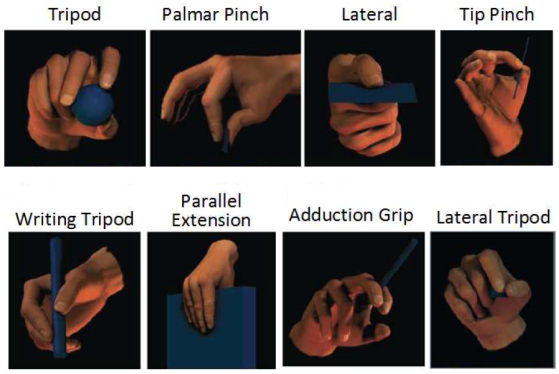

Fig. 2: 8 different kinds of grasps already implemented for iCub using synergies. Image from [5].

To alternate between these grasps, we do re-grasp by finger gaiting processes.

Some work on finger gaiting was developed in [4], [10], [3], [11]. The most approaches finger gaiting related, resort to rolling and/or sliding techniques. These techniques are used mostly when a forced closure grasp does not have removable fingers available to do the manipulation task. With our hand model we do not need rolling feature since we have 5 fingers to do grasp and re-grasp. In [2] it is also used rolling technique using both palm and fingers. Although, they use an anthropomorphic hand to develop the work. The usage of the palm, to do re-grasping, limits the degree of freedom of the fingers, once the palm is needed to keep a force closure grasp. In our approach we will not consider palm to do re-grasp.

Some existent approaches use multi-fingered grippers. The grippers are not the ideal manipulator to use with objects developed for anthropomorphic hands use them. The most of the manipulation tasks, feasible by grippers, are pick and place, object reorientation and finger substitution. Of course that, with more fingers, grippers can do re-grasping and finger gaiting.

Another tool widely used is the grasp trees. Although not all robots works with the same tree mainly because there are planar hands and anthropomorphic hands, both with the possibility to have a different number of fingers beside the degrees of freedom of each finger and other properties to take into account when doing manipulation. [8] suggests a new approach to build generic grasp trees. In [1] approach, it is built a switching graph similar to a grasp tree based in a 4-fingered dexterous hand. [1] presents a most similar case when compared with the approach of this paper. The differences between the present paper and the mentioned one is the use of rolling and sliding moves to achieve another grasp; it also bases in grasp trees building.

Our approach aims to achieve re-grasping task by finger gaiting. We need to relocate some fingers with a force closure grasp during all the time. This work fits in any of the previous mentioned works by helping to build the grasp map or defining new grasps enlarging the amount of grasps defined to re-grasp. It can be used also to build new grasp trees to more recent robotic hands with no grasp trees developed yet. Even with hands that does not allow rolling nor sliding techniques due to friction limitations.

IV. Outline of the Proposed Algorithm

In the literature there exists some algorithms that generate grasps that allow the robot to pick up objects. With this work, we intend to give the ability to robots change the position of the fingers, with the object held. The purpose is to reach some other grasp to use the object.

V. Outline of the Proposed Algorithm

First of all we should define the required inputs of this algorithm. We need a robotic hand, a set of, at least, two force closure grasps, one as the initial pose and another as the final pose and an object to perform those grasp poses. The object should be compatible with the selected grasps so the re-grasp task can be accomplished.

Given an initial force closure grasp, we start by the contact acquisition so we can identify which fingers are removable from the current grasp.

This identification in described in algorithm 1.

Algorithm 1 Identify Removable Fingers

1: $\mathbf{fingers_{names}} \leftarrow$ "names of fingers from topic positions_normals"
2: **for** $i = 1$ to $number_{fingers}$ **do**
3: $\quad \mathbf{V_{positions}} \leftarrow$ "positions of i-th finger from topic positions_normals"
4: $\quad \mathbf{V_{normal}} \leftarrow$ "normal of i-th finger from topic positions_normals"
5: **end for**
6: **if** ForceClosure($\mathbf{V_{positions}}, \mathbf{V_{normal}}$) **then**
7: $\quad$ **for** $finger_i = 1$ to $number_{fingers}$ **do**
8: $\quad\quad \mathbf{V_{normal_{aux}}} \leftarrow \mathbf{V_{normal}}$ "without normal from i-th finger"
9: $\quad\quad \mathbf{V_{positions_{aux}}} \leftarrow \mathbf{V_{positions}}$ "without positions from i-th finger"
10: $\quad\quad$ **if** ForceClosure($\mathbf{V_{positions_{aux}}}, \mathbf{V_{normal_{aux}}}$) **then**
11: $\quad\quad\quad \mathbf{V_{removable}} \leftarrow finger_i$
12: $\quad\quad$ **end if**
13: $\quad$ **end for**
14: **end if**
15: $publish\mathbf{V_{removable}}$ in $removable\_contacts$

It tests finger by finger, which can be removed from the grasp without loose the stability. If one of the fingers is not touching the object, that finger is automatically a removable one.

