Unions of Balls for Shape Approximation in Robot Grasping

Markus Przybylski, Tamim Asfour and Rüdiger Dillmann

Abstract— **Typical tasks of future service robots involve grasping and manipulating a large variety of objects differing in size and shape. Generating stable grasps on 3D objects is considered to be a hard problem, since many parameters such as hand kinematics, object geometry, material properties and forces have to be taken into account. This results in a high-dimensional space of possible grasps that cannot be searched exhaustively. We believe that the key to find stable grasps in an efficient manner is to use a special representation of the object geometry that can be easily analyzed. In this paper, we present a novel grasp planning method that evaluates local symmetry properties of objects to generate only candidate grasps that are likely to be of good quality. We achieve this by computing the medial axis which represents a 3D object as a union of balls. We analyze the symmetry information contained in the medial axis and use a set of heuristics to generate geometrically and kinematically reasonable candidate grasps. These candidate grasps are tested for force-closure. We present the algorithm and show experimental results on various object models using an anthropomorphic hand of a humanoid robot in simulation.**

I. INTRODUCTION AND RELATED WORK

The increasingly aging society will benefit from intelligent domestic robots that are able to assist human beings in their homes. The ability to grasp objects is crucial to many supporting activities a service robot might perform, such as serving a drink, tidying up or giving water to the flowers, for example. Human beings perform grasps intuitively on almost any kind of object. In contrast, grasping is a challenging problem for robots. Knowledge of hand kinematics, object geometry, physical and material properties is necessary to find a good grasp, making the space of possible candidate grasps intractibly large to search in a brute-force manner. This is especially the case for modern dexterous robot hands with an increasing number of degrees of freedom.

A. Grasp Planning

Many approaches for grasp planning have been developed in the past. Grasp synthesis on the contact level concentrates primarily on finding a predefined number of contact points without considering hand geometry [1]. Some work on automatic grasp synthesis focusses especially on object manipulation tasks ([2],[3]). Shimoga [2] presents a survey on measures for dexterity, equilibrium, stability, dynamic behavior and algorithms to synthesize grasps with these properties. Li et al. [4] recorded grasps for basic objects using motion capturing and used this information to perform shape matching between the inner surface of the hand and novel objects. The resulting candidate grasps were clustered and pruned depending on the task.

Since simulators such as GraspIt! [5], OpenRAVE [6] and Simox [7] have become available it is possible to simulate candidate grasps with robot hand models on object models, where hand kinematics, hand and object geometries as well as physical and material properties and environmental obstacles can be taken into account. In the recent past, many researchers developed grasp planning methods based on these simulation environments. Berenson et al. (see [8],[9]) developed a grasp scoring function that considers not only grasp stability but takes also environmental obstacles and kinematic reachability into account. In [10] an integrated grasp and motion planning algorithm is presented where the task of finding a suitable grasping pose is combined with searching collision free grasping motions. Ciocarlie et al. [11] introduced the concept of *eigengrasps* which allows for grasp planning in a low-dimensional subspace of the actual hand configuration space. Goldfeder et al. [12] used the eigengrasp planner to build a grasp database containing several hands, a multitude of objects and the associated grasps. They used Zernike descriptors to exploit shape similarity between object models to synthesize grasps for objects by searching for geometrically similar objects in their database. They extended this approach to novel objects [13], where partial 3D data of an object are matched and aligned to known objects in the database to find suitable grasps.

A number of simulator-based approaches to grasp planning rely on shape approximation of 3D object models. The basic idea underlying these approaches is that many objects can be decomposed into component parts that can be represented by simplified geometric shapes. Then rules are defined to generate candidate grasps on these components which allows for pruning of the search space of possible hand configurations. This concept is also known as *grasping by parts*. The first method in this context was presented by Miller et al. [14] who used boxes, spheres, cylinders and cones to approximate the shape of the object. However, the user has to perform the decomposition of the object into these primitives manually. Goldfeder et al. [15] presented a method that automatically approximates an object's geometry by a tree of superquadrics and generates candidate grasps on those. Huebner et al. [16] developed an algorithm that decomposes objects into a set of minimum volume bounding boxes. While these approaches significantly reduce the complexity of grasp planning, this comes at a price. Many grasps a human would intuitively use might not be found due to poor object geometry approximation. Especially box decomposition yields only a

This work was supported by EU through the project GRASP.

All authors are with the Institute for Anthropomatics, Karlsruhe Institute of Technology, Karlsruhe, Germany {markus.przybylski, asfour, dillmann}@kit.edu

