Grasp Planning for Interactive Object Manipulation

Efrain Lopez-Damian†, Daniel Sidobre†‡, Stephane de la Tour† and Rachid Alami†
{*edamian, daniel, sdelatou, rachid*}*@laas.fr*

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

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

***Abstract*— Nowadays, one important research area of robotics is the design of machines with capabilities to interact with humans or other robots. Service robots fall in this category of machines and they need a set of functions to accomplish their tasks. Future autonomous robots must be capable to work and take decisions in an automatic way. Such a system must be capable to handle most common objects in a dynamic human environment.**

The first problem to solve in every manipulation task is how to grasp the object under certain constraints. This paper presents a planner to grasp unknown arbitrary objects for interactive manipulation tasks.

***Keywords*— Grasp planning, object manipulation, object decomposition, interactive grasps.**

I. Introduction

Robotics research is focused in the development of machines capable to carry on tasks in an autonomous way in a constrained environment like the interior of a building or a house.

The interaction between robots and between robots and humans is a desirable functionality for service robots and humanoids. This interaction is beyond of just communication, detection and recognition of gestures made by the human but also the interaction to execute tasks in a collaborative way. One primary task in autonomous robotics is the manipulation of 3D arbitrary objects, see Fig. 1.

In general, manipulation begins with a grasp: pick and place, filling a glass, hand over task, using a tool, turning a crank handle. Sensors allow to obtain 3D data to polyhedral models of objects [1] and geometric algorithms for path planning are efficient [2]. One important point to solve in the motion planning for manipulation is then the automatic generation of grasps for two end-effectors on the object.

Planning for interactive manipulation when a human get involved becomes a very difficult problem because we do not known how the human is going to grasp the object. There are in literature several works trying to explain how human uses different hand configurations depending on the shape of the object, a taxonomy of these configurations can be found in [3]. We can divide these configurations in power grasps and fine grasps. In this paper we deal with fine grasps only. Many manipulation tasks involve more than one end-effector in the process like cooperation tasks.

In section III we present a random grasp planner, we explain how we can find grasps for a three-finger articulated hand and how can be extended for a simple human hand model. In section IV an object decomposition approach is used to plan two grasps for a interactive object manipulation tasks using the grasp planner described in previous section. Finally in section V and VI, results and conclusions are discussed, respectively.

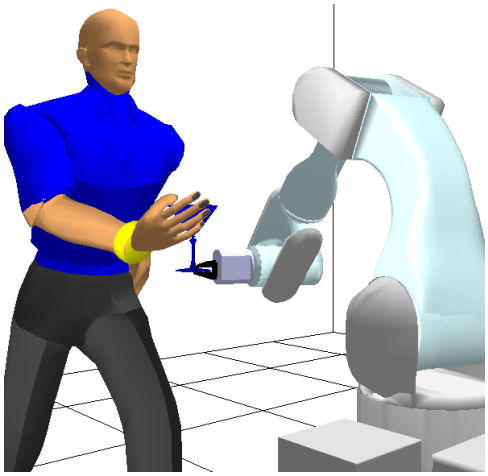


Fig. 1. Hand over operation.

II. Related work

In reference [4], an integrated system is presented. The complete manipulation task is to pick an object and hand over it to a human. The object is known a-priori and they do not plan a grasp site for the receiver. In [5], the authors use a simple heuristic to find grasps. They find a region as a result of the projection of one facet to another when they are parallel, a simple way to have stable grasps.

From the center of this region they find a grasp frame for the robot and a grasp frame for the human hand. The drawbacks are that only very simple objects can be grasped and the grasps are limited to stay always on the same surface. As the previous work, the objects are known and they consider a series of etiquettes to describe the function of the object and to indicate which features are not safety for grasping.

In literature, there are other approaches trying to obtain a solution to the problem of grasping objects for manipulation. These kind of methods are based on learning by demonstration, the robot observes how the human performs a task and try to replicate the sequence of motions to perform the task itself [6] [7].

