The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

# Grasping Novel Objects with Depth Segmentation

Deepak Rao, Quoc V. Le, Thanathorn Phoka, Morgan Quigley, Attawith Sudsang and Andrew Y. Ng

*Abstract*— **We consider the task of grasping novel objects and cleaning fairly cluttered tables with many novel objects. Recent successful approaches employ machine learning algorithms to identify points on the scene that the robot should grasp. In this paper, we show that the task can be significantly simplified by using segmentation, especially with depth information. A supervised localization method is employed to select graspable segments. We also propose a shape completion and grasp planner method which takes partial 3D information and plans the most stable grasping strategy. Extensive experiments on our robot demonstrate the effectiveness of our approach.**

## I. Introduction

We consider the task of robots cleaning a desk by grasping objects. Many challenges are posed by this task: the variation in shapes and orientations of new objects, lack of complete 3D information. An example of such scenarios is shown in Figure 1.

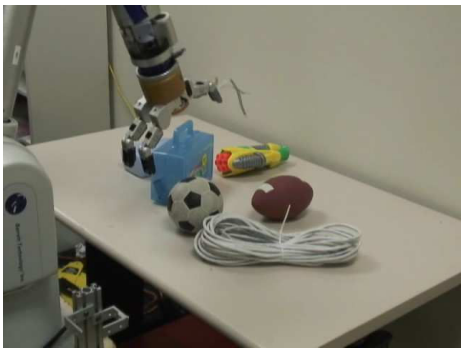


Fig. 1. An example scene which our STAIR2 robot tries to clean.

Recent successful methods for robotic grasping cast the problem as a machine learning task. Specifically, [1], [2], [3] show that grasping contacts can be learned and generalized among objects. The methods they developed, thus can be applied to grasp many novel objects. Most of these algorithms use local features and segmentation is avoided because color segmentation does not usually yield good results for a cluttered scene.

Cleaning a table by grasping, however, remains a very difficult task. In our experiments, grasping most of the objects is challenging without the knowledge of their shapes. For many objects, grasping at the centroid is most stable and having partial shape information makes the task very difficult and is one of the main reasons that leads to failures in grasping.

In this paper, we consider *segmentation* as a way to achieve better grasping and address the aforementioned failures. Unlike previous methods which consider only visible light images, we take advantage of depth cues along with visible light images to achieve better segmentation. With depth data, we can solve problems that are very difficult to solve by visible light images such as indistinguishable background, shadows etc. For this to work well in the case of grasping, we use an active triangulation sensor that gives very accurate and detailed depth data [4].

Finally, given all the segments for a scene, the robot has to distinguish between *graspable* and *ungraspable* segments. For this reason, we employ a supervised learning method to classify between whether a given segment can be grasped or not. For example, segments that are very large such as the walls or the tables cannot be grasped by the robot. Our approach is based on an intuition that ungraspable segments ("artifacts") usually have some common patterns such as being too flat on the tables or being too small, thin or large. We can therefore use such patterns to discard these bad segments.

Having good segmentation information can be very useful for grasping. To illustrate this fact, we consider a simple grasping method that perhaps makes best use of the segmentation algorithm. It estimates the 3D shape of the objects and samples good stable *antipodal* grasps from that.

The five steps of our method are shown in Figure 2. Note that the goal of step one ("Complete missing 3D information") is to fill in missing depth readings due to occlusions and bad readings. In this step, we employ the Gauss Siedel algorithm to smooth the 3D data and fill in missing values. The details of all the steps will be described in following sections.

Extensive experimental results show that the method developed in this paper achieves better grasping results for novel objects as well as cleaning a desk compared to previous work of our group. More specifically, the method developed in this paper enables the STAIR2 robot to achieve higher accuracy compared to competitive methods. Further, the robot also makes less failures for cleaning-a-desk scenarios.

## II. Related work

Prior work in robot grasping assumed complete 2D or 3D models of objects. Under such assumptions, many types of grasps can be modeled and computed, for example force closure [5], [6], [7], form-closure [8], equilibrium grasps [9],

Deepak Rao, Quoc V. Le, Morgan Quigley and Andrew Y. Ng are with the Computer Science Department, Stanford University. Thanathorn Phoka and Attawith Sudsang are with Department of Computer Engineering, Chulalongkorn University, Thailand. drao@cs.stanford.edu, quocle@cs.stanford.edu, thanathorn.p@gmail.com, mquigley@cs.stanford.edu, attawith@gmail.com, ang@cs.stanford.edu

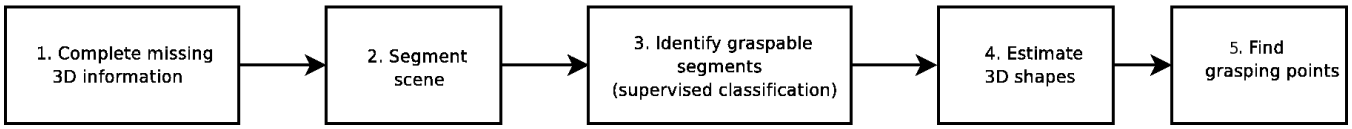



Fig. 2. The pipeline for our method.

