# Grasp Planning for Non-Convex Objects

Efrain Lopez-Damian†, Daniel Sidobre†‡, and Rachid Alami†
(edamian, daniel, rachid)@laas.fr

† LAAS-CNRS
7, avenue du Colonel Roche, 31077 Toulouse, France

‡ Université Paul Sabatier
118, route de Narbonne, 31062 Toulouse, France

*Robotics research aims also to develop autonomous systems in dynamically changing environments. One desirable functionality is manipulation that allows the robot to modify its surroundings. Generally, a grasp is the beginning of any manipulation task and robots must be capable to handle most common objects. The shape of these objects is varied and many are non-convex which make difficult to grasp them. This work presents a grasp planner that deals with this problem using an approximate convex decomposition of the object to plan grasps.*

## 1. INTRODUCTION

In the field of autonomous robotics, one kind of system that interest researchers today is service or personal robot. Such machines must have the capabilities to interact and make decisions in an automatic way depending on the constraint imposed by the always changing environment and the task to do. To reach this goal it is necessary to develop a set of functionalities. Among all the functions, the manipulation of objects has a decisive role to allow the robot to interact and modify its environment.

A grasp is generally the beginning of any manipulation task and robots must be capable to handle most common objects in the surroundings. The shape of these objects is varied and many are non-convex which difficult the grasp planning. The object model can be obtained by a stereovision or 3D laser sensors. These models are generally complex and composed by many details. Grasping an unknown object is one of the tasks that a personal robot must be able to accomplish. A clue point to solve is the automatic grasp of the object. In this paper we present a grasp planner for complex objects and more particularly the use of an approximate convex decomposition.

## 2. RELATED WORK

In literature, we can find work that propose algorithms for grasp planning of polygonal objects [12]. The grasp planning is converted into an optimization problem. In ref. [6] the planning for 2D curved objects is proposed, an antipodal grasps are computed in [5] for arbitrary shape objects. The problem becomes much more difficult in three-dimensional space. For polyhedral objects, work has been done to characterizes and compute three and four-finger grasps [13].

Based on the hypothesis that instead of compute optimal grasps a set of ranked grasps can be generated using a quality criteria, a good grasp can be chosen among grasps in a set [3]. The use of heuristics accelerate the process to find a grasp on the object. Borst [2], [8] uses a random generation strategy, they define an arbitrary point and frame inside the object. We have proposed a semi-random approach in ref. [11], where generation of grasps is guided by the mass center and inertial axis of the object.

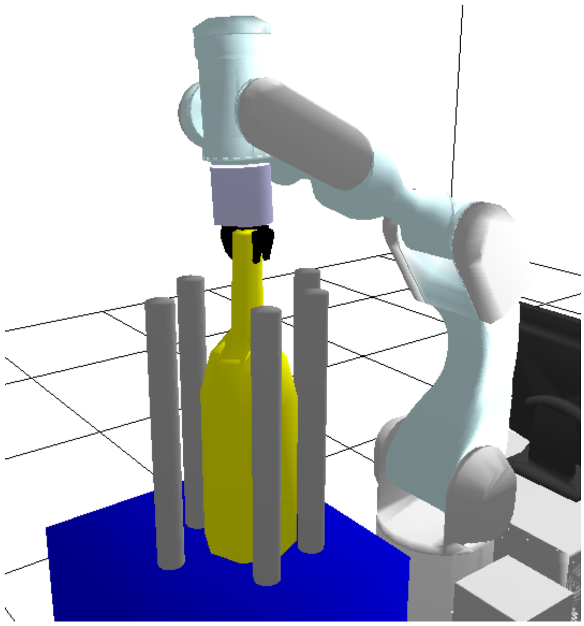


Fig. 1. Example where a bottle is surrounded by cylindrical obstacles, a grasp is found near the bottle neck.

## 3. GRASP PLANNING

We can define a grasp as the end-effector placement relative to the object and the contact points on the object surface. Then the grasp planning problem becomes the generation of feasible grasps on the object that satisfy some constraints. A grasp planner must produce a grasp that fulfill some fundamental criteria: the grasp must be stable, the grasp must be reachable and the grasp must be free of collision. Grasp planning is computationally intensive due to both the large search space and the incorporation of geometric and task-oriented constraints. The search space for an optimal grasp can be reduced by using certain heuristics. Shape of object, relative size of object and gripper are the principal geometric characteristics. Shapes are generally non-convex and constitute a more difficult problem to solve for a grasp planner. In next sections we describe a planner to deal with such kind of objects.

