Msc. Thesis

Using Retinanet to determine local graspability for a suction actuator

F.A.R. van Tilburg

TUDelft

FIZYR

Msc. Thesis

Using Retinanet to determine local graspability for a suction actuator

by

F.A.R. van Tilburg

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday August 26, 2021 at 14:00 AM.

Student number:	4166159	
Project duration:	May 1, 2018 – January 14, 2019	
Thesis committee:	Prof. dr. ir. M. Wisse,	TU Delft, supervisor
	Ass. Prof. dr. J. F. P. Kooij,	TU Delft
	Ir. M. Imre,	TU Delft
	Ir. V. Carpani,	Fizyr

This thesis is confidential and cannot be made public until August 31, 2021.

TU Delft FIZYR

Preface

This report describes the research I've completed as part of my MSc. study at the TU Delft. It was commissioned by Fizyr, a vision software company located in Delft working on automated picking of unknown objects, with the goal of improving upon the existing methods for grasping unknown objects.

The findings of the completed research have been compiled in a paper which can be found as Part 1 of this report. Appendices have been added as Part 2 to elaborate on the design choices made during the research. Beginning with the implementation and evaluation of the state of the art Dexnet grasp pipeline described in Appendix A. Followed by the creation of an annotation tool to obtain the data to train a newly proposed local grasp evaluator neural network and the training of this network described in Appendices B and C. Finishing with the experimental evaluation of the newly proposed grasp pipeline described in Appendices D-F.

I couldn't have completed this research on my own. The first person I want to mention and thank is Wenjie Duan for his collaboration on the implementation and experimental evaluation of the Dexnet grasp pipeline. Together this went faster and better than I could have done by myself. Secondly I want to thank my supervisors from the university Carlos Corbato and Martijn Wisse for their guidance and feedback. Finally I want to thank all the employees from Fizyr for their assistance and advice during the research. A few of them I wish to name specifically: Enrico Liscio for his guidance as supervisor within the company; Bas van Mil for his assistance related to the physical experimental setup; and finally Hans Gaiser, Maarten de Vries and Valerio Carpani for their assistance and suggestions related to the software.

F.A.R. van Tilburg
Delft, August 2021

Contents

I

Research paper

Using Retinanet to determine local graspability for a suction actuator

1st Floris van Tilburg
dept. Biomechanical Design
TU Delft

2nd Wenjie Duan
dept. Biomechanical Design
TU Delft

3rd Enrico Liscio
Fizyr

4rd Carlos Corbato
dept. Cognitive Robotics
TU Delft

5rd Martijn Wisse
dept. Cognitive Robotics
TU Delft

***Abstract*—Suction based robotic actuators have potential for the bin-picking industry, but are currently not usable due the needed speed, accuracy and ability to handle novel and adversarial objects. An evaluation of the state of the art grasp pipeline developed by Mahler *et al.* [1] for detecting grasps on novel objects lead us to split the problem of robotic grasp generation into a global and a local component. The state of the art solution had the ability to fill the role of global evaluator leaving a local evaluator to be developed. This local grasp evaluator was obtained by training a neural network on a suction based grasp dataset, which was created using a newly developed annotation tool. The proposed grasp pipeline obtained by combining these two showed a 95.27% pick success rate for a random setup and a 90.28% success rate for solely novel objects.**

***Index Terms*—novel objects, robotic bin picking, suction grasping**

I. Introduction

Automation in warehousing has come a long way with mobile storage solutions, but there is still a final problem to solve: the task of bin-picking still has to be completed by a human. The action that has to be executed fast and accurately is still too difficult for robotic systems due to the chaotic environment caused by the immense amount of unknown objects and all their possible orientations. However, solving this problem would allow robotic systems to function in a large variety of situations making it an important topic of research. Therefore, work is being done on this problem by researchers as well as companies such as the one behind this research Fizyr; a vision software company specializing in automated picking.