[10], [11], stable grasps [12], compliant grasps [10]. More recent approaches use machine learning to combine more information for better grasping using SVMs [13] or for better controlling using reinforcement learning [14], [15].

The main drawback of these methods is that it is hard to extend them to real-world data where capturing complete 3D models for objects is difficult. For example, given a static scene, the back face of objects cannot be captured using a stereo camera. This realization leads further developments in robotic grasping where methods have to consider more realistic sensory data, for example, intensity images, point clouds, haptic feedbacks. With such sensory data, researchers have to take into account sensory noises and partial shapes.

Using local visual features, methods are proposed to find *planar* grasps, i.e., looking for 2D locations where the robot can place its fingertips [16], [17], [18], [19]. For non-planar grasps, a schema structured method is presented to deal with simple objects [20]. Also with schema structured learning, Platt et al. [21] proposes a method that assumes segmentation and fits ellipsoids to objects. Edsinger and Kemp [22] designed an algorithm to grasp cylindrical objects with power grasp using visual servoing.

Learning to grasp novel objects has also received special attention in recent years [23] [2] [24] . The key idea is that there are certain common 2D and 3D cues that humans consider when grasping objects. These cues are so robust such that they can be generalized for grasping objects that we have never seen before. Saxena et al. [2], for example, apply this idea to the robotic context and employ machine learning algorithms to figure out the cues which enable robots to grasp new objects. Further, Le et al. [3] extends the above methods and use 3D information that adapts well to the configuration of the robot hand.

The above approaches do not consider shape information of the objects. This appears to be advantageous because finding the shape information of the objects in a scene is a very challenging computer vision task. In fact, segmentation is still an active area in computer vision despite many years of effort. However, as we observe in many experiments, having shape information is essential in order to have a high success rate for grasping. In this paper, we will show that segmentation can be significantly simplified if we use an accurate 3D sensor and that grasping can be made remarkably easy once segmentation is achieved.

Segmentation with depth data is a relatively old idea. One of the most well known method for range segmentation is described in [25]. Specifically, Trucco and Verri [25] illustrate a surface extraction method called HK segmentation that fits parametric shapes to range data. Parametric shape fitting and segmentation is also the focus of [26]. Most of the methods in the previous cases are used in the context of robotic navigation [27], [28] where the required accuracy is low.

Robotic manipulation, especially grasping, with segmentation is a focus of [29], [30]. Specifically, the authors consider common cases where objects can be segmented easily because they are in isolation on a flat surface such as floors, desks, tables. Once segmentation is achieved, grasping can be made trivial even with very simple rules. Our work builds on this insight and extends further to the case where objects, their shadows and background cannot be easily segmented using only intensity images.

Likewise, visible-light image segmentation for grasping has been considered in the work of [21]. Specifically, the authors consider fitting parametric shapes to objects and then apply learned decision rules for particular shapes. The method we developed in this paper, as mentioned above, considers segmentation with depth and hence achieve better results. Furthermore, we also employ a supervised learning algorithm that identifies ”bad” segments and discards them. As a result, our method can be applied successfully to more sophisticated scenarios than the ones considered by [21].

## III. Filling Missing Depth Data

Our system makes use of both depth and visible light image data. To capture depth data, we used an active triangulation sensor [4] (see Figure 3). An important feature of this sensor is that it gives very detailed depth data (also called depth map or point cloud).

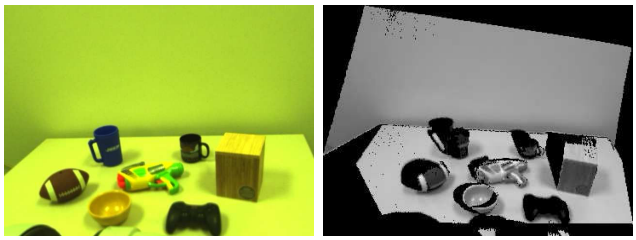


Fig. 3. Image and depth data captured by our robot.(Note: Missing back face of objects

Unlike visible light images, depth data usually contains mis-readings [4]. Mis-readings occur if the camera cannot see the laser beam because of occlusions or variation in lighting conditions. For segmentation and grasping to work, we employ a simple algorithm to fill in missing values for depth readings. Here, we describe the algorithm in full detail.

Specifically, we apply the successive overrelaxation method (SOR) to iteratively extrapolate the 3D data of the missing pixels. This iterative extrapolation method alternates

between two steps. The first step computes values of the missing pixels given their neighbourhood while the second step computes values of the missing pixels given their previous values. The method terminates when the values of the missing pixels do not change. In this work, initial values for missing pixels are set to zero.

