

On the Generation of a Variety of Grasps

Sahar El-Khoury, Miao Li, Aude Billard

*Learning Algorithms and Systems Laboratory (LASA),
Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland*

Abstract

In everyday life, people use a large diversity of hands configurations while reaching out to grasp an object. They tend to vary their hands position/orientation around the object and their fingers placement on its surface according to the object properties such as its weight, shape, friction coefficient and the task they need to accomplish. Taking into account these properties, we propose a method for generating such a variety of good grasps that can be used for the accomplishment of many different tasks. Grasp synthesis is formulated as a single constrained optimization problem, generating grasps that are feasible for the hand's kinematics by minimizing the norm of the joint torque vector of the hand ensuring grasp stability. Given an object and a kinematic hand model, this method can easily be used to build a library of the corresponding object possible grasps. We show that the approach is adapted to different representations of the object surface and different hand kinematic models.

Keywords: Grasping, force-closure, hand kinematics, implicit object representation

1. Introduction

The human hand is an amazing tool. People are capable of grasping a variety of objects of different shapes and sizes in a variety of ways depending on the task's requirements. Consider for example the different grasps in figure (1) and notice how the placement of the hand and fingers varies along different tasks. Such variety of grasps is possible in part thanks to the dexterity of the human hand and its large number of degrees of freedom and in part thanks to our marvellous control strategy.

Even today, it is difficult to explain how the Central Nervous System (CNS) is able to control the high number of degrees of freedom of the human hand. Evidence from Neuroscience indicates that such control is taking place in a reduced dimensional space [28]. While grasping a number of familiar objects, static hand posture of several subjects were measured by recording their finger joint angles [28]. Principal component analysis showed that the first two components account for more than 80% of the finger joint variance. The remaining components provide information about the shape of the grasped object. These results imply that the finger joint angles do not vary

Email addresses: sahar.elkhoury@epfl.ch (Sahar El-Khoury), miao.li@epfl.ch (Miao Li), aude.billard@epfl.ch (Aude Billard)



Figure 1: Several grasps employed in everyday life. Notice the variation of the hand position/orientation across the different grasps.

independently and thus the control of the hand posture involves two main postural synergies regulating the hand preshape. A finer control mechanism provides afterwards subtle adjustments of the joints to perform the final grasp. This study did not take into account the hand position/orientation relatively to the object placement, it only accounts for the finger joints angles.

When such strategies are applied to robotic grasping, they lead to defining preshapes for the robotic hands inspired from those of the human hand [4, 18, 7]. To perform a grasp, the fingers are positioned in a pregrasp shape and then closed around the object. The hand starting position and approach vector are either defined manually according to the object shape [18], are learned from human demonstration [7] or determined through optimization [4]. Such approaches choose first the hand preshape and then find the hand position/orientation matching this finger posture with the object shape. An appropriate finger posture, however, needs to be combined with a correct hand configuration in order to be relevant for a specific task. Take for example the task of flipping and picking up an object, see last column in figure (1). A cylindrical hand preshape is used for both tasks, the only difference is in the hand orientation. Thus, the optimality of the grasp depends on the placement of the hand relatively to the object as well as on the finger configurations. Notice as well that a preshape explains only 80% of a grasp finger joint values [28]. The remaining 20% are related to a fine adjustment of the fingers that depends on the object shape and thus cannot be obtained by simply closing the fingers around the object.

This paper shows that grasp synthesis can be formulated and solved as a large scale, non-linear, constraint-based optimization problem taking into account parameters related to the hand and finger configurations as well as the object shape, weight and friction coefficient. First results were published in our paper [10] where we synthesized a variety of grasps for a cylindrical object described as a superquadric shape, using the iCub hand. This paper extends our previous work by showing the ability of the algorithm to cope with more complex objects (described by a single function using a probabilistic method) and different hand kinematic models.

2. Related work

Napier, in his paper [22], distinguishes between power grasps and precision grasps. Power grasps lead to large areas of contact between the palm, the fingers, and the object providing means of holding an object robustly into one's palm. Precision grasps are particularly useful during manipulation since they offer more dexterity. In robotics, when a robot is provided with a hand

mimicking the human hand dexterity, precision grasps become particularly relevant. Usually, these hands have a much larger degrees of freedom number than more classical robot grippers. Using such complex hands to generate a precision grasp requires determining a position and an orientation for the hand and fingers that are feasible and that guarantee the grasp stability. This is difficult to achieve given the high-dimensionality of the grasping space and the non-linearity of the constraints. In the following, we provide an overview of precision grasp synthesis related literature.

2.1. Determining stable grasps

The force-closure criterion has been extensively used in the grasping literature in order to guarantee grasp stability [1, 23]. A grasp is said to achieve force-closure when the fingers can apply appropriate forces on the object to produce wrenches in any direction [29]. This condition is equivalent to stating that the origin of the wrench space must lie strictly inside the convex hull of the finger contact wrenches [21]. Several approaches have been proposed to compute force-closure grasps [23, 24, 16, 8]. These methods propose sufficient conditions for computing stable grasps on 3D objects. However, they rely on a secondary mechanism to test whether the grasping points can be reached by a particular hand kinematics and do not convey any information about how good the grasp is. A variety of criteria were introduced to give a measure of the quality of stable grasps, see [19, 31] for a study on various existing grasp metrics. Several works express this problem as constraint-based optimization. We briefly review these next.

2.2. Optimizing force-closure grasp synthesis

Mirtich and Canny [17] proposed two optimality criteria to measure a three finger grasp's ability to resist moments within or normal to the grip plane defined by the three contacts on the object. When the first (second) criterion is used, the maximum circumscribing (inscribing) equilateral triangle defines the optimum grasp of a 3D polyhedral object. To determine the optimal grasp, one can then compute all triples of vertices of a n -vertices polyhedron. This algorithm is efficient when the number of faces of the object is small. However, commonly used objects are not necessarily polyhedral and can rarely be modeled with a limited number of faces. Hence, when a polyhedral grasp synthesis approach is applied to these objects, a huge computational effort is required to study all combinations across constituent faces.

When objects are smooth, such as ellipsoids, the primitive wrenches of the grasp are also smooth functions of the grasp configuration and different methods are applied for optimizing the grasp quality [34, 33]. For instance, in [34], a numerical test is proposed to quantify how far a grasp is from losing force-closure and is solved through linear programming. Optimal force-closure grasps are obtained by gradient descent in the grasp configuration space, starting from a configuration where all the contact points are on the object surface. Zhu and Wang [33] proposed a similar algorithm in which gradient descent is performed to determine the minimum of the derivative of the Q-norm (the Q-norm measures the minimum scale factor required for a convex set to contain a given point a , i.e. it quantifies the maximum wrench that can be resisted in a predefined set of directions given by the corresponding convex set).

Because of the complexity yielded by the inclusion of the constraints of kinematic feasibility, all solutions to force-closure reviewed above did not consider these constraints during optimiza-

tion. Obviously, doing so is crucial for grasp execution but difficult to achieve considering the large number of possible hand configurations. In these approaches, the optimal grasp is defined solely by the locations of the contact points on the object's surface. Hence, the solution found through optimization may easily violate the finger joint constraints and thus may not be feasible for the robotic hand. When trying to optimize the grasp quality by taking into account the hand kinematics constraints, some approaches start from different initial grasps where the fingers are in contact with the object and then optimize the grasps according to different criteria using for example genetic algorithms [5]. Such approaches need, however, to generate these initial grasps. As we stated previously, this is complex due to the high dimensional grasp space.

2.3. Grasp generation given the hand kinematics

The problem of how to include kinematic constraints in the search for optimal grasp is known as the *configuration issue* [26]; given a specific grasp, defined by a number of contact points/regions on the object surface as well as contact points/regions on the hand surface, find a configuration of the hand permitting contact between each hand region and its corresponding object region [26, 12, 2]. These approaches solve the grasp synthesis problem in two steps: they first find contact points locations yielding a good grasp according to a quality criterion, then find the corresponding hand configuration if it exists, otherwise generate another set of contact locations for the fingers, etc. Searching through each set of contact locations until one finds a feasible grasp may be a long process. Hence, these methods are not optimal from a computation time point of view.

Instead of directly searching the high dimensional configuration space of robotic hands for a grasp, other studies reduce this space by generating a set of hand starting positions/orientations and finding afterwards all possible finger contacts on the object [32, 38], or by generating a set of hand preshapes [18, 7] or eigengrasps [4] that can then be tested on the object model. The former approaches require a good position/orientation of the hand in order to enable the fingers to touch the object yielding a good grasp, this is not simple to achieve since the grasp quality depends on both the hand and finger configurations and thus should be taken into account in the early stages of grasp planning. This applies as well to the latter approaches that need to carefully choose the initial approach direction of the hand. These approaches also induce a reduction in the accessible hand postures by reducing considerably the dimensionality of the hand configuration space.

3. The proposed approach

In this paper, we take advantage of novel developments in optimization techniques and propose a one shot algorithm for grasp synthesis. While this comes at the cost of finding locally optimal solutions, we show that in practice, this yields, even for simple shaped objects, a variety of good grasps that can be used for the accomplishment of many different tasks. Having at hand several locally optimal feasible grasps allows us to compare their quality metrics and pick out the best one. The originality of the method is that it considers all the variables involved in the grasping process, from the finger joint angles/torques to the palm position/orientation till the object's shape, size, weight and friction coefficient to generate possible grasps. Grasp synthesis can then be formulated as a non-linear, constraint-based optimization problem that takes into account the grasp quality from the very first step of the grasp computation procedure and does not rely on a

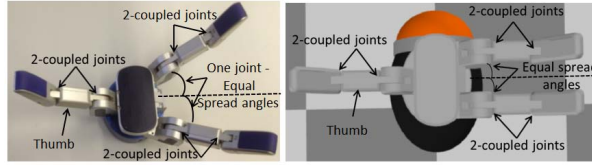


Figure 2: The real Barrett hand and its corresponding model in our RoboToolKit simulator.

pre-specification of a particular hand pose, orientation or preshape.

