Visual Robotic Object Grasping Through Combining RGB-D Data and 3D Meshes

Yiyang Zhou[1], Wenhai Wang[1], Wenjie Guan[1], Yirui Wu[2], Heng Lai[1], Tong Lu[1(✉)], and Min Cai[3]

[1] National Key Lab for Novel Software Technology, Nanjing University, Nanjing, China
zyy34472@gmail.com, wangwenhai362@163.com, terry_guanwenjie@163.com, 134lhforever@gmail.com, lutong@nju.edu.cn

[2] College of Computer and Information, Hohai University, Nanjing, China
wuyirui1989@163.com

[3] Riseauto Intelligent Tech, Beijing, China
caimin@riseauto.cn

Abstract. In this paper, we present a novel framework to drive automatic robotic grasp by matching camera captured RGB-D data with 3D meshes, on which prior knowledge for grasp is pre-defined for each object type. The proposed framework consists of two modules, namely, pre-defining grasping knowledge for each type of object shape on 3D meshes, and automatic robotic grasping by matching RGB-D data with pre-defined 3D meshes. In the first module, we scan 3D meshes for typical object shapes and pre-define grasping regions for each 3D shape surface, which will be considered as the prior knowledge for guiding automatic robotic grasp. In the second module, for each RGB-D image captured by a depth camera, we recognize 2D shape of the object in it by an SVM classifier, and then segment it from background using depth data. Next, we propose a new algorithm to match the segmented RGB-D shape with predefined 3D meshes to guide robotic self-location and grasp by an automatic way. Our experimental results show that the proposed framework is particularly useful to guide camera based robotic grasp.

Keywords: 3D mesh · Registration · Robotic grasping · 3D matching

1 Introduction

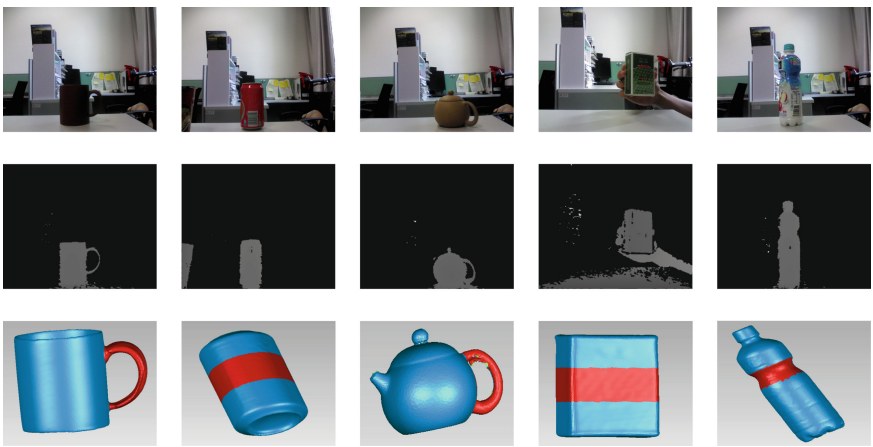
In the past years, robots have become much more popular in real-life applications such as industry, health care, human service and education. One reason is that new types of sensing hardware which have lower costs but higher computational performances are now becoming much cheaper nowadays. For example, RGB-D cameras (e.g., Microsoft Kinect and Intel Realsense) allow a robot to see different objects in an indoor scene with per-pixel depth information besides RGB data [1], which is particularly useful in developing environment perception systems

L. Amsaleg et al. (Eds.): MMM 2017, Part I, LNCS 10132, pp. 404–415, 2017.
DOI: 10.1007/978-3-319-51811-4_33

to make robots more intelligent to walk around in unknown scenes automatically. Therefore, more researchers have been attracted in RGB-D camera based robotics explorations such as indoor environment 3D modeling [2], robot self-localization [4] and navigation [5] in the multimedia community today.