## IV. Segmentation with depth data

The problem of image segmentation remains a great challenge for computer vision, having received continuous attention since the birth of the field. There have been a large number of fairly successful methods for segmenting visible image data. In this paper, we consider a graph-based segmentation algorithm described in [31] which our method will be based on.

In detail, Felzenszwalb and Huttenlocher [31] design a graph based representation of an image and find the evidence of a boundary between regions based on $L_2$-Norm between the intensity of pixels.

$$||I(p_i) - I(p_j)||_2 > \tau$$

where $I(p_i) \in R^3$ is the intensity of the pixel $p_i$ and $\tau$ is the threshold function.

If the $L_2$ Norm of the intensities is greater than the threshold function then the pixels are considered to be in different regions.

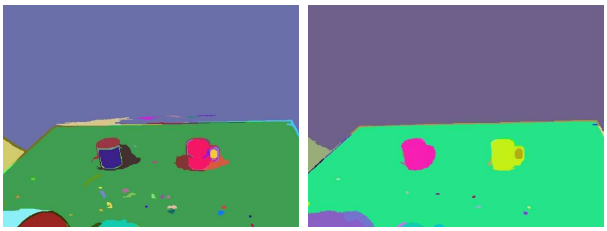
Despite its robustness, the method has some limitations. Since the algorithm is based entirely on pixel intensities it works poorly in scenes having shadows or more than one light source. This is because segmenting an image on the basis of color information is not an ideal approach. An advantage of depth data allows us to extend the approach in [31]. We follow the same scheme of representing an image as a graph but updated the metric for finding boundaries between regions to include the depth data.

$$||W^T * (F(p_i) - F(p_j))||_2 > \tau$$

where $F(p_i) \in R^4$ is the intensity of the pixel $p_i$ having an extra dimension of depth value corresponding to that location in 3D space, W $\in R^4$ is the weight vector assigning weights to different elements of F and $\tau$ is the threshold function.

The intuition behind including weights in the equation was to rank the elements in order of their importance. In our experiments, we observe that setting greater weight to the depth value than color intensities often gives much better results than setting equal weights to all of them. We found that setting weights to $0.1$ for the color channels and $0.7$ for the depth channel gives the best results.

Figure 4 shows a comparison between the two segmentation approaches – segmentation without and with depth information. It can be seen in Figure 4(a) that although the method proposed in [31] works well to segment the scene, it is not robust to shadows and tiny scratches on the table. This creates a large number of artifacts on the table and the wall which cannot be grasped by the robot. In contrast, segmentation with depth gives much better segments.



(a) Segmentation without depth (b) Segmentation with depth

Fig. 4. Comparison between the two approaches: segmentation without and with depth.

## V. Localizing Graspable Segments

Given all segments for a particular scene, obtained by the previous step, the robot has to distinguish between which segment can be grasped. For example, if we run the segmentation algorithm, we may find large segments like the walls, tables, floors which are very large and cannot be grasped by the robot hand. Likewise, the segmentation algorithm, even with accurate depth data, can produce some small segments such as textures on the table.

We employ a supervised learning classifier to localize all graspable segments from the previous unsupervised segmentation stage. The basic intuition here is that ungraspable segments have common simple patterns compared to graspable segments. The following features are used in our classifier:

*a) Geometric 2D features:* These features are computed from the image plane. For every segment, we compute the width, height and area of the segment from the visible image data. The intuition behind using these features is that graspable objects have to be close to the robot and thus can never appear very small or very large.

*b) Color image features:* These features are computed from the color information provided by the visible image. We first convert RGB image to LAB image which is more insensitive with lighting conditions. We then compute the variance in L, A and B channels of the new LAB image.

*c) Geometric 3D features:* These features are computed from the point cloud data. They are the variance in depth, variance in height and range of height. Compared to the geometric features from the image plane, geometric features from 3D point cloud are more sensitive to the coordinate axes. To address this issue, we use the robot frame as the main coordinate axes. In this frame, the XY plane is the flat horizontal plane which corresponds to table or desk surfaces. Computations for height, depth etc in this frame are therefore considerably simplified.

The intuition for above features is that ungraspable segments tend to be either very small or very large and tend to have lesser variations in color composition as compared to graspable segments or objects. We further illustrate the discriminative power of the above features in Figure 6.

To train the classifier, we collected 200 scenes of 6 objects (shown in Figure 5). We used the Support Vector Machine (SVM) algorithm with Gaussian Radial Basis Function (RBF) kernel.

In the prediction phase, once all the segments are classified or "localized", we pick the graspable segment that is closest to the robot and also farthest away from other graspable objects in the scene.

Fig. 5. Objects used for training the supervised localization classifier.

## VI. Grasping Overview

In this section, we will describe how our grasping algorithm works. The data provided to the algorithm is a 2D image, a 3D point cloud and a 2D bounding box from the segmentation algorithm described above.