In the following paragraphs, we derive the steps by which we can include the kinematic constraints of the robotic hand to describe finger placement on the object surface yielding a grasp of a good quality. Section 4 details the kinematic structure of the Barrett and iCub hands; Section 5 describes our representation of the object surface; Section 6 describes the constraints that must be satisfied to yield stable and high quality grasps on 3D objects; Section 7 shows that grasp computation can be formulated as a non-linear optimization problem and gives the corresponding solution; Section 8 illustrates the performance of the approach; Section 9 concludes.

4. Hand kinematics

Two different robotic hands are employed in our study, the Barrett hand and the iCub anthropomorphic hand. This paragraph details the kinematic structure of each of these hands describing their corresponding degrees of freedom.

4.1. The Barrett hand kinematics

The Barrett hand is a three-fingered robotic hand. While one finger often called the thumb is stationary, the other two fingers can spread synchronously up to 180 degrees about the palm. Each finger has two joints but only the proximal link is actuated. The distal link is coupled to the proximal one and it moves with it at a fixed rate. This hand has in total 4 degrees of freedom. The maximum voluntary force that can be applied by one of its fingertip is about 15 N. Figure (2) shows the real Barrett hand and its corresponding model in our simulator.

4.2. The iCub hand kinematics

The iCub robot has an anthropomorphic dextrous hand which can adopt a wide range of configurations and, hence, of grasp typologies. Each finger has three phalanges and their articulations mimic the *MP* (Metacarpophalangeal), *PIP* (Proximal Interphalangeal) and *DIP* (distal Interphalangeal) joints of the human hand (Fig. 3). Moreover, according to this hand physiology, the *PIP* and the *DIP* joints are coupled in each finger. Thus, the thumb, index and middle fingers have two degrees of freedom each to control for flexion/extension movements. The thumb has one additional degree of freedom for its opposition movement. The little and the ring fingers are fully coupled together. In summary, the iCub hand has 9 degrees of freedom distributed as follows: 3 for the thumb, 2 for the index, 2 for the middle finger, 1 for the ring and little fingers and 1 for the adduction/abduction movement. The motors used to actuate each finger have a torque limit of $21.5 N \times mm$. Given that the average length of one finger is about 70 mm, the maximum voluntary

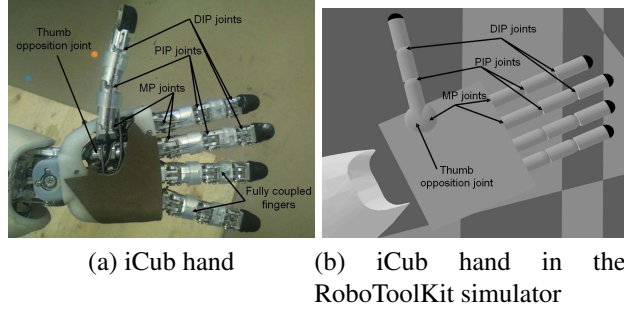


Figure 3: The real iCub hand and its corresponding model in our RoboToolKit simulator.

force that can be applied by the iCub fingertip is about $0.3 N$. Thus, this hand is only capable of grasping light weight objects.

Since the little and the ring fingers are fully coupled together, in the remainder of the paper, we generate grasps using only the thumb, index and middle fingers.

4.3. Finger kinematics

In order to compute the fingertip position of each hand, a kinematic model for each finger is needed. Figure 4 shows a detailed model of the iCub thumb. The revolute joints are represented as cylinders aligned with their axes, noted as \mathbf{r}_i^j ¹. An orthonormal reference frame $R_i^j = (\mathbf{e}_i^j, \mathbf{m}_i^j, \mathbf{r}_i^j)$ is attached to each joint, where $i = 1, \dots, q^j$, $j = 1, \dots, 3$ stand respectively for the joint and finger numbers², q^j is thus the number of DOFs of the j -th finger. The R_i^j are expressed according to an orthonormal reference frame, $R_h = (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3)$, attached to the hand palm. R_h is expressed in the object reference frame, $R_o = (\mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3)$, which is the origin of the system. A similar model can be defined for the Barrett hand. In this case, each finger has only two joint angles that are coupled. We modeled these joint angles separately and then add their coupling as a constraint in our optimization.

5. Object modeling

We want to solve grasp synthesis as an optimization problem starting from a position of the hand away from the object and converging towards a good grasp. An implicit representation of the object surface is needed in order to test at each step of the optimization whether the fingers are inside, outside or placed on the object surface. Implicit surfaces are contours or isosurfaces, which can be described as the set of all the $x \in \mathcal{X} \subseteq \mathbb{R}^d$ for which the function $g : \mathcal{X} \rightarrow \mathbb{R}$ equals to zero. Thus g gives a description of the object shape by telling for each location in space, \mathbf{p} , whether it is part or not of the object. A point \mathbf{p} is defined by its 3D cartesian coordinates (p_1, p_2, p_3) :

¹ In the rest of this document, we use bold, normal and capital letters to represent respectively vectors, scalars and matrices.

²In our simulation, we use the D-H (Denavit-Hartenberg) parameters of the iCub's hand that are available in the RobotToolkit simulator to compute these quantities, see <http://lisa.epfl.ch/RobotToolkit/>

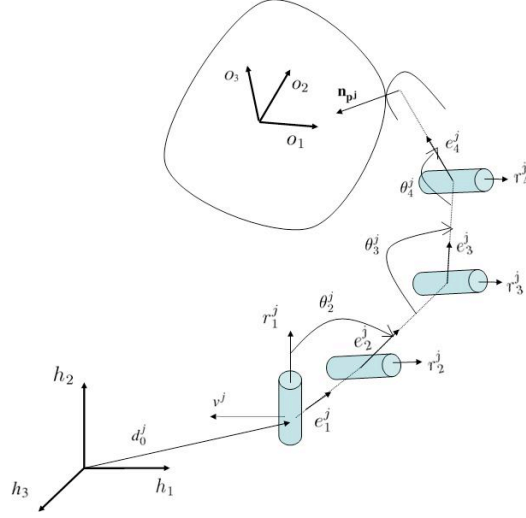


Figure 4: Kinematic structure of the iCub thumb, adapted from [26]. $R_h = (\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3)$ and $R_o = (\mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3)$ are the reference frames attached respectively to the hand palm and the object. The revolute joints are represented as cylinders aligned with their axes, noted as r_i^j . The $R_i^j = (e_i^j, m_i^j, r_i^j)$ are the reference frames attached to each joint. The m_i^j vectors are not shown for simplicity on the display. R_1^j represents the revolute joint responsible for the thumb opposition movement. R_2^j, R_3^j and R_4^j represent respectively the MP, PIP and DIP joints. n_p^j is the normal vector at the contact point. d_0^j is the finger anchor point defined relative to the hand reference frame. $\{\theta_i^j\}$ are the joint angles with $i = 1, \dots, 4$ in the thumb case. v^j is a vector whose orientation is fixed relatively to the palm, it is useful for measuring θ_1^j .

$$\begin{aligned}
 g : \mathbb{R}^3 \rightarrow \mathbb{R}; g(\mathbf{p}) &= g(p_1, p_2, p_3) \\
 &= \begin{cases} < 0, & \mathbf{p} \text{ is inside the object} \\ 0, & \mathbf{p} \text{ is on the object surface} \\ > 0, & \mathbf{p} \text{ is outside the object} \end{cases}
 \end{aligned}$$

We use a probabilistic method to represent complex object shapes as a single implicit function.

5.1. Modeling objects using Gaussian Processes

Gaussian Process (GP) is a standard method in machine learning that has been extensively applied to regression and classification problems [25]. GP have been used as well for object category recognition [6], or to fuse several sensorial information (visual, tactile, laser) in a probabilistic manner to represent object shape [14]. Starting from a 3D point cloud representing the object, we use GP to learn the implicit function g that represents the object surface. The different steps of the training procedure are given in algorithm (1). We detail next each step of the algorithm.

1- Data collection:

The first step towards learning an implicit function representing the object surface is the collection of a set of training data $\{\mathbf{p}^i \in \mathbb{R}^3, y^i \in \mathbb{R}\}_{i=1, \dots, m}$ by randomly sampling m 3D points \mathbf{p}^i belonging to the object cloud model (corresponding y^i value is 0) or lying outside or inside the object shape (corresponding y^i values are set to 1 and -1 respectively). Objects in this paper were trained by

Algorithm 1: Implicit surface reconstruction using GP

1 Data collection

Randomly select a set of training data points $\{\mathbf{p}^i, y^i\}_{i=1, \dots, m}$

2 Model learning

Learn the kernel parameters λ and σ by minimizing eq.(4)

3 Model testing

Generate a set of testing points $\{\mathbf{p}^* \in \mathbb{R}^3\}$

For All \mathbf{p}^* that satisfy $y^* = g(\mathbf{p}^*) = 0$

compute $s = \max cov(y^*)$, refer to eq.(6)

End

If $s > Threshold$

add a new training data point $\{\mathbf{p}, y\}$ belonging to the object point cloud model

Go to step 2

End

4 End

randomly sampling 20 data points from outside the object surface (sampled from a sphere englobing the object shape) and one data point from inside (object center). The number of data points belonging to the object surface is incrementally increased until the modelling error is below a threshold.

