Empty the Basket - A Shape Based Learning Approach for Grasping Piles of Unknown Objects

David Fischinger and Markus Vincze

Abstract— **This paper presents a novel approach to emptying a basket filled with a pile of objects. Form, size, position, orientation and constellation of the objects are unknown. Additional challenges are to localize the basket and treat it as an obstacle, and to cope with incomplete point cloud data. There are three key contributions. First, we introduce Height Accumulated Features (HAF) which provide an efficient way of calculating grasp related feature values. The second contribution is an extensible machine learning system for binary classification of grasp hypotheses based on raw point cloud data. Finally, a practical heuristic for selection of the most robust grasp hypothesis is introduced. We evaluate our system in experiments where a robot was required to autonomously empty a basket with unknown objects on a pile. Despite the challenging scenarios, our system succeeded each time.**

I. INTRODUCTION

Grasping unknown objects is a fundamental necessity of robot-world interaction. Domestic robots now and in future times need the capability to manipulate novel objects in very cluttered scenes. To the best of our knowledge we are the first to present a fully automated system for emptying a basket by grasping novel objects from a pile using vision alone. One of the key issues in robotic grasping has been the lack of availability of complete 3D visual data. Despite limited 2.5D data (obtained from a single view in most setups), object attributes like transparency, reflection, absence of texture, material or color limit the quality of received data depending on the method of perception.

Given complete 3D models of all objects in a scenario, default grasp quality metrics like force or form closure ([1],[2],[3],[4]) and other grasp quality metrics ([5],[6]) have been implemented in simulation environments like OpenRAVE/OpenGRASP [7]/[8] or GraspIt [9] for detecting high quality grasps. There are several ways to handle these knowledge gaps and gain full 3D models. Miller et al. [10] model objects as sets of shape primitives, such as spheres, cylinders, cones or boxes. Varadarajan and Vincze [11] use the more general Super Quadric Fitting after object part segmentation. The approach of Bohg et al. [12] is based on the observation that many objects possess symmetries. They use a mirroring technique to complete point clouds and reconstruct a closed surface mesh. Rao et al. [13] also assume symmetry for object completion. Wohlkinger [14] proposes a system that covers automated model acquisition from the web for object recognition and object classification. The idea for grasping unknown objects is that if the underlying database is large enough, the nearest model (metric based on shape similarity of the visible part of an object) is sufficient for detecting a suitable grasp.

This work was conducted within the EU Cognitive Systems project GRASP (FP7-215821) funded by the European Commission.

D. Fischinger and M. Vincze are with Vision4Robotics Group, Automation and Control Institute, Vienna University of Technology, Austria
[df,vm] @ acin.tuwien.ac.at

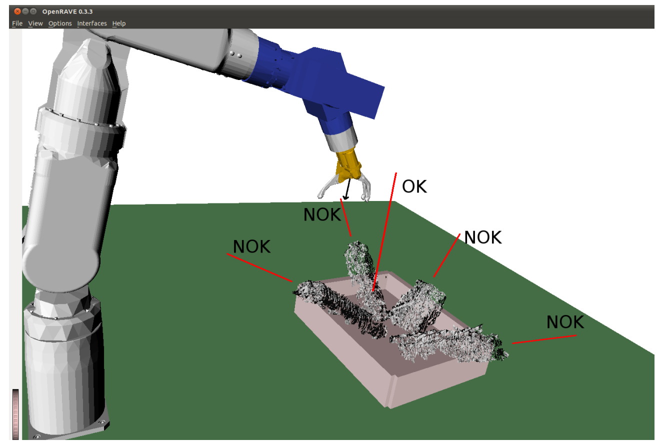


Fig. 1: Scene is showing 4 infeasible grasps due to the nonexistence of a kinematic solution. The basket is placed at an optimal position for top grasps regarding kinematic solutions for our robot arm.