Our algorithm consists of three main steps. It first transforms the bounding box from 2D to 3D to take advantage of provided 3D data. It then constructs a mesh which is an approximation of the shape of the object. Finally, by assuming the symmetry of the object, we perform sampling to find contact points that are accessible to the robot.

In our algorithm, we consider *antipodal grasps* in which the robotic manipulator grasps an object by placing its fingers at two opposite points on the object. Note that 3D data is not adequate because the backside cannot be captured by the sensors. To address this problem, we assume symmetry of objects and construct a triangular mesh for the current object. The algorithm will then use this mesh to find grasping points such that no collision will occur.

### A. 3D Bounding Box

To find a 3D bounding box, we first transform 3D data to a new coordinate system, where one axis corresponds to gravity force, i.e., the vertical direction, the other axis is the horizontal direction, and the third is the cross product of the two.

We project every point in the 2D bounding box to this coordinate system and use the minimum and maximum values for every coordinate to define the 3D bounding box.

### B. Triangular Mesh Construction

In this step, we construct a triangular mesh from the depth data and then use it to check collisions between the hand and the object.

We construct a mesh of the object using the 3D data of all pixels in the 2D bounding box. A mesh for an object consists of many local meshes. Each local mesh is constructed using four adjacent pixels. If any points corresponding to these pixels lie outside the 3D bounding box, they will be projected to the nearest face of the 3D bounding box.

### C. Antipodal Grasp

Once a triangular mesh is constructed, we are now ready to find contact points for the robot to grasp the objects. Our goal is to find points that make the grasps most stable. In this section, we will elucidate a simple method that makes best use of the given segmentation information and hand configuration.

We consider the case of grasping with two contact points. This is because manipulators with two fingers are quite common and also our algorithm is significantly simplified under such configuration. To extend this method to three-fingered configuration, we use the thumb to reach the first point and configure the second point to be in the middle of the other two fingers.

First, our method samples random initial grasp points. For each grasping point, it computes the normal vector at the grasping point based on nearby surface information. Specifically, the normal vector can be computed by principal component analysis of depth data around the current grasping point. We denote the grasping point by $P$ and the normal vector by $\vec{n}$ (see Figure 7a).

Next, by assuming *object symmetry* along the vertical axis of the object, when a grasping point is acquired, its opposite grasping point is assumed to have a normal vector pointing in the direction opposite to the normal vector of the initial grasping point. The opposite grasping point, under the symmetry assumption, is therefore obtained by taking the intersection of the line formed by $P$ and $n$ with the opposite side of the object.

Once the two points are chosen, we have one remaining degree of freedom, which is the approach angle of the hand to the pair. This degree of freedom will be further solved by collision detection.

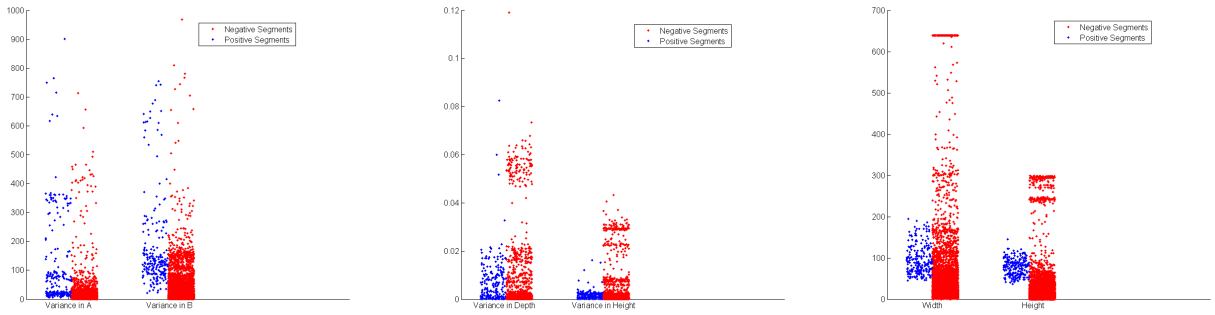
Specifically, according to the configuration of the hand, a finger is in contact with the object when the finger is pointing into the normal direction $\vec{n}$ which is perpendicular to the approaching direction $\vec{a}$ of the hand. Since such direction is perpendicular to the normal vector of the grasping point, this degree of freedom forms a plane. This plane will be used for collision checking between the hand and the object.

We discretize this degree of freedom (by angles) and search for one that has no collision with nearby objects. Among all searched angles, we choose the angle that has the maximum distance between the intersection and the line formed by the grasping point and the normal vector. If this distance ($d_g$ in Fig. 7(b)) is not adequate for the palm placement $d_h$ in Fig. 7(b)), the grasping point is rejected. If the space between these two points is enough to place a finger ($d_p < d_s$ and $d_t < d_f$), the initial grasping point $\vec{p}$ and the first reached point $\vec{p_1}$ are reported as a valid grasp.