ISBN 970-769-070-4, ISRA 2006 Proceedings

We can find planners for cooperative manipulation for virtual characters that consider the geometric constraints imposed by the environments, they successfully find a trajectory to execute the task. The body of the virtual character or mannequin is divided depending of the function that is intended to do, legs and pelvis are for locomotion and arms for manipulation [8] [9] [10], but they do not plan the grasps, a set of them are specified and used. Our grasp planner could be used in addition to have a more general motion planner for service robots.

III. PLANNING GRASPS

We propose a simple solution to plan grasps for interactive manipulation tasks. These tasks define the interaction between two arms belonging to the same robot or two robots. We have divided our solution in robot-robot interaction and robot-human interaction. A humanoid presents the same difficulty as the human if its end-effector is a five-fingered hand.

For the case of a human, we propose to consider the human as another robot to simplify the planning. We propose to use the same strategy applied for the case of a three-fingered hand to grasp an object. We make the simple assumption that if we find a grasp for the robot in this way, we assure that in reality human can grasp the object and he probably can find a better grasp using all his dexterity, compliance and intelligence.

A. Random Grasp planner

The grasp planner is organized in a modular way. Here we give a briefly description, and how we use it to implement the proposed solution to plan grasps for interactive manipulation tasks.

In the literature on grasp planning, we can identify two general approaches. The first one consists in finding the optimal grasp that satisfies the force closure property [11], [12], [13]. Such methods work fine but they are computationally complex to implement.

The second approach assumes that we can generate a sufficient number of candidate grasps and choose the best among them. Reference [14] shows that it is not necessary to generate optimal grasps. An average quality grasp is an acceptable good grasp.

Our grasp planner [15] is based on the second approach and it was thought to handle the case of unknown arbitrary objects, this is an essential function since the robot has to interact with a highly changing environment and we avoid the use of not robust recognition algorithms. This implies that the object has to be modeled by a 3D sensor, a stereo camera or a 3D laser and the resulting model is represented through a triangular mesh representation [1] [16]. The algorithm uses this model to find several grasps that must satisfy some constraints, desirable and necessary conditions. The basic steps are presented in algorithm 1.

Algorithm 1: Grasp Planning Algorithm Framework

```
input  : Geometric Model: gripper Grp, the object O,
         the environment E and the robot R
output : Grasp G
begin
    COMPUTE_SIX_INERTIAL_AXES;
    COMPUTE_MASS_CENTER;
    COMPUTE_GRASP_FRAME G_F;
    forall Inertial Axes do
        COMPUTE_SEVERAL_ORIENTATIONS(G_F);
        COMPUTE_SEVERAL_DISPLACEMENTS(G_F);
        CP ←COMPUTE_CONTACT_POINTS(G_F, Grp);
        APPLY_FILTERS;
        ASSIGN_QUALITY_VALUE;
        G ← [CP, G_F];
    end
    CHOOSE_BEST_GRASP;
end
```

For the generation of grasps we use a simple but powerful heuristics, as the object is subject to external force of gravity and acceleration that act on the mass center, it is a good idea to grasp the object around this point. In the same sense, good approach directions candidates for grasps are given by the principal inertial directions of an object. The axes of inertia are computed from the geometric representation of the object, computing the volume integrals and considering a uniform density [17]. Based on these axes of inertia and the end-effector geometry, the planner generates candidate grasps with different orientations and locations. For a gripper with two fingers and three fingertips, the first contact point is the intersection between a ray drawn from the grasp frame to the object surface, the second contact point is the point given by the intersection between a circle with a radius equal to the fingertips separation in one of the fingers of the gripper centered at the first contact point. Finally a ray is drawn from the middle point between the two first intersection points to the opposite object surface for the third contact point. The complete description for a gripper with three fingertips can be found in [15].

The grasp is submitted to a set of filters like kinematics feasibility, force-closure condition and collision checking. This is a fast way to eliminate bad grasps from the beginning, resulting in a better computational time. After the grasp has passed the filter step, a quality value is assigned to it. The value is a measure made in the wrench space [18]. A list of grasps is generated and the algorithm gives as output the best grasp whole robot configuration.

