"Less is More": Simplifying Point Clouds to Improve Grasping Performance

Vasco Lopes*, Luís A. Alexandre† and Miguel Fernandes‡
Instituto de Telecomunicações, Universidade da Beira Interior
Rua Marquês d'Ávila e Bolama, 6201-001, Covilhã, Portugal
*Email: vascoferrinholopes{at}gmail.com
†Email: luis.alexandre{at}ubi.pt
‡Email: killazfern{at}gmail.com

***Abstract*—Object grasping is a task that humans do without major concerns. This results from self learning and by observing of other skilled humans doing such task with previous information. However, grasping novel objects in unknown positions for a robot is a complex task which encounters many problems, such as sub-optimal performance rates and the time consumption. In this paper we present a method that complements the state-of-the-art grasping algorithms with two segmentation steps, the first one which removes the largest planar surface in the point cloud of the world before the grasp detector receives them and the second one that complements this segmentation with another segmentation that calculates where the object is located and segments the point cloud by executing a crop around the object. The proposed method significantly improves the grasping success rate (100% improvement over the baseline approach) and simultaneously is able to reduce the time consumption by 23%.**

I. Introduction

The task of grasping novel objects for a robot is very complex and with many problems, and that is the main reason why it's an important area and with active and extensive research. In this paper we present a method to improve a current state-of-the-art algorithm [4] that gives a robot the capability of grasping novel objects in unknown positions. Robots are getting more and more present in our daily basis, but some tasks still encounters many barriers, which is the case of grasping novel objects. The most predominant problems in the state-of-the-art methods are the incapacity of achieving high performance in detecting grasps and the time spent on processing the algorithm for detecting such grasps. These problems may render the robot useless in practice because in a real-life situation, if the robot fails a grasp it can damage itself or harm persons that are around it, and if it spends too much time processing, the world can change and it executes movements that are not correct anymore and may collide with objects.

The paper is organized as follows: the next section discusses some of the related work, section III presents the proposed method, section IV presents the experiments conducted, section V contains a discussion about the obtained results and the final section contains the conclusions.

978-1-5386-5346-6/18/$31.00

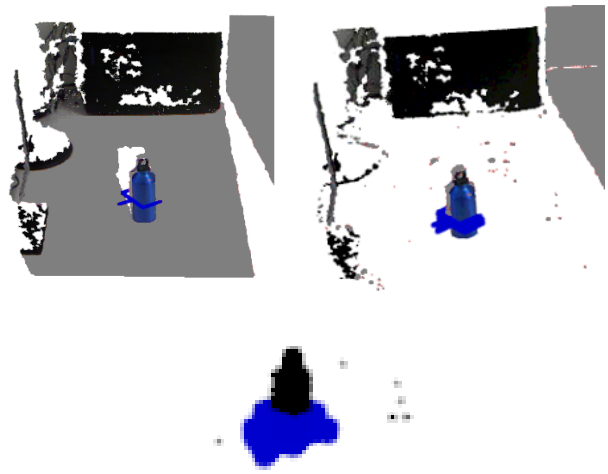

Figure 1. Three examples in which the grasp detection was successful. Top left: original approach; Top right: result of the application of the first pre-processing step; Bottom: result of the application of both pre-processing steps. The blue lines represent possible grasping positions for a two-finger parallel gripper.

II. Related Work

Extensive work has been done to improve the performance of grasps, some of them even including segmentation of the point clouds in order to improve those performances.

Varadarajan [11] presents a a framework for robotic grasping, providing segmentation and detection, and as an additional feature the automatic generation of grasps for unknown or objects based on known parts of a object. It also does various types of detections, such as concavity detection, cavity detection and others used for grasp generation even in novel objects.

Richtsfeld *et al.* [8] also presents a framework for segmentation of RGB-D images. The method starts by pre-segmenting the images, in order to find object surfaces hypotheses. After that RANSAC method is used to estimate the various surfaces, following this, the resultant meshes is then grouped into object hypotheses.

Cavallo [1] presents a method for segmentation and classification for grasping tasks. It uses fingertip contact force

sensors, aiding the segmentation and interpretation of motion, as it is based on a singular value decomposition of data that is gathered from the observation of humans. This observation consists on the 22 Degrees Of Freedom of the human hand joint angles, the hand pose with respect to a world frame, and fingertip contact forces. The interaction between the hand and object with the physical world is needed for the segmentation and motion interpretation.