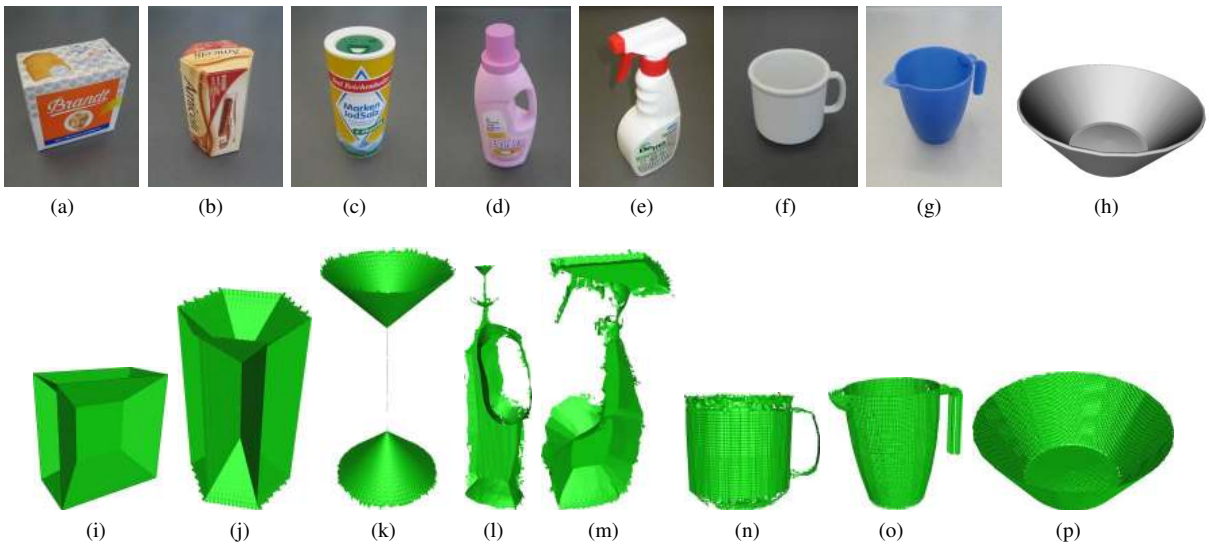


Fig. 1: Our test objects and their medial axes

relatively small number of candidate grasps for an object. This might be too restrictive for a real robot, as obstacles in the environment or kinematic constraints of the robot might turn many grasps infeasible. We believe for these reasons that it is desirable to carefully evaluate geometric information of the object to produce only high-potential candidate grasps, so costly validation of candidate grasps by collision-checking and testing for force-closure can be reduced to a minimum and therefore a large number of good grasps can be efficiently generated. We achieve this goal by representing the object's shape by the medial axis, which allows for efficient generation of geometrically and kinematically reasonable candidate grasps that have a high likelihood to be stable.

B. Medial Axis

Three-dimensional shapes can be approximated by inscribing balls of maximal diameter, i.e. balls that touch the shape boundary at least at two different points. The union of these balls' centers is called the *medial axis*. Together with the radii of these maximally inscribed balls it is referred to as the *medial axis transform*. The medial axis transform is a complete shape descriptor, i.e. it contains all necessary information to reconstruct the original object's shape. The medial axis was originally introduced by Blum [17] as a means for biological shape recognition and its computation and applications have been a fruitful area of research [18]. As it provides a compact representation of shapes, their features and connectivity it has been applied in numerous domains including CAD model simplification [19], tool-path creation in CAM [20], routing in sensor networks [21] and feature extraction in geometric design [22]. Yet we are not aware of any previous attempt to use the medial axis in grasp planning. In this paper, we show that the medial axis contains high-level symmetry information of an object that can be easily exploited to produce big numbers of candidate grasps that are very likely to be stable. This makes it possible to significantly reduce costly validation of candidate grasps. Fig. 1 shows the objects we use for our experiments and their respective medial axes.

The rest of the paper is organized as follows. In section II, we explain our grasp planning algorithm in detail. In section III, we present experimental results using a robot hand model and various object models in the OpenRAVE [6] simulation environment. In section IV, we draw conclusions and present ideas for future research.

II. MEDIAL AXIS GRASP PLANNING

In this section, we describe our grasp planning method in detail. In the first step, we sample the object's surface. This results in a 3D point cloud we use to compute the medial axis of the object in the second step. Then we analyze the medial axis by searching for familiar structures in the third step. In the fourth step, we use the extracted information to generate candidate grasps which are then tested for force-closure.

A. Sampling the object surface

Before we are able to compute the medial axis, we need to sample points on the object surface. We adopt and modify the method from [8] for this purpose. We calculate an axis-aligned bounding box for the object, sample this bounding box uniformly and calculate vectors from the sampling points on the bounding box perpendicular to the surfaces of the bounding box. We then use these vectors to perform ray collision on the object. The points where the rays intersect with the object are a good sampling of the object surface (Fig. 2a). For objects with openings like cups we additionally generate rays perpendicular to an imaginary cylinder surface inside the object to obtain sampling points on the inner surface of the object. We are aware that more sophisticated sampling techniques might be necessary for more complex objects, but as the focus of this paper is not on sampling, we

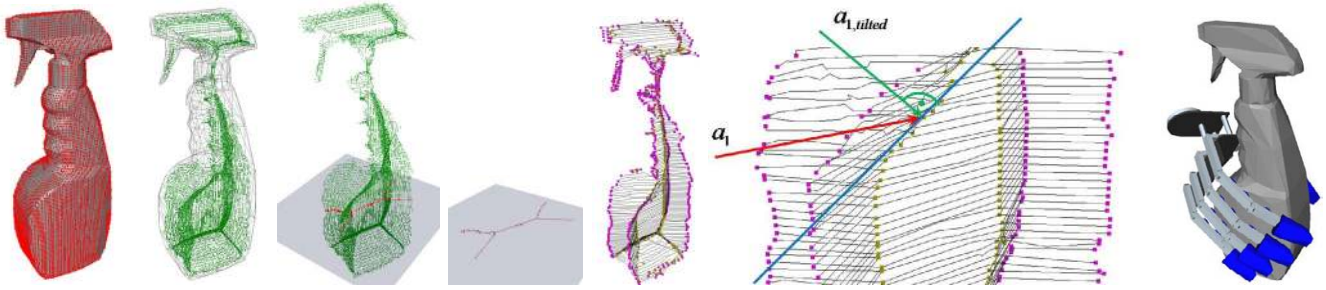


Fig. 2: Steps of our algorithm. Sampling of the objects' surface (a), computation of its medial axis (b), analysis of slices of the medial axis (c), minimum spanning tree of a slice (d), an object as a collection of minimum spanning trees (e), computation of approach directions (f), example of a resulting grasp (g). Figures best viewed in colour.

leave the evaluation of more advanced sampling methods to future work.