## 4. NON-CONVEX OBJECT GRASP PLANNER

The approach that we proposed, firstly try to generate grasps on the whole object. Secondly, decompose the object in smaller components and compute a set of grasps for each component generated. Every grasp is ranked depending on a quality criterion, the grasp with best quality is chosen as the output of the planner.

We present here the basic algorithm framework of the semi-random generator of grasps for non-convex objects. The grasp planner is based on the work done in [11].

*Algorithm Framework*

*loop:*object decomposition
*for* each component
    1. Compute inertial axis from component
    2. Semi-random generation of grasps
    3. Apply filters
    4. Assign a quality value
    5. Choose the best feasible grasp
    *if* grasp founded
    *then* Stop planner and output grasp
    *end if*
*end for*
*end loop*

The grasp planner take as an input the geometric model of the environment, including the object and the geometric and kinematics models of the robot and gripper. The object decomposition process is the first step of the algorithm, here two strategies for the planner can be followed, call a complete decomposition process that gives as a result the series of components of the object and generate a set of grasps for each component in the list or as we present in the framework, at each iteration of the decomposition process two components of the input are produced, we choose one of the two components and we try to generate a feasible grasp on it. If a grasp is founded the planning process is terminated and the grasp is given as output. If not we take the second component and we call again the grasp planner. In case that a feasible grasp cannot be generated for both components a new iteration is performed.

This process is repeated until one of two conditions arrives: a grasp is founded in one component or decomposition is terminated. A feasible grasp can be found before complete decomposition. The last components found are generally small and less interesting. Next we give a briefly description for each part of the grasp planner. A more detailed description of certains parts can be founded in [11]. The description of the object decomposition is given in next section.

*Inertial Axes:* the location of the center of mass and the inertia tensor can be computed by the conversion of the integrals of mass into the volume integrals. We suppose the polyhedron (P) has a mass m and a uniform density $\rho$, we can relate the volume as $m = \rho V$.
*Semi-random Generation:* First we define a frame at or near the inertial frames and secondly we compute contact points according to the gripper used. A method for a gripper with two fingers and three contact points is presented in [11]. A specialized grasp generator is needed for each different gripper to compute the contact points from the grasp frame position.
*Filter:* As quality determination is computationally expensive, we introduce the filter step to reject as soon as possible unfeasible grasps. Some constraints are imposed by the system itself, the grasp must be kinematically reachable by the robot and free of collision with the possible obstacles in the environment, simulation tools are used for this [14]. To guarantee that the object is firmly held with no slippage, we use a force closure filter.
*Quality Measure:* Several grasps can be produced after the first two steps are executed. The final step is the assignment of a quality measure to the grasps.

### 4.1 Force-closure filter

One of the most important properties for a grasp is the notion of force closure. A grasp is defined geometrically

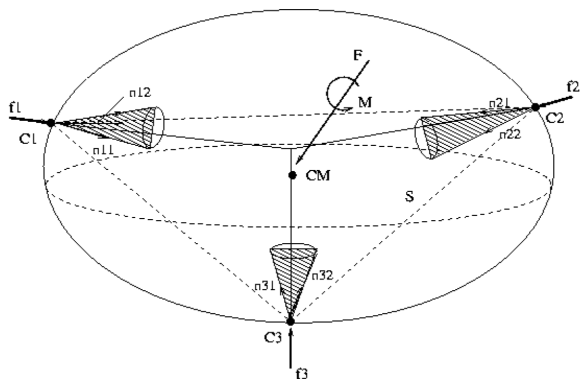



Fig. 2. 3D Grasp with three contact points can be see as a plane grasp considering the sections of friction cones by the plane containing the three points.