2- Model learning:

Given the training data set $\{\mathbf{p}^i, y^i\}_{i=1, \dots, m}$, the implicit surface function representing the object using GP is given by:

$$g(\mathbf{p}) = \sum_{i=1}^m \alpha^i k(\mathbf{p}^i, \mathbf{p}) \quad (1)$$

$$k(\mathbf{p}^i, \mathbf{p}^j) = \lambda^2 \exp\left\{-\frac{\|\mathbf{p}^i - \mathbf{p}^j\|^2}{\sigma^2}\right\} \quad (2)$$

$$[\alpha^1, \dots, \alpha^m]^T = (K + B)^{-1} \mathbf{y}, \quad \text{with } \mathbf{y} = [y^1, \dots, y^m] \quad (3)$$

Where $k : \mathbb{R}^3 \times \mathbb{R}^3 \rightarrow \mathbb{R}$ is the gaussian kernel as defined in equation(2); λ and $\sigma \in \mathbb{R}$ are the hyperparameters to learn; $B \in \mathbb{R}^{m \times m}$ is a diagonal matrix, representing the noise on the output; K is a $m \times m$ real matrix, composed in each entry of the kernel $k(\mathbf{p}^i, \mathbf{p}^j)$, $i, j \in 1, \dots, m$ computed at each pair of points.

The learning procedure consists in determining the parameters λ and σ by maximizing the likelihood or minimizing the negative log marginal likelihood \mathcal{L} , [25]:

$$\begin{aligned}\mathcal{L} &= -\log p(\mathbf{y}|\lambda, \sigma) \\ &= \frac{1}{2} \log \det(K + B) + \frac{1}{2} \mathbf{y}^T (K + B)^{-1} \mathbf{y}\end{aligned}\quad (4)$$

3- *Model testing:*

Once the parameters λ and σ are determined, the predicted value $y^* \in \mathbb{R}$ of the implicit function g at a new testing point $\mathbf{p}^* \in \mathbb{R}^3$ and its variance $cov(y^*) \in \mathbb{R}$ can be written as:

$$y^* = \sum_{i=1}^m \alpha^i k(\mathbf{p}^i, \mathbf{p}^*) = \mathbf{k}_*(K + B)^{-1} \mathbf{y}, \quad (5)$$

with $\mathbf{k}_* = [k(\mathbf{p}^*, \mathbf{p}^1), \dots, k(\mathbf{p}^*, \mathbf{p}^m)] \in \mathbb{R}^m$

$$cov(y^*) = k(\mathbf{p}^*, \mathbf{p}^*) - \mathbf{k}_*(K + B)^{-1} \mathbf{k}_*^T \quad (6)$$

The testing points are generated by uniformly sampling a box englobing the object. The uncertainty of the model is denoted as the maximal variance (equation 6), s , of the testing points belonging to the estimated object surface. If s is larger than a threshold, a new data point is sampled from the object surface and is added to the training data. The threshold value is set empirically to 0.01. Figure(5) illustrates this notion of uncertainty on a 2D object. Starting from an object illustrated as a red contour, the learned object model using GP is represented in black. The yellow dots show the training data points. Notice that the variance, used to check the model accuracy, decreases in regions where many training data points were sampled and increases in regions where only few training data points were considered. The uncertainty of the model is the maximum variance along the object contour. For example an uncertainty of 0.04 in the 2D object example means that the real contour can be located between the two curves labeled 0.2 and -0.2 in Figure(5).

We tested the GP modelling technique with a simple one-part object, a cylinder, with multi-part objects such as a light bulb and a bottle, with more complex objects such as a cup, a duck and a spray bottle which shape has several fine features (Fig. 6). Table (1) shows the number of points used for training a GP model for each object as well as its corresponding uncertainty and computation time on a 8 GB machine with a CPU at 2.4 GHz.

6. Constraints for generating feasible grasps of good quality

This paragraph details the different constraints required for the generation of feasible grasps of good quality. Such constraints include the fingertip placement on the object surface, stability of the grasp through the force-closure criterion, collision avoidance and the choice of a quality measure.

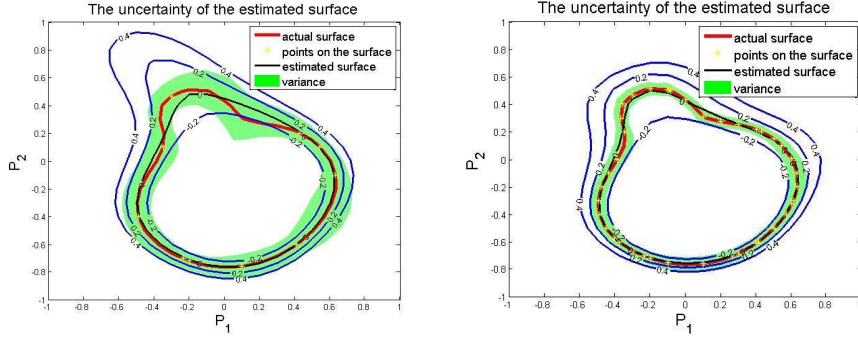


Figure 5: Illustration of 2D object GP model uncertainty. 2D object contour is in red. The yellow dots are the learning data points. The learned GP model is shown in black. The variance along the learned contour is shown in green. The blue contours are the iso-contours of the learned model. Left: model learned using 10 data points on the object contour. Right: model learned using 40 data points.

Table 1: The number of training data points used to generate a GP model for the object, the GP model uncertainty and computation time.

objects	Number of data points	uncertainty	Time (sec)
cylinder	191	0.0026	7.3
duck	241	0.0075	15.1
bottle	301	0.0075	23.5
cup	211	0.0458	13.4
lightBulb	171	0.0031	13.0
sprayBottle	361	0.0292	26.6

6.1. Computing fingertip locations

Using the hand kinematic model described in section 4, we can express the positions \mathbf{p}^j , $j = 1, \dots, 3$ of the fingertips in the object's frame of reference:

$$\mathbf{p}^j = \mathbf{d}_h + R_h \cdot \sum_{i=0}^4 \mathbf{d}_i^j \quad (7)$$

\mathbf{d}_h is the palm's position in the object's reference frame and \mathbf{d}_i^j is the vector connecting two successive reference frames R_i^j and R_{i+1}^j . A precision grasp is obtained when a contact is established between each considered fingertip and the object's surface. Since we have at our disposal an analytical function $g : \mathbb{R}^3 \rightarrow \mathbb{R}$ that describes the object's surface, see Section 5, when each finger is in contact with the object, each \mathbf{p}^j satisfies:

$$g(\mathbf{p}^j) = 0, \quad j = 1, \dots, 3 \quad (8)$$

Equation (8) forms a set of 3 constraints that, if satisfied, guarantee that all three fingers are in contact with the object. However, this is not sufficient to ensure that the resulting grasp is feasible for our robot hand. To ensure that this is the case, we need an additional set of constraints

that take into account the range of motion of each of the joint angles. Since these constraints are easily derived from standard transformations across frames of reference, we have moved them into appendix 13.1.

6.2. Generating Force-closure grasps

To recall, a grasp is said to be force-closure when the origin of its wrench space is contained inside the convex hull of the contact wrenches [21]. Computing force-closure grasps requires to compute the location of the fingertips on the object surface as well as their corresponding contact normals. Figure 7 shows a force-closure grasp on a bottle. Each finger j , $j = 1 \dots 3$, is described by its position \mathbf{p}^j and its normal $\mathbf{n}_{\mathbf{p}^j}$ at the object's surface. We assume a frictional contact point model and thus three contacts are sufficient to achieve force-closure [17]. The approach, however, can be easily generalized to compute n -finger grasps. To simplify the expression of the normals at the surface, we compute the fingertips position relative to a frame of reference attached to the object. The origin is located at the center of mass and the axes are aligned with the object's main axes of symmetry.

Next we briefly redefine the expression of friction cone and wrench for our particular problem and use this to derive the constraints to guarantee force-closure.

6.2.1. The friction cone

Assuming static friction at all contact points, for each fingertip, we can define a *friction cone* at \mathbf{p}^j centered about the contact point internal normal $\mathbf{n}_{\mathbf{p}^j}$. To ensure that, once in contact, the finger will not slip along the surface of the object, the force \mathbf{f}^j applied by the finger at point \mathbf{p}^j must satisfy Coulomb's law [15]. This law states that \mathbf{f}^j must lie within the *friction cone*. Generally, the friction cone is linearized by a polyhedral convex cone with m sides. Under this approximation, the grasp force can be represented as:

$$\mathbf{f}^j = \sum_{i=1}^m \lambda_i^j \mathbf{l}_i^j, \quad \lambda_i^j \geq 0 \quad (9)$$

where \mathbf{l}_i^j represents the i th edge vector of the polyhedral convex cone, see Figure 7. The coefficients λ_i^j are non negative constants and are determined by the friction coefficient μ of the surface material, see Appendix 13.2.

6.2.2. Contact wrenches

A contact wrench, $\mathbf{w}^j \in \mathbb{R}^6$, is the combination of the grasp force \mathbf{f}^j and its corresponding torque:

$$\mathbf{w}^j = \begin{pmatrix} \mathbf{f}^j \\ \mathbf{p}^j \times \mathbf{f}^j \end{pmatrix} = \sum_{i=1}^m \lambda_i^j \begin{pmatrix} \mathbf{l}_i^j \\ \mathbf{p}^j \times \mathbf{l}_i^j \end{pmatrix} = \sum_{i=1}^m \lambda_i^j \mathbf{w}_i^j \quad (10)$$

Where $\mathbf{w}_i^j \in \mathbb{R}^6$, $i = 1, \dots, m$, is called a primitive contact wrench associated to the j th contact point \mathbf{p}^j . In the case of a three-finger grasp, one has $3 \times m$ primitive contact wrenches.