B. Computing the medial axis

In this step, we compute the medial axis of the object based on the surface sampling points we generated. The robust computation of the medial axis is quite difficult and still subject to research [18]. Therefore, we use the *Tight Cocone* software [23] for this purpose. Tight Cocone computes an approximation of the medial axis as a set of points together with a triangulation. Fig. 2b shows an object with its medial axis inscribed. In the following we refer to the set of points that approximates the medial axis as M.

C. Analysis of the medial axis

In this step, we analyze the object's medial axis representation we generated in the previous step in order to find familiar structures which can be exploited for generating candidate grasps. Before we continue with the details of our algorithm, we introduce the following terminology which will be used in the remainder of this paper:

- p denotes a plane that contains a projection of a subset of the medial axis points.
- $M(p)$ denotes the medial axis points projected into a plane p.
- The medial axis points in a plane p may be grouped into clusters c_i. $M(c)$ denotes the medial axis points associated to a cluster c. For planes p_i that contain only one cluster, $M(c)$ and $M(p)$ may be used as synonyms.
- $H_{conv}(X)$ denotes the convex hull of a set of points X.
- $B(H_{conv}(X)) \subset X$ denotes the subset of points $x_i \in X$ that lie at the boundary of $H_{conv}(X)$.
- $vol(H_{conv}(X))$ denotes the volume of $H_{conv}(X)$.
- $d_{min}(X, CoG(X))$ denotes the minimum distance of all points $x_i \in X$ to the center of gravity (CoG) of a cluster c.

In the proposed method we will also use methods from graph theory. In this context we will use the following terms:

- $deg(v)$ is the degree of a vertex $v \in V$ of a graph $G = (V, E, w)$ with a set of vertices V, a set of edges E and a weight function w.
- A *branching vertex* is a vertex $v \in V$ with $deg(v) \geq 3$.
- $MST(G)$ is the minimum spanning tree of a graph G. $MST(c)$ denotes the minimum spanning tree of the induced subgraph $G' = (V', E', w')$ of G where V' contains all medial axis points of the cluster c.

1) Subdividing the medial axis point cloud: As a complete medial axis is rather difficult to interpret, we subdivide it into slices and analyze each slice of the medial axis individually. Therefore, we define a set of equidistant planes p_i that are parallel to a virtual supporting surface beneath the object and obtain slices $M(p_i)$ of M by projecting each point $x_i \in M$ into its nearest neighbor plane p. A projection plane p_i and the corresponding slice $M(p_i)$ of the medial axis are illustrated in Fig. 2c and Fig. 3a.

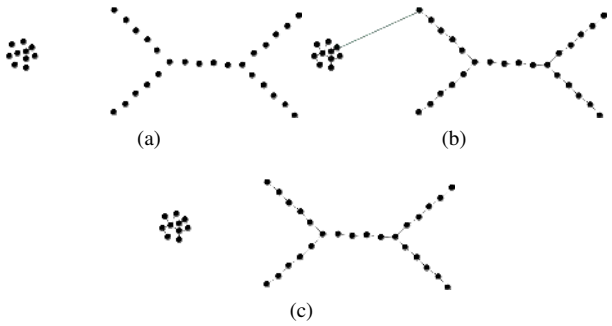


Fig. 3: Processing a slice of the medial axis: Points of a single slice (a), MST connecting all points with their respective nearest neighbours (b), two clusters obtained by pruning an edge with a distance weight $w > d_{cut}$ from the MST (c).

We interpret the medial axis data in a projection plane p as a weighted complete graph $G = (V, E, w)$, where the set of vertices V is given by the medial axis points $M(p)$ and the weight function w is defined by the pairwise euclidean distances of the points of the medial axis in the projection plane p. In order to determine each point's nearest neighbors we compute the minimum spanning tree $G_{MST} = MST(G)$ of this graph and obtain a tree structure of the medial axis data in each projection plane (see Fig. 3b). If the medial axis

of the object branches projection planes p_i exist where the points of the medial axis form clusters c_i. We detect these clusters by pruning all edges of the minimum spanning tree that have a distance weight w exceeding a certain cutoff threshold d_{cut} (see Fig. 3c). Then each cluster c has its own $MST(c)$ which is further analyzed.

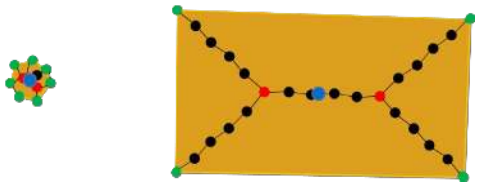


Fig. 4: Two clusters in a slice of the medial axis with their MSTs and several features highlighted in colour. Ocher: Convex hull $H_{conv}(c)$ of a cluster c. Green: Vertices at boundary of the convex hull $B(H_{conv}(c))$. Red: Branching vertices of the MST. Blue: Cluster's center of gravity (CoG). Figure best viewed in colour.