by the position $C_i$ of $d$ hard fingers or contact points on the object surface, with $i = 1, ..., d$. Hard finger contact model and Coulomb friction are assumed between the object and the fingers. Each finger exerts in $C_i$ a force $f_i$ and a moment $C_i \times f_i$ with respect with some point on the object, the center of mass in this case. Force and moments are combined and form a six-dimensional vector called wrench $w_i = [f_i, C_i \times f_i]^T$. A grasp achieves equilibrium when the sum of the wrenches is zero $\sum_{i=1}^{d} w_i = 0$. A grasp is force closure if it can balance any external forces and moments ($w$), exerted on the object

$$w + \sum_{i=1}^{d} w_i = 0 \tag{1}$$

The forces applied by the finger $f_i$ must remains in the friction cone to avoid the slippage, see Fig. 2. The grasp then is force-closure if and only if there exists a force in each friction cone such that the sum of the corresponding wrenches is zero. For a three-finger grasp as in Fig. 2, a necessary condition for force closure is that there exists a point in the intersection of the plane formed by the three contact points with the friction cones at these points [13] [9].

### 4.2 Quality Measure

The planning of a good grasp is important when the robot has to take an object in a firmly way, for this a quality criterion has been developed in [7]. The criterion try to quantify the notion of a good grasp for a force closure grasp. Again a hard finger contact model and Coulomb friction are assumed. We must discretize the cone of friction to represent it by a finite set of *m* vectors.

For the quality measure, Ferrari [7] consider that the sum of the magnitude of the forces applied by the gripper at the *n* contact point is 1, then $f_i$ can be written as:

$$f_i = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{i,j} f_{i,j} \tag{2}$$

with $\alpha_{i,j} \geq 0$. Similarly we have that the total wrench applied on the object is expressed by:

$$w = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{i,j} w_{i,j} \tag{3}$$

and the set of all wrenches is:

$$W = CHULL\left(\bigcup_{i=1}^{n} w_{i,1}, ..., w_{i,m}\right) \tag{4}$$

The quality measure is the distance of the nearest facet of the Convex Hull from the origin.

Other criteria can be added, for example to try to favor grasps that contact points are in the middle of the facets and defines a more stable grasps.

## 5. OBJECT DECOMPOSITION

Sometimes the grasp planner doesn't find any valid grasp, caused by several reasons. Size and geometry complexity of the object are the most common reasons for the planner to fail. One possible solution is to decompose the object in several smaller parts.

Convex polyhedra decomposition is not new, in the computational geometry community a solution is given for convex decomposition [1] [4]. The algorithms make a partition of the model into convex components. Two main problems arise in the use of these kind of algorithms: an unmanageable number of parts and the tiny size of these parts. Both results are undesirable in grasp planning because of the excessive computing time and the impossibility to grasp tiny parts.

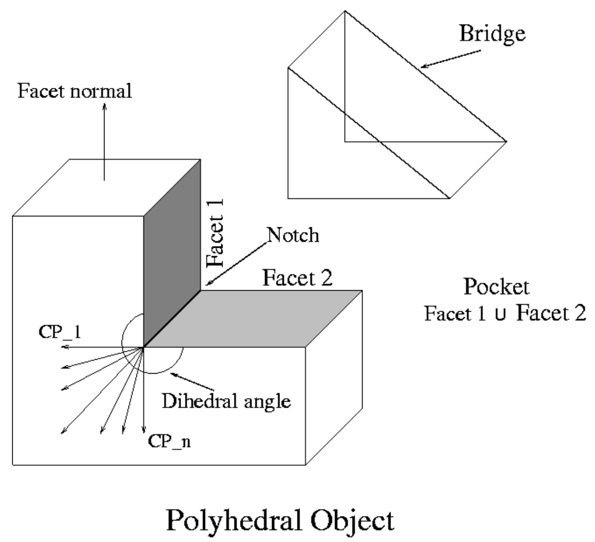



Fig. 3. Example of Polyhedral object, bridges and pockets are indicated. The notch is the reflex edge formed by the two facets with an internal angle greater than $180°$. A series of cutting planes CP_1 ... CP_n are shown.

An alternative approach is the use of an approximate convex decomposition(ACD) [10] to partition the object in approximate convex pieces.

### 5.1 Approximate convex decomposition

The ACD takes as an input the geometric model of a component, this can be the object or a part of the object, represented by a polyhedron $P_H$ and a tunable convex tolerance $\lambda$.

