2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Minimum Volume Bounding Box Decomposition for Shape Approximation in Robot Grasping

Kai Huebner, Steffen Ruthotto and Danica Kragic

Abstract— **Thinking about intelligent robots involves consideration of how such systems can be enabled to perceive, interpret and act in arbitrary and dynamic environments. While sensor perception and model interpretation focus on the robot's internal representation of the world rather passively, robot grasping capabilities are needed to actively execute tasks, modify scenarios and thereby reach versatile goals. These capabilities should also include the generation of stable grasps to safely handle even objects unknown to the robot. We believe that the key to this ability is not to select a good grasp depending on the identification of an object (e.g. as a cup), but on its shape (e.g. as a composition of shape primitives). In this paper, we envelop given 3D data points into primitive box shapes by a fit-and-split algorithm that is based on an efficient Minimum Volume Bounding Box implementation. Though box shapes are not able to approximate arbitrary data in a precise manner, they give efficient clues for planning grasps on arbitrary objects. We present the algorithm and experiments using the 3D grasping simulator *GraspIt!* [1].**

I. INTRODUCTION

In the service robot domain, researchers and programmers provide each robot with manifold tasks to do in order to aid and support, e.g. clearing a table or fill a dishwasher after lunch. The knowledge about such aims might be either hard-coded or learned in a more intelligent manner, e.g. by a person teaching the robot how to clear a table. Such scenarios are known as Learning- or Programming-by-Demonstration applications. However, whether in an office, in health care or in a domestic scenario, a robot has to finally operate independently to satisfy various claims. Thus, the handling of objects is a central issue of many service robot systems. Robot grasping capabilites are therefore essential to actively execute tasks, modify scenarios and thereby reach versatile goals in an autonomous manner.

For grasping, numerous approaches and concepts have been developed over the last decades. Designing grasping systems and planning grasps is difficult due to the large search space resulting from all possible hand configurations, grasp types, and object properties that occur in regular environments. Early work on contact-level grasp synthesis focused mainly on finding a fixed number of contact locations without regarding hand geometry [2]. Considering specifically object manipulation tasks, the work on automatic grasp synthesis and planning is of significant relevance [3], [4], [5]. The main issue here is the automatic generation of stable grasps assuming that the model of the hand is known and that certain assumptions about the object (e.g. shape, pose) can be made. Taking into account both the hand kinematics as well as some a-priori knowledge about the feasible grasps has been acknowledged as a more flexible and natural approach towards automatic grasp planning [4]. It is obvious that knowledge about the object shape, as also the task on hand, is quite meaningful for grasp planning [6].

This is important for our scenario, in which we aim at providing a robot actuator system with a set of primitive actions, like *pick-up*, *push* or *erect* an arbitrary object on a table. For performing such basic actions, an object has to be modeled from 3D sensory input, e.g. from range or dense stereo data. However, we state the question up to which detail this is necessary in terms of grasping.

All authors are or were with the KTH – Royal Institute of Technology, Stockholm, Sweden, as members of the Computer Vision & Active Perception Lab., www home page: http://www.csc.kth.se/cvap, e-mail addresses: {khubner, ruthotto, danik}@kth.se.

II. MOTIVATION

Modeling range data is a crucial, but also difficult task for robot manipulation. The source data offered by range sensors or dense stereo camera systems is a more or less distorted and scattered cloud of 3D points of the scenario. A higher-level representation of these points as a set of shape primitives (e.g. planes, spheres or cylinders) obviously gives more valuable clues for object recognition and grasping by compressing information to their core. Most approaches that consider this problem are likewise bottom-up, starting from point-clouds and synthesizing object shapes by using superquadrics (SQs). Superquadrics are parametrizable models that offer a large variety of different shapes. Considering the problem of 3D volume approximation, only superellipsoids are used out of the group of SQs, as only these represent closed shapes. There is a multitude of state-of-the-art approaches based on parametrized superellipsoids for modeling 3D range data with shape primitives [7], [8], [9], [10].

Assuming that an arbitrary point cloud has to be approximated, one SQ is not enough for most objects, e.g. a screw or an office chair (see Fig. 1). The more complex the shape is, the more SQs have to be used to conveniently represent its different parts. However, good generality is not possible with few parameters for such cases [7]. Besides the advantages of immense parametrization capabilities with at least 11 parameters, intensive research on SQs has also yielded disadvantages in two common strategies for shape approximation. The first strategy is region-growing, starting with a set of hypotheses, the *seeds*, and let these adapt to the point set. However, this approach has not proved to be effective [8] and suffers from the refinement problem of the seeds [10]. The second strategy uses a split-and-merge

978-1-4244-1647-9/08/$25.00 ©2008 IEEE.

technique. Splitting up a shape and merging parts again is more adapted to unorganized and irregular data [8].