In this paper we propose Height Accumulated Features (HAF) to abstract shape data from very cluttered scenes in a form that enables learning grasp hypotheses based on point cloud data. The framework introduced by us has a number of key benefits:

- *No Segmentation Necessary:* Segmentation is not only a very hard problem, but it is an unsolvable problem using vision alone. For instance, it is not possible to decide from visual perception if two adjacent boxes are glued together. Our approach has the full capability to solve the complex problem of emptying a basket with novel objects as standalone system. It can also be seen as complementary approach for methods which need segmented input e.g. for Superquadric fitting or as a preprocessing module for object recognition by separating one object from a heap of items.
- *Integrated Path Planning:* In the majority of recently published grasping approaches such as [11], grasp planning and path planning are done independently. Grasp points and grasp approach directions are calculated first, computation of inverse kinematic solutions and obstacle avoidance for path planning are done later.
  - Our machine learning approach is trained to only pick grasp hypotheses which results in collision free local paths for the manipulator used and the

given approach vector.
  - A second benefit is the possibility to restrict chosen grasp hypotheses by selection of an appropriate approach vector direction - for grasping out of a box, grasps from the side will be of no use.

  Due to the limited workspace for practical work with fixed robot arms (see Fig.1), these enhancements for pre-selection of grasps simplify the work of the inverse kinematic solver and improve the computing time for the final grasp configuration immensely.
- *Use Known Shape Data Only:* Although a complete and correct reconstruction of objects is clearly an advantage for calculation of grasps, the state-of-the-art algorithms do not work reliably enough for a complex scene like the one dealt with in this paper. Our approach concentrates on grasps on perceived surfaces where the manipulator can approach the object without the need to estimate the surface of the object facing away from the camera.

The next section discusses related work. Section III introduces the Height Accumulated Features and gives a detailed overview of our learning framework, methods for exploring the relevant grasp search space, and the heuristic for choosing the best grasp hypothesis out of many. In Section IV, we describe the working of the system modules, together with the hardware setup and present results. Finally, we give a short conclusion.

II. RELATED WORK

Among popular approaches, Klingbeil et al. [15] closely relate to the work presented in this paper. Klingbeil et al. propose an approach for grasp selection for a two finger end-effector to autonomously grasp unknown objects based on raw depth data. Their approach is based on finding the place where the manipulator shape fits best and does not require neither given object models nor a learning phase. Furthermore, they have integrated their grasping methods to create an autonomous checkout robot by reading barcodes. Saxena ([16],[17]) used supervised learning with local patch-based image and depth features for grasping novel objects in cluttered environments. Since this approach only identifies a single grasp point - that is, the 3D position (and 3D orientation), it is best suited for pinch-type grasps on items with thin edges. Jiang et al. [18] used an approach with many types of rectangular features for learning grasps, including one based on the comparison of depth values. Another approach based on SVM learning has been used by Pelossof et al. [19]. Berenson et al. [20] also considered grasping in cluttered scenes but with known 3D models. Calli et al. [21] propose a grasping algorithm which uses curvature information obtained from the silhouette of unknown objects. Pitzer et al. [22] showed that human interaction for deciding where to segment and hence where to grasp surprisingly enhances grasp performance for known objects in complex scenes.

III. APPROACH

Our approach is based on the observation that for grasping from top, parts of a manipulator have to enclose an object and hence go further down than the top height of the object. There must be space around the object where the manipulator fingers can be placed. Our idea is to define small regions and compare there average heights using discretized point cloud data. The height differences gives an abstraction of the objects shape that enables classifier training (supervised learning) if grasping would succeed for a given constellation. This classifier is used to explore the complete relevant grasp space. A weighting system evaluates all found grasps, preferring grasps which are centered in an area of potentially successful grasps.

Algorithm 1 shows the pseudo code of the empty the basket scenario. In the next subsection we are introducing the Height Accumulated Features (Algorithm 1: line 9) on which our learning module (III-B, Algorithm 1: line 10) is based. In the next subsection we present a heuristic for selecting one grasp hypothesis from many possible ones (Algorithm 1: line 11). Subsection III-D shows the working of our binary classifier for grasps with fixed hand roll and approach vector, thereby exploring the whole relevant grasp space (pseudo code Algorithm 1 lines 5-7). Finally, III-E describes methods for fine calculation of the grasps using the OpenRAVE simulation environment (Algorithm 1: line 17).