Identified those fingers, we start with the re-grasp task. This re-grasp task is based on finger gaiting. The gait process is done as shown in algorithm 2.

At each contact changing, we update the force closure result so we can have a force closure grasp all the way. At each true result of force closure, the removable fingers might change, since they are updated at each finger gait.

Algorithm 2 Identify Removable Fingers

1: **while** $current\_pose \neq final\_pose$ **do**
2: $\quad$ **removable_fingers** $\leftarrow$ "names of fingers from topic removable_contacts"
3: $\quad$ **if removable_fingers** $\neq \emptyset$ **then**
4: $\quad\quad$ **if** $\exists$**removable_fingers**$(i) \in$ **fingers_to_reposit**(j) **then**
5: $\quad\quad\quad$ **if** $finger\_done(j) = 0$ **then**
6: $\quad\quad\quad\quad Lift\_finger($**removable_fingers**$(i))$
7: $\quad\quad\quad\quad Reposition\_finger($**removable_fingers**$(i))$
8: $\quad\quad\quad\quad Land\_finger\_in\_obj($**removable_fingers**$(i))$
9: $\quad\quad\quad\quad finger\_done(j) \leftarrow 1$
10: $\quad\quad\quad$ **end if**
11: $\quad\quad$ **end if**
12: $\quad$ **end if**
13: **end while**

In figure 3 we have a flowchart to illustrate the sequence of actions done by the algorithm.

At each update of the force closure test and the removable fingers set, the fingers that could not be relocated in previous iterations and now can, are selected to be relocated until they reach the final pose.

The algorithm gives priority to reposition the needed fingers for the final grasp pose. After all the fingers from the final grasp are in the corresponding final poses, the fingers not required are lifted. The final pose is reached and the algorithm ends.

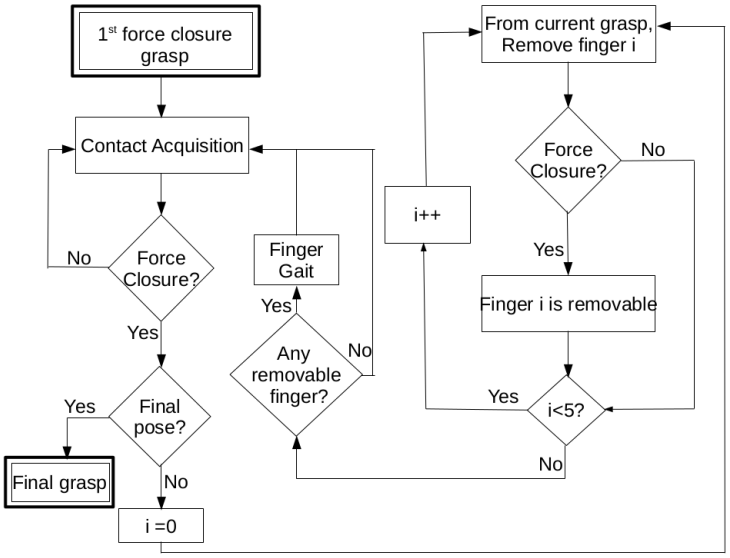

Fig. 3: Flowchart illustrating the proposed algorithm (see text).

As outputs from this algorithm we have the finger gaits that lead each of the fingers, from the initial grasp, to the final grasp. Each finger is relocated at a time so we can have a

forced closure grasp in every iteration of this algorithm. The final result will be the robotic hand holding the object with the configurations of the final grasp.

VI. Hand Model For Gazebo Simulator

The hand model to use is a model based on the hand of iCub. This model differs from the real one, once this has one controller for each joint and the real model has some coupled joints that requires fewer controllers. The figure 4 shows the model developed from the meshes present in RVIZ, just used to visualize iCub robot. We adjust some parameters and we needed to create some elements, required so the model could be shown in Gazebo.

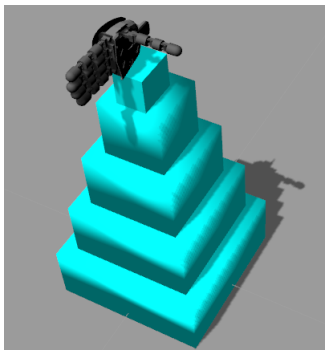

Fig. 4: Hand model on Gazebo.