Omi *et al.* [6] propose an approach for encountering the correspondence between objects in a before and after occlusion state, by detecting the grab and release of those objects by the same hand. Due to that, segmentation of object's body is also made via the detection of the grasping and release of the object. The hand is important because objects do not move by themselves, but are moved by human hands. This is useful in surveillance in public environments or tracking daily items in indoor environments, such as offices or bedrooms, when they go missing and need to be located.

Kehoe *et al.* [5] used cloud computing as a computation powerhouse and data storage. Due to the training data shared and aggregated by multiple robots it can provide a faster learning experience comparing with a single robot, serving as a way to address the novel object problem, with the addition that the cloud processing decreases the run time.

Saxena *et al.* [10], presents a solution for the problem of grasping new objects that the robot is perceiving for the first time. A learning algorithm is proposed that does not require a 3D view of the object, instead, the algorithm identifies a set of points in 2D images that correspond to a good area to grasp the object, and with that area, it then uses triangulation to obtain a 3D position to attempt the grasp.

Although there are various methods for segmentation and grasping, as referred in the previous paragraphs, there is no method that presents a perfect solution. We focus on improving the state-of-the-art algorithm by creating a simplification of the input point cloud that helps the grasp detection both running faster and more accurately, as discussed in the following section.

III. Proposed Method

The proposed method, works on the Robot Operating System (ROS) [7] Linux-Based middle-ware. ROS serves as an established interface for controlling various robotic hardware, from simple homemade robots to industrial ones. We also used the Point Cloud Library [9], a project used for 2D/3D image and point cloud processing. This library is used to aid with processing the RGB-D images provided by the existing hardware.

We based our proposed method in the improvement of a state-of-the-art grasp detection algorithm [4], which receives a point cloud with information of the world from a depth sensor, samples a pre-determined number of random points, n, and for each point generates possible grasp candidates and classifies them as a viable or not viable grasp. Our approach improves this algorithm by pre-processing the point cloud before it reaches the grasp detector algorithm, segmenting it two times

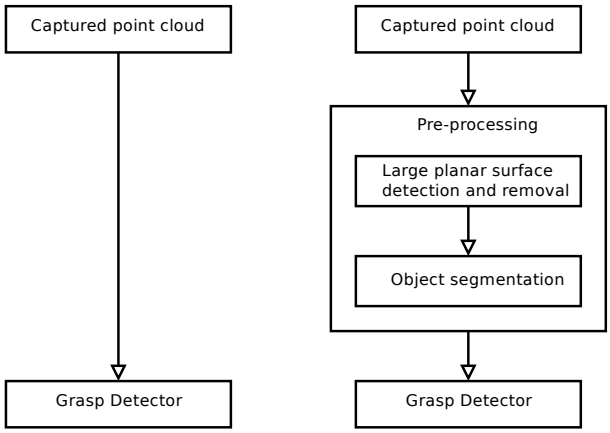

Figure 2. Left: original method. Right: proposed method.

and sending only to the grasp detector the point cloud region of the object to be grasped. This approach leads to a performance boost in terms of success rate and time consumption, since we can reduce the number of points to be processed and the algorithm will not spend time calculating grasps for points that do not contain objects. The reason behind this performance boost is that the point cloud contains less points and the points it contains are the ones that are important to consider.

In figure 2, we compare, in a simple way, the original method, in which the information of the world goes directly to the grasp detector, and the method we propose in this paper, where the information of the world passes through a segmentation process before the information goes to the grasp detector.

The first part of the method, receives the point cloud and finds the largest planar surfaces using RANSAC [2] and removes them. RANSAC, RANdom SAmple Consensus, is an iterative method that, using a set of data, estimates parameters of a mathematical model. It assumes that all data consists of inliers and outliers, where the inliers can be explained by the mathematical model and the outliers can not. The Point Cloud Library provides a wrapper for the implemented SAC model for detecting planes and RANSAC algorithm.