Algorithm 1 Pseudo Code for Empty the Basket

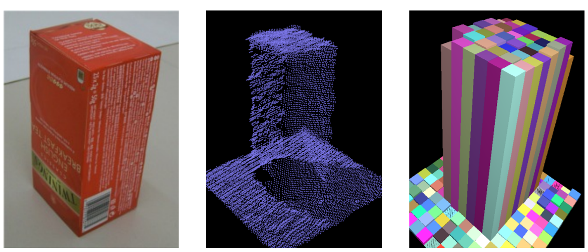
Require: Raw depth data of scene
Ensure: The basket gets emptied
1: $pcBox \leftarrow GetBoxAndObjectsPointCloudData()$
2: $pc \leftarrow DetectAndDeleteBox(pcBox)$
3: **while** basket not empty **do**
4:    $GH = \emptyset$ #GraspHypotheses
5:    **for** $\alpha_{tilt} = 0;\ \alpha_{tilt} < maxTilt;\ \alpha_{tilt}$ += $tiltStep$ **do**
6:       **for** $\beta_{roll} = 0;\ \beta_{roll} \leq 2\pi;\ \beta_{roll}$ += $rollStep$ **do**
7:          $heightGrid \leftarrow MakeHeightGrid(pc)$
8:          $accumHeightGrid \leftarrow Accum(heightGrid)$
9:          $HAF \leftarrow CalcHAF(accumHeightGrid)$
10:          $graspGrid \leftarrow SVMClassifier(HAF)$
11:          $GH \leftarrow AppendTopGrasps(GH, graspGrid)$
12:       **end for**
13:    **end for**
14:    $success \leftarrow false$
15:    **while** success == false **and** GH $\neq \emptyset$ **do**
16:       $topGH \leftarrow GetAndRemoveTopGH(GH))$
17:       $topGrasp \leftarrow FineCalculation(topGH)$
18:       $success \leftarrow TryExecuteTopGrasp(topGrasp)$
19:    **end while**
20:    $pcBox \leftarrow GetBoxAndObjectsPointCloudData()$
21:    $pc \leftarrow DetectAndDeleteBox(pcBox)$
22: **end while**

A. Height Accumulated Features

Inspired by Haar-like features from [23], we developed Height Accumulated Features fitting for grasp manipulation. Here, we illustrate our idea on a simple example of a box shaped object (Fig. 3(a)). Fig. 3(b) shows the extracted

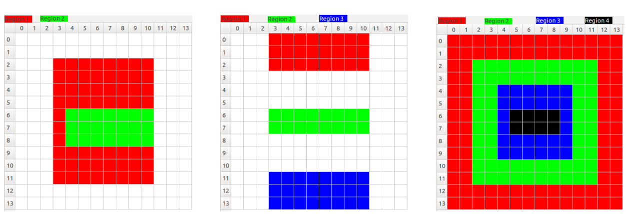
Fig. 2: Even for such scenarios the basket orientation and pose detection has to work robustly. Beside that, the basket as additional circumfluent obstacle reduces the number of possible grasp points and approach directions and is responsible for occlusions of lower object parts near the border.

point cloud of the box. For our approach we now calculate a height grid H of a 14cm x 14cm scene part which is appropriate for the used manipulator. For different sized manipulators, the point cloud would be scaled accordingly to the manipulator size (if needed x- and y-axes independently) as a preprocessing step, giving the opportunity to use smaller or bigger manipulators with only minor changes of the system. Every cell in Fig. 3(c) shows the highest z-value (height from base of the table top surface) of the discretized point cloud displayed as height bars.



(a) Tea Box (b) Point Cloud (c) Height Grid H

Fig. 3: Preprocessing steps for HAF



(a) 2 regions (b) 3 regions (c) 4 regions

Fig. 4: Examples of HA-Features with 2 overlapping, 3 disjunct and 4 overlapping regions

Height Accumulated Features are defined in a fashion similar to Haar Basis functions from [24]. Fig. 4(a) shows an example of one HAF feature. We have defined two overlapping rectangular regions R_1 (red+green cells) and R_2 (green) on the height grid H. All height grid values of each region R_i are summed up. These region sums r_i are individually weighted by w_i and then summed up. Regions R_i and weights w_i are dependent from the HA-Feature. So the jth HA-Feature value f_j is calculated by

$$f_j = \sum_{i=1}^{nrRegions_j} w_{i,j} \cdot r_{i,j} \tag{1}$$

with

$$r_{i,j} = \sum_{k,l \in \mathbb{N}: H(k,l) \in R_{i,j}} H(k,l) \tag{2}$$

One HA-Feature is defined by 2-4 rectangular regions (and region weights), used in (1) to calculate one HA-Feature value. The HA-Feature vector **f** is the sequence of HAF values:

$$\mathbf{f} = (f_1, f_2, \ldots, f_{nrFeatures}) \tag{3}$$

For the choosen example with two overlapping regions the weights are choosen such that an intuitive interpretation of one feature value f_j is possible and as follows. f_j indicates if the center of the height grid is a good position for a grasp with approach vector perpendicular to the grid plane:

$$\begin{aligned} &f_j < 0 \text{ green region in average higher (good)} \\ &f_j > 0 \text{ red region in average higher} \\ &f_j = 0 \text{ average region height is equal} \end{aligned} \tag{4}$$