1) Force-Closure Filter: A grasp is defined geometrically 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. 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. 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.

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 [18]. The criterion tries to quantify the notion of a good grasp for a force closure grasp. We must approximate the friction cone to represent it by a finite set of *m* vectors.
For the quality measure, Ferrari [18] 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:

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

The quality measure is the distance of the nearest facet of the Convex Hull from the origin.
Each different end-effector needs its own candidate grasp generation module, specifying the hand and the robot associated with it, we can use the right module for any case.

B. Grasp generation for an articulated hand

In this section we present the generation of grasps for a three-fingered hand.

The manipulation robot hand for our simulations is the Barrett Hand. This hand is composed by three fingers, each one with two joints. One finger is fixed and the other two can spread synchronously from 0 to 180 degrees around the palm. This mechanism is controlled by four motors which implies that each finger has one actuated inner link and a coupled outer link that moves with a ratio τ regarding the inner link. The fourth motor controls the motion of the two fingers from the palm.

1) Grasp Generator Algorithm: Our planner considers the geometry and kinematics of the hand in the generation process. As two of the fingers of the Barrett hand have a motion concerning the palm of the hand, we generate several start configuration grasps only moving these fingers. Then we have to generate a grasp frame and find the contact points on the object surface.

Grasp Frame (G_F): The location of the G_F is given by the mass center the first time, then we generate a set of locations along the axes of inertia.

Contact Points: We reduce the problem to the intersection of the fingers with the object surface considering each finger like a planar robot manipulator with two linked joints.

We generate two finger trajectories described by circles. The first trajectory with radius R_f corresponds to the case where the finger is in its maximal extension and we compute the intersection between the circle and the plane defined by the object facet. With the resulting line, we compute now the intersection between this and the triangular facet that gives us one intersection point P_1. Next, we generate a second trajectory with radius r_f, that is the minimal distance when the finger joints take their maximal values, the finger is at its minimal extension. We find a second intersection point P_2. As we can see, both intersection points do not correspond to the real contact point, the actual point is found with the line formed by P_1 and P_2 that is located on the object surface and the geometric equation of the planar manipulator, see Fig. 2. We assume that both intersection points P_1 and P_2 are lying on the same facet.

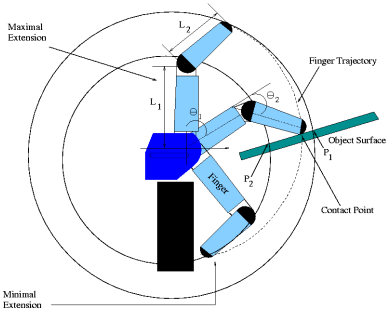


Fig. 2. Taking maximal-minimal configuration of finger, two circular trajectories are generated to compute the contact point when real finger trajectory intersects the object surface

Intersection point: the Barrett hand has three independent fingers, each finger has a motored inner link and a coupled outer one $\theta_2 = \tau\theta_1$. The position of the fingertip for each finger is given by

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} cos\theta_1 \\ sin\theta_1 \end{bmatrix} L_1 + \begin{bmatrix} cos((1+\tau)\theta_2 \\ sin((1+\tau)\theta_2 \end{bmatrix} L_2 \tag{5}$$

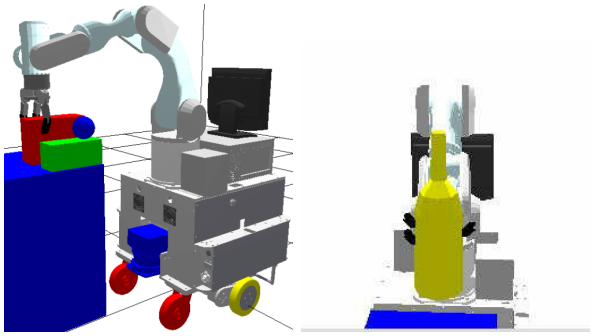
We have to consider the initial position of the finger given by θ_0. Finally taking the parametric equation of the line $P = P_1 + \mu(P_2 - P_1)$ in a matrix form we have:

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} A \\ C \end{bmatrix} + \mu \begin{bmatrix} B \\ D \end{bmatrix} \tag{6}$$

We make some operations and we find an equation with one unknown.

$$\begin{aligned} D(L_1 \cos\theta_1 + L_2 \cos(\theta_0 + (1+\tau)\theta_1) - A) - \\ - B(L_1 \sin\theta_1 + L_2 \sin(\theta_0 + (1+\tau)\theta_1) - C) = 0 \end{aligned} \tag{7}$$

We solve this equation with a numerical method for θ_1, we use the equations above and we find the coordinates for the contact point. We do the same for the others fingers. We can see in Fig. 3(a) the result of the grasp planner for a simple object and in Fig. 3(b) for a more complex object.

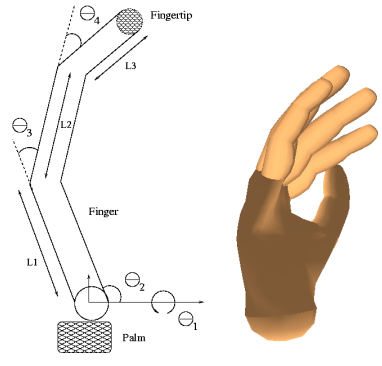


(a) Grasp for a cube (b) Grasp for a bottle

Fig. 3. a) Grasp generation for an object with obstacles, b) Grasp for a more complex object.

2) Human hand model extension: We can see that for a human hand or a five-fingered hand, the approach described for the Barrett hand could be used after some modifications. We find some methods for four finger grasps [12] [19] and how an n-finger grasp can be transformed to a force-closure grasp knowing at least m finger contact points and finding the position for the other fingers [20] [11].

We do not know how to handle easily the problem for five fingers, but we propose to use the thumb, the index and the medium fingers to form a three force-closure grasps and the other two fingers we assume that they will increase the stability of the grasp. The second difference is the number of links for each finger, see Fig. 4, if we consider the human hand we have three links with four joints instead of two as the Barrett hand, there is a dependency between the links caused by a tendon that passes through the finger. This dependency of the last two links can be approximated with a linear relationship given by $\theta_4 = \frac{2}{3}\theta_3$ [21]. We simplify the human hand model considering a second dependency between θ_2 and θ_3 given by $\theta_3 = \tau\theta_2$.



(a) Finger (b) Hand

Fig. 4. Geometric model of a human finger and hand

As for the Barrett case, we generate two trajectories taking the minimal and maximal extensions of the finger, generating two intersection points P_1 and P_2. A set of finger points are located between the previous two intersection points, given that we can generate a set of finger trajectories by setting τ with different values, contrary to the Barrett hand. For the values of θ_1, we apply the same strategy of the Barrett hand and we set from the beginning the value of this joint that has the function to spread the fingers.

At the end, we can find an equation with only one unknown as the Barrett hand case.

$$\begin{aligned} &D(L_1\cos\theta_2 + L_2\cos(\theta_{inner} + (1+\tau)\theta_2) + \\ &+ L_3\cos(\theta_{outer} + (1+\frac{5}{3}\tau)\theta_2)) - A) - \\ &- B(L_1\sin\theta_2 + L_2\sin(\theta_{inner} + (1+\tau)\theta_2) + \\ &+ L_3\sin(\theta_{outer} + (1+\frac{5}{3}\tau)\theta_2)) - C) = 0 \end{aligned} \quad (8)$$

We consider the current position of the finger given by the inner link joint θ_{inner} and θ_{outer} for the outer joint link.

IV. Object Decomposition

A practical and fast solution to decompose an object to grasp is to partition it into some few components C_i, constructing a set of cutting planes passing through the axes of inertia C_{pi}. A sketch of the inertial axes decomposition algorithm IAD is presented in algorithm 2. A polyhedron P can be described by a set of vertices, edges and facets. It is important to say here that the facets we considered are triangles, affecting the split process because a triangulation has to be made.