2) Extracting structural information from the medial axis: For each cluster c we identify the structure of its medial axis points $M(c)$ by evaluating its minimum spanning tree $MST(c)$ and the measures defined above (for an illustration see Fig. 4). We introduce the following structures:

- *Circle*: We classify the structure of a cluster c as a *circle* if $d_{min}(M(c), CoG(c))$ exceeds a threshold d_{thresh} and there are more than k_1 points at the boundary of its convex hull, i.e. if $|B(H_{conv}(M(c)))| > k_1$ holds true, where k_1 is a constant (see Fig. 5a).
- *Star with ring*: In contrast, we classify the structure as a *star with ring* if $d_{min}(M(c), CoG(c)) > d_{thresh}$ but $|B(H_{conv}(M(c)))| < k_2$, where k_2 is a constant (see Fig. 5c and Fig. 5d).
- *Tree*: We distinguish two types of trees. If $MST(c)$ contains exactly one branching vertex we call c a cluster with a *star* (see Fig. 5e and Fig. 5f). If $MST(c)$ contains exactly two branching vertices we call c a cluster with a *preference direction* (see Fig. 5b).
- *Symmetry axis element*: If $|B(H_{conv}(M(c)))| < k_2$ and the volume of the convex hull is very small, i.e. $vol(H_{conv}(M(c))) < k_3$, we classify the structure as a *symmetry axis element* (see Fig. 5g).

We give a short motivation and some example values for the constants introduced above. In this paper we choose $k_1 = 40$, $k_2 = 10$, $k_3 = 0.0001$, $d_{thresh} = d_{cut} = 5\rho$, where ρ is the sampling resolution for sampling the object's surface in the first step of our algorithm. d_{thresh} and d_{cut} depend on ρ because the density of the medial axis points roughly equals the surface sampling resolution ρ. The choice of k_2 is motivated by the idea that the structures with plane symmetries we want to exploit for grasp planning typically have a limited number of vertices at the boundary of their convex hull. Structures with more than 10 such vertices rather indicate the lack of such plane symmetries but instead suggest the presence of a symmetry axis. Or differently put: If you keep adding vertices to a convex polygon you will eventually end up with a circle. The reason for introducing k_3 is noise and discretization errors in the medial axis approximation. In theory, the presence of a symmetry axis should be indicated by the presence of single isolated medial axis points in a slice. In practice however, there will be a set of medial axis points clustered in a very small area.

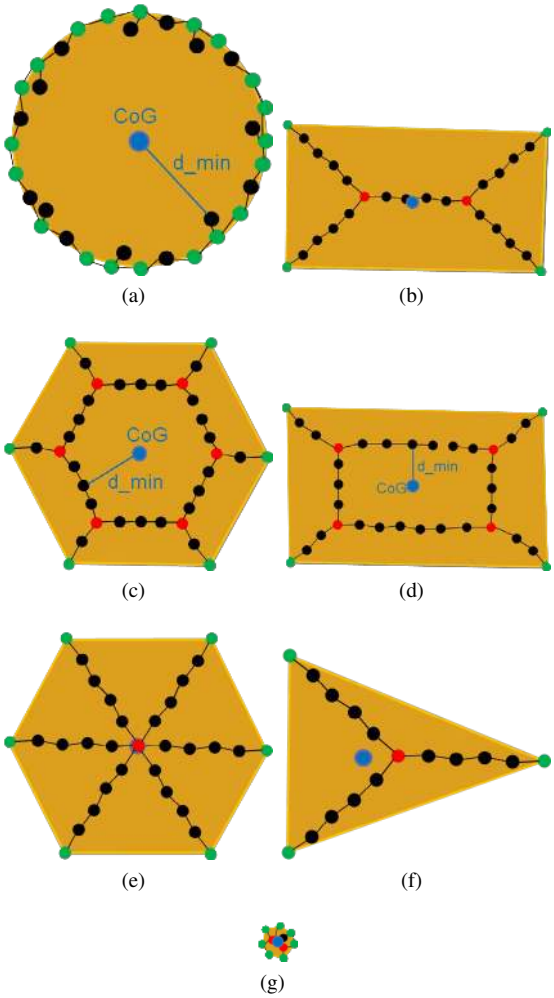


Fig. 5: Structures in slices of the medial axis: Ring (a); Preference direction (b); Stars with ring (c),(d); Stars (e),(f); Symmetry axis element (g). Figures best viewed in colour.

Fig. 2d and Fig. 5b show minimum spanning trees $MST(c)$ with a preference direction. Fig. 2e depicts an object as a collection of $MST(c_i)$ with highlighted branching and end vertices. Using the structural information we extracted from the medial axis we are able to generate promising candidate grasps in the next step.

D. Generating candidate grasps

In this section we present a set of heuristics describing how promising candidate grasps can be generated based on the structural information extracted from the medial axis in the previous step. Before we describe these heuristics, we explain which parameters we use to describe a candidate grasp. In [8], Berenson et al. defined a set of parameters describing a candidate grasp:

- An approach direction P_d of the hand
- A 3D target point P_t on the surface of the object the hand is approaching
- A roll angle P_r of the hand about the approach direction
- A vector of joint values P_p defining the preshape of the hand

For generating candidate grasps, the method described in [8] samples the object's surface by casting rays from a bounding box of the object. Each point where a ray intersects with the object is used as a target point P_t and the surface normal at P_t is used as an approach direction P_d. The palm normal of the hand is orientated towards P_d and moves toward the object until collision occurs. Then the hand is closed in order to wrap the fingers around the object. For each approach direction, a set of different roll angles P_r and preshapes P_p is tested. We adopt the grasp parameters from [8] but we use a different policy to generate them. Instead of sampling the object's surface, we carefully choose approach directions P_d and target points P_t based on the structural information we extracted from the medial axis. In addition to that, we are able to choose such roll angles P_r and preshapes P_p that have a high probability to result in a stable grasp when the hand collides with the object and the fingers wrap around it.