35,000 features were randomly selected from about 350,000 automatically generated features using 2 rectangular regions with the only restrictions that the first region includes the center of the grid and the smaller second region is completely inside the first region. For these 35,000 features the weighting is done s.t. (4) holds. We added 500 manually created features with 2-4 rectangular regions. Examples for HA-Features with three and four regions can be seen in Fig. 4(b) and Fig. 4(c). For the latter 500 features weighting was perturbed to diversify the features and enhance the shape descriptive power of our system. Using F-score feature selection technique with SVM learning from [25], the 300 most characteristic features were chosen for the classification task.

Of significant importance to our system is the representation of height grids. To expedite computation, we use accumulated height values for the given scene. This principle was first introduced as summed area tables in [26] for texture mapping in computer graphics and was very successfully renamed to integral images by [23] in the vision community.

Instead of an initial height grid H we calculate an accumulated height grid AH once (the terminology emphasizes the suitability of the feature for height related representation) where each location x,y of AH contains the height sum above and to the left of x,y in the grid.

$$AH(x,y) = \sum_{x' \le x, y' \le y} H(x',y') \tag{5}$$

Using height accumulated rectangular regions, each region sum can be computed with four or less array references, see Fig. 5.

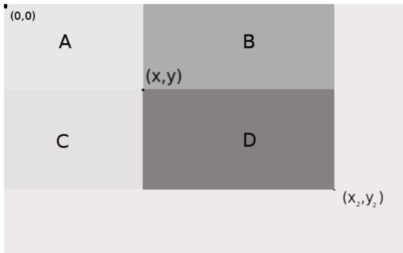


Fig. 5: To calculate the accumulated heights of region A a single AH reference is needed: AH(A) = AH(x,y), Area D requires four: $AH(D) = AH(x_2, y_2) - AH(x_2, y) - AH(x, y_2) + AH(x, y)$

B. Learning Approach – SVM

For learning good grasps Support Vector Machines with radial basis function (RBF) kernel were used based on the implementation described by [27]. The SVM was trained to classify grasp points using the HAF vectors. Fig. 6 illustrates

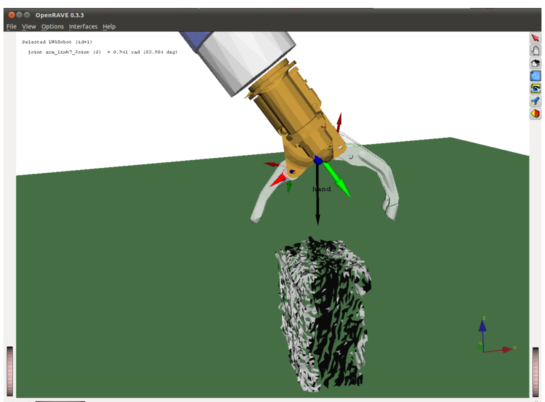


Fig. 6: For grasp classification it is learned (supervised learning) if a hand movement in approach direction (black arrow) with subsequent closing of the fingers would lead to a stable grasp

the actual classification task. For a manipulator with finger tips aligned to the world x-axis (i.e. for opened hand the line between thumb tip and forefinger tip is parallel to the x-axis) and given hand approach vector (black arrow in picture) parallel to the vertical world coordinate z-axis, a manual evaluation by a human observer is done if the hand movement along the approach vector with subsequent closing of fingers would lead to a successful grasp (which we then refer as 'good grasp constellation'). For training purposes we manually gathered point cloud data for 450 scenes representing good grasp constellations and 250 for bad ones. Using techniques like mirroring about x-,y- xy-axis, cutting point cloud heights or inverting heights for getting negative examples from positive ones, 8300 positive and 12800 negative training examples were generated. The enormous descriptive power of HAF is evident from the high success rates on a series of test data sets: 99.2%-100%.

For the test scene with a basket B, we decide for each integer coordinate pair (x, y) inside the basket if this x,y-position would fit for the center of a top grasp with predefined roll angle of manipulator. A typical grasp classification outcome for the box example (Fig. 3(a)) is shown in Fig. 7 where the green area on the left side symbolizes good grasp positions.