The approximate convex decomposition removes at each iteration the most significant non-convex feature called notch (a reflex edge where the inner dihedral angle subtended by two incident facets is greater than $180°$) by defining a cutting plane and partitioning the object in exactly two components. The resulting components are the input for the next iteration of the algorithm. The process is repeated until all components $C_i$ satisfy the convex tolerance $\lambda$. The definition of a $\lambda$-approximate convex component is a polyhedron whose concavity is at most $\lambda$. The convex decomposition is defined as:

$$CD(P_H) = \{C_i \subset P_H | concavity(C_i) \leq \lambda\} \; with \\ \cup_{i=1}^{n} C_i = P_H and \; C_i \cap C_j = \emptyset, \forall_{i,j} \; i \neq j \qquad (5)$$

For remove the most significant notch we need a measure of its concavity. In contrast of some measures as radius, area and volume, concavity is not a well defined measure. Here, we use the concavity as a distance from the concave features to the convex hull of the object. To measure concavity, the notions of bridges and pockets of polyhedron are used, see Fig. 3.

$$Bridge(P_H) = \{f_{C_H} | f_{C_H} \in C_H \wedge f_{C_H} \backslash S_{P_H} \neq 0 \\ Pocket(P_H) = \{f_{C_H} | f_{C_H} \in S_{P_H} \wedge f_{C_H} \backslash C_H \neq 0 \qquad (6)$$

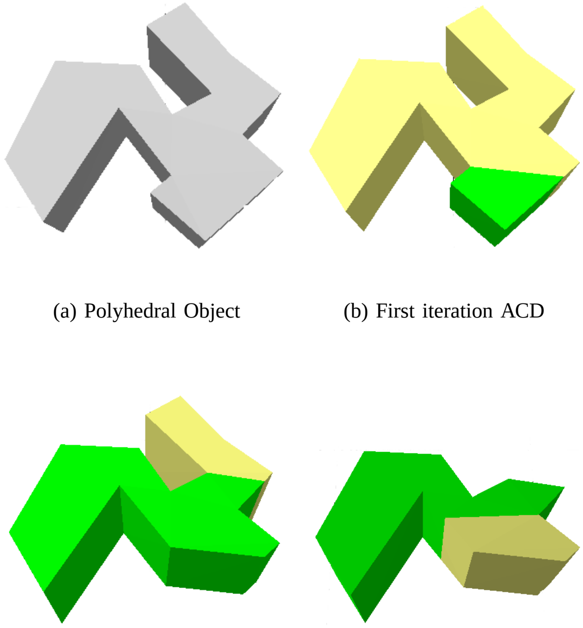
where, $C_H$ is the convex hull of the polyhedron, $f$ represents a facet and $S_{P_H}$ is the surface of the polyhedron $P_H$ composed by a set of facets. The concavity measure is then the distance between the middle point of each notch to the bridge.

The bridges are facets of the $C_H$ that are not part of polyhedron. The algorithm identify the match between facets of $C_H$ and $P_H$ and discard them. Pockets are facets of polyhedron $P_H$ and they are not part of the convex hull. As the facets of $C_H$ are not generated from the facets of $P_H$, it is to say, if $P_H$ would be convex its correspondent $C_H$ will be the same $P_H$, a test is required to find the facets that correspond to pockets. Such test is done by a user tolerance in the distance between the $P_H$ facet and the $C_H$ facet and a tolerance in the orientation between facets planes, after test we obtain the pockets.

When bridges and pockets are computed and a concavity measure is assigned to each notch. The notch with the highest concavity is selected and a series of cutting-planes ($C_p$) are formed to remove the notch. We select one $C_p$, taking as criterion that the cut has the smallest surface. Once the cut is made, we generate the set of facets that are on the $C_p$ and we build the two polyhedra.

## 6. RESULTS

We present the first results of the non-convex grasp planner. In Fig. 5 we can observe the inertial axes of a mug, that were used to generate the grasps on the object. These axes do not give any valid grasp. After the first call to the object decomposition process, the grasp planner found one feasible grasp on one of the generated component. We test the algorithm with a glass, see Fig. 6. A bottle object is used as third example, we can see the bottle size is big with respect of the gripper, and the only possibility is to grasp the bottle neck, see Fig. 7. Finally in Fig. 1, we