Independent of the strategy used, the models and seeds, respectively, have to be fitted to the 3D data. This is usually done by least square minimization of an inside-outside fitting function, as there is no analytical method to compute the distance between a point and a superquadric [9]. Thus, SQs are though a good trade-off between flexibility and computational simplicity, but sensitive to noise and outliers that will cause imperfect approximations. This is an important issue, as our work is oriented towards the use of dense stereo accompanied by highly distorted and incomplete data.

We observed that modeling 3D data by shape primitives is a valuable step for object representation [11]. Sets of such primitives can be used to describe instances of the same object classes, e.g. cups or tables. However, it is not our aim to focus on such high-level classifications or identification of objects, but on grasping. We moreover approach a deeper understanding of objects by interaction instead of observation for that purpose, e.g., if there is an object that can be picked up, pushed and filled, it can be used as a cup. Processing an enormous number of data points takes time, both in approaches that use the raw points for grasp hypotheses and in those that approximate as good as possible by shape primitives. In this context, a question remains: *how rudimentary can a model of a thing be in order to be handled successfully and efficiently?* While comparable work is placed mostly at extrema of this scale, e.g. by using pairs of primitive feature points [12] or a-priori known models for each object [11], we are interested in looking into which primitive shape representations might be sufficient for the task of grasping arbitrary, unseen objects.

We believe that a mid-level solution is a promising trade-off between good approximation and efficiency for this purpose. Complex shapes are difficult to process, while the simple produce worse approximation. However, we can access valuable methods to handle approximation inaccuracies for grasping like haptic feedback, visual servoing and advanced grasp controllers for online correction of grasps. We prefer general fast online techniques instead of pre-learned offline examples, thus the algorithm's efficiency is the more important issue. Unknown objects are hardly parametrizable but need real-time application for robot grasping. A computation in terms of minutes for a superquadric approximation is therefore not feasible.

We adopt these motivations to propose an algorithm based on boxes as a mid-level representation. In our approach, we combine different incentives on simplicity of boxes, efficiency of hierarchies and fit-and-split algorithms:

1) We aim for *simplicity* stating the question if humans approach an apple for grasping with their hand in another way as they approach a cup, or a pen in another way as a fork? While there are surely differences in fine grasping and task dependencies, differences in approaching these objects seem quite marginal.
2) Computational efficiency of *hierarchies* was pointed out in several other approaches that compose models with use of superquadric primitives [7], [9], [13].
3) While seed growing as a bottom-up strategy has several drawbacks, and a split-and-merge strategy both needs top-down (split) and bottom-up (merge), *fit-and-split* algorithms are purely top-down and thereby iteratively implementable in a one-way hierarchical manner.

Following our primary incentive, we chose boxes as a very simple and roughly approximating representation.

III. ALGORITHM

A. Computing Bounding Boxes

The algorithm of minimum volume bounding box computation proposed by Barequet and Har-Peled [14] will form the base for our approach. Given a set of n 3D points, the implementation of the algorithm computes their Minimum Volume Bounding Box (MVBB) in $O(n \log n + n/\varepsilon^3)$ time, where ε is a factor of approximation. The algorithm is quite efficient and parametrizable by several optimizations. Performing the computation on an arbitrary point cloud, a tight-fitting, oriented MVBB enclosing the data points is produced (see the example in Fig. 1).

B. Decomposition of MVBBs

Based on this algorithm, we aim at iteratively splitting the box and the data points, respectively, such that new point sets yield better box approximations of the shape. Iterative splitting of a root box corresponds to the build-up of a hierarchy of boxes. Gottschalk *et al*. [15] present the OBBTree (Oriented Bounding Box Tree) for this purpose, where the goal is efficiently collision detection between polygonal objects. The realization of the splitting step is quite straightforward: each box is cut at the mean point of the vertices, perpendicular to the longest axis. This is done iteratively, until a box is not dividable any more. Similar work on division of polygonal structures for grasping has been proposed by others [16], [17]. In our case, these strategies are suboptimal or less applicable. Splitting into many small boxes is against our aim of approximating a shape with as few boxes as possible. Additionally, though the MVBB algorithm is efficient, a fitting step after each splitting consumes valuable computation time. Finally, in our application both the splitting at the mean point is not optimal and we can not access polygonal structures, but point clouds only. Thus, another heuristic to find a "good" split is needed.