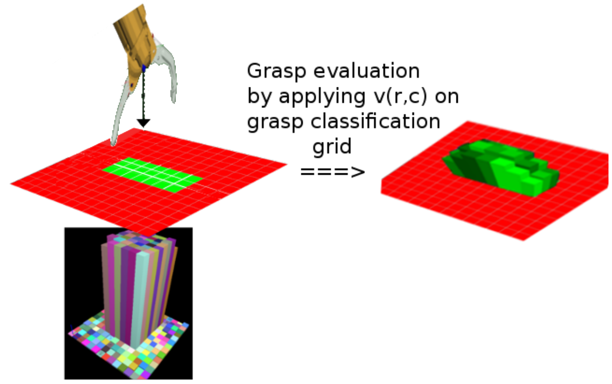


Fig. 7: Grasping Classification Grid GCG(center,left): green identifies spots where a grasp would be successful. The height of the green bars on the right, symbolizes the evaluation value received by applying the weight function from (6).

C. Heuristic for Grasp Selection

For selecting the most promising grasp point we developed a simple heuristic. For simple grasp scenarios like the box scenario but also for far more complex constellations the grasp classifier from III-B often delivers not only single grasp points, but whole areas where grasping would be successful (see green area on the left in Fig. 7). Obviously in most cases it is advantageous to take a grasp point which is centered in an area where grasping would be successful, e.g. for a more balanced weight distribution and hence a more stable grasp. We developed a weighting system for selecting grasp points out of many possible which is highly valuable for practice. For each identified grasp in our grasp classification grid GCG we weight the location considering adjacent grasps by the following evaluation function

$$v(r, c) = \sum_{x,y \in \mathbb{N}} I_{grasp}(x, y) \cdot w_{r,c}(x, y) \tag{6}$$

where r, c are indicating the actual row and column of the grasp location. I is the indicator function for a grasp point:

$$I_{grasp}(x, y) = \begin{cases} 1 \text{ if grasp at location (x,y)} \\ 0 \text{ if no grasp at location (x,y)} \end{cases}$$

and $w_{r,c}$ is the weighting function with values displayed in Table I with respect to one grasp hypothesis GH.

		1	2	3	2	1		
		2	3	4	3	2		
1	1	3	4	GH	4	3	1	1
		2	3	4	3	2		
		1	2	3	2	1		

TABLE I: Weighting values for evaluation of grasp hypothesis GH

D. Exploring Grasp Space

By now we have a classifier for top grasps with one hand orientation (hand roll). Our technique to explore the whole relevant grasp space is as follows.

1) Roll: To get grasps for different hand rolls β, i.e. different angles for manipulator rotations about the manipulators approach direction, we rotate the initial point cloud iteratively (by $rollStep$ = 15 degrees) about the vertical z-axis up to 180 degrees, make a new accumulated height grid and start the HAF based grasp point detection on this data. After selection of the top grasp points for the rotated scene, grasp points where transformed to the original world coordinate system. By use of a roll angle range $[\beta - rollStep/2, \beta + rollStep/2]$ in the simulation environment and testing with manipulator rotation β and $\beta + 180°$ simultaneously, we achieve a sound exploration of all rolls.

2) Tilt: In order to widen the domain of grasps from grasps with vertical approach direction to grasps with tilted approach direction, we transform the point cloud analog the roll calculation with $tiltStep = 20°$. After detection of good grasp points on this data, the transform of grasp points and tilted approach vectors is inverted and we get even grasps from side direction. Due to the camera view and the basket as grasping obstacle we prefer grasps with vertical approach direction. Additionally for these grasps the search for inverse kinematic solutions do not fail so often. Kinematic solutions often do not exist for grasps with defined approach vectors and fixed roll, see Fig. 1.

E. Getting actual Grasp Points using OpenRAVE

Using the heuristic from Section III-C we take the best evaluated grasp hypothesis from all roll-tilt combinations and use OpenRAVE for grasp and path planning. OpenRAVE tries to approach the object mesh using the calculated approach vector and manipulator roll angle until a collision occurs. Than it sets back the manipulator by a standoff value which is dependent from the object position: if a standoff value of $1mm$ leads to a collision of the gripper fingers with the basket, the standoff is increased until the closing fingers do not collide with the basket anymore. Then the actual grasp points, i.e. contact points of fingers with object, are calculated. From the resulting hand position OpenRAVE calculates the manipulator position 7cm away and searches for a collision free path to place the manipulator there. For the last 7cm to the object OpenRAVE calculates a straight path to the object if one exists. To make the system more flexible the calculated approach vector and manipulator roll angle are varied, currently by ± 10 resp. ± 7.5 degrees, to overcome unsolvable situations like the non existence of a kinematic solution.