6.2.3. Force-Closure criterion

Force-closure is achieved when the origin of the wrench space lies strictly inside the convex hull of $\{\mathbf{w}_i^j\}$. This condition can be formulated as follows [21]:

$$\begin{aligned} \exists \phi_i^j \in \mathbb{R}, \phi_i^j > 0, \sum_{i,j} \phi_i^j = 1, j = 1, \dots, 3, i = 1, \dots, m \\ \text{s.t. } \sum_{i,j} \phi_i^j \mathbf{w}_i^j = 0 \end{aligned} \quad (11)$$

Where ϕ_i^j are positive scalars ensuring that the primitive wrenches can positively span \mathbb{R}^6 and hence resist any external wrench.

6.3. Force feasibility and Grasp quality criterion

In order to select an optimal grasp among a set of force-closure grasps, the magnitude of the external wrench that can be equilibrated in the worst direction with unit contact forces is generally used in the literature as a grasp quality criterion, see [31] for a review. In other words, this criterion tries to equilibrate external wrenches in all directions. But specific tasks may require the hand to generate wrenches in particular directions. Hence to allow generating such task-specific grasps, given a desired external wrench, \mathbf{W}_{ext} , we use as quality measure the minimum joint torque required to equilibrate that wrench. Minimizing that measure induces a minimization of the energy required in the joint space in order to accomplish the corresponding task [20]. This quality can be formulated as:

$$Q(\mathbf{p}) = \sum_{i,j} \|\tau_i^j(\mathbf{p})\| \quad (12)$$

Where τ_i^j stands for the i th joint torque of the j th finger. When considering such a quality criterion, one needs to include constraints related to the limitation of the finger joint torque and force equilibrium. These can be written as:

$$\tau_i^j \in [\bar{\tau}_i^j, \hat{\tau}_i^j] \quad (13)$$

$$\tau^j = J^{jT} \mathbf{f}^j \quad (14)$$

$$\sum_{i,j} \alpha_i^j \mathbf{w}_i^j = -\mathbf{W}_{\text{ext}}, \alpha_i^j \geq 0 \quad (15)$$

Where J^j is the jacobian matrix of the j th finger, α_i^j are positive scalars and $\bar{\tau}_i^j, \hat{\tau}_i^j$ are respectively lower and upper bounds on τ_i^j . Note that equation (15) is different from equation (11) in two aspects: first it balances one specific external wrench instead of trying to equilibrate wrenches in any direction and second it takes into account the limitation on the finger contact forces instead of assuming normalized contact forces. The external wrench considered in our case is the one due to gravity.

6.4. Collision avoidance

Our objects are described by an implicit surface function. This formulation allows to detect whether a point is inside, outside or on the surface of the object. Therefore, collision avoidance between the object and the hand is obtained by sampling several points \mathbf{c}^{kj} , on the finger phalanges and palm; we compute the distance from these points to the object surface and add a new set of constraints to ensure all points are outside the object’s surface¹:

$$g(\mathbf{c}^{kj}) > 0, \text{ for all } k_j \quad (16)$$

Where k_j is the number of sampled points on the j th finger. Collision avoidance between the fingers is taken into account in the kinematic feasibility of the hand by considering the finger joint limits. This condition was enough to avoid collisions between the fingers in the case of the Barrett and iCub hands.

7. Solving Grasp Synthesis as an optimization problem

Given a hand kinematical model and a representation of the object, this paragraph details the formulation of grasp synthesis as an optimization problem guaranteeing stability through the generation of force-closure grasps and optimality through choosing a force-related quality measure as an objective function. In other terms, our objective is to find a hand position/orientation and fingers joints configurations that guarantee a good force-closure grasp on a 3D object approximated by an implicit function. This yields a constraint-based minimization for the set of parameters $\theta = \{\phi_i^j, i = 1..m, j = 1..3, R_h, \mathbf{d}_h, R_i^j, i = 1..q^j, j = 1..3\}$, given by:

$$\underset{\theta}{\operatorname{argmin}} Q(\mathbf{p})$$

under the constraints (8), (16), (11), (13 – 25)

While our objective function is convex, the constraints are not linear and hence this becomes a difficult optimization problem. A number of primal-dual interior-point methods have been proposed to solve such problems. However, most of these methods converge to infeasible solutions when dealing with problems having two or more equality constraints and a total number of constraints exceeding the dimension of the space [36] (as is the case in our problem). The Interior Point OPTimizer (IPOPT) method proposed by Wächter and Biegler [36] offers an alternative method for non-linear optimization that is robust to change in initial conditions (as for instance when starting a reach and grasp motion very far away from the target and in a very awkward hand position). We successfully applied this optimization technique on our grasp synthesis problem. The algorithm is written in the Modeling Language for Mathematical Programming (*AMPL*) for the specific case of the iCub and Barrett hand models. Note that by formulating the problem in *AMPL*, no analytical gradient computation is required. We exploit the fact that the IPOPT solver generates locally optimal solutions to generate multiple possible solutions. As stated before, this diversity in the possible grasps is a richness in manipulation capability that may prove useful in complex manipulation.

¹ We sampled, for collision detection, 3 points on each finger yielding a granularity of 23.3mm and one on the palm.

8. Experimental Results

The experiments we conducted aimed at testing the algorithm’s ability to generate for one object, a variety of grasps that are feasible for the robotic hand and result in a high quality according to the metric given in Eq. (12). This variety of grasps allows the hand to be shaped differently according to the task to accomplish. For example, many objects from our daily life have a graspable cylindrical-shaped part but are grasped differently by this part according to their functionality such as pens, cups, toothbrushes, wine glasses, frying pans, knives, tea storage pots, etc. In order to be able to grasp these different objects accurately, a grasping algorithm should be able to synthesize this variety of hand/fingers placement. Therefore, we choose to test our algorithm on a cylinder and make sure that the computed grasps explore the grasps solution space instead of converging only towards the standard cylindrical grasp (side grasp where the thumb is opposing the other two fingers and where the palm is placed in a parallel position to the cylinder main axis). In a second experiment, we test the ability of the algorithm to generate grasps on much more complex shapes. The experiments were run on an AMD Opteron machine with 47 GB and a CPU at 2.4 GHz.







8.1. Generating a variety of grasps for a cylinder

In order to test the robustness of the algorithm to changes in the initial conditions, we generate 42 orientations of the palm that span uniformly the different directions in space, initialized from 3 different locations (the palm is positioned on the diagonal of the reference system attached to the center of the object, the palm aligned with the middle of the object height and the palm placed above the object), for a total of 126 different initial hand postures, (fig. 8). Starting from each initial configuration, we computed grasps on a cylinder, for 5 different friction coefficients varying from 0.1 (corresponds to the coefficient between steel and graphite) to 0.9 (corresponds to the coefficient between rubber and paving) and 3 different weights, yielding a total of 1890 trials. For simplicity reasons and to avoid introducing unnecessary imprecision, we use a superquadric model [9] instead of a GP one to represent the cylinder. We use a 8-sided pyramid to represent a linear model of the friction cone. In the following, we present the results obtained for the Barrett and iCub hands.

8.1.1. Results for the Barrett hand

The maximum voluntary force and maximum torque that can be applied by each fingertip of the Barrett hand are respectively about 15 N and $1.92N \times m$. By varying the object weight from 10 N to 30 N, we vary the weight to maximal force ratio from 0.22 to 0.66. For the 1890 trials, the algorithm converged to 791 locally optimal solutions that are both force-closure and satisfied the kinematic constraint of the Barrett’s hand. Many of these grasps were similar in terms of hand/fingers configurations, thus, in order to have an idea about the variety of grasps configurations obtained, we clustered these into 11 groups, based on the hand position, orientation and joint angles, using the K-means algorithm. The number of clusters was chosen empirically through visual inspection of the clusters obtained after training. Figure (9) shows an example of a typical grasp for each of the 11 clusters. The average computation time to find a solution was 12.14 s with a standard deviation of 12.27 s. We provide in figure (10) the number of grasps configurations

Table 2: Quality (minimum torque in $N \times m$) of the grasps generated with the Barrett hand (friction coefficient = 0.9, weight to maximal force ratio = 0.44)

<i>Label</i>	3	8	4	11	7	10
<i>Quality</i>	4.80	4.80	4.41	4.02	3.76	3.38
<i>Normalized_Quality</i>	0.83	0.83	0.77	0.70	0.65	0.59
<i>Grasp</i>						

obtained as a function of the friction coefficient for the 3 different weights. Unsurprisingly, more diversity in the grasps configurations appears when increasing the friction coefficient. For a weight to maximal force ratio of 0.44 and with a friction coefficient equal to 0.1, the algorithm yielded only one grasp belonging to the cluster 10. With a friction coefficient of 0.5, two additional grasp types (clusters 3, 7) are obtained. Finally, for a high friction coefficient of 0.9, grasps from categories 3, 4, 7, 8, 10, 11 were found. Table 2 summarizes the grasps quality obtained for a friction coefficient of 0.9. The normalized quality is also given and is the ratio between our quality criterion which is the joint torque required for lifting the object with a specific grasp and the maximum joint torque the hand is capable of. This is equal to $1.92 \times 3 = 5.76(N \times m)$. The optimal grasp in this case is the one labeled 10 and is the one obtained for a low friction coefficient. As we increase this coefficient, the algorithm starts to converge to a variety of grasps including the optimal ones as well as grasps of a lower quality.









Notice that the typical preshape to grasp a cylinder is illustrated in the cluster labeled 5. The other clusters generate interesting grasps, which exploit the dexterity of the three fingers of the hand, by orienting/positioning the hand relative to the object to make these finger configurations feasible. In the following paragraph, we show several computed grasps for the iCub anthropomorphic hand. The resemblance between the iCub and human hands will illustrate more clearly the relevance of the generated grasps to the accomplishment of everyday life tasks.