However, unlike automatic path plan research, there are few reported works on robotic grasp planning by using visual RGB-D data. This is because of the following difficulties. First, detecting proper regions on real scene objects for grasping is challenging due to the inherent limitations of vision based methods like the variations on viewpoint, size or distance of cameras and the influences from illumination conditions, which make it very difficult to decide proper points on objects for grasping. For a comparison, other systems like 3D modeling of indoor environments [2,3] need only generate 3D maps with large loops by RGB-D mapping, the accuracy of which is much lower than that of an intelligent grasp plan system. Second, objects in real scenes may have varied 3D surface shapes, which make the grasp plan task more challenging. Note that generally there are no other sensors are used for guiding grasping objects except for visual cameras here; however, a lot of new sensors can be selected in developing other systems like automatic navigation for robots.

In this paper, we present a novel method for detecting grasp regions on indoor scene objects by combing both RGB-D data and 3D mesh priors. A depth camera of Intel Realsense is used to capture source visual data composed of RGB and depth information, which are denoted by *Depth* and *Color*, respectively. Assume there are N visual object shapes of different categories for grasping, the overview of the proposed framework is shown in Fig. 1. We first categorize these objects



(a) Cup (b) Can (c) Teapot (d) Box (e) Bottle

Fig. 1. The first row shows the color data of five different objects captured by a Realsense camera, the second row presents their corresponding depth data, while the third row illustrates their corresponding 3D meshes with pre-defined grasp regions, which are marked the red color for guiding robotic grasp in the images of the first row. (Color figure online)

into representative shapes, and model their shape surfaces by 3D meshes as the prior knowledge of these objects. For example, we pre-define proper grasp regions on different 3D meshes as shown in the second row of Fig. 1. This is necessary due to viewpoint variations of the camera or shape variations of the object, only RGB-D data as shown in the first and the second rows of Fig. 1 are not reliable to drive accurate robot grasp. Next, after recognizing and segmenting scene objects using an SVM classifier $Classifer_{obj}$, we propose a novel algorithm to register the point clouds generated by the intrinsic parameters from *Depth*. In this step, we adopt the RANSAC strategy with the FPFH feature for matching. Finally, proper grasp regions are determined from camera captured RGB-D data by an accurate and robust way to overcome the inherent shortcomings of vision based methods for robotics. To the best of our knowledge, this is the first work for exploring grasp planning for robots by combining both RGB-D data and 3D meshes to overcome the inherent difficulties such as viewpoint and distance variations of vision based methods for accurate robotic self-location or grasp.

2 Related Work

There are a number of research using RGB-D data in developing robotic systems. These approaches can be roughly categorized into the following two classes consisting of 3D navigation and vision driven robotic hand grasping.

3D navigation systems use stereo cameras to align consecutive frames and detect loop closures for robots. Iterative Closet Point (ICP) algorithms are explored to match the best alignment between frames. Henry *et al.* [3] investigate how RGB-D cameras can be used for building dense 3D maps of indoor environments for robot navigation. Points in a source cloud are matched with their nearest neighboring points in a target cloud by a rigid transformation through minimizing the sum of squared spatial error between associated points. Prusak *et al.* [4] combine a time-of-flight camera with a CCD camera for robot self-localization. To solve the problem of no proper illumination, Prakhya *et al.* propose Sparse Depth Odometry (SDO) for keypoint detection, which comprises two keypoint detectors, namely, SURE and NARF, to augment RGB-D camera data for robots. Peasley and Birchfield [5] further use geometric and color information to overcome the significant limitation, namely, the sole reliance upon geometric information, of ICP algorithms used by Kinect Fusion.

Robotic object grasping is increasingly hard as the robot or the environment is not constrained. Due to the richness of stimulus, generally only visual cameras can be used to assist this task [12]. Saponaro [13] describes an approach for real-time preparation of grasping tasks based on the low-order moments of object shape on a stereo pair of images. Horaud *et al.* [14] present an algorithm to compute a set-point automatically from conic features interactively selected by an operator onto the object, and then use a vision based algorithm to guide grasping cylindric parts with a 6 degree of freedom robot arm. Bergamasco *et al.* [6] propose a technique for fitting elliptical shapes in 3D space by performing an initial 2D guess on each image followed by a multi-camera optimization. These

methods show their effectiveness in developing robot systems by using visual cameras; however, knowledge priors of 3D objects are not used for guiding more accurate grasping on different types of objects.