IV. EXPERIMENTS

We demonstrate the capability of our approach by grasping objects from a basket. Tests are done for ten different scenarios (Fig. 8(a)-8(e)). To the best knowledge of the authors we are the first to present an automatic system for grasping from piles of unknown objects in a basket.

A. Test Setup

For grasp execution we use a Schunk 7-DOF robot arm with Otto Bock hand prosthesis SensorHand Speed. Perception is done using a Microsoft Kinect camera with a PrimeSense sensor. Optionally a second camera can be used. Our approach is aimed to work with input data from one top view only. Since more data from a second camera is useful for path planning, a second camera was used for testing purposes.

ROS (Robot Operating System, www.ros.org) is used for module communication. Point cloud manipulations are done with PCL (Point Cloud Library, www.pointclouds.org).

B. Results

Table II shows the results for the ten trials to emptying the basket. In all cases the basket was successfully emptied without any user interaction after placing the basket and starting the system. Five out of ten times the pile was resolved without a single grasp failure. In the current implementation 2-3 seconds are needed for grasp calculation and about one second for grasp and path planning with OpenRAVE.

Fig. 9: Objects used for test runs

Run	Objects Removed	Basket Emptied	Grasp Failures
1	7/7	yes	0
2	7/7	yes	9
3	7/7	yes	0
4	7/7	yes	2
5	7/7	yes	0
6	7/7	yes	0
7	7/7	yes	2
8	7/7	yes	0
9	7/7	yes	7
10	7/7	yes	2
Sum	**70/70**	**10/10**	**22**

TABLE II: Empty the basket results for all trials

Table III gives a detailed overview of grasp failures per trial and object. Regarding only first tries to grasp an object our approach succeeded in 61 out of 70 cases.

(a) Test Run 1 (b) Test Run 7 (c) Test Run 8 (d) Test Run 9 (e) Test Run 10

Fig. 8: Examples of test scenarios for empty the basket

Obj\ Run	1	2	3	4	5	6	7	8	9	10	Sum
Ball	-	-	0	0	0	0	-	-	5	0	5/11
Bowl	0	-	-	-	-	-	-	-	0	-	0/2
Car	-	-	-	-	-	-	-	-	0	1	1/3
Cereal	0	-	0	0	0	-	-	-	-	-	0/4
Cube	-	0	-	-	-	-	-	-	-	-	0/1
CubeFoam	-	0	-	-	-	-	-	0	-	-	0/2
Cuboid	0	0	0	0	0	0	2	0	1	-	3/12
CuboidF.	-	-	-	-	-	-	0	-	-	-	0/1
CylinderF.	-	-	-	-	-	-	0	0	-	0	0/3
EdgeF.	-	-	-	-	-	-	0	0	-	-	0/2
Elephant	0	2	0	0	0	0	0	0	1	0	3/13
Milk	-	7	-	2	-	-	-	-	-	-	9/11
Pig	-	-	-	-	-	0	-	-	0	0	0/3
Plasticine	0	0	0	0	0	0	-	-	-	1	1/8
SoftPads	0	-	0	0	0	0	0	0	-	-	0/7
DrinkBox	0	-	-	-	0	-	-	-	-	-	0/2
Whey	-	0	0	-	-	0	0	0	0	0	0/7
Sum	**0**	**9**	**0**	**2**	**0**	**0**	**2**	**0**	**7**	**2**	**22/92**

TABLE III: Grasp failures per object for 10 trials. F. = Foam. Entry of last column is number of failures divided by number of tries.

Table IV shows a grasp error analysis. Three main issues were identified causing grasp failures.