Just removing the planar surfaces in the point cloud still maintains other unwanted features present that need to be filtered out, and that is what the second part of the pre-processing addresses. The second pre-processing part uses min-cut based segmentation to isolate the closest object to the camera from the rest of the scene. The min-cut based segmentation makes a binary segmentation of the given input point cloud. It considers a given point and a radius around that point. This divides the cloud in two parts, a named foreground and background, meaning, the points that belong or are neighbors of the given point, and points that are not close enough to that point. The Point Cloud Library implementation of the min-cut based segmentation was used and that implementation is thoroughly explained in Golovinskiy and Funkhouser [3]. As a summary, the method builds a k-nearest neighbor graph, assuming a background and foreground constraints, where every point in

the beginning belongs in the background group, then, min-cut method searches around the initial given area for neighbor points and finds the min-cut for separating the background and foreground regions of the point cloud. After the separation of the closest object in the point cloud from the rest of the data, only this portion of the point cloud is sent to the grasp detector method that uses it to find possible grasps for the object.

This grasp method selects n points at random to check if they are possible points of grasp for the object, considering the provided gripper constraints.

The justification for removing the largest planar surface is that it usually represents a table top, the floor or a wall, and not the object to be grasped. By removing this large planar surface we are reducing the amount of data to be processed by the grasping algorithm. This can have two benefits: first, the potential grasps will not appear on the removed plane, increasing the probability that they are correctly placed on the object to be grasped. Second, since the potential grasps are more likely to be correct, the algorithm can work with a smaller number of attempts, to achieve the same grasping success rate, but using less time to do it. This first simplification of the point cloud enables us to improve the correct prediction of the object in the second part of the pre-processing step, removing possible errors and the time consumption of the algorithm.

Figure 1 shows an example of a successful gasp detection, first without pre-processing, with the first pre-processing, and both steps, respectively. It is one of the example objects used in our data set, from a total of eight, as shown in figure 3. The grasps are represented as a blue parallel jaw gripper.

IV. Experiments

A. Setup

We conducted 4 experiments, each one using the same setup. The setup used was a desktop computer with the following specifications:

- CPU: Intel(R) Core™ i7-2600 CPU @ 3.40GHz
- GPU: GeForce GTS 450
- RAM: 12 GB
- OS: Ubuntu 14.04.03 LTS with ROS Indigo
- Microsoft XBox 360 Kinect RGBD-Camera/sensor

To ensure every experiment was conducted with the same characteristics, we created a database using a set of 8 objects, which can be seen in figure 3. For each object, 20 point clouds were captured with a Kinect in our lab, each point cloud simulating a possible scenario where a robot needs to grasp an object on a table. Between each capture of a point cloud, the position and the orientation of the object were changed so that each point cloud is different from the others. After the point clouds were captured, they were segmented in two ways: the first one so that we can test the performance of the method with point clouds that were segmented in order to remove the largest planar surfaces, and the other one by segmenting the point clouds removing the largest planar surface (the method can be easly changed to remove all the surfaces that are bigger than a threshold) and by cropping the image around the closest object, considerably reducing the number of points to be processed.

Figure 3. Set of 8 objects used to create the database.

The first pre-processing step, removing the largest planar surface from the point clouds, takes on average 0.21 seconds, and both pre-processing steps took, on average, 0.33 seconds (0.21 of the first step plus 0.12 seconds for the second step that segments the closest object in the point cloud).

With the original and segmented point clouds, we created a database that consists in 480 point clouds (20 original, 20 with only the first pre-processing step and 20 with both steps, per object), which is represented in figure 4, in which the first line represents one original image per object, second line represents one image per object after the first pre-processing step and the third and last line represents a point cloud after the application of both pre-processing steps, for each object.

Each experiment had the same number of trials, 20 per object, using either the original images, the segmented images or the segmented and cropped images. In these experiments, the method [4] identifies if a grasp is viable or not. A trial in the experiments is classified as successful if the algorithm detects at least one viable grasp, otherwise, the trial is classified as a failure.

B. Experiment 1

In order to have a baseline comparison, we've evaluated the performance of the original method with the original point clouds of the objects. In this experiment we set the number of sampling points in the grasp generator, n, as 5000, this means that the grasp detector will sample 5000 points and determine grasps for those points.

The results for this experience can be seen in table I. The average run time was 2.63 seconds, which means that for each trial, the original method calculated the grasp candidates and classified them in 2.63 seconds, with a success rate of 45.63%. This experiment shows that the original method is capable of detecting a viable grasp in 73 of the 160 trials for the database we created.