Initially we choose to work with a known model of a robotic hand, the hand from iCub. In the most recent model of the iCub for Gazebo, there is no hand. We implement and complete an existent model of the iCub hand mentioned above. We complete it by adding inertia, collision and sensor elements in the description file of the robot. These elements are required so it can be simulated with physic values and, with sensors, we can use detection feature of Gazebo.

VII. Contact Computation

When sending commands to a certain joint, an effort is applied and the joint bends. With a set of effort joints to each finger we can create contacts. The contact model acquired from Gazebo is based on soft finger contact model. Although, the contact information received from Gazebo is not stable. During simulation, when there is some contact, the finger that is touching the object is trembling due to the singularities, associated to the contact and, as consequence, the contact is not constant.

This acquisition is done using a temporal filtering since samples are not constant through time as desired. This filtering allows us to acquire consistent information about all existing contacts in simulation. Such adaptation to the simulator data is not that simple since, the interval of samples acquired is not always the same. If we update the contact information based in just a sample at once, we would not have all contact information existing in the environment and barely have a force closure grasp situation.

With this solution we create another problem: we need to detect when a contact is removed from simulation. To solve this issue we implement a counter to detect how many samples are transmitted without a certain contact. When a certain value of samples without registering of a contact, is reached, the contact is considered as removed and it does not count for the next computations.

TABLE I: Set of time samples from the topic we acquire contact information. Notice the inconsistency of information during time without any motion of the fingers during the acquisition.

Samples	**Detected Contacts**	**Bodies in collision**
1	0	-
2	1	box from base vs ground
3	0	-
4	2	object vs right index fingertip, box from base vs ground
5	4	object vs right index fingertip, box from base vs ground, object vs right ring fingertip, object vs right middle fingertip
6	1	object vs right index fingertip
7	2	object vs right middle fingertip, object vs right ring fingertip
8	0	-
9	3	object vs right index fingertip, object vs right ring fingertip, object vs right middle fingertip
10	2	object vs right ring fingertip, object vs right little fingertip
11	5	object vs right index fingertip, box from base vs ground, object vs right ring fingertip, object vs right middle fingertip, object vs right thumb fingertip
12	1	object vs right little fingertip
13	4	object vs right index fingertip, object vs right ring fingertip, object vs right middle fingertip, object vs right thumb fingertip
14	3	box from base vs ground, object vs right ring fingertip, object vs right thumb fingertip
15	6	object vs right index fingertip, box from base vs ground, object vs right ring fingertip, object vs right middle fingertip, object vs right thumb fingertip, object vs right little fingertip

In the table I we can see that the set of samples from the topic does not give us synchronized information within a unique sample. We need to post-process that information so we can get all the contacts from the simulated grasp. We acquire the information from all samples and ignore the not relevant contacts to the grasp, such as the contact between the base box and the ground. We put all relevant information in a structure. Since the positions and normal received are in the reference frame of the correspondent link, we need to transform all data to a common reference frame.

When some contact is transmitted, that contact has many points of contact and the correspondent normal. There are many points because Gazebo allows a little penetration between objects due to some imprecision. These points correspond to the points in common between the object surface and the link surface touching the object.

To close the finger to grasp the object we apply a method that distribute the weight of closing each joint proportionally to a geometric series, so we can find the best closed pose:

$$\sum_{n=1}^{20} \left(\frac{1}{2}\right)^n, \tag{8}$$

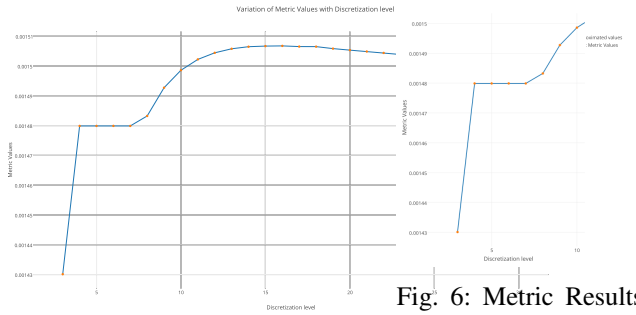

Fig. 5: Metric Results using different levels of discretization.