3 Overview of the Framework

The overview of the proposed framework is shown in Fig. 2. Let $model_i$ denote the *i-th* (i is from 1 to N) pre-defined 3D mesh in the mesh set $Models$. For $Model_i$, the pre-defined grasp regions are represented by gp_i ($1 \leq i \leq N$). As a result, the mesh set $Models$ consists of $model_i$ and gp_i for each i in $1, 2..., N$ by

$$Models = \{< model_i,\ gp_i >\},\ (1 \leq i \leq N) \tag{1}$$

For a depth image, its *Depth* information is first used to segment it into object and background regions with the help of *Color* information. For the extracted regions, we further extract key points depending on the characteristics of each object type, e.g., for a cuboid, we search for its visible vertices and edges in the image. RGB-D data are then converted to point cloud data in this step. Next, we propose a novel method for matching camera captured RGB-D data with accurately pre-defined 3D meshes for determining proper regions for grasping in the RGB-D space. By this way, image data captured by the RGB-D camera on the robot can be used to guide self-localization or grasping by a robust and accurate way.

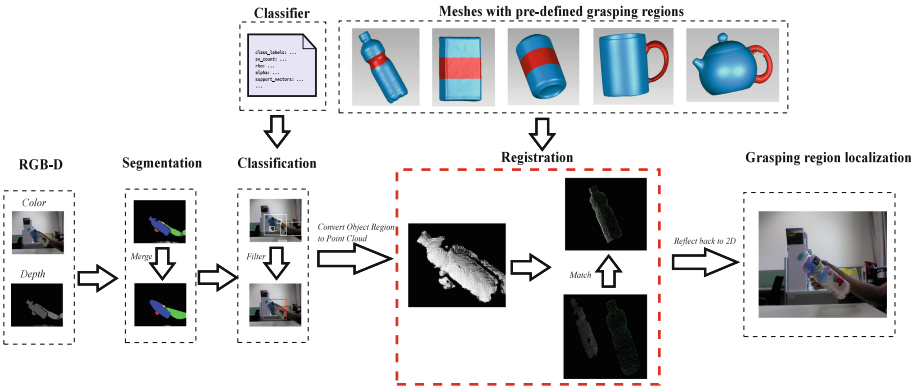


Fig. 2. The overview of the proposed framework for robotic grasping on visual objects by combining RGB-D data and 3D mesh priors. (Color figure online)

4 Pre-processing

In the pre-processing stage, the following two preparations are required, that is, (1) creating 3D meshes for each object category and define grasping region priors on these 3D meshes, and (2) training a classifier to detect and recognize visual objects from input 2D RGB images.

Object Detection from 2D Images. In this step, we train an SVM (Support Vector Machine) classifier with HOG features as in [8] for object detection. For this target, we build a dataset with the depth camera and manually label the ground truth for grasping on objects. We train altogether $N+1$ classifiers by considering indoor backgrounds as a special kind of negative class. The classifiers are denoted by $Classifer_{obj}$, the output of each classifier is $0, 1, 2, ..., N$ to give the categorization of each indoor object.

Pre-defining Grasping Priors on 3D Meshes. As well known, accurate estimation of size or shape of an object from 2D images is very challenging due to the variations on viewpoint or distance of cameras. For robotics grasping of different types of objects, a better way is to first create their 3D meshes and pre-define proper grasp regions as knowledge priors for different object types accordingly, and then use such knowledge to guide robotic self-localization and grasping by matching real time RGB-D data with the pre-defined 3D mesh. Assume there are N objects of different shapes, we build their 3D meshes separately, and annotate grasp regions gp_i on $model_i$ $(1 \leq i \leq N)$ on each mesh $model_i$.