Algorithm 2: IA Decomposition Algorithm Framework

```
input  : Polyhedron P
output : polyhedra P1,P2,..,Pn
begin
    IAi ←COMPUTE_INERTIAL_AXES;
    forall Each Inertial Axis IAi do
        Cpi ←COMPUTE_CUT_PLANE(IAi) ;
        Ci ← SPLIT(P, IAi);
    end
    ASSIGN_COMPONENT_TO_POLYHEDRA_SET;
end
```

The object decomposition is made by generating cut planes and by separating the object. Inertial axes are used to construct cutting planes. Each axis gives us the normal of the plane and the other axes generate the plane itself. The process of splitting is a basic one, finding all the facets and edges that intersect the cutting plane C_p. At each iteration, a partition gives only two components that belong to the set C_i and then is used to form two new subpolyhedra P_1 and P_2. As a result, we assign each vertex depending on which side of the cutting plane it lies. For the vertices that lie on C_p, they form a polygon, this one is triangulated and included in both polyhedra data structures.

The algorithm can be used recursively, taking each component of the polyhedra set as an input; a series of subpolyhedra will be generated.

In Fig. 5(a) we can see the model of one object and its correspondent axes of inertia. A candidate plane is defined by the mass center and one axis of inertia. A cutting plane example can be seen in Fig. 5(b).

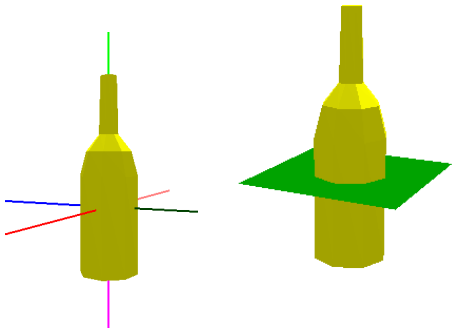


Fig. 5. a) Inertial Axes, b) Cut plane taking one inertial axis for object decomposition

The decomposition of the object can be defined as:

$$DP(P_H) = \{C_i \subset P_H \ with \ \cup_{i=1}^{n} C_i = P_H\} \\ and \ C_i \cap C_j = \emptyset, \forall_{i,j} \ i \neq j \qquad (9)$$

An example of the cut process is shown in Fig.6.

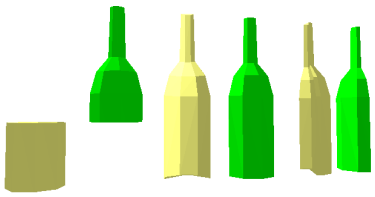


Fig. 6. Result of decomposition of a bottle by the three inertial axes planes, each plane produces two parts.

A. Planning Interactive Grasps

Algorithm 3: Interactive Grasps Algorithm

input : the robots $R1$ and $R2$, the environment E, the object O

output : the grasps G_1 and G_2

begin

  while not *StopCondition* ($Ntry == LIMIT$ *or* G_1 *and* G_2 *or* $CD = \emptyset$ **do**

    $[C_1, C_2] \leftarrow$OBJECT_DECOMPOSITION;

    $[G_1, G_2] \leftarrow$RANDOM_GRASP_PLANNER(C_1, C_2);

    if $G_1 \neq True$ *or* $G_2 \neq True$ **then**

      INCREMENT_NTRY;

    end

    UPDATE_ROBOT_CONFIGURATIONS;

  end

end

The planning for interactive grasps takes as an input the geometric model of the environment, including the object and the geometric and kinematic 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. The first one calls for a complete decomposition process that gives as a result the series of components of the object and generates a set of grasps for each component in the list. The second one, as we present in algorithm 3, 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 found, the planning process continues to compute a grasp for the second robot or end-effector taking the second component, if a second grasp is found, the process ended and the whole robot configurations for the interactive grasps are given as output. If no feasible grasps are found or only one grasp has been found, we take the components and we perform a new iteration until both grasps have been generated or the algorithm reaches a limit of tries.