In this context we would like to emphasize the importance of the hand's roll angle and preshape. One of the most basic classes of grasps for human hands are grasps where the thumb as the first virtual finger opposes the other fingers which form a second virtual finger. In our opinion it is therefore a promising strategy to preferably generate *geometrically meaningful* candidate grasps where these two virtual fingers are likely to contact the object at opposing sides. Accordingly, if not stated otherwise, we suggest to use a parallel preshape for most of the candidate grasps we generate in the following and adjust the roll angle of the hand with respect to the symmetry information from the medial axis. We would like to state that favoring candidate grasps with two opposing virtual fingers does not impose any restriction on the kinematics of the robotic hand to be used. To the extent of our knowledge, all currently available robotic hands are capable of moving the thumb to a position opposed to the other fingers.

In the following we present various heuristics to generate candidate grasps depending on the structures we detected in slices and clusters of the medial axis. The general approach is always to look at several adjacent planes p_i and to derive promising approach directions and roll angles for the hand from the information in these p_i.

1) Clusters with a preference direction: If in a plane p we have a cluster c with a *preference direction*, there are exactly two branching vertices v_1, v_2. The vectors $a_1 = v_2 - v_1$ and $a_2 = v_1 - v_2$ are very promising approach directions P_d. For a_1 we use v_1 and for a_2 we use v_2 as the target point. If there are neighbor planes to p that contain also clusters with *preference direction* we can use the coordinates of their branching vertices to calculate tilted approach directions $a_{1,tilted}$ and $a_{2,tilted}$ to make our hand better conform to the surface of the object (see Fig. 2f, 2g). We also use this information to chose roll angles P_r that make it probable that the fingers will be able to wrap around the object.

2) Clusters with a star structure: If a cluster contains a *star* structure, we proceed in a similar way but we generate an approach direction P_d from every spike of the star to its branching vertex. There are two exceptions to this rule. For stars with three spikes we check if the star is symmetric with regard to one of its spikes. In this case, we treat the star as a structure with a preference direction. The second exception is for stars with four spikes. If every two opposing spikes in such a star define an axis of symmetry the profile of the object in this plane or cluster is likely to be a square for which we use the angle bisecting vectors of the spikes as approach directions.

3) Clusters with a symmetry axis element: Clusters with a *symmetry axis element* structure indicate that the profile of the object in this cluster is circular. If we find a cluster of this type we search in adjacent planes for more clusters with a *symmetry axis element* structure and use this information to generate approach directions perpendicular to the axis defined by the *symmetry axis element* structures.

4) Clusters with a circle structure: For clusters with a *circle* structure, we search for clusters with a *circle* structure in adjacent planes and generate approach directions perpendicular to the symmetry axis defined by the mean coordinates of the $M(c_i)$ in these planes.

5) Treating two clusters together: In general, candidate grasps can be individually generated for each cluster of a plane. Yet if there are exactly two clusters that are located close to each other the process of generating individual candidate grasps for each cluster will probably produce many unsuccessful candidate grasps. Therefore, we treat both clusters together. If both clusters are small in terms of the volume of their convex hull $vol(H_{conv}(M(c_i)))$ we use the clusters' centers of gravity (COG) as target points and the vectors b_1 and $-b_1$ connecting the COGs as approach directions. If one cluster is significantly smaller than the other we treat the bigger cluster independently and generate for the smaller cluster only an approach direction that is directed from the COG of the smaller cluster towards the COG of the bigger cluster.

6) Candidate grasps at the top and the bottom of objects: In order to generate candidate grasps for the top and the bottom of an object, we distinguish a number of different cases that take into account the structure of the medial axis in planes near the top and the bottom of the object. If there are planes with a *star with ring* structure, followed by planes with *star*, as for the prismatic chocolates box, or *preference direction* structure, as for the bread box object, we generate candidate grasps with roll angles P_r aligned with the preference direction or the spikes of the star. If there are *circles* or *symmetry axis element* structures, we generate candidate grasps in such a way that the approach direction coincides with the symmetry axis and use various different roll angles. For these candidate grasps it also makes sense to use a spherical hand preshape if this is supported by the hand model. If *circle* structures are present we also check by

TABLE I: Candidate grasps tested and percentage of force-closure (FC) grasps found

Objects	Our method		Method from [8]	
	Candidates	FC	Candidates	FC
Bread box	632	86.2%	13440	15.5%
Prismatic box	1344	90.7%	8512	36.0%
Salt can	2144	96.9%	7904	45.7%
Detergent	1996	65.9%	12672	26.2%
Spray	1304	55.1%	11200	21.2%
Cup	1428	59.5%	6688	37.0%
Pitcher	1124	47.0%	15504	25.9%
Salad bowl	504	68.5%	13648	4.5%

ray collision if the object has an opening. For open objects we generate candidate grasps at the rim of the object with roll angles of the hand that make it probable that at the final position of the hand the rim is between the thumb and the other fingers.

7) Size of the object: In our attempt to minimize costly testing of candidate grasps we also consider the size of the object with respect to the hand. We use ray collision to roughly estimate the width of the object geometry perpendicular to our approach direction and compare this value to the diameter of the biggest ball our hand can stably grasp. If the structure is too wide for the hand we do not have to generate a candidate grasp as it will fail.