Fig. 6: Metric Results using different levels of discretization.

and the resulting angle at n iterations is given by:

$$\alpha_n = \alpha_{n-1} \pm \alpha_{max} \times \left(\frac{1}{2}\right)^n \ , \ \alpha_0 = 0. \tag{9}$$

If the finger is in collision, we subtract the next iteration, if it is not, we sum (unless it reaches the maximum angle of closing). At the end, if there is any collision we subtract 0.1% of the maximum angle until it is a collision free pose and then we do the motion planning. This percentage is low because we do not want the finger go too far from the object.

For collision detection, we use the FCL, [6] from MoveIt package, available in ROS.

VIII. Force Closure Computation

The force closure condition needs to be verified, so we can identify if a grasp is stable. To do so, we will test the set of contacts received, with the equations in II.

With the points of contact established between the fingers and the object we will verify if the force closure condition is satisfied.

To do so, we will test the set of contacts received with the equations in II. In this test we considered each point of contact as a unique and compute the friction cone to each of them.

To compute the friction cone, we approximate the base by a polygon on x edges. The value of x is named as the level of discretization of the friction cone.

With different discretization levels of the friction cone, the result also changes. We perform some tests with the discretization level changing from 3 to 30. In figure 5, we have the resulting values from one of those tests. To acquire this results we compute the force closure test with one set of contacts corresponding to a unique grasp pose. We can see, from figure 6, a zoomed in image of 5. The discretization level is stable between the level 4 and 7 and then increases and start oscillating. The purpose is to have a value as stable as possible so we will choose a level of discretization between 5 and 6. The time interval took to compute the force closure result increases linearly so the best value is the smallest possible. Once we also intend to use a stable value in a safe zone and smallest as possible. We use the value 5 for the discretization level of the friction cone to calculate the force closure metric.

Instead of having one friction cone associated to each finger, we have as many friction cones as contact points in each finger. This approach gives us better force closure results. This choice relies in the fact that simulations does not give us constant data to process so we decide to consider all contact points of each finger. This decision allows us to have better result because having just one point for each contact, it could give us a false negative force closure result due to position and normal direction of that point and with this we are sure that we are computing a more accurate result of force closure. This is our variety of soft finger contact.

We adopted the work done in [9] and by adjusting some parameters like static and torsional friction coefficients and discretization level from fiction cone we achieve better results of force closure tests.

Since is a method needed sporadically, we put the code into a service node and once we need to know the result of force closure test, we fill the request, create a client and call the service by that client.

IX. Experimental Results

In this work it was developed a method to perform re-grasp actions using finger gaiting by identify which fingers are expendable to a certain grasp so the finger gaiting can be made. To do that, we need to know which fingers we can reposition on the object.

To have that information we need to know if it is safe to remove some fingers. Those fingers need to be moved to another position. We use force closure tests to make sure that the grasps are always forced closure grasps during the finger gaiting.

In this chapter we establish the baseline for the solution to the re-grasping problem with finger gaiting. We will also show some choices made for some adjustable parameters.

We used an anthropomorphic hand with 5 fingers. Four of the five fingers have 2 degrees of freedom and one has 3 degrees of freedom present in figure 4. The model used was base on the iCub robotic hand. All joints have an individual controller associated. This tool was developed in ROS. The version used was ROS Indigo combined with Gazebo Simulator 2.2. This is not the latest version from Gazebo and it has not all the characteristics from the latest version. Once we use a more recent version of ROS compatible with Gazebo, we can use a more recent version of Gazebo. Perhaps with a larger amount of tools available to use. We needed to simulate with a static object, which difficult the task to find two grasp configurations to test our approach. So we created a new, unusual, grasp to perform the re-grasp with a static object that we decide to call IR2M, index ring to middle, represented in figure 9.

The tool was written in C++ and Python.

A. Experimental setup

To approach the re-grasping task using finger gaiting, we need a hand model, one object and, at least, two different grasps to perform in-hand manipulation.

The hand model chosen is the model from the iCub robot, present in figure 4. The objects used are illustrated in figure 7.

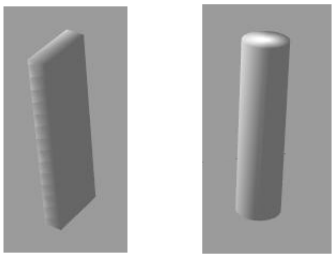

Fig. 7: Objects used in this algorithm.

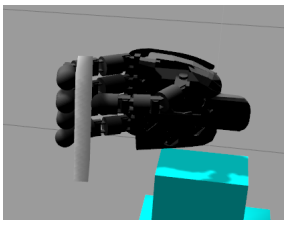

Fig. 8: The Version of Parallel extension grasp used in this work.