We can repeat the above process to find more valid grasps for all sampled points. In this work, we find 30 valid grasps and rank them by their distance to the centroid of the object.

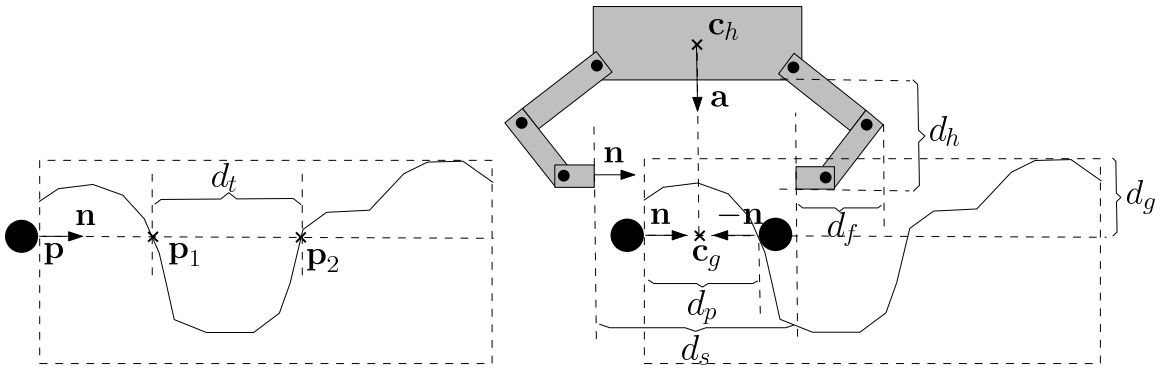
## VII. Descriptions of the Robot

We performed our experiment on the STanford AI Robot (STAIR2). This robot has a 7-DOF arm (WAM, Barrett




(a) Variance in A and B in the LAB color space (b) Variance in depth and height of the segments in the point cloud data (c) Width and height of the segments in the 2D image.

Fig. 6. Feature Vector Plots for a subset of features employed by our classifier. Blue dots correspond to graspable segments (positive class) and red dots correspond to ungraspable segments (negative class).




(a) (b)

Fig. 7. Finding an antipodal grasp. (a) Two grasping pairs $(p, p_1)$ and $(p, p_2)$ formed by $p$ and $\vec{n}$ and their intersection with the object boundaries. (b) Grasping pair $(p, p_1)$ is considered because it best matches the hand configuration and no collision occurs.

Technologies) and a three-fingered 4-DOF hand.

As mentioned earlier, to capture depth data, the robot uses an active triangulation sensor which contains a laser projector and a camera [4]. The camera returns a 640x480 color image. The active triangulation sensor returns a very dense depth map for most pixels in the image. Figure 3 shows data captured by the camera and the active triangulation system.

The whole system (arm, camera, laser) is calibrated by our recently proposed algorithm for joint calibration [32]. The average calibration errors of the entire system are often less than 5mm.

## VIII. Experiments

### A. Offline experiments with segmentation

In the first experiment, we would like to test the performance of our localization algorithm on a offline labelled dataset. Our goal of this experiment is to understand how well the segmentation and supervised classifier (Stage 2 and Stage 3 in Figure 2) perform without actual grasping. For this experiment, we collected a test set that consists of 100 fairly cluttered scenes of **novel** objects and a 100 scenes of single **novel** objects. Novel objects are the ones which do not belong to the training set or more informally the ones that "the robot has never seen before." Each scene generally has up to ten such objects. For each scene, we labelled all graspable objects by drawing a contour segment along the boundary of the objects. Whenever the segmentation and classifier return a segment, that overlaps 70% with a labelled segment, we consider that a "success"; otherwise we consider that a "failure".

We record the Max F1-score and average precision on this test set and report it in Table I. As can be seen from the Table, our segmentation with depth gives significantly better results than segmentation without depth. Moreover, our algorithm gives very high accuracy before any grasping takes place. We also plot the results in Figure 8. We also visualize the effectiveness of segmentation with depth and color data in comparison with standard color segmentation and show the results in Figure 9. This visualization confirms previous numerical results: segmentation with depth is more robust than segmentation without depth. As a result, the first two steps give very high accuracy even with rather cluttered tables.