III. EXPERIMENTS

In this section, we present some experiments for our grasp planning method. We perform experiments on the set of typical household objects depicted in Fig. 1. Except for the spray (Fig. 1e) and the detergent (Fig. 1d) and the cup (Fig. 1f), where the object model was generated using a 3D laser scanner ([24],[25]) all object models are handmodelled. We use the hand model of our humanoid robot ARMAR-III [26] to evaluate our algorithm in OpenRAVE. Each candidate grasp is evaluated by moving the hand with its palm facing the object along the approach direction until collision is detected. Then the fingers of the hand close until all fingers have contact with the object or the joint limits are reached and no finger link can move any more. Then we evaluate grasp quality by computing the commonly used worst-case epsilon measure for force-closure, as described by Canny and Ferrari [27]. For the force-closure computations we use a friction coefficient of 0.5. For each candidate grasp we test two different alternatives. The first alternative is the one described above. For the second alternative we move the hand after contact with the object a small distance d_{back} away from the object along the approach direction, before we close the fingers. This way, we are able to test candidate grasps where only the finger links have contact with the object. We choose $d_{back} = 2.5cm$ for all objects. In the special case of the pitcher's handle, we choose $d_{back,1} = 7cm$ and $d_{back,2} = 8cm$ to evaluate candidate grasps at the handle.

Some representative force-closure grasps produced by our algorithm are depicted in Fig. 6. We note that in our opinion the grasps produced by our method look quite intuitive for a human being.

Fig. 7 shows a visualization of the force-closure quality of all candidate grasps generated by our method. The rays indicate the approach directions of the hand towards the object. Each sphere indicates the final wrist position of a candidate grasp. Each blue sphere represents a force-closure grasp. For these grasps the diameter of the sphere is proportional to the force-closure score of the respective grasp, i.e. the biggest spheres represent the grasps with the highest force-closure rating. Red spheres represent grasps that do not fulfill the force-closure criterion.

To demonstrate the advantages of our method, we also compare it to Berenson's grasp planning algorithm [8] that uses only minimal knowledge of the object's geometry. As described above, Berenson's algorithm uniformly samples the object surface and uses the surface normals at the sampled surface points directly as approach directions for the hand. For Berenson's method as wells as for our own, we choose a surface sampling resolution ρ_1 of one ray per centimeter. In case of the bowl, we choose a sampling resolution ρ_2 of one ray per two centimeters, as the object is very big. Above that, we evaluate grasps with eight different roll angles of the hand around the approach direction, i.e. with roll angles α_i where

$$\alpha_i = 0.125\pi k, \quad k \in [0, 7]. \tag{1}$$

We also test $d_{back,1} = 0cm$ and $d_{back,2} = 2.5cm$ for all candidate grasps as also described above for our proposed method. The results of the comparison are given in Table I. We note that while Berenson's method produces far more candidate grasps, our algorithm has a significantly higher likelihood that the generated grasps are force-closure. The differences in efficiency are especially striking for objects like the bread box (Fig. 1a) and the salad bowl (Fig. 1h) which are too big to be grasped from arbitrary directions or with arbitrary roll angles of the hand. For most other objects in our experiments the proposed method still outperforms Berenson's method by a factor of two or more in terms of the fraction of force-closure grasps among the generated candidate grasps. This is especially interesting because it shows that our method significantly reduces the computation time overhead for collision detection and force-closure testing.

IV. DISCUSSION AND CONCLUSION

In this paper we presented a novel grasp planning algorithm based on the medial axis of 3D objects. The only requirement of this algorithm to be met by a robotic hand is the capability to oppose the thumb to the other fingers which is fulfilled by all hand models we know. We performed experiments on a set of household objects that show that our method effectively reduces the computational overhead for costly collision detection and force-closure testing by generating and evaluating only candidate grasps that - from a geometric point of view - have a high probability to be stable. In contrast to related work that uses boxes or superquadrics to approximate an object's geometry, the medial