8.1.2. Results for the iCub hand

The iCub hand has 9-dof. Each finger has a torque limit of $21.5 N \times mm$. Given that the average length of one finger is about $70 mm$, the maximum voluntary force that can be applied by the iCub fingertip is about $0.3 N$. By varying the object weight from $0.2 N$ to $0.6 N$, we vary the weight to maximal force ratio from 0.22 to 0.66 similarly to the Barrett hand case. For the 1890 trials, the algorithm converged to 612 locally optimal solutions that are both force-closure and satisfied the kinematic constraint of the iCub's hand. We clustered these into 20 groups using the K-means algorithm. Figure 11 shows an example of a typical grasp for each of the 20 clusters².

²For some grasps, the finger phalanges seem to penetrate the object. This is normal and does not violate our collision avoidance constraints, since we sampled, for collisions, points that are located on the finger links without

Table 3: Quality (minimum torque in $N \times mm$) of the grasps generated with the iCub hand (friction coefficient = 0.9, weight to maximal force ratio = 0.44)

<i>Label</i>	15	9	7	2	11	5	6	8
<i>Quality</i>	53.2	52.7	48.0	40.9	40.6	37.5	32.9	30.3
<i>Normalized_Quality</i>	0.82	0.82	0.74	0.63	0.63	0.58	0.51	0.47
<i>Grasp</i>								

The typical preshape grasp is used in this case in 7 of the 20 clusters. Here are some of the interesting, non-standard grasps generated by the algorithm. Figure 12 shows that the hand configuration in the grasp number 10 resembles that adopted when we pick a champagne glass. Grasp number 13 is useful when holding a takeaway mug with a lid. Grasp 8 is typically chosen when carrying a flower pot from the floor. Figure 13 shows how that the grasp number 4 is used when flipping an object upside down using only one hand. Figure 14 illustrates the usability of grasps 19, 20 and 16. Grasp 19 is useful when opening a box or a bottle while grasp 20 is a polite hand configuration when offering something to someone. Grasp 8 is often used by waiters since it enables them to free the ring and small fingers for holding a second object.

The average computation time to find a solution was 2.65 s with a standard deviation of 1.82 s. We provide in figure 15 the number of grasps configurations obtained as a function of the friction coefficient for 3 different weights. Similarly to the Barrett hand example, the number of grasps configurations increases with the friction coefficient. For a too high weight ratio and too low friction coefficient, no feasible solutions were found, (Fig. 15). As the friction coefficient increases, more diversity in the grasps appears. Table 3 summarizes the grasps quality obtained for a friction coefficient of 0.9 and a weight to maximal force ratio of 0.44. The optimal grasps in this case are the ones labeled 6 and 8.

8.2. Generating grasps on complex objects

At this point, we showed the algorithm ability to compute a variety of grasps on a simple cylindrical shaped object modeled as a superquadric. In this experiment, we show the ability of the algorithm of to cope with more complex objects. These objects are modeled using Gaussian Processes.

Figure (16) shows grasps generated on different multi-part objects modeled with Gaussian Processes as a single implicit continuous function. The average computation time it takes the solver to generate a feasible solution on a GP object model is given in table (4) for the Barrett hand and in table (5) for the iCub hand. Notice for example the average computation time for generating a grasp for the iCub hand on a cylindrical shaped object when modeled using GP with 191 training

taking into account the radius of the cylinders covering them. These cylindrical shapes were included later on into our RobotToolKit simulator.

Table 4: Grasp computation time (for Barrett hand) Vs data points number used for modelling the object.

	Nb. points	mean (s)	std (s)
<i>Spray_bottle</i>	361	18.13	7.37
<i>Toy_duck</i>	241	12.38	6.35
<i>Cup</i>	271	10.35	6.78

Table 5: Grasp computation time (for iCub hand) Vs data points number used for modelling the object.

	Nb. points	mean (s)	std (s)
<i>Cylinder</i>	191	11.80	15.76
<i>bottle</i>	301	29.50	23.64
<i>Toy_duck</i>	241	19.83	18.42
<i>Lightbulb</i>	171	5.30	5.05

data points (11.8s) and when modeled using superquadrics with only 5 parameters (2.65s). Notice as well the variation of this computation time across different object shapes. When a complex object such as a spray_bottle is modeled using GP, it requires more training data points than a cylinder and thus more time is required to generate a grasp.

The grasps computed for the cylindrical shaped object in the simulator were very precise, in the sense that the placement of the hand allowed the fingers to be positioned exactly on the object surface. No further planning or adjustments were needed to perform the grasp. For complex objects modeled with GP, we realized that sometimes the fingers may not reach exactly the object surface. This is due to the imprecision in the object model which is not uniform across the object shape as shown in Figure (5). In this case, to ensure a contact with the object surface, the finger joints should be incrementally increased until a good contact with the object is reached. This is tackled in the next section.

9. Implementation on the real robots

In the following, we will detail the implementation of the grasping strategy on the real iCub robot and Barrett hand. Objects are modelled using GP and their position and orientation were sensed using the OptiTrack Vision system. Note that, the GP model is needed here solely for computing off-line the object model and feasible grasps (refer to tables 1, 4, 5 for the GP model computation time). For this reason, the markers are needed in order to localize the object in the scene (Figure 17).

9.1. Implementation on the Barrett hand

Figure 18 shows the implementation of the computed grasps for the flask bottle on the Barrett hand mounted on the KUKA LightWeight Robot (LWR) . As discussed in the previous section,

the fingertips may not be positioned exactly on the object surface due to imprecisions from the object modeling; Even when all the three fingertips are in contact with the object surface, the object may still slip or be deformed because of too low or too large contact forces applied at the fingertips. Thus, a pure position control of the robotic hand is not sufficient for the implementation of the computed grasps and there is a need for controlling the forces applied at the fingers. Our optimization algorithm computes the hand position, orientation, finger joint angles and also gives the optimal grasping forces $f^j, j = 1..3$, that need to be applied by each fingertip on the object (equation 14). Since it is not possible to precisely control the Barrett hand joint torques, we mounted the Syntouch tactile sensors on its fingertips [37]. Once calibrated, these sensors allow us to control the contact forces applied on the object surface enabling a compliant control of the Barrett hand. The controller for each finger is implemented as:

$$\delta\theta^j = K(f_d^j - f^j), \quad (17)$$

Where f_d^j and f^j are respectively the desired and current normal forces at the j th fingertip³, θ^j is the proximal joint angle of the j th finger⁴ and K is a scalar. The joint angles are controlled according to a force feedback from the fingertips⁵. When f^j is less than f_d^j , $\delta\theta^j$ is positive and the finger keeps closing until the desired contact force is reached.

9.2. Implementation on the iCub hand

The grasps computed for the duck, bottle and lightbulb objects, modelled using GP, were implemented on the real iCub robot. Once the object configuration is determined using the vision system, the robot grasps it shaping its hand according to one of the pre-computed optimal grasps. Notice that in the real world, the robot hand configuration relatively to the object could not be exactly identical to the precomputed one. We had sometimes to slightly adjust it to match the one in simulation. This is due mainly to the design of the iCub hand which only gives a rough estimation of the hand reference frame localisation, to a slack in the iCub wrist joint, and to cumulative imprecisions in the iCub arm forward kinematic chain. Figure 19 illustrates the grasps obtained on these real objects. No force control was performed on the iCub hand; on one hand there is no possibility to control the hand joint torques and on the other hand the syntouch sensors cannot be mounted on the iCub fingers.

10. Discussion

In spite of the non-linearity and the high-dimensionality of the grasping space, the previous results illustrate the ability of the proposed algorithm to compute kinematically feasible force-closure grasps of high quality. This problem was addressed previously through techniques based

³These normal forces correspond to the normal component of f^j .

⁴The proximal and distal joint angles are coupled. Only the proximal one can be controlled.

⁵A video showing the reaching and grasping of the flask bottle can be found following this link: lasa.epfl.ch/~miao/grasp_demo_RAS.wmv. The reaching algorithm is based on a coupled dynamical system between the two systems driving the hand position/orientation and finger motions. [30].

on preshape of the hand [18, 7]. There, a specific shape of the fingers is predetermined by the object's shape, which is known to be close to the optimum. The fingers need only to close on the object to reach the desired grasping points. Such techniques are intuitive and draw from an abundant literature on similar control in biology. However, this strategy may not yield successful grasps when the hand must adopt very particular postures, as shown in our example of Figure 12. Our approach offers the possibility to generate a large variety of hand/finger postures. Which of these should be used when, is then to be decided depending on the task requirement. Of course, this still relies on a controller to bring the hand from its current position to the position found by our solver. This position may not necessarily be the position that is the closest to the current robot's position. This could however be tackled by adding a further term in the objective function and solve the same way as done here (since IPOPT is not constrained to using convex objective function). We also developed a simple compliance controller in order to deal with the uncertainties on the object model. For a static grasp, as considered in this paper, this controller is sufficient to guarantee the grasp stability. However, in a dynamical context, where the object is subject to a perturbation or when the object weight is changing (pouring water example), a more advanced compliance controller should be employed. This controller should adaptively change the compliance behavior according to the object properties or external perturbations. This is a topic we are currently working on.

As for the approach computation time, it is primarily influenced by the complexity of the search space and the difficulty faced by the optimizer to find a feasible and optimal solution. The number of degrees of freedom affect the complexity of this search space, but it is not possible to provide a notion of growth in complexity as an effect of this number. In our study, we computed grasps for the same cylindrical part modeled with Gaussian Processes (GP) using both the iCub and the Barrett hand. The Barrett hand has 7 degrees of freedom, 4 of them fully actuated and the iCub hand has 9 actuated degrees of freedom. The average computation time for the Barrett hand to find a solution was 12.14 sec and the one for the iCub hand was 11.8 sec. For the two hands considered in this work, we see that the computation time is not affected by an increase in the degrees of freedom and reflects primarily the complexity of the search space. Another point worth mentioning is that the applicability of the method on-line in a real world scenario without using any markers will depend on the capability to compute GP models or other analytical object models online (to our knowledge there is no such online object modeling methods). We here chose to use a slow method to compute optimal grasps. While slow, the method had the advantage to allow us to express a set of new constraints to take into account the kinematics of the hand. A follow-up paper of ours developed a technique to build a probability distribution function representation of these feasible grasps which can then be sampled in real time, [13]. To generate a grasp, one no longer needs the GP model. Having solely the position and orientation of the object is sufficient to choose the most appropriate grasp.