### B. Visualizing grasping points

We further analyze the performance of our algorithm by visualizing the grasping points produced by Stage 4 and 5

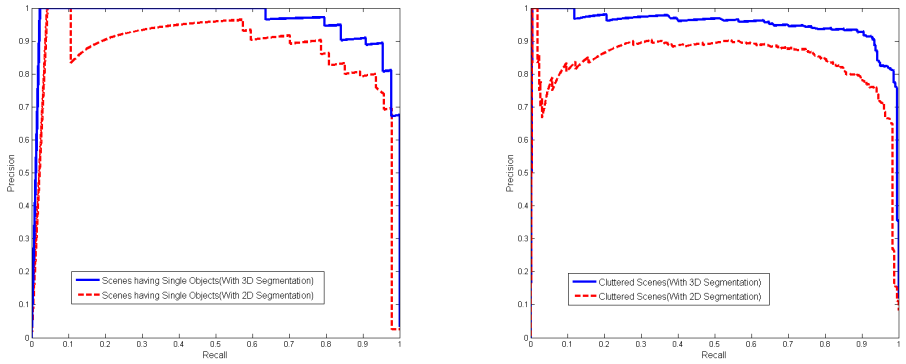



Fig. 8. Precision Recall curves for object localization in simple (Left) and cluttered scenes (Right).

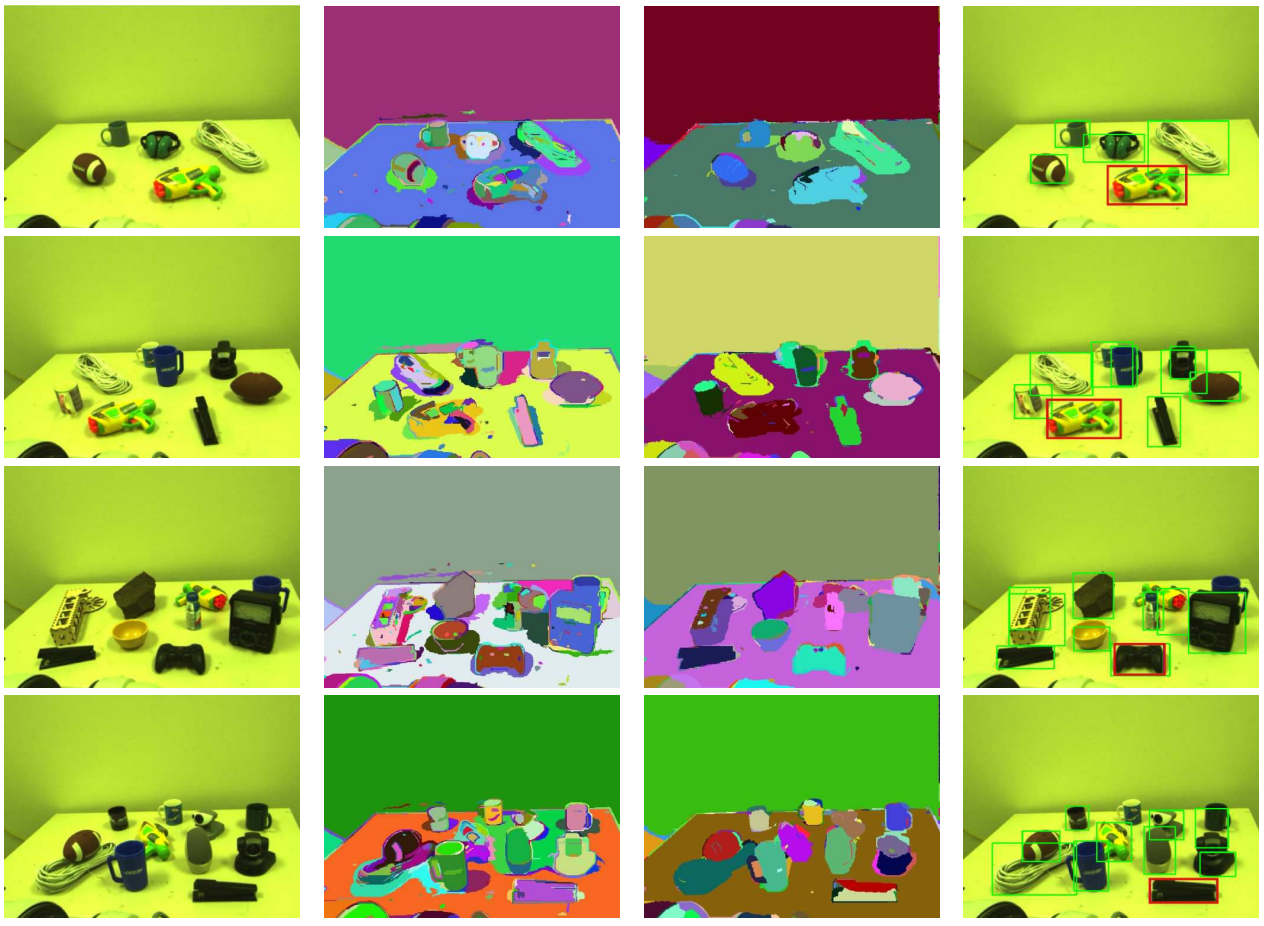


Fig. 9. Segmentation with and without depth and localization for a cluttered table with many novel objects. Column 1: original scenes. Column 2: segmentation without depth data. Column 3: segmentation with depth data. Column 4: Localization with depth segmentation – objects in red rectangles will be grasped first. The execution of grasping will result in the simplification of the scenes.

in the processing pipeline (see Figure 2).