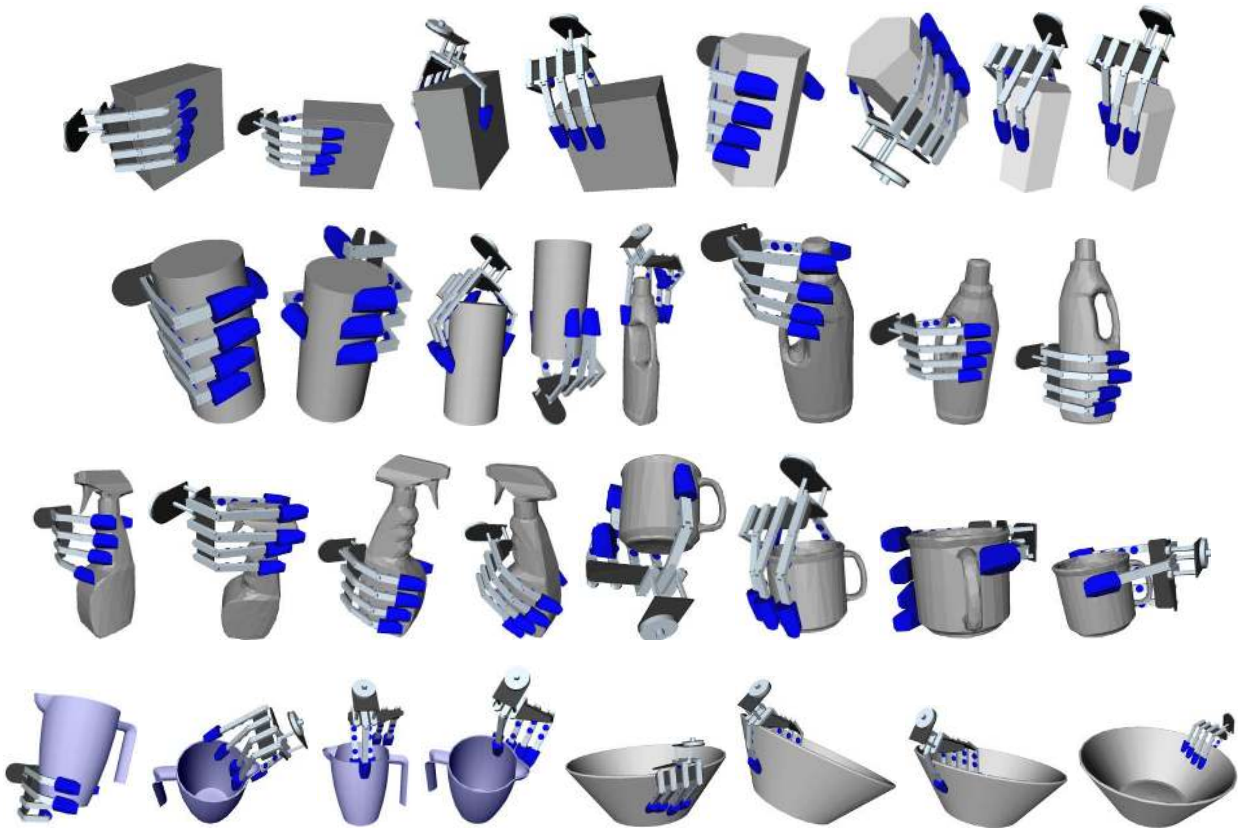


Fig. 6: Some representative force-closure grasps generated by our algorithm

axis representation of an object does not sacrifice potentially promising candidate grasps to poor geometry approximation. Instead, we showed that the medial axis provides valuable structural and symmetry information that can be easily used to generate big quantitities of force-closure grasps in an efficient manner. As we derived the heuristics for candidate grasp generation directly from geometric object properties our method produces many grasps that, in our opinion, seem quite intuitive to a human. We also emphasize that we consider the method presented in this paper as an extendable framework. If necessary, additional heuristics for candidate grasp generation can be easily defined.

In our current implementation of the presented method, we assume that the objects stand on a virtual surface and we use planes parallel to this virtual surface to subdivide the object's medial axis into slices. Yet this is no real limitation to the approach of medial axis-based grasp planning. Arbitrarily oriented planes may be used to subdivide the medial axis data and it might be interesting to investigate the benefits of such a strategy. A point where further improvement is possible are the criteria we currently use to classify structures of the medial axis in the planes. More sophisticated classification techniques might reduce the number of necessary parameters. During the experiments we also noticed that it is quite difficult to find force-closure grasps on structures that are small compared to the hand like the lid of a bottle. Extracting further information from the medial axis to calculate promising distances between palm and target point on the object as well as a more active consideration of hand kinematics might solve this issue and further increase the efficiency of our method. We also plan to combine the proposed approach with other methods and to evaluate this work on our humanoid robot ARMAR-III.

V. ACKNOWLEDGMENTS

We wish to thank Professor Tamal Dey from The Ohio State University for providing his Tight Cocone software and Rosen Diankov from Carnegie Mellon University for excellent support in all questions regarding the OpenRAVE simulator.

REFERENCES

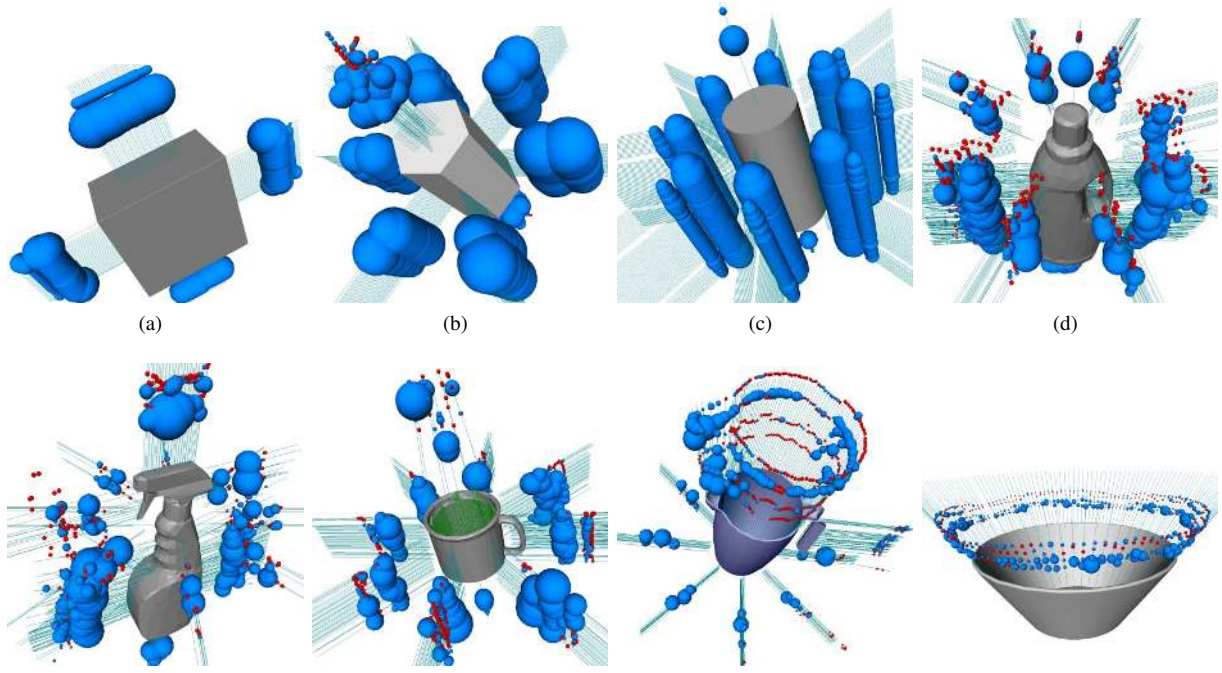
[1] Y. H. Liu, M. Lam, and D. Ding, "A complete and efficient algorithm for searching 3-d form-closure grasps in discrete domain," *IEEE Transactions on Robotics*, vol. 20, no. 5, pp. 805–816, 2004.