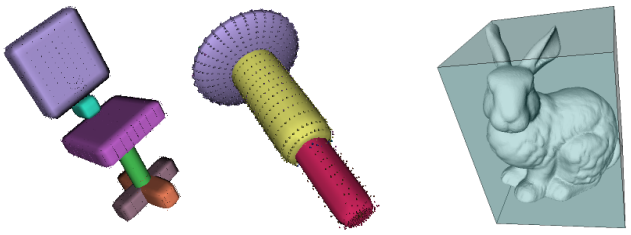


Fig. 1. Left: Examples of range data approximated by sets of superquadrics [8]. Right: The Stanford bunny model and the root MVBB of its vertices.

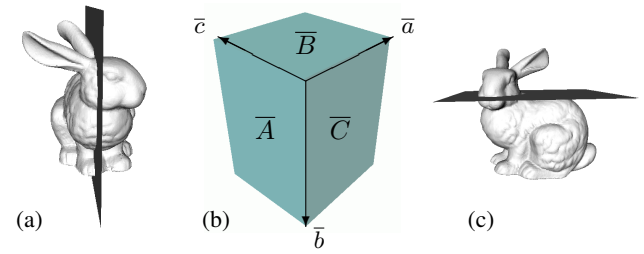


Fig. 2. (a) A mean cut of the bunny model. (b) We restrict to box parallel cutting planes. (c) A good cut parallel to the root MVBB plane $\overline{B}$.

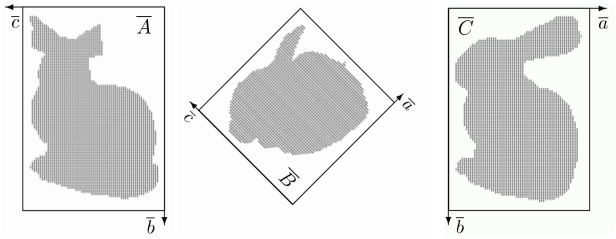


Fig. 3. Bunny sample projections onto the three faces of the root box (Fig. 1) according to the face-parallel cutting scheme in Fig. 2(b).

Therefore, we will have to define what a "good" split is. Fig. 2(a) shows a mean cut, which obviously is not good for our task. It does not improve the approximation with boxes of both new halfs, but is also not intuitive in terms of dividing the bunny in semantic parts, e.g. head and body. Even with planar cuts, finding the best intuitive one would correspond to an extensive search and comparison of a lot of planes, differing in position and orientation. Therefore, we decide to test only the planes parallel to the parent MVBB, like Fig. 2(b) and (c) show. As a measure of a good split, we can consult the relation of the box volume before and after the splitting: a split of the parent box is the better, the less volume the two resulting child MVBBs will include. This is intuitively plausible, as shape approximation is better with highly tight-fitting boxes.

C. Computing the Best Split

As motivated, we just test planes parallel to the three box surfaces for the best splitting plane. Each MVBB has six sides, whereof opposing pairs are parallel and symmetric. Inbetween each of these pairs, we can shift a cutting plane. Fig. 2(b) depicts this restriction on a splitting parallel to $\overline{A}$, shifted by a distance a, and $\overline{B}$ by b and $\overline{C}$ by c, respectively. A computation of new MVBBs for each value of the *split parameters* a, b and c would take a lot of computational effort. Therefore, we estimate the best cut by first projecting the data on 2D grids which correspond to the surfaces $\overline{A}$, $\overline{B}$ and $\overline{C}$. The bunny sample data projections onto the three surface grids of the root MVBB are shown in Fig. 3, reducing the problem of splitting a 3D box by a surface-parallel plane to splitting a 2D box by an edge-parallel line. For the sake of efficiency, it is thereby abstracted from the real 3D volume of the shape. The figure shows that there are six valid split directions left, two for each of the surfaces $\overline{A}$, $\overline{B}$ and $\overline{C}$.

As mentioned above, we define the best split as the one that minimizes the summed volume of the two partitions. Thus, we now test each discretized grid split along the six axes, using the split parameters. We define a split measure $\theta(\overline{\mathcal{F}}, \overline{f}, i)$ with $\overline{\mathcal{F}} \in \{\overline{A}, \overline{B}, \overline{C}\}$ being the projection plane to split, $\overline{f}$ being one of the two axes that span $\overline{\mathcal{F}}$, and i as the grid value on this axis that defines the current split. Consequently, we have six possible split measures

$$\begin{aligned}
&\theta_1(\overline{A}, \overline{c}, i_1),\ i_1 \in \mathbb{N}^{<c_{\max}}, \theta_2(\overline{A}, \overline{b}, i_2),\ i_2 \in \mathbb{N}^{<b_{\max}},\\
&\theta_3(\overline{C}, \overline{a}, i_3),\ i_3 \in \mathbb{N}^{<a_{\max}}, \theta_4(\overline{C}, \overline{b}, i_4),\ i_4 \in \mathbb{N}^{<b_{\max}},\\
&\theta_5(\overline{B}, \overline{a}, i_5),\ i_5 \in \mathbb{N}^{<a_{\max}}, \theta_6(\overline{B}, \overline{c}, i_6),\ i_6 \in \mathbb{N}^{<c_{\max}} \qquad (1)
\end{aligned}$$