Note that, the approach proposed here is used to generate precision grasps. It could in principle be extended for generating power grasps. In this case, desired contact points on the palm and the different phalanges of the fingers need to be specified and corresponding constraints need to be added to the optimization algorithm. This would however increase the complexity of the problem. This would be a valuable extension of this work.

The reader should notice that IPOPT does not guarantee convergence to the global optimum

and consequently different starting configurations result in different grasps. While local optimality is often seen as a drawback, we saw that in our experiments, this led to a variety of high quality force-closure grasps. The fact that the algorithm converges to several solutions rapidly is encouraging and indicates that one could add task-specific criteria in the objective functions to constrain further the search. One important detail influencing the computation time is the way we formulate the constraints. For generating grasps on a cylinder modeled as a superquadric with the Barrett hand, we use the non-linear functions cosine and sine in the computation of the kinematical chain permitting us to calculate the fingertip positions. For generating grasps with the iCub hand, we replaced these non-linear functions with their equivalent inner product between vectors, see appendix(13.1) for more details. With these two different formulations of the same constraints, we noticed that the average computation time for generating a grasp with the Barrett hand was of 12.14s, while the one with the iCub hand, having more degrees of freedom, was of 2.65s. Even though the IPOPT solver is able to handle non-linear constraints, one can accelerate the computation time by replacing non-linear formulations with their equivalent linear ones when possible.

We showed in this paper grasps computed on two kinematically different hands, the Barrett and the iCub ones. Our approach is not restricted to a specific hand kinematic structure. The constraints in Equations (18-25) could be adapted to describe any other hand kinematics. Similarly, the use of superquadrics or Gaussian Processes to describe the object's surface could be replaced by other analytical functions describing the shape of the object. Note that, superquadrics or Gaussian Processes are only an approximation of the object surface and thus the optimization algorithm will not guarantee the placement of the fingertips exactly on the object surface. However, in order to have a better approximation of the local contact area between the fingers and the object surface, one could rely on a meshing of these two surfaces and some heuristics to compute distances between points in the mesh to determine potential contact. While this may help to rule out some impossible grasps, this should be balanced with the increased costs in computation while not giving any true guarantee that there would be a good contact. Consequently, there will be in all cases an amount of imprecision when it comes to approximating the shape of the fingers and of the local object surface. As mentioned previously in this section, there will always be a need to rely on tactile feedback and compliant control for a placement of the fingers guaranteeing stability.

11. Conclusion

This paper shows that a large variety of good grasps can be obtained by formulating and solving grasp synthesis as a non-linear optimization problem taking into account all the degrees of freedom involved, those of the hands as well as those related to the object. We performed experiments on two kinematically different hands based on two different descriptions of the object surface, showing the ability of the algorithm to cope with any hand kinematic structure and different object models. In contrast to existing optimal force-closure grasps generation methods that compute first several grasps and then select the optimal one among a set of these, our approach generates in one optimization round grasps of high quality that are feasible for the hand kinematics and adapted to a large set of tasks.

12. Acknowledgments

We would like to thank François Margot for advices on efficiently applying the IPOPT algorithm to our problem and for giving us access to the computing facilities at the Tepper School of Business at Carnegie Mellon University, where the simulations were run. The research leading to these results has received funding from the European Community's Seventh Framework Program FP7/2007-2013 under grant agreement no288533 ROBOHOW and the Swiss National Science Foundation through the National Centre of Competence in Research (NCCR) in Robotics.

13. Appendix

13.1. Joint limit and axes orientation constraints

In order to have a grasp that is feasible for the hand, the corresponding joint angles $\{\theta_i^j\}$, $i = 1, \dots, q^j$, $j = 1, \dots, 3$ should be within the joint limits. Two different representations of the joint angles were employed. The first one uses the non-linear functions cosine and sine in the computation of the kinematical chain. The second one replaces cosine and sine with the inner product between the relevant vectors. Let $c_i^j = \cos(\theta_i^j)$, and $s_i^j = \sin(\theta_i^j)$, we have the following relationships:

$$c_1^j = \mathbf{v}^{jT} \cdot \mathbf{r}_2^j, \quad \mathbf{s}_1^j \cdot \mathbf{r}_1^j = \mathbf{v}^j \times \mathbf{r}_2^j \quad (18)$$

$$c_2^j = \mathbf{r}_1^{jT} \cdot \mathbf{e}_2^j, \quad \mathbf{s}_2^j \cdot \mathbf{r}_2^j = \mathbf{r}_1^j \times \mathbf{e}_2^j \quad (19)$$

$$c_3^j = \mathbf{e}_2^{jT} \cdot \mathbf{e}_3^j, \quad \mathbf{s}_3^j \cdot \mathbf{r}_3^j = \mathbf{e}_2^j \times \mathbf{e}_3^j \quad (20)$$

$$c_4^j = \mathbf{e}_3^{jT} \cdot \mathbf{e}_4^j, \quad \mathbf{s}_4^j \cdot \mathbf{r}_4^j = \mathbf{e}_3^j \times \mathbf{e}_4^j \quad (21)$$

Satisfying joint angle limits induce a limitation on c_i^j and s_i^j . Their corresponding lower and upper bounds are respectively $c_{low, s_{low}}$ and $c_{up, s_{up}}$:

$$c_{low}^j \leq c_i^j \leq c_{up}^j, \quad i = 1, \dots, q^j, \quad j = 1, \dots, 3 \quad (22)$$

$$s_{low}^j \leq s_i^j \leq s_{up}^j, \quad i = 1, \dots, q^j, \quad j = 1, \dots, 3 \quad (23)$$

For each finger j , the revolute joints responsible for flexion/extension have parallel axes and the abduction/adduction axis, defined by the vector \mathbf{r}_1^j is orthogonal to \mathbf{r}_2^j . These constraints can be expressed by the following equations:

$$\mathbf{r}_2^j = \mathbf{r}_3^j = \mathbf{r}_4^j, \quad j = 1, \dots, 3 \quad (24)$$

$$\mathbf{r}_1^{jT} \cdot \mathbf{r}_2^j = 0, \quad j = 1, \dots, 3 \quad (25)$$

13.2. Friction cone segments

The friction cone is linearized by a polyhedral convex cone with m sides \mathbf{I}_i^j . In the reference frame of the object, \mathbf{I}_i^j is given by:

$$\mathbf{I}_i^j = R_h \cdot R_{q^j}^j \cdot \begin{bmatrix} \mu \cdot \cos\left(\frac{2\pi i}{m}\right) \\ 1 \\ \mu \cdot \sin\left(\frac{2\pi i}{m}\right) \end{bmatrix}, \quad \text{where } i = 1, \dots, m \quad (26)$$

Where μ is the friction coefficient. Vectors \mathbf{I}_i^j are then normalized as follows: $\mathbf{I}_i^j = \frac{\mathbf{I}_i^j}{\|\mathbf{I}_i^j\|}$. That assumes that all the finger forces have the same limit.

References

- [1] A. Bicchi, *On the closure properties of robotic grasping*, International Journal of Robotics Research, vol. 14, no. 4, pp. 319-334, 1995.
- [2] C. Borst, M. Fischer and G. Hirzinger, *Calculating hand configurations for precision and pinch grasps*, IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2, pp. 1553-1559, 2002.
- [3] L.Chen and N.G. Georganas, *Region-based 3D Mesh Compression Using an Efficient Neighborhood-based Segmentation*, Journal Simulation, Vol. 84, no. 5, pp. 185-195, 2008.
- [4] M.T. Ciocarlie and P.K. Allen, *Hand posture subspaces for dextrous robotic grasping*, International Journal of Robotics Research, Vol. 28, no. 7, pp. 851-867, 2009.
- [5] N. Daoud, Jean-Pierre Gazeau, Sad Zegloul, Marc Arsicault, *A fast grasp synthesis method for online manipulation*, Robotics and Autonomous Systems 59(6): 421-427, 2011.
- [6] S. Dragiev, M. Toussaint, and M. Gienger, *Gaussian Process Implicit Surfaces for Shape Estimation and Grasping*, International Conference on Robotics and Automation, 2011.
- [7] S. Ekvall and D. Kragic, *Learning and Evaluation of the Approach Vector for Automatic Grasp Generation and Planning*, IEEE International Conference on Robotics and Automation, pp. 4715-4720, 2007.
- [8] S. El-Khoury and A. Sahbani, *On computing robust N-finger force-closure grasps of 3D objects*, IEEE International Conference on Robotics and automation, pp. 2480-2486, (2009).
- [9] S. El-Khoury and A. Sahbani, *A new strategy combining empirical and analytical approaches for grasping unknown 3D objects*, Robotics and Autonomous Systems, vol.58, no. 5, pp. 497-507, 2010.
- [10] S. El-Khoury, M. Li and A. Billard, *Bridging the Gap: One Shot Grasp Synthesis Approach*, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012.
- [11] J. Fagard, E.S. Spelke and C. von Hofsten, *Reaching and grasping a moving object in 6-, 8-, and 10-month olds: laterality aspects*, Infant Behavior and Development, vol. 34, pp. 137-146, 2009.
- [12] P. Gorce and N. Rezzoug, *Grasping posture learning with noisy sensing information for a large set of multi-fingered robotic systems*, Journal of Robotic Systems, vol. 22, no. 12, pp. 711-724, 2005.
- [13] B. Huang, S. El-Khoury, M. Li, J.J. Bryson and A. Billard, *Learning a Real Time Grasping Strategy*, IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, May 6-10, 2013.
- [14] A. Kapoor, K. Grauman, R. Urtasun and T. Darrell, *Gaussian Processes for Object Categorization*, International Journal of Computer Vision, 2009.
- [15] P.R. Kraus, V.I. Kumar, and P. Dupont., *Analysis of frictional contact models for dynamic simulation.*, IEEE International Conference on Robotics and Automation, 1997.
- [16] Y. H. Liu, *Qualitative test and force optimization of 3-D frictional form closure grasps using linear programming*, IEEE Transactions on Robotics and Automation, vol. 15, no. 1, 1999.
- [17] B. Mirtich and J. Canny, *Easily computable optimum grasps in 2D and 3D*, IEEE International Conference on Robotics and Automation, vol. 1, pp. 739-747, 1994.