- Insufficient point cloud data leads to bad grasp hypotheses: on the milk package the brand name was covered by black tape for TV recording. The sensor delivered no data for the taped area. Taking into account the object position adjacent to another object, the resulting grasp points looked reasonable in the simulation environment, but failed in the real world seven times in a row. (Fig. 10). Each time the milk package was moved a bit, until the hard constellation was cleared and the milk package separated a bit from the adjacent object. Stable grasp points could than be found despite the misleading input data.
- Path planning was two times the reason for grasp failures. In both cases the arm stopped in simulation before it reached the expected position. This problem is related to finding kinematic solutions and needs further investigation.
- The HAF learning approach identified grasp points that lead to unstable grasps. Consequently, objects slipped out of the manipulator's fingers. This happens if objects have no obvious grasp points, e.g., when object top surfaces have the same height and touch each other. From all 22 failed grasps calculated there was none where already in simulation it was obvious for human observer, that the grasp can't succeed. However, in all cases the grasp trial touches at least one object, resulting in perturbing these constellations and thus creating a situation such that autonomous emptying proceeds.

Note that out of 22 failed grasps, 18 happened when only 1-2 objects where left in the basket. Particularly two hard constellations with seven and four grasp failures in a row were responsible for that, but it shows that our weighting system is capable to decide which are simple grasps to execute first. It also shows that the basket brings a complication which should not be underestimated, since most of these 18 grasp failures were related to objects adjacent to the basket border. It also reveals potentials for enhancements of the HAF learning system. To avoid obstacles (i.e. basket boarders or currently not grasped objects), the system chooses grasp points near the edges of an object, which can result in objects slipping out of the manipulator's fingers.

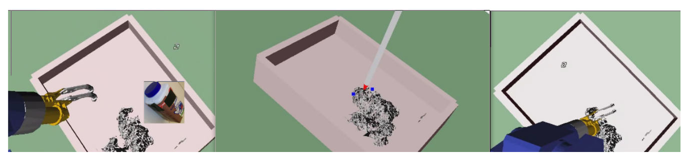


Fig. 10: Unstable grasp points due to insufficient data
1) misleading hole in data mesh 2) grasp points and approach direction 3) grasp execution in simulation

Run	Object	Failures	Data	PP	HAF
2	Milk	7	x		
9	Ball	4	x		
2	Elephant	2			x
4	Milk	2	x		x
7	Cuboid	2	x		x
9	Ball	1			x
9	Cuboid	1			x
9	Elephant	1			x
10	Car	1		x	
10	Plasticine	1		x	x
Sum		**22**	**15**	**2**	**10**

TABLE IV: Analysis for grasp failures per object and test run. Failures are caused by insufficient point cloud data (Data), wrong path planning (PP) or unstable grasp points (HAF)

V. CONCLUSION

In this paper, we proposed Height Accumulated Features, a new method to abstract shape information from cluttered scenes to enable learning of grasp hypotheses. We explained a method to explore the relevant grasp space using a trained SVM classifier. A practical weighting system enhances the robustness of calculated grasps. The reliability and robustness of our system was shown by a test series, emptying a basket with a pile of unknown objects fully autonomous. The complexity of tested scenes is beyond all comparable test cases the authors are aware of. Videos at www.youtube.com/user/emptythebasket demonstrate our results on evidently arbitrary piles with even more than 7 objects uncut and a higher success rate as in Section IV-B.

Beside a highly efficient way of calculating HA-Features and the avoidance of segmentation, our system has the benefit to be extensible. The use of other features additional to HAF can improve results, e.g. by taking into account detected (side) faces to maximize grasp stability or profit from the additional data the second camera view delivers.

References

[1] M. T. Mason and J. K. Salisbury Jr., *Robot Hands and the Mechanics of Manipulation*. The MIT Press, 1985.

[2] Z. Li and S. S. Sastry, "Task-oriented optimal grasping by multifingered robot hands," *IEEE Journal on Robotics and Automation*, vol. 4, no. 1, pp. 32–44, 1988.

[3] A. Bicchi and V. Kumar, "Robotic grasping and contact: a review," *Proceedings 2000 ICRA Millennium Conference IEEE International Conference on Robotics and Automation Symposia Proceedings*, vol. 10, no. 4, pp. 348–353, 2000.

[4] N. S. Pollard, "Closure and Quality Equivalence for Efficient Synthesis of Grasps from Examples," *The International Journal of Robotics Research*, vol. 23, no. 6, pp. 595–613, 2004.

[5] K. Hsiao, L. P. Kaelbling, and T. Lozano-Perez, "Grasping POMDPs," *ICRA*, pp. 4685–4692, 2007.