to compare. Their minimum gives reason to the best split. The minimization of each $\theta(\overline{\mathcal{F}}, \overline{f}, i)$ is implemented as follows. For each i that cuts $\overline{\mathcal{F}}$ perpendicular to $\overline{f}$ in two rectangular shapes, we compute the two resulting minimum grid areas by lower and upper bounds. The i that yields the minimum value is the best cut of $\overline{\mathcal{F}}$ along $\overline{f}$. $\theta(\overline{\mathcal{F}}, \overline{f}, i)$ is computed as the fraction between the whole projection rectangular and the sum of the two best cut rectangles. Though this is a very approximative method, it is quite fast, as rectangle volume and bounds are easy to generate. The best bunny cuts for which rectangular volume and the corresponding values $\theta_{1\ldots6}$ are minimal are shown in Fig. 4.

D. Building a Fit-and-Split Hierarchy

According to the best split θ^*, which would be θ_1 or θ_2 in this exemplary case, the original point cloud can be divided into two subsets of the data points. These can be used as inputs to the MVBB algorithm to produce two child MVBBs of the root MVBB. In this way, the complete fit-and-split method can iteratively be performed. It is important to note that by MVBB re-computation, the MVBBs will greatly differ in orientation and scale from the box cuts in Fig. 4.

Additionally, the previous step of cutting along one of the six directions is just equal to computing an approximative gain value, for the purpose of efficiency. As an iteration breaking criterion, we now subsequently test the real MVBB volume gain Θ^* of the resulting best split measure θ^*. Therefore, we compute the gain in volume defining

$$\Theta^* = \frac{V(\mathbf{C}_1) + V(\mathbf{C}_2) + V(\mathbf{A}^{\setminus\mathbf{P}})}{V(\mathbf{P}) + V(\mathbf{A}^{\setminus\mathbf{P}})}, \qquad (2)$$

where $\mathbf{A}$ is the complete set of boxes in the current hierarchy, $\mathbf{P}$ is the current (parent) box, $\mathbf{C}_1$, $\mathbf{C}_2$ are the two child boxes produced by the split, and V being a volume function.

We decide further process on two constraints. First, if the gain is too low, a split is not valuable. For this purpose, we include a threshold value t. The precision of the whole approximation can be parametrized by simply preventing a split if Θ^* exceeds t. Second, we do not preserve boxes in the hierarchy that include a very low number of points. By this process, noise in the point data can be handled.

It might also be important in this context that in Fig. 4, θ_6 would intuitively be a probably valuable next cut below the bunny's ear. However, the best split computation presented (Section III-C) will not find this cut. Finding this cut is not that simple, especially when distorted, sparse and insecure

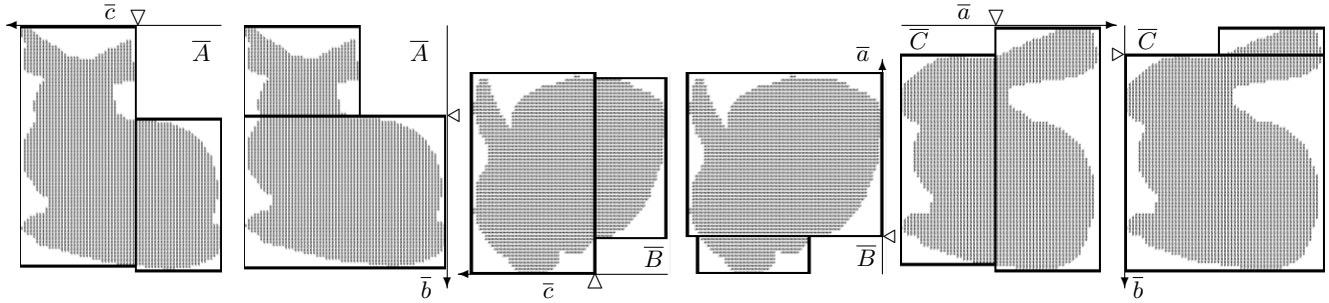


Fig. 4. Best cuts along the six box directions and cut positions i marked by triangles. The corresponding volume values $\theta(\overline{\mathcal{F}},\overline{f},i)$ are presented below.

data is provided. An add-on for the solution of this problem would therefore be more complex and time-consuming. The bunny is a very ideal model, as it is artificial, complete, and data points are very dense. As it is our aim to evaluate our algorithm also on real sensory data, we can not assume such ideal conditions and do not handle such situations presently.