5 Visual Robotic Object Grasping

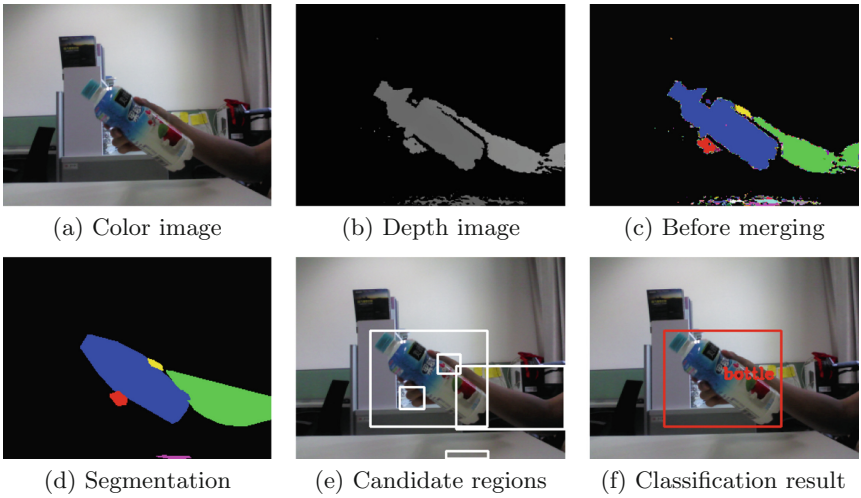
After detecting scene objects, a novel method for matching these objects with accurately pre-defined 3D meshes for determining proper regions is proposed.

5.1 Shape Analysis from RGB-D Data

We first segment each object shape from the input RGB-D image. After classifying each object into a known type, we determine grasp regions by matching real-time RGB-D data with the priors pre-defined on 3D meshes.

Object Segmentation Using Depth Information. A depth camera which is manufactured by infrared rays generally represents unreliable points as black. A black point here means that the distance from its position is either out of range or uncertain. That is, there is a recommended distance for either Kinect or Realsense cameras, the range out of which appears as black as shown in Fig. 3(b) (see the red region). Besides this, sometimes black regions are also brought due to diffuse reflections.

Considering this, we first connect non-black points in *Depth* into groups using a region-growing method. Note that the points in *Depth* are also clustered into groups according to whether the distance is near or far. As a result, we divide the input image into several depth connected regions DCR and black connected regions BCR with *Depth*. The segmentation results are shown in Fig. 3(c). According to the size of each region, we sort DCR by a descending order and select the *topk* regions as the main segments, which are denoted by



(a) Color image (b) Depth image (c) Before merging

(d) Segmentation (e) Candidate regions (f) Classification result

Fig. 3. (a) and (b) respectively show color and depth source captured by Realsense. The initial segmentations with $topk = 5$ are shown in (c). After merging small regions and black regions, we obtain the results as shown in (d). The white rectangle bounding boxes in (e) denote the *topk* color candidate regions to calculate HOG features, which will be classified by an SVM classifier. The recognition results in (f) show that rectangle regions filtered by the classifier are marked red, while the rest white regions are removed from candidate regions after classifying. (Color figure online)

MS. We further search for convex hulls of the main segments, and define the average distance of the points in a segment as the distance $Distance$ of it.

Next, the rest small regions and black points in MS are further processed as follows. We group small regions to MS_i, $(1 \leq i \leq topk)$ as follows. If a small region sr belongs to more than one regions, we allocate sr to the nearest main region by $\arg\min_{ms}(|Distance(sr) - Distance(ms)|)$. For BCR, the points in black regions may also belong to more than one convex hull of MS. However, it is most likely that the black region belongs to MS which has the minimum distance. As a result of black points with the default value 0, the distance of black region $Distance(br)$ equals to 0. We find that black regions are due to the minimum distance Main Segmentation as well. Thus we define that br belongs to $\arg\min_{ms}(|Distance(br) - Distance(ms)|)$. By this way, we add those small regions and black regions to the same set.