In this experiment, we took several cases in the previous algorithm, ran the segmentation and localization algorithm to produce the bounding boxes. We then visualized the 3D meshes and the grasping points produced by our algorithm in Figure 10.

### C. Grasping novel objects

In this experiment, we executed all the steps of our algorithm in order to grasp novel objects using the STAIR2 robot. A success case is recorded if the robot can grasp an object on a table and drop it into the bin; otherwise we record that as a failure. Our grasping pipeline was executed

TABLE I

LOCALIZATION STATISTICS FOR CLUTTERED AND SINGLE-OBJECT SCENES.

| | Cluttered Scenes | | Single Object Scenes | |
|---|---|---|---|---|
| | 3D | 2D | 3D | 2D |
| Max. F-1 Score | 0.9102 | 0.8323 | 0.9231 | 0.8627 |
| Average Precision | 0.6021 | 0.5240 | 0.6584 | 0.6159 |

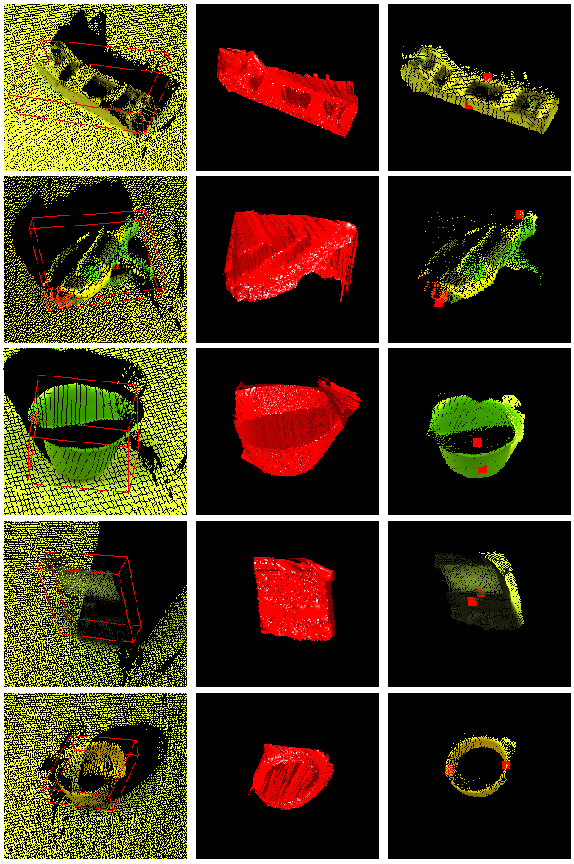


Fig. 10. Visualization of the grasping points produced by the grasping algorithm (best viewed with colors). Left: point cloud with 3D bounding boxes around segmented objects. Middle: 3D mesh of segmented objects. Right: grasping points (red dots) produced by the grasping algorithm.

under different test conditions placing the robot at different positions with respect to the table. We report our grasping results in Table II and compare them with results in [1] and [3]. The results show that despite its simplicity, the algorithm performs very well in grasping novel objects. Note that the 3D mesh is an approximation of the shape of the object, hence the robot hand in some cases collides with objects while grasping.

### D. Robot cleaning a table

We also applied the above algorithm to successfully clean tables with several new objects. In general, our algorithm is more robust than the algorithm described in [3]. The video attachment with this paper shows two such cases.

TABLE II

GRASPING NOVEL OBJECTS WITH THE STAIR2 ROBOT.

| Objects | Method in [1] | Method in [3] | Our Method |
|---|---|---|---|
| Football | 50% | 70% | 95% |
| Mug | 80% | 90% | 95% |
| Nerf Gun | 50% | 75% | 65% |
| Helmet | 90% | 75% | 95% |
| Robot Arm | 70% | 80% | 80% |
| Foam | 70% | 85% | 90% |
| Cup | 70% | 85% | 85% |
| CD Holder | 75% | 95% | 95% |
| Mean/Std | 69.3 $\pm$ 13.7% | 81.8 $\pm$ 8.4% | 87.5 $\pm$ 10.6% |

## IX. DISCUSSION AND CONCLUSION

In this paper, we considered the task of grasping novel objects and cleaning a cluttered table. We showed that grasping can be significantly simplified if segmentation is employed. To achieve good segmentation, we proposed to use depth data in combination with visible light image data. We also designed a supervised classifier to select only graspable segments. Given the segments, 3D construction of the object can be carried out. As a result, we can use a very simple grasping algorithm and achieve very high accuracy compared to other competitive methods.

## ACKNOWLEDGMENT

This research is financially supported in part by the Thailand Research Fund through the Royal Golden Jubilee Ph.D. program under grant No. Ph.D. 1.O.CU/49/D.1 and the 90th Anniversary of Chulalongkorn University Fund through the Ratchadapiseksomphot Fund, both are greatly appreciated.