IV. EXPERIMENTS

A. MVBB Splitting Evaluation

In the following, we present some experiments for the proposed fit-and-split algorithm. For all experiments, we fix the two original MVBB approximation parameters (see [14]). The grid parameter defines orientations that induce bounding box approximation in an exhaustive way, so we keep it small at 3. We decide to sample sets of 200 points, so even very large point clouds are reduced and efficiently handled. We found that these settings provide a good trade-off between quality and efficiency of each split for our application. The main parameter that we are going to change in the experiments is the gain threshold t.

We evaluate the behaviour of the algorithm on several types of input data by taking both ideal point clouds emerging from complete and unnoisy simulative vertice models (4 models) and real laser scan excerpts (5 scans) as input data. The latter is therefore incomplete and noisy, but at least regular due to the scan sampling. One sample is produced from a stereo vision system that offers three-dimensional points by disparity, including incomplete, noisy and irregular data. Fig. 5 shows these samples divided with different gain thresholds $t \in \{0.90, 0.94, 0.98\}$. The corresponding overview on point sets, computation time and number of boxes for the groups is given in Tab. I.

B. MVBB Grasping Evaluation

The best way to find a good grasp is said to be grasp candidate simulation [4], [9]. Miller *et al*. have simulated pre-models and shape primitives using their public grasp simulation environment *GraspIt!* [4]. So we also base our evaluation on model-based grasping in *GraspIt!*.

The first iteration, performed as proposed in Section III-D, yields the root node of the box tree. The root box has six faces, each of which we use for four grasp hypotheses parallel to its spanning edges. For symmetric grippers these could be reduced to two grasp hypotheses, but as we will use an asymmetric 5-finger hand model [18] in our simulation, we take these four. After the grasps on the root box have been performed, we apply the decomposition algorithm to produce MVBB approximations with gain parameters 0.90, 0.94 and 0.98 for the pure model data, Fig. 5(a)-(c), only. All faces are then collected from a final approximation, before occluded and ungraspable ones are removed. The applied grasping method is simple here: each initial position is set to a constant distance from the face's center aligned to its normal, i.e. the approach vector is the negative face normal. The hand is set to an intuitively good pose to have a large opening angle towards the object. We let the hand approach along the normal until a contact is detected. After contact, the hand retreats a small distance before we call GraspIt!'s auto-grasp function which uniformly closes the fingers of the hand. When all fingers are in contact with the object, we evaluate the two standard grasp quality measures that come with GraspIt!: ε, a worst-case epsilon measure for force-closure grasps, and V, an average case volume measure [1].

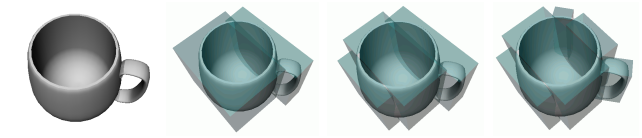
To compare the grasps that we get from this sequence, we compute a random "spherical" grasp evaluation for each model. Initial hand positions are placed on a sphere, with the approach vector oriented towards the object's center of mass and two spherical coordinates and a hand orientation angle configuring the hand's pose (discretized by steps of 10 deg).

We find that geometrical detection of blocked faces reduces the number of graspable faces drastically. Each spherical evaluation includes 22104 grasps. Referring to the grasp quality comparison between spherical and box evaluation for our models (Fig. 6), a_1 resulted from a test of only f=6 valid face grasps from the t =0.94 decomposition. Same pairs for

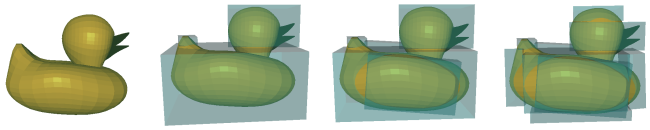
TABLE I

STATISTICS OF THE EXPERIMENTS PRESENTED IN FIG. 5.