C. Experiment 2

In the second experiment we tested the performance of the grasp detector using point clouds with the largest planar

Figure 4. Examples of the database created. The first line represents one original point cloud per object, second line represents one point cloud per per object after the application of the first step of the pre-processing and the third line represents a point cloud of each object after the application of both pre-processing steps.

Table I
RESULTS OF THE EXPERIMENT USING NON-PROCESSED POINT CLOUDS WITH n=5000.

	Success	Failure	Average run time (s)	Success Rate (%)
Blue Canteen	19	1	2.65	95.0
Cardboard Box	11	9	2.69	55.0
Cardboard Cup	9	11	2.63	45.0
Clay Cup	11	9	2.66	55.0
Gel Tube	11	9	2.58	55.0
Headphones	6	14	2.64	30.0
Paper Holder	1	19	2.49	5.0
Water Bottle	5	15	2.69	25.0
Overall	73	87	2.63	45.63%

Table II
RESULTS OF THE EXPERIMENT USING POINT CLOUDS AFTER LARGE PLANAR SURFACE REMOVAL, WITH n=5000.

	Success	Failure	Average run time (s)	Success Rate (%)
Blue Canteen	20	0	3.09	100.0
Cardboard Box	12	8	3.28	60.0
Cardboard Cup	19	1	3.08	95.0
Clay Cup	17	3	3.08	85.0
Gel Tube	18	2	3.17	90.0
Headphones	12	8	3.17	60.0
Paper Holder	8	12	2.94	40.0
Water Bottle	9	11	2.90	45.0
Overall	115	45	3.09	71.88%

surface removed and with the same number of sampling points, n, as the previous experiment. This experiment achieved a success rate of 71.88% and an average run time of 3.09 seconds. These results can be seen in table II. The total mean run time cost for this experiment is 3.3 seconds, which represents the 3.09 of the grasp detector run time plus the 0.21 seconds spent removing the large planar surfaces from the point clouds. In this experiment, the tested method was capable of segmenting a point cloud, generate grasp candidates and classifying them with success in 115 of the 160 trials in an average run time of 3.3 seconds.

D. Experiment 3

In this experiment the method tested was the same as in the second experiment, but with less sampling points. The number of points to be sampled, n, in the point cloud that represents the world was cut in half, from 5000 points to 2500. This experiment reduced the average run time of creating grasp candidates and classifying them to 1.62 seconds and achieved an success rate of 43.13%, as described in table III. The total average run time is 1.83 seconds (1.62 seconds of the grasp detection run time and 0.21 seconds for large planar surface removal) and the grasp detector was able to detect viable grasps in 69 out of 160 trials.

Table III
RESULTS OF THE EXPERIMENT USING POINT CLOUDS AFTER LARGE PLANAR SURFACE REMOVAL, WITH n=2500.

	Success	Failure	Average run time (s)	Success Rate (%)
Blue Canteen	20	0	1.71	100.0
Cardboard Box	6	14	1.74	30.0
Cardboard Cup	10	10	1.60	50.0
Clay Cup	12	8	1.58	60.0
Gel Tube	12	8	1.64	60.0
Headphones	4	16	1.58	20.0
Paper Holder	1	19	1.51	5.0
Water Bottle	4	16	1.58	20.0
Overall	69	91	1.62	43.13%

E. Experiment 4

The fourth and final experiment was conducted using point clouds after the application of both pre-processing steps. In this case, the number of sampling points to be selected from the point clouds was 2500. Table IV shows that this experiment

had an overall success rate of 90% and an average run time of 1.73 seconds, meaning that the overall run time is 2.06 seconds (1.73 seconds from the run time, 0.21 from the first pre-processing step and 0.12 seconds for the second step). The proposed method was capable of detecting a viable grasp in 144 of 160 trials.

Table IV
RESULTS OF THE EXPERIMENT USING POINT CLOUDS AFTER THE APPLICATION OF BOTH PRE-PROCESSING STEPS, WITH n=2500.