Finally, we segment camera captured object from background. The proposed segmentation algorithm from the input RGB-D image is shown in Algorithm 1. Note that owing to the fact that depth data record distance for every pixel, the classic Region-Growing [9] algorithm helps us utilize depth data effectively to well segment scene objects from indoor backgrounds. After segmentation, we obtain *topk* regions, and the example results of this step are shown in Fig. 3(d).

Algorithm 1. Segmentation from the input RGB-D image

Input:
Depth data: $Depth$
Number of Main Segmentations: $topk$
Output:
Main Segmentations: MS
$< BCR, DCR >$ ← Region-Growing($Depth$)
Sort DCR according to the number of points
Select $topk$ segmentations as MS from DCR
for $seg \in (DCR - MS) \cup BCR$ **do**
  for $p \in seg$ **do**
    $CS \leftarrow \{ms | ms \in MS \wedge p \; in \; ConvexHull(ms)\}$
    $SEG \leftarrow \arg\min_{ms}(distance(ms), ms \in CS)$
    $SEG \leftarrow SEG \cup \{p\}$
  end for
end for
return MS

Classifying Object Types. After segmenting from the input RGB-D image, we need further determine object class for each segmented region. As shown in Fig. 3(e), we calculate a white rectangle bounding box for each segmented region. These bounding boxes actually represent the ROI (Region of Interest) information, which contain our interested indoor scene objects for further verifying. For the $topk$ ROIs, we determine their object classes using classic linear SVM classifier with one-versus-rest strategy. For the $i\text{-}th$ segment, we first calculate its rectangle bounding box $BoundBox_i$, $(1 \leq i \leq topk)$ from the convex hull which we obtain in segmentation step. The $topk$ rectangle regions are then classified by $Classifer_{obj}$. Note that for each region, it is resized to the same scale of the training samples for feature extraction (we resize it to 64×64 according to experiments). Additionally, we adopt HOG (Histogram of Oriented Gradient) [7] as the feature because HOG is based on gradient and the margin of a regular indoor object generally has significant gradient characteristic. Moreover, HOG is inexpensive and suitable for discriminating regular objects. With the extracted feature vectors, a predicted label, whose range is from 0 to N, is generated by $Classifer_{obj}$. As a result, we obtain rectangle regions labeled by our interested object shape types of $1, 2, ..., N$ as shown in Fig. 3(f) by

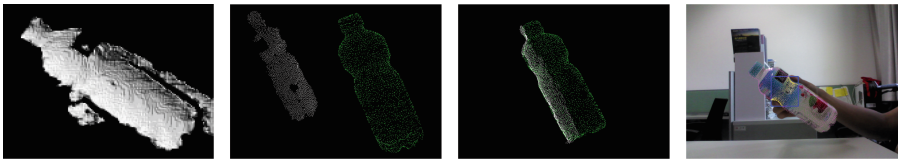
$$Regions = \{< r, l > | r \in Rect \wedge 0 < l \leq N\} \tag{2}$$

where $|Regions|$ represents how many regular object types need to be classified, and $rect$ denotes the position and size of an ROI in the 2D image.

5.2 Matching Point Cloud with Pre-defined 3D Meshes for Guiding Robotic Grasp

In this section, we introduce how to match a recognized visual object with a pre-defined 3D mesh for driving robust robotic grasp plan. We adopt FPFH

(Fast Point Feature Histograms) [10] as the descriptor for extracting features. FPFH is actually an efficient approximation of PFH and reserves the most useful information of the latter. Essentially, FPFH has 6-DOF invariance and is robust for neighbourhood noises at different sampling densities. FPFH thus is suitable for matching point clouds here (Fig. 4).