[6] M. Ciocarlie, C. Goldfeder, and P. Allen, "Dimensionality reduction for hand-independent dexterous robotic grasping," *2007 IEEERSJ International Conference on Intelligent Robots and Systems*, vol. on29, pp. 3270–3275, 2007.

[7] R. Diankov and J. Kuffner, "OpenRAVE : A Planning Architecture for Autonomous Robotics," *Tech. Rep. CMU-RI-TR-08-34, Robotics Institute*, no. July, 2008.

[8] B. Leon, S. Ulbrich, R. Diankov, G. Puche, M. Przybylski, A. Morales, T. Asfour, S. Moisio, J. Bohg, and J. Kuffner, "OpenGRASP: A Toolkit for Robot Grasping Simulation," in *Simulation Modeling and Programming for Autonomous Robots Second International Conference SIMPAR 2010*, ser. Lecture Notes in Computer Science, N. Ando, S. Balakirsky, T. Hemker, M. Reggiani, and O. V. Stryk, Eds., vol. 6472 LNAI. Springer, 2010, pp. 109–120.

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

[10] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," *ICRA*, vol. 2, pp. 1824–1829, 2003.

[11] K. M. Varadarajan and M. Vincze, "Object Part Segmentation and Classification in Range Images for Grasping," in *International Conference On Advanced Robotics ICAR 2011*, 2011, pp. 21–27.

[12] J. Bohg, M. Johnson-roberson, B. Le, J. Felip, X. Gratal, N. Bergstr, D. Kragic, and A. Morales, "Mind the Gap - Robotic Grasping under Incomplete Observation," *ICRA*, pp. 686–693, 2011.

[13] D. Rao, Q. V. Le, T. Phoka, M. Quigley, A. Sudsang, and A. Y. Ng, "Grasping Novel Objects with Depth Segmentation," *Image Rochester NY*, pp. 2578–2585, 2010.

[14] W. Wohlkinger and M. Vincze, "3D Object Classification for Mobile Robots in Home-Environments Using Web-Data," *International Workshop on Robotics in AlpeAdriaDanube Region RAAD*, pp. 247–252, 2010.

[15] E. Klingbeil, D. Rao, B. Carpenter, V. Ganapathi, A. Y. Ng, and O. Khatib, "Grasping with Application to an Autonomous Checkout Robot," *ICRA*, pp. 2837–2844, 2011.

[16] A. Saxena, L. L. S. Wong, and A. Y. Ng, "Learning Grasp Strategies with Partial Shape Information," *AAAI*, vol. 3, no. 2, pp. 1491–1494, 2008.

[17] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic Grasping of Novel Objects using Vision," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2008.

[18] Y. Jiang, S. Moseson, and A. Saxena, "Efficient Grasping from RGBD Images : Learning using a new Rectangle Representation," *ICRA*, pp. 3304–3311, 2011.

[19] R. Pelossof, A. Miller, P. Allen, and T. Jebara, "An SVM learning approach to robotic grasping," *IEEE International Conference on Robotics and Automation 2004 Proceedings ICRA 04 2004*, vol. 4, no. April, pp. 3512–3518, 2004.

[20] D. Berenson, R. Diankov, and J. Kuffner, "Grasp planning in complex scenes," *2007 7th IEEERAS International Conference on Humanoid Robots*, pp. 42–48, 2007.

[21] B. Calli, M. Wisse, and P. Jonker, "Grasping of unknown objects via curvature maximization using active vision," *IEEERSJ International Conference on Intelligent Robots and Systems*, pp. 995–1001, 2011.

[22] B. Pitzer, M. Styer, C. Bersch, C. DuHadway, and J. Becker, "Towards perceptual shared autonomy for robotic mobile manipulation," in *2011 IEEE International Conference on Robotics and Automation*, Robert Bosch LLC at the Research and Technology Center North America, Palo Alto, CA 94304, USA. IEEE, 2011, pp. 6245–6251.

[23] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[24] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," *International Conference on Computer Vision*, vol. 6, no. January, pp. 555–562, 1998.

[25] Y.-w. Chen and C.-j. Lin, "Combining SVMs with Various Feature Selection Strategies," *Strategies*, vol. 324, no. 1, pp. 1–10, 2006.

[26] F. C. Crow, "Summed-area tables for texture mapping," in *Proceedings of SIGGRAPH*, 1984, pp. 18(3):207–212.

[27] C.-c. Chang and C.-j. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1—-27:27, 2011.