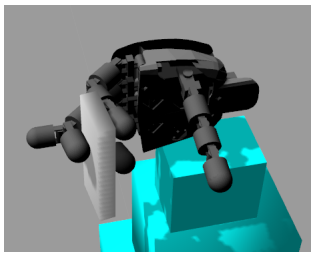

Fig. 9: Index Ring to Middle grasp (IR2M).

The re-grasp task consists in changing the contacts between some fingers and the object, to reach a different grasp. We used the grasps present in figures 8 and 9.

B. Reaching the First Grasp Pose

The first grasp is created by moving the fingers against the object like explained in VII. We implemented a motion planning method to achieve the version of parallel extension grasp. With this grasp pose we can show the robustness of the removable fingers identification since we could perform a force closure grasp with just the middle and the thumb fingers but we use all fingers to show that some fingers can be removed.

Initially, the hand model and the object are like in figure 10a). Next, we apply the motion planning method and the fingers are placed on object surface like showed in figure 10b). All fingers are touching the object.

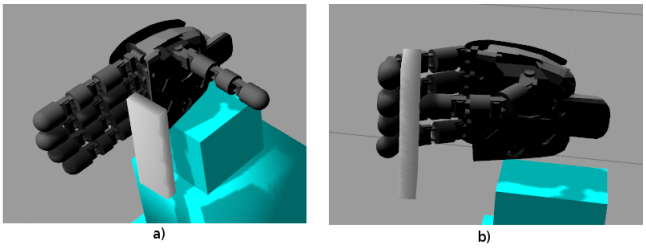

Fig. 10: a) The hand and the object in the simulation environment. b) First grasp pose.

C. In-Hand Manipulation

In figure 3, we showed the flowchart scheme of this algorithm. Figures 11 and 12 shows us 4 different ways of re-grasp, with the algorithm that leads to the same result using the box from figure 7.

During the test phase we mostly used the first sequence of the grasp phase, the upper sequence from figure 11. It starts with the grasp pose from figure 10b), reposition the index and the ring fingers, follow one of the sequences, reaching the grasp when all fingers are in contact with the object. From there, it only needs to remove the little and the thumb fingers, by any order to reach the final pose. It can perform any sequence. The main advantage of performing any re-grasp sequence reflects in the case of we need to remove, for example, index finger and it is not an expendable finger. Then, we can remove ring finger instead of the opposite may also happen. With this, we have more chance of achieving the final grasp pose. The algorithm gives preference to reposition the finger needed for the final grasp pose. When they are repositioned, it remove the fingers no required for the final pose.

We show the robustness of the algorithm by using another object with a different shape: the cylinder from figure 7. Performing this algorithm with an object like a cylinder is challenging because its surface is not plane as a rectangle, which implies a more accurate re-grasp task to ensure force closure grasp phases when re-grasping. Figure 12 illustrates the result of applying this approach with a different object. It can perform one of the 4 possible sequences, present in figure 12 of the re-grasping task.

Notice that the movement of the index and ring fingers are the best to perform the re-grasp task described since the fingers are shrunken at the maximum so they can pass behind any object. If the fingers collide with the object, then the hand should be slightly moved away from the object so the fingers can be repositioned.

D. Force Closure Computation

In addition to the unsynchronized information, we also have noisy contact normal and positions. This issue increases the chances to get inconsistent data to compute the force closure test, corrupting the results of the test. To deal with this problem, we have two options: compute the mean of all contact normal and positions from each link and compute the force closure with that mean value or, we use all contact positions, and respective normal, to compute the force closure result.

We test both options, based on the sequence a, b1, c1, d, e1, f from figure 11. For each grasp phase we acquire the set of contact normal and positions and compute the force closure test with the mean calculation and without it.

In figure 13 we have the force closure results with and without the mean of the contact normal and positions. This test was made using the box from image 7 as the object to grasp. For a better understanding of the x axis:

- First letter correspond to the grasp phase in the sequence shown in figure 11:
  - a - initial grasp pose
  - b1 - changing the index finger
  - c1 - index finger in the final position, ring finger in lifted
  - d - index and ring fingers in the final positions
  - e1 - little finger lifted