(a) Points Cloud (b) Before registration (c) Registration (d) Localization

Fig. 4. We generate point cloud data from the depth information using Realsense, and then grasp regions on camera captured point cloud are guided for robots. (a) point clouds after segmentation and classification results, (b) the point cloud colored green denotes the pre-defined 3D cup mesh after downsampling, while the white shape represents the point cloud obtained by the RGB-D camera, and (c) gives the shape matching process. The grasping region is shown by yellow points in (d), and the blue rectangle show minimum bounding box of grasping region which we used to compare with ground truth. (Color figure online)

We first estimate the normals for each point both in the 3D mesh coordinate system and the world coordinate system. Specifically, for each pre-defined mesh, we estimate the normals for each point on it through its nearest neighbors. We set *radius* to search for the neighbours with an octree to speed up this process. With these normals, we then extract the FPFH features for each point on a 3D mesh. We then use camera calibration to obtain point cloud data for the segmented object from RGB-D data. Note that these point cloud data generally only contain partial 3D shape information since the back side of an object is always invisible for a RGB-D camera. Similarly, we calculate normals and extract FPFH for each point from the point cloud.

Next, we register the candidate 3D mesh with point cloud data using RANSAC [11] strategy, that is, matching FPFH descriptors to recover transform matrices, which consist of a rotation matrix R and a translation vector $\boldsymbol{t}$. As a result, we get the rough registration and further adopt classic ICP with Euclidean distance to fine tune the registration results. The details of the method can be seen in Algorithm 2. Note that we use cosine distance to calculate the similarity between FPFH features extracted from each 3D mesh and real-time captured RGB-D data. Additionally, when the result rotate matrix R outputs $dig(1, 1, 1)$ and the translation vector equals to $(0, 0, 0)^T$, the matching process fails and a new viewpoint of the camera on the robot is required. With R and $\boldsymbol{t}$, we gradually convert pre-defined grasp regions on a matched 3D mesh to point

Algorithm 2. Registration with pre-defined 3D meshes for guiding robotic grasp

Input:
Point Cloud Region: $PointCloud$
3D Object Models: $Model_{label}$
Min Similarity: sim
Max Acceptable Distance: mad
Max Iterations: mi
Acceptable Number: an

Output:
Rotate Matrix: R, Translation Vector: $\boldsymbol{t}$

$R \leftarrow dig(1,1,1)$, $\boldsymbol{t} \leftarrow (0,0,0)^T$
repeat
$mrs \leftarrow$ Random-Select($Model_{label}$)
$pcrs \leftarrow$ Random-Select($PointCloud$)
$similar\text{-}pairs \leftarrow \{< p1, p2 > | Similarity(FPFH_{p1}, FPFH_{p2}) > sim \wedge p1 \in mrs \wedge p2 \in pcrs\}$
R, $\boldsymbol{t} \leftarrow$ Estimate-TransformMatrix($similar\text{-}pairs$)
$inliers \leftarrow \{p| < p, * > \in similar\text{-}pairs\}$
$alsoinliers \leftarrow \{p|min(distance(R * p + \boldsymbol{t}, q), q \in pcrs) < mad \wedge p \in mrs - inliers\}$
until $|alsoinliers| > an$ or Reach mi
R_{icp}, $\boldsymbol{t}_{icp} \leftarrow Classic\text{-}ICP(R * Model_{label}, PointCloud)$
$R \leftarrow R_{icp} * R$
$\boldsymbol{t} \leftarrow R_{icp} * \boldsymbol{t} + \boldsymbol{t}_{icp}$
return R, $\boldsymbol{t}$

cloud captured from robot camera. The positions of a grasp point in the world coordinate can be expressed as follows:

$$position_{cloud} = \{\boldsymbol{p}_c | \boldsymbol{p}_c \leftarrow R\boldsymbol{p}_m + \boldsymbol{t}\} \tag{3}$$

where $\boldsymbol{p}_m = (x, y, z)^T$ represents the position of a grasping point on a 3D mesh.

Once the point cloud of the segmented object shape is successfully matched with a specific pre-defined 3D mesh, gp_{label} on $model_{label}$ will also be matched, which means grasp regions in the input image can be easily converted with the camera transform matrix for guiding robotic grasp.