	Success	Failure	Average run time (s)	Success Rate (%)
Blue Canteen	20	0	2.06	100.0
Cardboard Box	17	3	2.28	85.0
Cardboard Cup	18	2	1.71	90.0
Clay Cup	17	3	2.17	85.0
Gel Tube	18	2	1.28	90.0
Headphones	19	1	2.60	95.0
Paper Holder	15	5	0.56	75.0
Water Bottle	20	0	1.20	100.0
Overall	144	16	1.73	90.0%

V. RESULTS

In table V, the results of the four experiments are condensed, presenting the corresponding mean values.

Table V
MEAN VALUES OF EACH EXPERIMENT. PRE-PROCESS 1 CONSISTS ON USING ONLY THE FIRST PRE-PROCESSING STEP (PLANAR SURFACES REMOVAL), PRE-PROCESSING 2 CONSISTS ON USING BOTH PRE-PROCESSING STEPS (THE PROPOSED METHOD IN THIS PAPER).

	n	Total Run Time (s)	Success Rate (%)
Baseline	5000	2.63	45.63
Pre-process 1	5000	3.30	71.88
Pre-process 1	2500	1.83	43.13
Pre-process 2	2500	2.06	90.00

This set of results is explained with the way how the grasping method works. Since the grasping method has a smaller area of search, the method takes less time, even with the added processing and provides more accurate results due to the fact that the added processing time results in a smaller search field for the grasping method to work. And since the smaller search field is an area around the object that we want to grasp, the success rate is higher, due to the fact that the correct grasps are in the object itself, instead of a random place of the provided point cloud.

With this experiments, we show how the proposed method can improve both in time and success rate the original method, because as explained, it uses less points on the point cloud, achieving improved results, since those points are the ones that matter, in this case, points that contain one object that is to be grasped.

VI. CONCLUSIONS

Grasping novel objects in unknown positions is a task that is very complex and challenging. Normally, the methods that exist, take several seconds to process, each input point cloud, without guarantee of success which can be critical in a real-life scenario.

In this paper we present a method that pre-processes an input point cloud and provides the result to a grasping method. Our proposal can be used to determine a successful grasp faster and with a higher success rate, than the baseline approach. It's capable of significantly reducing the time consumption on average, from 2.6 seconds to 2.0 seconds per point cloud (23% improvement) and improving the success rate by a large margin, from 45% to 90% (100% improvement). Our code and the database used in this paper are freely available https://github.com/VascoLopes/Segmentation-Database-and-Code.

For future work we will try to reduce the run time of the whole process, improve the grasp detector so it is capable of, in less time, achieving same or better results and expand our database to more objects, which will lead to more detailed experiments.

ACKNOWLEDGMENTS

This work was supported by National Funding from the FCT - Fundação para a Ciência e a Tecnologia, through project UID/EEA/50008/2013.

REFERENCES

[1] A. Cavallo and P. Falco. Online segmentation and classification of manipulation actions from the observation of kinetostatic data. *IEEE Transactions on Human-Machine Systems*, 44(2):256–269, April 2014.

[2] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[3] Aleksey Golovinskiy and Thomas Funkhouser. Min-cut based segmentation of point clouds. In *IEEE Workshop on Search in 3D and Video (S3DV) at ICCV*, September 2009.

[4] Marcus Gualtieri, Andreas ten Pas, Kate Saenko, and Robert Platt Jr. High precision grasp pose detection in dense clutter. *CoRR*, abs/1603.01564, 2016.

[5] B. Kehoe, A. Matsukawa, S. Candido, J. Kuffner, and K. Goldberg. Cloud-based robot grasping with the google object recognition engine. In *2013 IEEE International Conference on Robotics and Automation*, pages 4263–4270, May 2013.

[6] T. Omi, K. Kakusho, M. Iiyama, and S. Nishiguchi. Segmentation and tracking of object when grasped and moved within living spaces. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 3147–3152, Oct 2017.

[7] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully B. Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. ROS: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[8] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich, and M. Vincze. Segmentation of unknown objects in indoor environments. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4791–4796, Oct 2012.

[9] R.B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[10] Ashutosh Saxena, Justin Driemeyer, and Andrew Y. Ng. Robotic grasping of novel objects using vision. *The International Journal of Robotics Research*, 27(2):157–173, 2008.

[11] K. M. Varadarajan and M. Vincze. Object part segmentation and classification in range images for grasping. In *2011 15th International Conference on Advanced Robotics (ICAR)*, pages 21–27, June 2011.