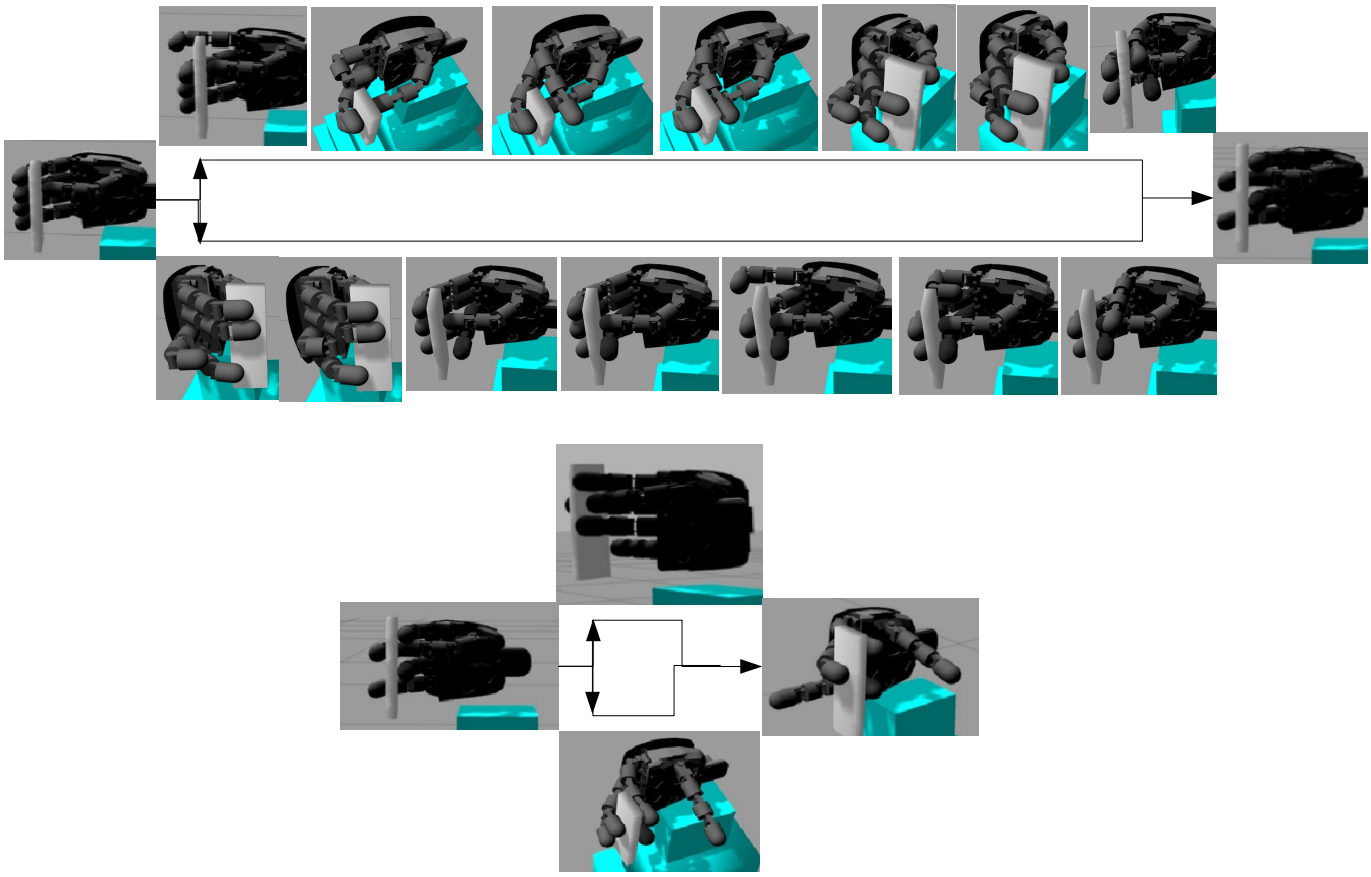

Fig. 11: 4 possible re-grasp sequences tested with the algorithm.

  - f - both little and thumb fingers lifted; final grasp pose
- Second letter means the finger that we are testing with the test:
  - i - index finger
  - l - little finger
  - m - middle finger
  - r - ring finger
  - t - thumb finger

If the metric value is negative, that means that the force closure result is False. If the metric value is positive, that means that the force closure result is positive.

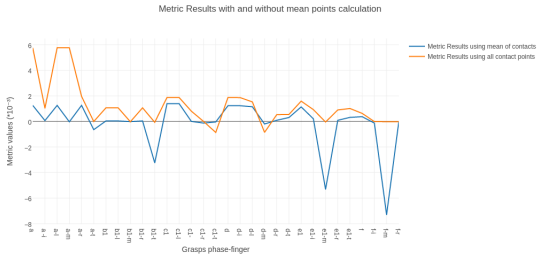

Fig. 13: Metric Results using mean calculation, in blue, and using all contact points, in orange.