- [18] A.T. Miller, S. Knoop, P.K. Allen and H.I. Christensen, *Automatic grasp planning using shape primitives*, IEEE International Conference on Robotics and Automation, vol.2, pp. 1824-1829, 2003.
- [19] B. Mishra, *Grasp metrics: Optimality and complexity*, First Workshop on Algorithmic Foundations of Robotics (WAFR), A.K. Peters, pp. 137-165, 1995.
- [20] W. Yang, M. Li and X. Zhang, *Robust Robotic Grasping Force Optimization with Uncertainty*, International Conference on Intelligent Robotics and Applications, pp. 264-275, 2010.
- [21] D.J. Montana, *The condition for contact grasp stability*, IEEE International Conference on Robotics and Automation, pp. 412-417, 1991.
- [22] J. Napier, *The prehensile movements of the human hand*, Journal of Bone and Joint Surgery, vol. 38, no. 4, pp. 902-913, 1956.
- [23] V.D. Nguyen, *Constructing stable grasps in 3D*, IEEE International Conference on Robotics and Automation, pp. 234-239, 1987.
- [24] J. Ponce and S. Sullivan and A. Sudsang and J.P. Merlet, *On computing four-finger equilibrium and force-closure grasps of polyhedral objects*, International Journal of Robotics Research, vol. 16, no. 1, pp. 11-35, 1997.
- [25] C.E. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, MIT press, 2006.
- [26] C. Rosales, L. Ros, J.M. Porta and R. Suarez, *Synthesizing grasp configurations with specified contact regions*, International Journal of Robotics Research, vol. 30, 2011.
- [27] A. Sahbani, S.El-Khoury and P. Bidaud, *An Overview of 3D Object Grasp Synthesis Algorithms*, Robotics and Autonomous Systems, 2011.
- [28] M. Santello, M. Flanders and J.F. Soechting, *Postural hand synergies for tool use*, Journal of Neuroscience, vol. 18, no. 23, pp. 105-115, 1998.
- [29] J.K. Salisbury and B. Roth, *Kinematic and force analysis of articulated hands*, Journal of Mechanisms, Transmissions and Actuation in Design, vol. 105, pp. 33-41, 1983.
- [30] A. Shukla and A. Billard, *Coupled dynamical system based arm-hand grasping model for learning fast adaptation strategies under real-time perturbations*, Robotics Science and Systems, 2011.
- [31] R. Suárez, M. Roa, and J. Cornella, *Grasp quality measures*, Universitat Politècnica de Catalunya, Institut d'Organització i Control de Sistemes Industrials, Technical Report IOC-DT-P, 2006.
- [32] Y. Zheng, W.H. Qian, *On some weaknesses existing in optimal grasp planning*, Mechanism and Machine Theory, vol. 43, no. 5, pp. 576-590, 2008.
- [33] X. Zhu and J. Wang, *Synthesis of force-closure grasps on 3D objects based on the Q distance*, IEEE Transactions on robotics and Automation, vol. 19, no. 4, pp. 669 - 679, 2003.
- [34] X. Zhu and H. Ding, *Planning force-closure grasps on 3D objects*, IEEE International Conference on Robotics and Automation, pp. 1258-1263, 2004.
- [35] A. Wächter and L.T Biegler, *Failure of global convergence for a class of interior point methods for nonlinear programming*, Mathematical Programming, vol. 88, no. 2, pp. 565-574, 2000.
- [36] A. Wächter and L.T. Biegler, *On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming*, Mathematical Programming, vol. 106, no. 1, pp. 25-57, 2006.
- [37] N. Wettels, V.J. Santos, R.S. Johansson, and G.E Loeb, *Biomimetic tactile sensor array*, Advanced Robotics, vol. 22, no. 8, pp.829-849, 2008.
- [38] Z. Xue, J.M. Zoellner and R. Dillmann, *Grasp planning: find the contact points*, IEEE International Conference on Robotics and Biomimetics, pp. 835-840, 2007.

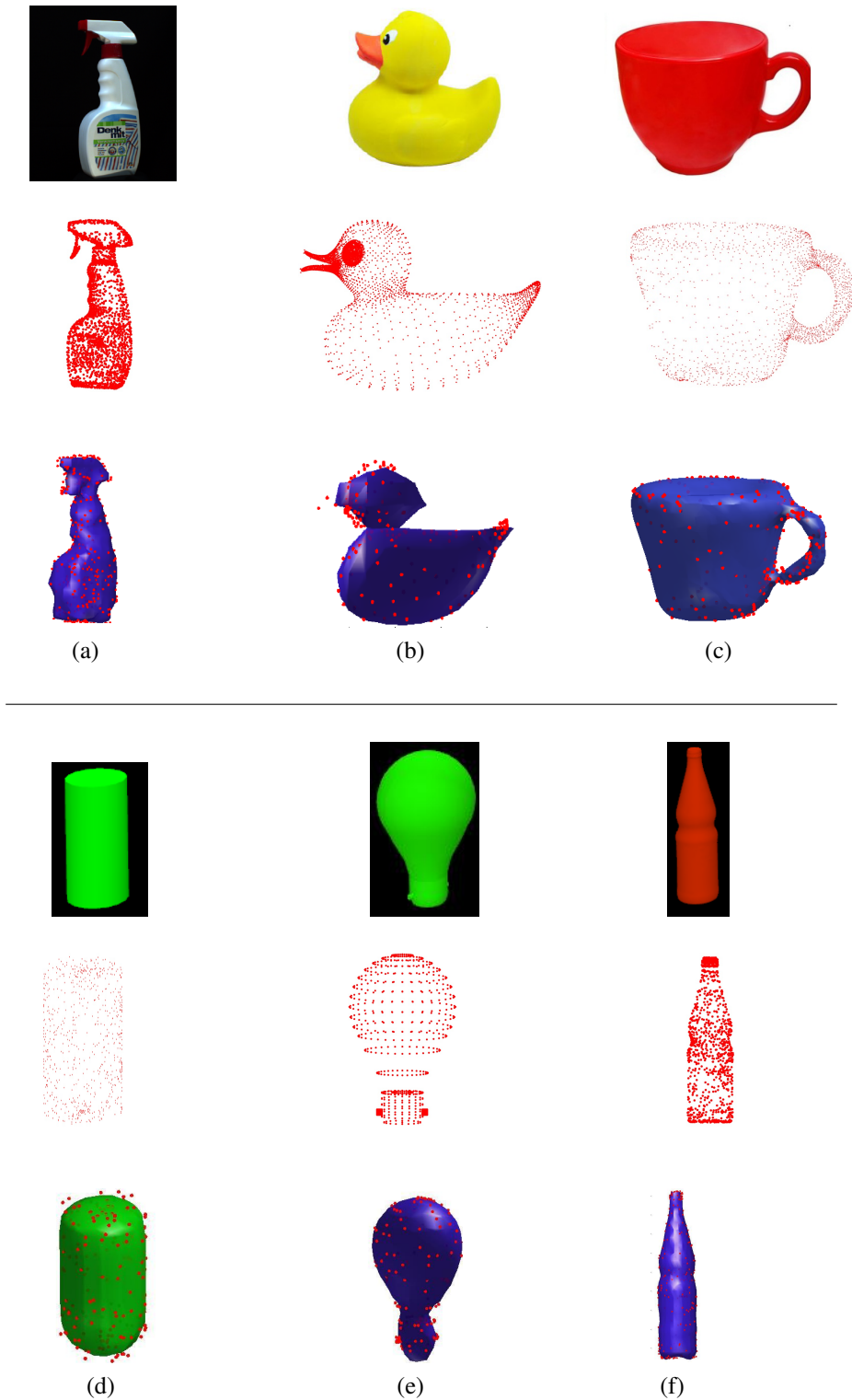


Figure 6: Six 3D objects and their corresponding GP representations. The second and fifth rows show 3D point cloud object models, (a) a spray bottle, (b) a toy duck, (c) a cup, (d) a cylinder, (e) a light bulb and (f) a bottle. The third and sixth rows show object modeled using GP. Dots on the object model are the 3D points on the object surface used to train the GP model.

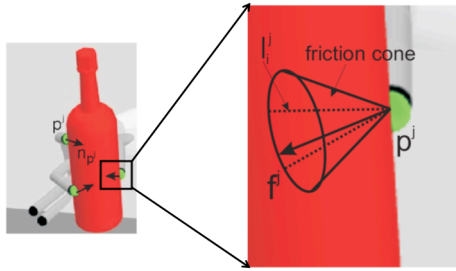


Figure 7: A force-closure grasp on a bottle. The contact point position associated to the j -th finger is noted as \mathbf{p}^j and its corresponding normal as $\mathbf{n}_{\mathbf{p}^j}$. The friction cone at contact point \mathbf{p}^j is linearized with m sides, \mathbf{l}_i^j .