V. RESULTS

The algorithms were implemented and tested in the motion planning platform Move3D [2]. The tests were performed using a 500 MHz Solaris SunBlade machine.

In Fig. 7, we show the grasps for a glass model when we make a decomposition of the object, we see that the grasp planner arrives to plan a grasp for each part of the object. The plan cut the object in two by the center of mass. One component is the base of the glass and the other component is the high part of the glass.

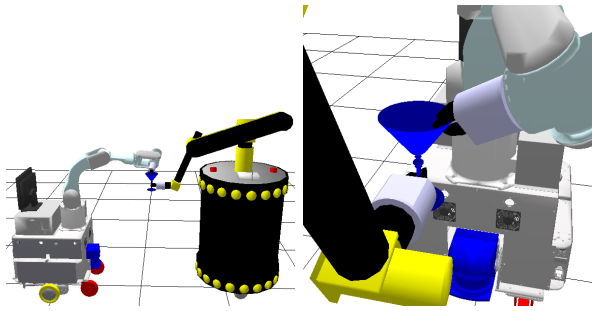
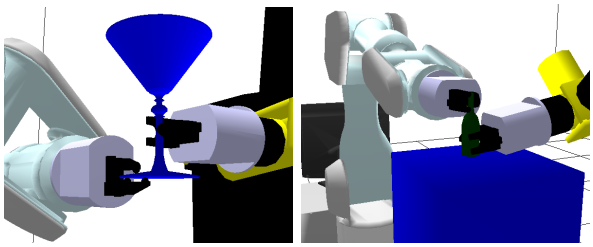


Fig. 7. Grasps for a decomposed object

We can see in Fig. 8(a) two grasps founded by the planner for two similar robots. The second grasp is located near to the first one when the whole polyhedron is taking into account. In Fig. 8(b), the planner was executed for a bottle but the object is on the table, the grasps are founded in such a way that the robots cannot collide with the table. In Fig. 9, we show the grasps for a T-shape object.

In table I we show the computing times for the grasp planning and object decomposition algorithms.



(a) (b)

Fig. 8. a) Grasps for similar robots, b) Planning avoiding obstacle

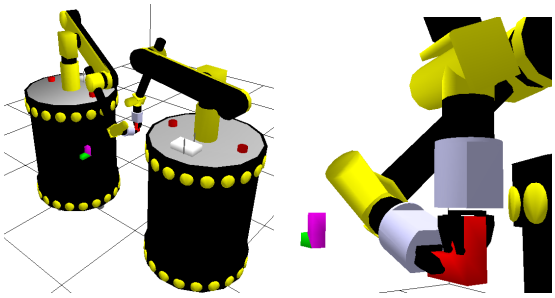


Fig. 9. Double grasp operation

TABLE I

GRASP PLANNER COMPUTING AND OBJECT DECOMPOSITION TIME.

Object	Cut Time	Grasp Time	Total Time	Facets
Bottle	6.11s	42.86s	48.97s	80
Glass	21.54s	301.2s	322.74s	2750

VI. CONCLUSION

In the paper we have presented a planning strategy to find grasps on objects for interactive manipulation tasks, for example, the hand over operation. Such grasps satisfy the force-closure criterion, the collision free condition and they are kinematically feasible. If a human is involved in a hand over task when a robot presents the object, the algorithm only can assure that the human will find at least one place where to grasp the object. A more complicated problem is when the human grasps the object and presents it to the robot, a remodeling of the object and human hand have to be made to plan the robot grasp correctly. An interaction between the planner and a vision module is necessary. The extension to compute grasps on the object for a human hand or a humanoid based on the strategy of the barrett hand is a simple and we believe that is a good option, instead of trying to compute the inverse kinematics of the whole hand. We think that the solution proposed here and the results obtained give us a base to continue developing algorithms for a more robust planner for service and humanoid robots. The planner can be extended to model sensors to consider, for example, the incertitude in the location of the object due to the noise in the data produced by these sensors.

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).