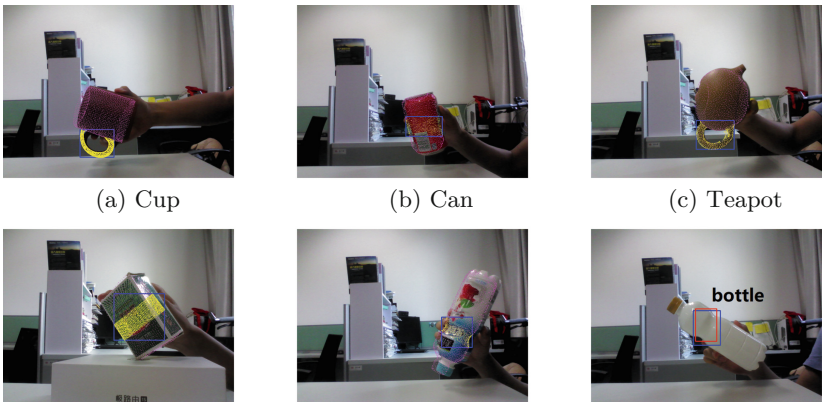
6 Experimental Results and Discussion

To evaluate the effectiveness of the proposed framework, we collect five desktop objects as the dataset due to there are no such benchmark datasets comprising both camera captured images and scanned 3D models for guiding robotic grasp.

6.1 Dataset

We used an *Intel Realsense F200* camera on a Dell Inspiron 660 S with *Intel(R) Core(M) i3-2130 CPU@3.40 GHz*. Five regular desktop object types consist of

Cup, Can, Teapot, Box and Bottle are collected in our dataset; meanwhile, the 3D mesh of each object type is pre-scanned using a 3D scanner. We manually annotated grasp regions on each 3D mesh, which are considered as pre-defined knowledge for guiding automatic grasp in the proposed method (see the red regions of the third row in Fig. 1). The groundtruth for evaluating is composed of both object category and grasping region.



(a) Cup (b) Can (c) Teapot

(d) Box (e) Bottle (f) Groundtruth

Fig. 5. (a)-(e) show the grasping region by our method. Yellow: predicated regions for grasping, the bounding box for each region is also annotated. Note that (f) gives the groundtruth example for Bottle, the bounding box of which is marked red. (Color figure online)

6.2 Evaluation Criteria

As shown in Fig. 5(f), grasping groundtruth are manually marked by red rectangles, while the blue rectangles are the automatically calculated results using the proposed framework. To evaluate the results, we define the following criteria using Jaccard Similarity:

$$J = \frac{|P \bigcap G|}{|P \bigcup G|} \tag{4}$$

which P is the predicted region by algorithm, G is groundtruth and $|P|$ represents the area of rectangle P.

6.3 Results

In our experiment, we adopt different registration methods for comparisons. We select the classic ICP, With-Normals ICP that exploits a transformation estimated based on Point to Plane distances, and Non-linear ICP which is ICP

variant that uses Levenberg-Marquardt optimization backend. We random select about 1/3 samples in our dataset to test and the left samples we utilize to train the HOG-SVM classifier for object classification. Then we take *Jaccard Similarity* as evaluation criteria to test the performance of our algorithm. In object classification step, we train a SVM classifier with train data set. Before registration, we classify detected object with SVM classifier. If the SVM classifies the sample different from the category groundtruth, (e.g. if the category is *bottle*, but the SVM output *can*) we decide *Jaccard Similarity* of sample as 0. The first columns in Table 1 gives the accuracies of object classification. The left columns in Table 1 are the average *Jaccard Similarity* with different registration algorithms. It can be found that the proposed algorithm achieves higher *Jaccard Similarity* results comparing with other ICP based methods, which shows that the proposed method is effective for automatic robotic grasp.