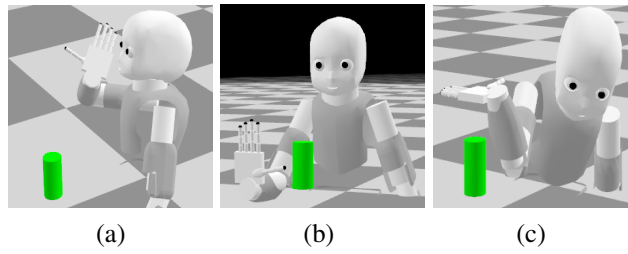


Figure 8: Different initial positions/orientations of the hand are illustrated. In (a), the palm is positioned on the diagonal of the reference system attached to the center of the object. In (b), it is aligned with the middle of the object height and in (c) it is placed above the object.

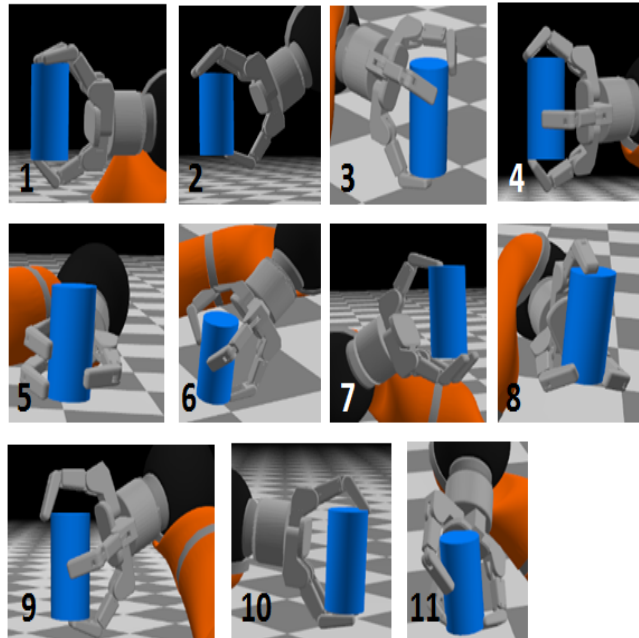


Figure 9: The different Barrett hand configurations to grasp a cylinder, labeled from 1 to 11.

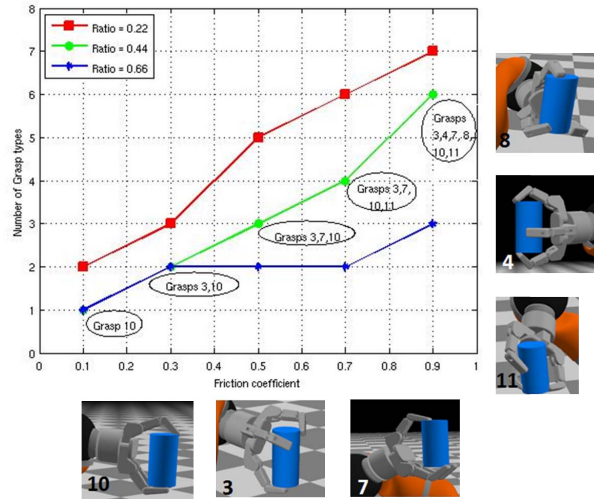


Figure 10: The number of grasps obtained for the Barrett hand as a function of the friction coefficient for different weight to maximal force ratio.



Figure 11: The different iCub hand configurations to grasp a cylinder, labeled from 1 to 20.

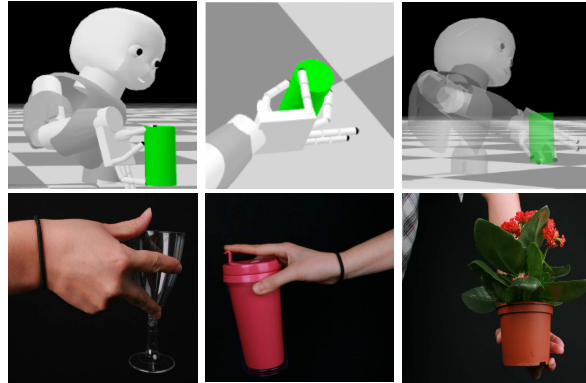


Figure 12: Examples showing the importance of grasps labeled 10, 13 and 8 for every day objects manipulation.

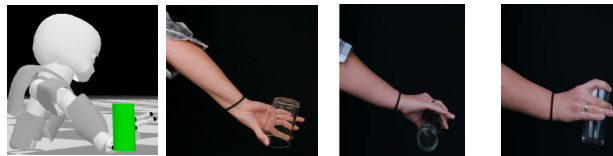


Figure 13: Examples showing the importance of grasp number 4 for flipping an object upside down using only one hand.

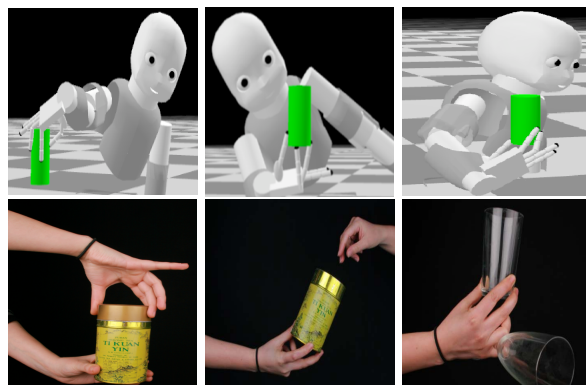


Figure 14: Examples showing the importance of grasps labeled 19, 20 and 16 for every day objects manipulation.

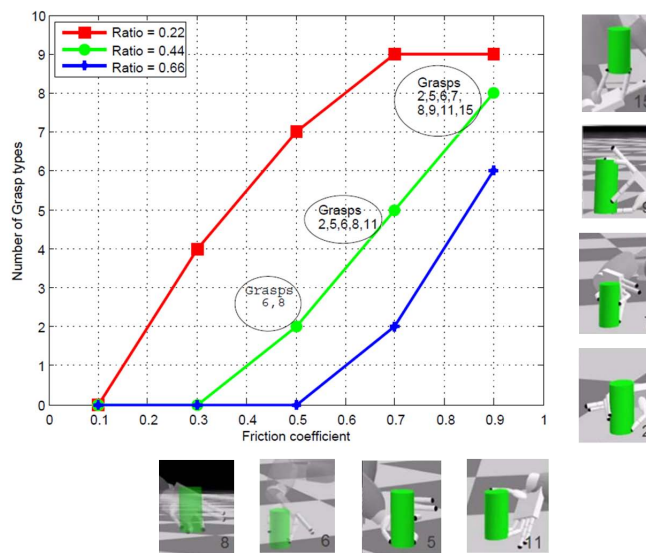


Figure 15: The number of grasps obtained for the iCub hand as a function of the friction coefficient for different weight to maximal force ratio.

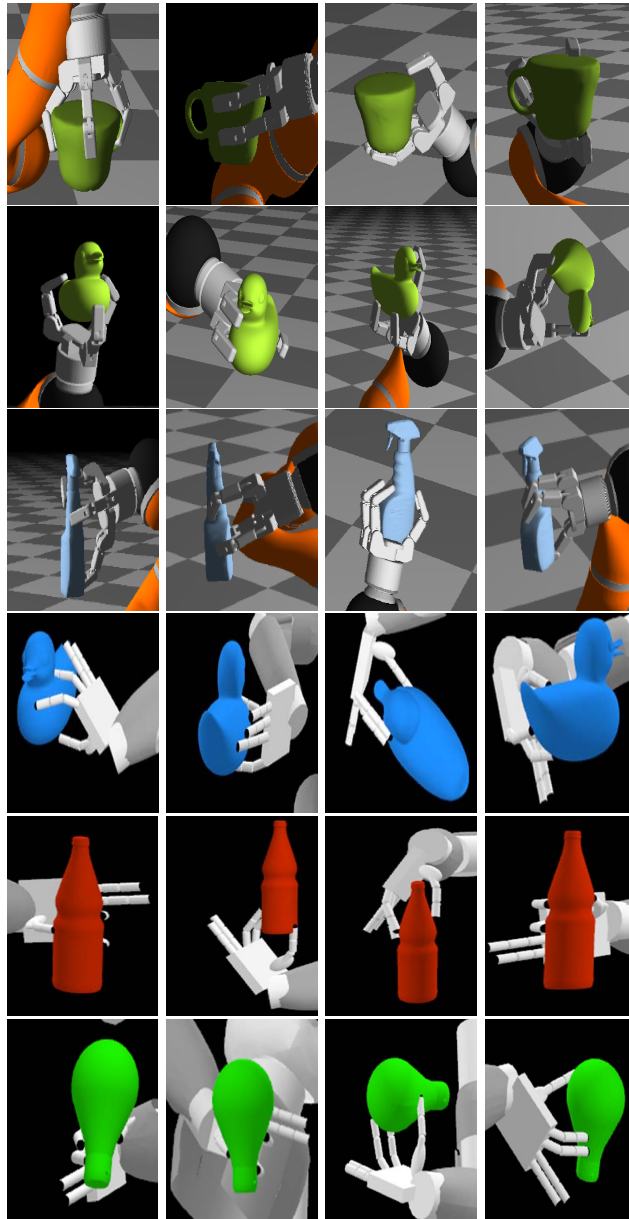


Figure 16: Grasps computed on some realistic objects modeled with Gaussian Processes.



Figure 17: Implementation of a computed grasp for the flask bottle on the Barrett hand.



Figure 18: Implementation of several grasps generated for the flask bottle on the Barrett hand.



Figure 19: Grasps computed for the iCub hand on some real objects modeled with Gaussian Processes.