[2] K. B. Shimoga, "Robot grasp synthesis algorithms: A survey," *The Int. Journal of Humanoid Robotics*, vol. 15, no. 3, pp. 230–266, 1996.

[3] A. Morales, E. Chinellato, A. H. Fagg, and A. P. del Pobil, "Using experience for assessing grasp reliability," *Int. Journal of Humanoid Robotics*, vol. 1, no. 4, pp. 671–691, 2004.

[4] Y. Li, J. L. Fu, and N. S. Pollard, "Data-driven grasp synthesis using shape matching and task-based pruning," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 13, no. 4, pp. 732 –747, July-Aug. 2007.

[5] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *Robotics Automation Magazine, IEEE*, vol. 11, no. 4, pp. 110 – 122, Dec. 2004.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 7: Force-closure quality of the tested candidate grasps. Rays indicate the approach directions of the hand towards the object. Blue spheres indicate the wrist positions of the hand where grasps with a positive force-closure rating were found. The diameter of a blue sphere is proportional to the force-closure score of the respective grasp. Red spheres indicate grasps where the force-closure rating was zero.

[6] R. Diankov and J. Kuffner, "Openrave: A planning architecture for autonomous robotics," Robotics Institute, Tech. Rep. CMU-RI-TR-08-34, July 2008. [Online]. Available: http://openrave.programmingvision.com

[7] [Online]. Available: http://www.sourceforge.net/projects/simox

[8] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, "Grasp planning in complex scenes," in *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*, 29 2007-Dec. 1 2007, pp. 42 –48.

[9] D. Berenson and S. S. Srinivasa, "Grasp synthesis in cluttered environments for dexterous hands," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, Dec. 2008, pp. 189 –196.

[10] N. Vahrenkamp, M. Do, T. Asfour, and R. Dillmann, "Integrated grasp and motion planning," in *ICRA*, Anchorage, USA, Mai 2010.

[11] M. Ciocarlie, C. Goldfeder, and P. Allen, "Dimensionality reduction for hand-independent dexterous robotic grasping," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 29 2007-Nov. 2 2007, pp. 3270 –3275.

[12] C. Goldfeder, M. Ciocarlie, H. Dang, and P. K. Allen, "The columbia grasp database," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, May 2009, pp. 1710 –1716.

[13] C. Goldfeder, M. Ciocarlie, J. Peretzman, H. Dang, and P. K. Allen, "Data-driven grasping with partial sensor data," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, Oct. 2009, pp. 1278 –1283.

[14] A. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 2, Sept. 2003, pp. 1824 – 1829 vol.2.

[15] C. Goldfeder, C. Lackner, R. Pelossof, and P. K. Allen, "Grasp planning via decomposition trees," in *Robotics and Automation, 2007 IEEE International Conference on*, April 2007, pp. 4679 –4684.

[16] K. Huebner, S. Ruthotto, and D. Kragic, "Minimum volume bounding box decomposition for shape approximation in robot grasping," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, May 2008, pp. 1628 –1633.

[17] H. Blum, *Models for the Perception of Speech and Visual Form*. Cambridge, Massachusetts: MIT Press, 1967, ch. A transformation for extracting new descriptors of shape, pp. 362–380.

[18] D. Attali, J.-D. Boissonnat, and H. Edelsbrunner, *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, ser. Mathematics and Visualization. Springer Berlin Heidelberg, 2009, ch. Stability and Computation of Medial Axes: A State-of-the-Art Report, pp. 109–125.

[19] A. Thakur, A. G. Banerjee, and S. K. Gupta, "A survey of cad model simplification techniques for physics-based simulation applications," *Computer-Aided Design*, vol. 41, no. 2, pp. 65–80, February 2009.

[20] M. Held, "Vroni: An engineering approach to the reliable and efficient computation of voronoi diagrams of points and line segments," *Comput. Geom. Theory Appl.*, vol. 18, pp. 95–123, 2001.

[21] J. Bruck, J. Gao, and A. Jiang, "Map: Medial axis based geometric routing in sensor networks," *Wireless Networks*, vol. 13, no. 6, pp. 835–853, December 2007.

[22] M. Hisada, A. G. Belyaev, and T. L. Kunii, "A skeleton-based approach for detection of perceptually salient features on polygonal surfaces," *Computer Graphics Forum*, vol. 21, pp. 689–700, 2002.

[23] T. Dey and S. Goswami, "Tight cocone: a water-tight surface reconstructor," in *Proceedings of the eighth ACM symposium on Solid modeling and applications*. ACM, 2003, pp. 127–134.

[24] A. Kasper, R. Becher, P. Steinhaus, and R. Dillmann, "Developing and analyzing intuitive modes for interactive object modeling," in *International Conference on Multimodal Interfaces*, 2007.

[25] R. Becher, P. Steinhaus, R. Zöllner, and R. Dillmann, "Design and implementation of an interactive object modelling system," in *International Symposium on Robotics*, 2006.

[26] T. Asfour, K. Regenstein, P. Azad, J. Schröder, A. Bierbaum, N. Vahrenkamp, and R. Dillmann, "Armar-iii: An integrated humanoid platform for sensory-motor control," in *Proceedings of the 7th IEEE-RAS International Conference on Humanoid Robots*, 2006.

[27] C. Ferrari and J. Canny, "Planning optimal grasps," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, May 1992, pp. 2290 –2295 vol.3.