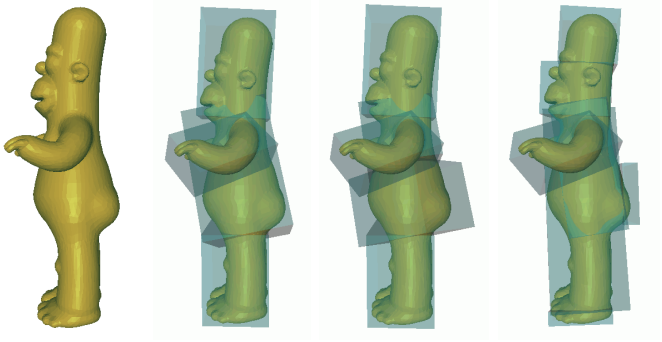
Model	#points	#boxes—sec (t=0.90)	#boxes—sec (t=0.94)	#boxes—sec (t=0.98)
Mug	1725	2—4	3—7	5—11
Duck	1824	3—7	5—9	9—14
Homer	5103	4—10	5—13	7—16
Bunny	35947	2—5	4—11	11—30
Stapler	313	2—2	2—2	2—2
Puncher	449	3—3	3—3	4—3
Can	1266	2—4	5—8	9—10
Phone	1461	3—5	4—5	9—12
Laptop	4199	3—7	4—8	6—15
Can2	9039	2—7	7—20	16—46



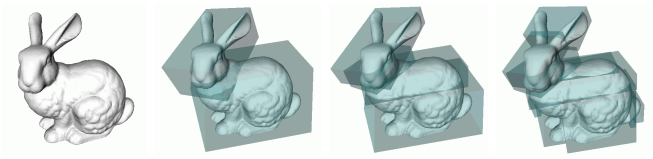
(a) Mug (model): MVBBs (2,3,5) produced with t=0.90, 0.94, 0.98.



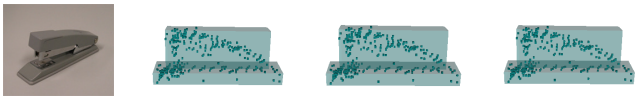
(b) Duck (model): MVBBs (3,5,9) produced with t=0.90, 0.94, 0.98.



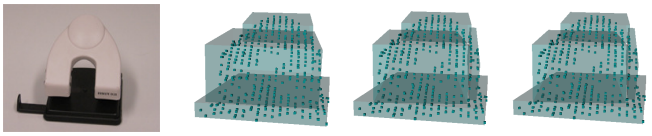
(c) Homer (model): MVBBs (4,5,7) produced with t=0.90, 0.94, 0.98.



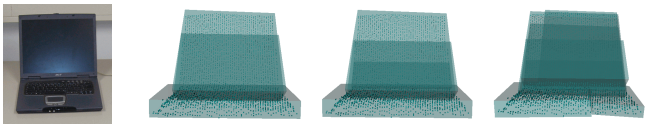
(d) Bunny (model): MVBBs (2,4,11) produced with t=0.90, 0.94, 0.98.



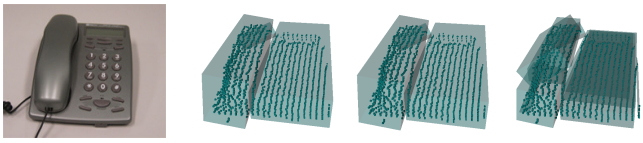
(e) Stapler (scan): MVBBs (2,2,2) produced with t=0.90, 0.94, 0.98.



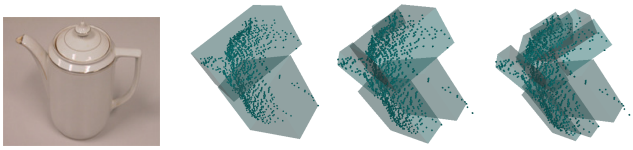
(f) Puncher (scan): MVBBs (3,4,4) produced with t=0.90, 0.94, 0.98.



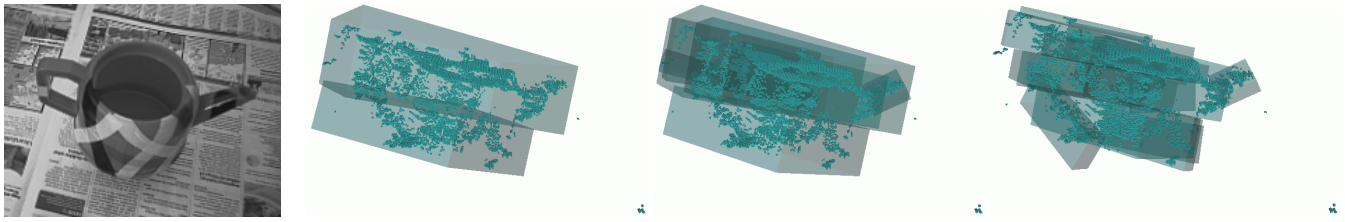
(g) Notebook (scan): MVBBs (3,4,6) produced with t=0.90, 0.94, 0.98.



(h) Phone (scan): MVBBs (3,3,9) produced with t=0.90, 0.94, 0.98.



(i) Can (scan): MVBBs (2,5,9) produced with t=0.90, 0.94, 0.98.