## REFERENCES

[1] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *IJRR*, 2008.
[2] A. Saxena, L. Wong, and A. Y. Ng, "Learning grasp strategies with partial shape information," in *AAAI*, 2008.
[3] Q. V. Le, D. Kamm, A. Kara, and A. Y. Ng, "Learning to grasp objects with multiple contact points," in *ICRA*, 2010.
[4] M. Quigley, S. Batra, S. Gould, E. Klingbeil, Q. Le, A. Wellman, and A. Y. Ng, "High accuracy 3D sensing for mobile manipulators: Improving object detection and door opening," in *ICRA*, 2009.
[5] V. Nguyen, "Constructing stable force-closure grasps," in *ACM Fall joint computer conference*, 1986.
[6] ——, "Constructing stable grasps," *IJRR*, 1989.
[7] J. Ponce, D. Sam, and B. Faverjon, "On computing two-finger force-closure grasps of curved 2d objects," *IJRR*, 1993.
[8] K. Lakshminarayana, "Mechanics of form closure," in *ASME*, 1978.
[9] J. Salisbury, "Active stiffness control of a manipulator in cartesian coordinates," in *IEEE Conference on Decision and Control*, 1980.
[10] ——, "Kinematic and force analysis of articulated hands," Ph.D. dissertation, Stanford University, 1982.
[11] M. Cutkosky, "Mechanical properties for the grasp of a robotic hand," CMU, Tech. Rep., 1984.
[12] H. Hanafusa and H. Asada, "Stable prehension by a robot hand with elastic fingers," in *Seventh Inter. Symp. on Industrial Robots*, 1977.
[13] R. Pelossof, A. Miller, and T. Jebera, "An SVM learning approach to robotic grasping," in *ICRA*, 2004.
[14] K. Hsiao and T. Lozano-Perez, "Imitation learning of whole-body grasps," in *IROS*, 2006.
[15] K. Hsiao, L. Kaelbling, and T. Lozano-Perez, "Grasping POMPDPs," in *International Conference on Robotics and Automation*, 2007.
[16] E. Chinellato, R. Fisher, A. Morales, and A. del Pobil, "Ranking planar grasp configurations for a three-finger hand," in *ICRA*, 2003.

[17] J. Coelho, J. Piater, and R. Grupen, "Developing haptic and visual perceptual categories for reaching and grasping with a humanoid robot," in *Robotics and Autonomous Systems*, 2001.
[18] D. Bowers and R. Lumia, "Manipulation of unmodelled objects using intelligent grasping schemes," *IEEE Trans. on Fuzzy Systems*, 2003.
[19] A. Morales, E. Chinellato, P. Sanz, and A. del Pobil, "Learning to predict grasp reliability for a multifinger robot hand by using visual features," in *International Conference AI Soft Computing*, 2004.
[20] R. Platt, R. Grupen, and A. Fagg, "Improving grasp skills using shema structured learning," in *ICDL*, 2006.
[21] R. Platt, A. H. Fagg, and R. Grupen, "Learning grasp context distinctions that generalize," in *IEEE-RAS International Conference on Humanoid Robots*, 2006.
[22] A. Edsinger and C. Kemp, "Manipulation in human environments," in *IEEE/RAS International Conference on Humanoid Robotics*, 2006.
[23] A. Saxena, J. Driemeyer, J. Kearns, and A. Ng, "Robotic grasping of novel objects," in *NIPS*, 2006.
[24] A. Saxena, "Monocular depth perception and robotic grasping of novel objects," Ph.D. dissertation, Stanford University, 2009.
[25] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.
[26] A. Bab-Hadiashar and N. Gheissari, "Range image segmentation using surface selection criterion," *IEEE Transactions on Image Processing*, 2006.
[27] A. Harati, S. Gchter, and R. Siegwart, "Fast range image segmentation for indoor 3d-slam," in *6th IFAC Symposium on Intelligent Autonomous Vehicles*, 2007.
[28] P. de la Puente, D. Rodriguez-Losada, R. Lopez, and F. Matia, "Extraction of geometrical features in 3d environments for service robotic applications," in *HAIS*, 2008.
[29] C. Kemp, C. Anderson, H. Nguyen, A. Trevor, and Z. Xu, "A point-and-click interface for the real world: Laser designation of objects for mobile manipulation," in *HRI*, 2008.
[30] H. Nguyen, C. Anderson, A. Trevor, A. Jain, Z. Xu, and C. Kemp, "El-e: An assistive robot that fetches objects from flat surfaces," in *Robotic Helpers Workshop at HRI*, 2008.
[31] P. F. Felzenszwalb and D. P. Huttenlocher, "Effecient graph-based image segmentation," *International Journal of Computer Vision*, 2004.
[32] Q. Le and A. Ng, "Joint calibration of multiple sensors," in *IROS*, 2009.