(a) Polyhedral Object (b) First iteration ACD

(c) Second iteration ACD (d) Third iteration ACD

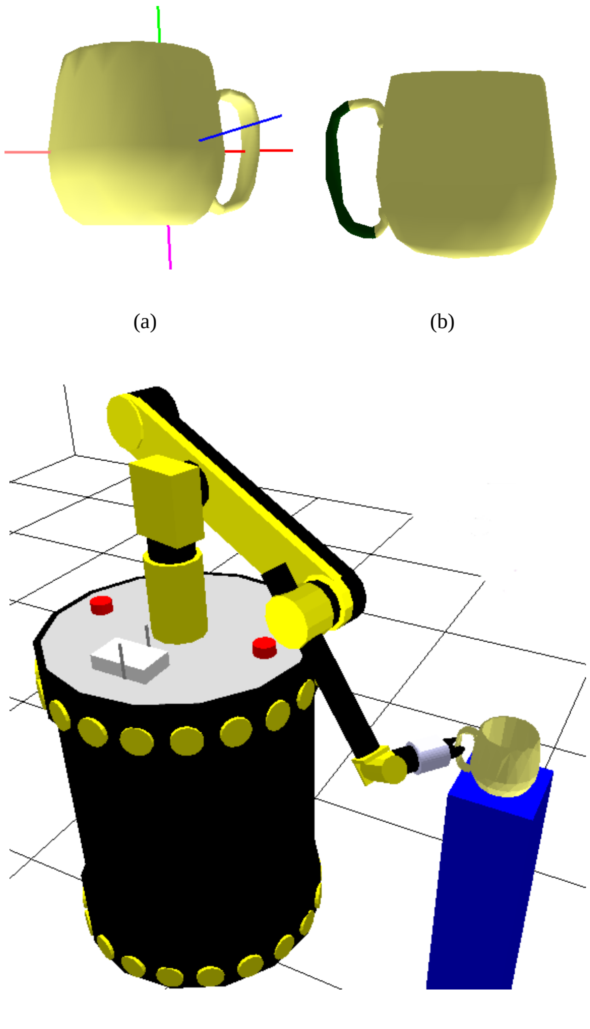
Fig. 4. The figure shows the output of the ACD algorithm. After three iterations, after first iteration is done, two components are produced. Second iteration taking as input one component of the first iteration generates two more subcomponents, the second component result of the first iteration is convex and it is not decomposed

can see how the planner is capable to compute a different grasp when the object is surrounded by obstacles. The grasp planner was implemented within the motion planner tool Move3D developed at LAAS-CNRS. Move3D counts with a series of motion planners, collision checkers and steering methods for non-holonomic car-like robots. More detailed description of the tool can be found in [14].

The experiments were performed using a 500 MHz Solaris SunBlade. The processing time that the algorithm requires to decompose an object is determined by the complexity of the object model. In Table 1 we can see the time after one iteration of the object decomposition algorithm for different object models.

Table 1 Results for Several Object Models

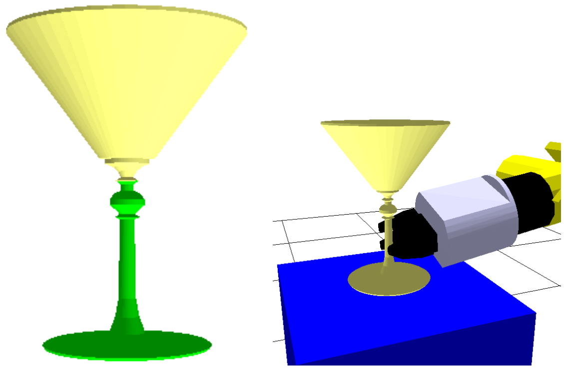
| Object | Time | Facets |
|---|---|---|
| Polyhedral object | 0.08 s | 60 |
| Bottle | 0.12 s | 80 |
| Mug | 0.71 s | 499 |
| Glass | 13.58 s | 2750 |



(a) (b)

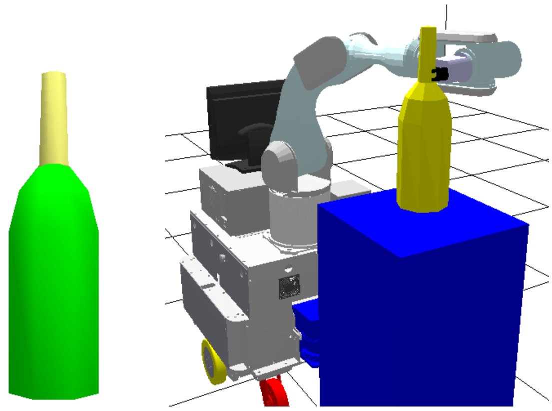
(c)