We notice that, without computing the mean, we get more accurate and credible results and while the method is running, when using the metric with mean, sometimes the result is false. This happens because we are computing the mean with the non constant data from the simulator. Without mean calculations we also do not get constant points but the fact that we are using all the points, the metric returns more constant results in terms of binary result so we use all points without averaging.

X. Limitations

We accomplished the proposed work, although, there is some issues to improve. We can start by describing some limitations of this method.

With the present implementation of the contact acquisition, there is no ambition to wait for the initial grasp pose. This issue might bring some troubles, namely the fact that some required fingers to reposition might not be available to reposition because they are not expendable from the grasp.

With another pair of grasps, this method is not able to verify if the sequence is feasible directly or indirectly. It could be feasible but the order of gaiting might be crucial or it might be necessary resort to another grasp pose that will connect the two sequences. The order of the gait can be done resorting to the metric value from the force closure test result. For example,

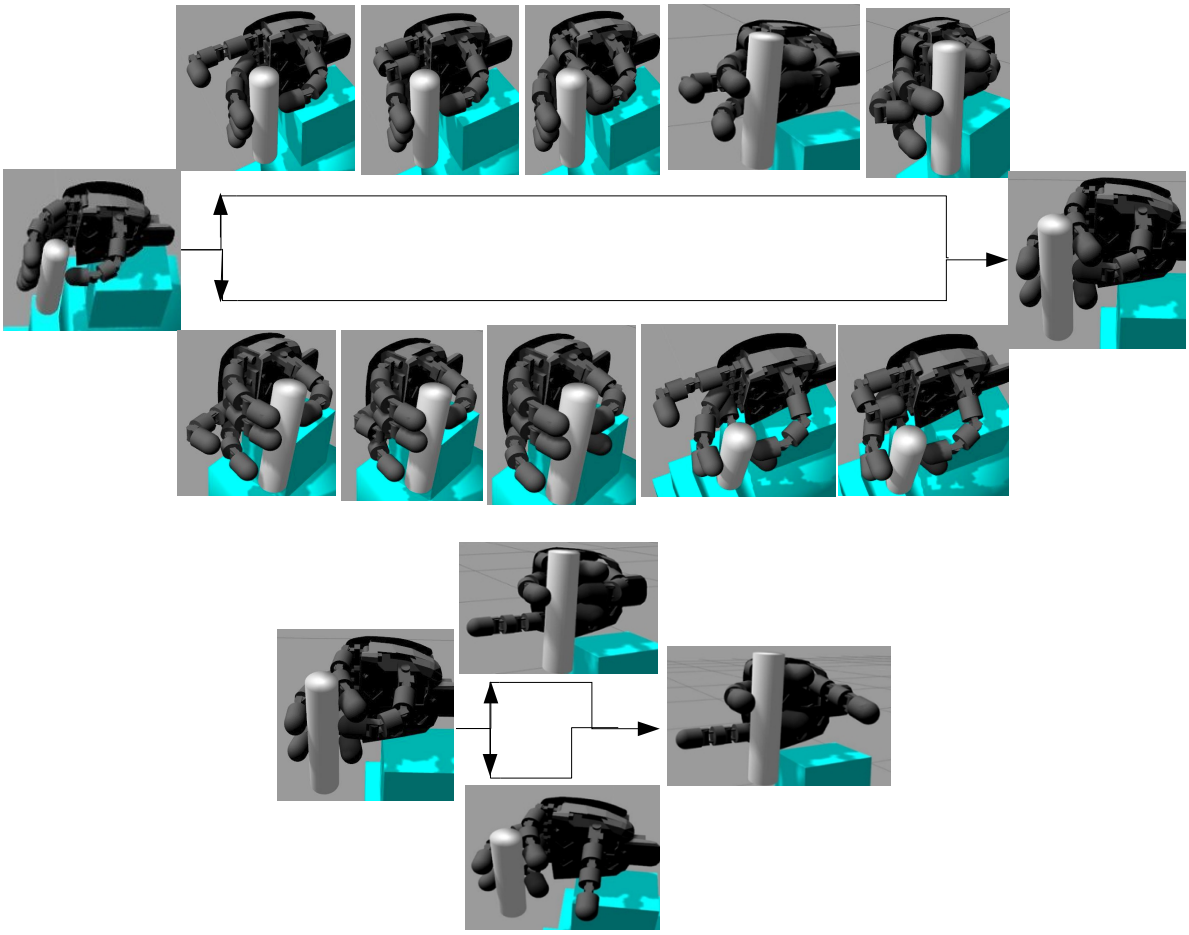

Fig. 12: 4 possible re-grasp sequences tested with the algorithm.

when two grasps can be reached used on a sequence of re-grasp, we might choose the safer one.