REFERENCES

[1] J. Badcock and R. Jarvis, "Wire-frame modelling of polyhedral objects from rangefinder data," *Robotica*, vol. 12, pp. 65–75, 1994.

[2] 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.

[3] M. Cutkosky, "On grasp choice, grasp models and the design of hands for manufacturing tasks," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 269–279, 1989.

[4] L. Petersson, P. Jensfelt, D. Tell, M. Strandberg, D. Kragic, and H. Christensen, "Systems integration for real-world manipulation tasks," in *In Proc. IEEE conf. on Robotics and Automation*, Washington D.C., 2002, pp. 2500–2505.

[5] J. kim, J. Park, Y. Hwang, and M. Lee, "Advanced grasp planning for handover operation between human and robot: three handover methods in esteem etiquettes using dual arms and hands of home-service robot," in *In Proc. International Conference on Autonomous Robots and Agents*, New Zealand, December 2004, pp. 34–39.

[6] S. Ekvall and D. Kragic, "Interactive grasp learning based on human demostration," in *Proc. IEEE conf. on Robotics and Automation*, 2004.

[7] S. Kang and K. Ikeuchi, "Toward automatic robot instruction from perception-mapping human grasps to manipulator grasps," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 1, pp. 81–95, 1997.

[8] G. Arechavaleta, C. Esteves, and J. Laumond, "Planning cooperative motions for animated characters," in *International Symposium on Robotics and Automation*, Mexico, 2004.

[9] C. Esteves, G. Arechavaleta, J. Pettre, and J. Laumond, "Animation planning for virtual mannequins cooperation," *ACM Transactions on Graphics*, 2006.

[10] J. Kuffner, "Autonomous agents for real-time animation," Ph.D. dissertation, Stanford University, December 1999.

[11] D. Ding, Y.H.Liu, and S. Wang, "Computing 3-d optimal form-closure grasps," in *In Proc. IEEE conf. on Robotics and Automation*, San Francisco, April 2000, pp. 3573–3578.

[12] A. Sudsang and J. Ponce, "New techniques for computing four-finger force-closure grasps of polyhedral objects," in *In Proc. IEEE conf. on Robotics and Automation*, vol. 1, Nagoya, Japan, 1995, pp. 1355–1360.

[13] J. Ponce and B. Faverjon, "On computing three-finger force-closure grasps of polygonal objects," *In IEEE Transactions on Robotics and Automation*, vol. 11, no. 6, pp. 868–881, December 1995.

[14] C. Borst, M. Fischer, and G. Hirzinger, "Grasping the dice by dicing the grasp," in *In Proc. IEEE/RSJ conf. on Intelligent Robots and Systems*, Las Vegas, Nevada, October 2003, pp. 3692–3697.

[15] E. Lopez-Damian, D. Sidobre, and R. Alami, "A grasp planner based on inertial properties," in *Proc. IEEE conf. on Robotics and Automation*, Barcelona, April 2005, pp. 766–771.

[16] J. Restrepo, "Modlisation d'objets 3d par construction incrmentale d'un maillage triangulaire dans un contexte robotique," Ph.D. dissertation, Universit Paul Sabatier, Janvier 2005.

[17] B. Mirtich, "Fast and accurate computation of polyhedral mass properties," *Journal of Graphics Tools*, vol. 1, no. 2, pp. 31–50, 1996.

[18] C. Ferrari and J. Canny, "Planning optimal grasps," in *In Proc. IEEE conf. on Robotics and Automation*, Nice, France, May 1991, pp. 2290–2295.

[19] J. Ponce and B. Faverjon, "On computing four-finger equilibrium and force-closure grasps of polyhedral objects," *International Journal of Robotics Research*, vol. 16, no. 1, pp. 11–35, February 1997.

[20] Y. Liu, "On computing n-finger force-closure grasps on polygonal objects," *International Journal of Robotics Research*, vol. 19, no. 2, pp. 149–158, 2000.

[21] H. Rijpkema and M. Girard, "Computer animation of knowledge-based human grasping," *In Computer Graphics*, vol. 25, no. 4, pp. 339–348, July 1991.