Fig. 5. a) Inertial axes used in grasp planner to generate a set of grasps on the object b) Decomposition of object after one iteration of the process c) The final grasp founded by the grasp planner after one iteration in the decomposition process



(a) (b)

Fig. 6. a) Decomposition of glass after one iteration of the process b) Feasible grasp generated after object decomposition



(a) (b)

Fig. 7. a) Decomposition of a bottle b) Grasp founded on the upper part of the bottle because bottom part is too big. As the quality criterion is global, the grasp computed is in the bottom of the bottle neck.

## 7. CONCLUSION

The grasp planning for non-convex 3D objects is a challenge for service robotics. In this paper we have proposed a grasp planner for these kind of objects. The idea to decompose the object in smaller parts seems to be a good solution. Strategy to generate grasps on components at each iteration of the decomposition process for the grasp planning allows to save computing time in the execution of the algorithm. The decomposition by removing concavities gives interesting results but a main point that would help to improve the algorithm is the selection of the cutting plane, until now the criterion used does not guarantee the best way to partition an object for grasping, a more intentional manner has to be found.

## 8. ACKNOWLEDGMENT

The work described in this paper was partially conducted within the EU Integrated Project COGNIRON ("The Cognitive Companion") and funded by the European Commission Division FP6-IST Future and Emerging Technologies under Contract FP6-002020. The authors acknowledge the support of the National Council of Science and Technology of Mexico (CONACyT).

## 9. REFERENCES


[1] C.L. Bajaj and T.K. Dey. Convex decomposition of polyhedra and robustness. *SIAM Journal of Computing*, 13(3):488–507, 1984.

[2] Ch. Borst, M. Fischer, and G. Hirzinger. A fast and robust grasp planner for arbitrary 3d objects. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1890–1896, Detroit, Michigan, May 1999.

[3] Ch. Borst, M. Fischer, and G. Hirzinger. Grasping the dice by dicing the grasp. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3692–3697, Las Vegas, Nevada, October 2003.

[4] B. Chazelle. Convex partitions of polyhedra: A lower bound and worst-case optimal algorithm. *SIAM Journal of Computing*, 13(3):488–507, 1984.

[5] I.M. Chen and J.W. Burdick. Finding antipodal point grasps on irregularly shaped objects. In *Proc. IEEE International Conference on Robotics and Automation*, pages 2278–2283, Nice, France, May 1992.

[6] B. Faverjon and J. Ponce. On computing two-finger force-closure grasps of curved 2d objects. In *Proc. IEEE International Conference on Robotics and Automation*, pages 424–429, Sacramento, California, April 1991.

[7] C. Ferrari and J. Canny. Planning optimal grasps. In *Proc. IEEE International Conference on Robotics and Automation*, pages 2290–2295, Nice, France, May 1991.

[8] M. Fischer and G. Hirzinger. Fast planning of precision grasps for 3d objects. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 120–126, Grenoble, 1997.

[9] J.W. Li, H. Liu, and H.G. Cai. On computing three-finger force-closure grasps of 2-d and 3-d objects. *In IEEE Transactions on Robotics and Automation*, 19(1):155–161, February 2003.

[10] JM. Lien and N. Amato. Approximate convex decomposition of polyhedra. Technical Report TR05-001, PARASOL LAB, Texas A&M University, January 2005.

[11] E. Lopez-Damian, D. Sidobre, and R. Alami. A grasp planner based on inertial properties. In *Proc. IEEE International Conference on Robotics and Automation*, pages 766–771, Barcelona, April 2005.

[12] J. Ponce and B. Faverjon. On computing three-finger force-closure grasps of polygonal objects. *In IEEE Transactions on Robotics and Automation*, 11(6):868–881, December 1995.

[13] J. Ponce, S. Sullivan, J.D. Boissonnat, and J.P. Merlet. On characterizing and computing three- and four-finger force-closure grasps of polyhedral objects. In *Proc. IEEE International Conference on Robotics and Automation*, pages 821–827, 1993.

[14] T. Siméon, J-P. Laumond, and F. Lamiraux. Move3d: A generic platform for motion planning. In *Proc. 4th International Symposium on Assembly and Task Planning (ISATP'2001)*, 2001.