(j) Can2 (stereo): MVBBs (2,7,16) produced with t=0.90, 0.94, 0.98.

Fig. 5. Examples of box decomposition using different gain thresholds t=0.90, 0.94, 0.98, where numbers in brackets correspond to numbers of boxes. (a)-(d) are complete, dense and unnoisy 3D models. (e)-(i) result from incomplete, dense and less noisy, but manually pre-segmented range scans. (j) is produced by an incomplete, sparse and noisy, automatically pre-segmented stereo disparity point cloud.

the other depicted samples are: b_1=(16,Root), b_2=(32,0.94), c_1=(48,0.98) and c_2=(22,0.90). Concluding, the box decomposition effectively produces very few hypotheses which still feature good grasp quality. Note that only force-closure grasps ($\varepsilon > 0$) are drawn in Fig. 6.

V. DISCUSSION AND CONCLUSION

In our approach, we combined motivations known from the shape approximation and grasping literature. We prune the search space of possible approximations by rating and decomposing bounding boxes. Related work uses more complex superquadrics as approximation elements and confirm that grasp planning on finer components is likely to find better grasps than returning the first stable grasp [9]. This intuitively corresponds to the "grasping-by-parts" strategy. This strategy also underlies the presented approach of MVBB decomposition. In this paper, we proposed MVBBs as an efficient and valuable box decomposition on a fit-and-split strategy. As the presented approach is hierarchical, it is also possible to use dependencies between boxes and granularities of different hierarchical levels for shape approximation. Thus, the processing of shape approximation can be controlled and run parallel to the execution of a grasp.

The trade-off of our approach is higher efficiency and simplicity for the price of precise shape approximation. However, we claim that exact approximation may not be necessary for grasping tasks. A wider evaluation of this claim will be one of our next steps. Our approach is therefore

grounded on box representation and decomposition with an efficient splitting criterion. The resulting box representation offers fast computational techniques for common problems, e.g. collision detection, neighborhood relations, etc., valuable for efficient further analysis. This analysis will become important for a next step towards grasping objects. Managing valid grasps will not only be dependent on the box faces, but also on the whole constellation of boxes.

Another issue in this context will be task dependency. A grasp might depend on different types of tasks, e.g. to pick up a cup and place it somewhere else might yield a different grasping action as to pick it up to show it or hand it over to someone. Such grasp semantics might be mapped to boxes in the set, e.g. "grasp the *biggest box* for a good grasp to stably move the object", "grasp the *smallest box* for a good grasp to show a most unoccluded object to a viewer / a camera" or "grasp the *outermost box* for a good grasp to hand over to another human / another robot", where the latter are said to be quite valuable for applications that are based upon interacting with objects before the exploration and recognition stage. Future work will focus on how the presented box representation provides a good and easy-to-use interface to such applications.

VI. ACKNOWLEDGMENTS

This work was supported by EU through the project PACO-PLUS, IST-FP6-IP-027657. The authors would like to thank Michael Wünstel (Universität Bremen) for providing the raw partial scan data and images used for Fig. 5(e)-(i).

REFERENCES

[1] 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, 2004.

[2] 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.

[3] K. Shimoga, "Robot Grasp Synthesis Algorithms: A Survey," *Int. Journal of Robotic Research*, vol. 15, no. 3, pp. 230–266, 1996.

[4] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic Grasp Planning Using Shape Primitives," in *IEEE Int. Conf. on Robotics and Automation*, 2003, pp. 1824–1829.

[5] 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.

[6] C. Borst, M. Fischer, and G. Hirzinger, "Grasp Planning: How to Choose a Suitable Task Wrench Space," in *Proceedings of the IEEE Int. Conf. on Robotics and Automation*, 2004, pp. 319–325.

[7] G. Biegelbauer and M. Vincze, "Efficient 3D Object Detection by Fitting Superquadrics to Range Image Data for Robot's Object Manipulation," *IEEE Int. Conf. on Robotics and Automation*, 2007.

[8] L. Chevalier, F. Jaillet, and A. Baskurt, "Segmentation and Superquadric Modeling of 3D Objects," *Journal of WSCG*, 2003.

[9] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof, "Grasp Planning Via Decomposition Trees," in *IEEE Int. Conf. on Robotics and Automation*, 2007.

[10] D. Katsoulas, "Reliable Recovery of Piled Box-like Objects via Parabolically Deformable Superquadrics," in *Proceedings of the 9th IEEE Int. Conf. on Computer Vision*, vol. 2, 2003, pp. 931–938.

[11] J. Tegin, S. Ekvall, D. Kragic, B. Iliev, and J. Wikander, "Experience based Learning and Control of Robotic Grasping," in *Workshop: Towards Cognitive Humanoid Robots*, 2006.