XI. Conclusion

This work proposed a methodology to perform in-hand re-grasping actions with finger gaiting. This approach aims, given an initial grasp pose, to changing the contacts between the fingers and the object to reach a new grasp pose. The re-grasp task should be performed always with a force closure grasp so the object does not fall.

In this approach we do finger gaiting by moving one finger at a time to a final pose. This motion of the fingers requires an algorithm of motion planning, with collision avoidance for not creating undesired collisions and perhaps, jeopardize the re-grasping process.

Moving one finger at a time requires the identification of the fingers that can be removed from the current set of contacts with the object. We identify the removable fingers by testing the stability of that grasp without considering the finger under analysis. If the result of the force closure test remains true, then the finger is removable from the grasp and we can move that finger freely to any other point of contact on the object surface, provided it does not slide or collide with the object.

From the initial grasp on an object, we identify the removable fingers from that grasp and start the finger gaiting process by repositioning a finger at each time. As the contacts change, the force closure test returns the result related to the updated contact information.

We considered that any part of the finger can be in contact with the object, in other words, we considered that all links from each finger have sensors on their surface allowing the acquisition of contact information from any part of the finger. It can be applied in another contexts, since it gives the information about all the fingers that can be moved.

We create a new unusual grasp pose that allows us to do re-grasp with a static object in simulation, IR2M - Index Ring to Middle.

This approach is different from those already developed since this one was developed in the ROS robot platform, giving our contributions for the ROS community and also allows adaptability to other robots. It is able to deal with sensors in all links in the fingers and it also works with sensors only on fingertips.

We showed our way to do the motion planning for finger gaiting with collision avoidance and, at the same time, create a desired contact situation with the object. We showed how to

deal with gazebo contact information using temporal filtering. We also showed that, with non-constant data acquisition, the best way to compute force closure tests is using all contact points between the object and the fingers.

Also, we contributed for the community by introducing a model of the iCub hand to use in the Gazebo simulator.

We showed the robustness level of this algorithm, not only in terms of variations of the object shapes, but also in terms of performing different sequences of finger gaiting, reaching the same final grasp pose.

A. Future Work

Future work will improve this research, by adding more features, like acquiring the object information from Gazebo, for instance, the surface area where each finger can reach. Another way to keep this research is to couple the fingers joint as in the real iCub hand model, favoring the use in the real robot. To improve this research, it could be introduced grasp skills, not only for the iCub but also to general dexterous hands giving the ability to being adaptable for other robotic hand models. Finally, we suggest adding rolling and sliding techniques to enhance re-grasp feature.

REFERENCES

[1] T. P. Attawith Sudsang. Regrasp planning for a 4-fingered hand manipulating a polygon, September 2003.

[2] Y. Bai and C. Liu. Dexterous manipulation using both palm and fingers. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1560–1565, May 2014.

[3] L. Han and J. Trinkle. Dextrous manipulation by rolling and finger gaiting. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 1, pages 730–735 vol.1, May 1998.

[4] L. Han and J. C. Trinkle. Object reorientation with finger gaiting, April 1998.

[5] M. A. M. Henriques. Controlo e planeamento de mos robticas antropomrficas utilizando sinergias. Master's thesis, Instituto Superior Técnico, July 2013.

[6] D. M. J. Pan, S. Chitta. *FCL: A General Purpose Library for Collision and Proximity Queries,*. in IEEE Intl. Conf. on Robotics & Automation Magazine, Maio 2012.

[7] S. S. S. Richard M. Murray, Zexiang Li. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[8] J.-P. Saut, A. Sahbani, and V. Perdereau. A Generic Motion Planner for Robot Multi-fingered Manipulation. *Advanced Robotics*, 25(1-2):23–46, Jan. 2011.

[9] F. Veiga. Robotic grasp optimization from contact force analysis. Master's thesis, Instituto Superior Técnico, April 2012.

[10] J. Xu, T. Koo, and Z. Li. Finger gaits planning for multifingered manipulation. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 2932–2937, Oct 2007.

[11] J. Xu, T.-K. J. Koo, and Z. Li. Sampling-based finger gaits planning for multifingered robotic hand. *Auton. Robots*, 28(4):385–402, May 2010.