Table 1. Experiment of comparison different registration methods

Category	Test sample	Classification accuracy	Jaccard(Average)			
			Proposed method	ICP-classic	ICP-withnormals	ICP-nonlinear
Bottle	116	94.48%	0.657	0.309	0.307	0.316
Box	111	92.72%	0.619	0.225	0.241	0.289
Can	98	89.76%	0.597	0.445	0.372	0.329
Teacup	54	94.44%	0.695	0.042	0.132	0.177
Teapot	32	90.63%	0.658	0.165	0.237	0.215

7 Conclusion

We propose a novel framework for guiding robotic grasp plan by combing both RGB-D camera videos and pre-defined knowledge in 3D meshes. A new algorithm is proposed to match a segmented RGB-D shape with predefined 3D mesh to guide robotic self-location and grasp. Experimental results show the effectiveness of the proposed framework based on our collected dataset. To the best of our knowledge, this is the first work for exploring grasp planning for robots by combining both RGB-D data and 3D meshes to overcome the inherent difficulties such as viewpoint and distance variations of vision based methods. Our future work includes include more object types for robust robotic grasping.

Acknowledgment. The work described in this paper was supported by the Natural Science Foundation of China under Grant No. 61672273, No. 61272218 and No. 61321491, the Science Foundation for Distinguished Young Scholars of Jiangsu under Grant No. BK20160021.

References

1. Han, J., Shao, L., Xu, D., Shotton, J.: Enhanced computer vision with microsoft kinect sensor: a review. IEEE Trans. Cybern. **43**(5), 1318–1334 (2013)
2. Cheng, H., Chen, H., Liu, Y.: Topological indoor localization and navigation for autonomous mobile robot. IEEE Trans. Autom. Sci. Eng. **12**(2), 729–738 (2015)
3. Henry, P., Krainin, M., Herbst, E., Ren, X.F., Fox, D.: RGB-D mapping: using kinect-style depth cameras for dense 3D modeling of indoor enviroments. **31**(5), 647–663 (2012)
4. Prusak, A., Melnychuk, O., Roth, H., Schiller, I., Koch, R.: Pose estimation and map building with a time-of-flight-camera for robot navigation. Int. J. Intell. Syst. Technol. Appl. **5**, 355–364 (2008)
5. Peasley, B., Birchfield, S.: RGBD point cloud alignment using Lucas-Kanade data association and automatic error metric selection. IEEE Trans. Robot. **31**(6), 1–7 (2015)
6. Bergamasco, F., Cosmo, L., Albarelli, A., Torsello, A.: A robust multi-camera 3D ellipse fitting for contactless measurement. In: International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission, pp. 168–175 (2012)
7. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection, computer vision and pattern recognition (2005)
8. Chapelle, O., Haffner, P., Vapnik, N.: Support vector machines for histogram-based image classification (1995)
9. Snyder, E., Cowart, E.: An iterative approach to region growing using associative memories. IEEE Trans. Pattern Anal. Mach. Intell. **5**(3), 349–352 (1983)
10. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (FPFH) for 3D registration. In: ICRA, pp. 3212–3217 (2009)
11. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)
12. Zaharescu, A.: An object grasping literature survey in computer vision and robotics
13. Saponaro, G.: Pose estimation for grasping preparation from stereo ellipses
14. Horaud, R., Dufournaud, Y., Long, Q.: Robot stereo-based coordination for grasping cylindric parts
15. Jiang, Y., Moseson, S., Saxena, A.: Efficient grasping from RGBD images: learning using a new rectangle representation. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 3304–3311. IEEE (2011)
16. Montesano, L., Lopes, M.: Active learning of visual descriptors for grasping using non-parametric smoothed beta distributions. Robot. Auton. Syst. **60**(3), 452–462 (2012)
17. Saxena, A., Driemeyer, J., Ng, A.Y.: Robotic grasping of novel objects using vision. Int. J. Robot. Res. **27**(2), 157–173 (2008)
18. Lenz, I., Lee, H., Saxena, A.: Deep learning for detecting robotic grasps. Int. J. Robot. Res. **34**(4–5), 705–724 (2015)