[12] D. Aarno, J. Sommerfeld, D. Kragic, N. Pugeault, S. Kalkan, F. Wörgötter, D. Kraft, and N. Krüger, "Early Reactive Grasping with Second Order 3D Feature Relations," in *ICRA Workshop: From Features to Actions*, 2007, pp. 319–325.

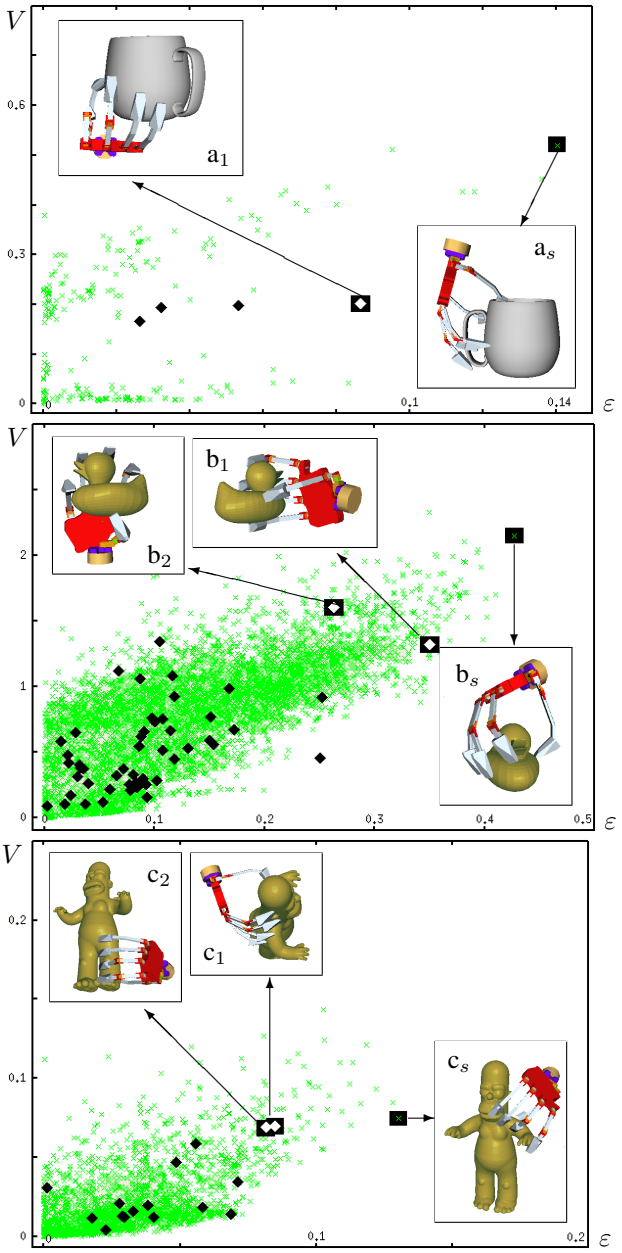


Fig. 6. Grasp quality spaces (ε,V) for models from Fig. 5(a)-(c). Note the depicted samples as a contrast of box grasps and random (spherical) grasps. Legend: × sample of the spherical grasp; ■ best ε-sample of the spherical grasp; ◆ sample of the box grasp; ◘ selected best sample of the box grasp.

[13] H. Zha, T. Hoshide, and T. Hasegawa, "A Recursive Fitting-and-Splitting Algorithm for 3-D Object Modeling Using Superquadrics," in *14th Int. Conf. on Pattern Recognition*, vol. 1, 1998, pp. 658–662.

[14] G. Barequet and S. Har-Peled, "Efficiently Approximating the Minimum-Volume Bounding Box of a Point Set in Three Dimensions," *Journal of Algorithms*, vol. 38, pp. 91–109, 2001.

[15] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: A Hierarchical Structure for Rapid Interference Detection," *Computer Graphics*, vol. 30, no. Annual Conf. Series, pp. 171–180, 1996.

[16] J.-M. Lien and N. M. Amato, "Approximate Convex Decomposition of Polyhedra," Texas A&M University, Tech. Rep. TR06-002, 2006.

[17] E. L. Damian, "Grasp Planning for Object Manipulation by an Autonomous Robot," Ph.D. dissertation, Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS, 2006.

[18] A. Morales, P. Azad, T. Asfour, D. Kraft, S. Knoop, R. Dillmann, A. Kargov, C. Pylatiuk, and S. Schulz, "An Anthropomorphic Grasping Approach for an Assistant Humanoid Robot," in *Int. Symposium on Robotics (ISR)*, 2006.