Grasp pipelines nowadays make use of neural networks to determine possible grasp areas followed by heuristics to determine the best grasp point, allowing for sub-second calculation times. A new method proposed by Mahler *et al.* [1] suggests to flip this around by using a heuristic generation of possible grasp areas followed by a neural network, trained on the wrench resistance based Dexnet dataset, to evaluate which of the generated grasp points will most likely result in a successful grasp attempt. Especially exciting about this new Dexnet based grasp pipeline is that, while comparative slower, it achieves a higher accuracy on novel adversarial objects compared to the general approach.

In this paper we decided to build further on the high picking accuracy on novel adversarial objects achieved by this Dexnet based grasp pipeline by evaluating its shortcomings and suggesting a potential improvement: the addition of a second neural network to work sequentially with the existing Dexnet based one. This new neural network would function as local grasp evaluator, i.e. can a suction seal be achieved, while the Dexnet based neural network, which is trained on wrench resistance, would function as global evaluator, i.e. can a graspable point result in a successful pick with the forces and torques acting on it. The data required for the training of the proposed local evaluator network is obtained using a newly developed annotation tool as presently no such tool exists for the creation of a suction based grasp database.

This paper makes four contributions:

1) An evaluation of the state of the art Dexnet based grasp pipeline [1].
2) The creation of an annotation tool for suction based grasps.
3) A dataset of suction based grasps that includes additional information for alternative applications.
4) A new hybrid grasp pipeline with experimentally obtained data on picking accuracy and speed.

II. Related work

A decent amount of research exists for robotic grasping, but unfortunately most of it is focused on using pinch grippers leaving the work on robotic suction grasping lacking [5]. The research that has been done for suction grasping can generally be divided in two types: analytical and empirical. Analytical approaches generate grasps using models and/or constraints such as the gripper models by Domae *et al.* [6] and the surface models by Vona and Kanoulas [7]. Such methods can potentially work for adversarial objects, but are restricted by strict heuristic thresholds and can take multiple seconds to determine a grasp pose. Empirical approaches on the other hand make use of a knowledge base that has been acquired previously such as feature detection by Saxena *et al.* [8] and the use of deep learning such as done by Zeng *et al.* [9]. The method of using neural networks has been the most popular in recent years due to it's sub-second calculation

times. The recent work by Mahler *et al.* [1] shows especially good results. Achieving close to a 98% success rate for basic objects, defined as prismatic solid shapes in [1], and a 58% success rate for adversarial ones, defined as complex geometric shapes in [1], which could be pushed to 81% at the cost of the success rate on simpler objects. This high accuracy for the most difficult type of object to pick in addition to their wrench force based approach made us decide to evaluate the created grasp pipeline to figure out what would be the best direction to expand on it.

III. Problem Statement

We want to use a single-view point cloud or depth image as input to get a point on a novel object where it can be grasped in such a way that suction can be achieved and the object can be lifted and moved without being dropped. Only a single, top-down, viewpoint is desired compared to multiple as the latter would likely increase the time between potential picks due to having to record and parse multiple images per grasp attempt.

For this task we use the following assumptions:

- The evaluated objects consists of a material that can be grasped using a suction/vacuum based actuator, so no porous materials.
- The objects can be unknown to the grasp planner.
- The point cloud or depth image is taken by a single 3D camera mounted above the workspace.
- The point cloud is segmented so the grasp planner only sees a singular object when multiple objects are present in the workspace.
- The objects pose is stable and can be grasped in its current orientation without shifting.
- A suction cup with a diameter of 20 mm is used.

The objective of this research is to develop a grasp pipeline that is able to find a grasp that maximizes the likelihood of a successful robotic pick for novel objects irrelevant of their structural difficulty.

IV. Dexnet 3 evaluation

Because of the great results shown by Mahler *et al.* [1] compared to the rest of the scientific work on robotic suction grasping we've made the decision to use their Dexnet based grasp pipeline as a starting point for this research. In order to have a better idea of the direction to take this research we started with an experimental evaluation to find the grasp pipelines' limitations and possible areas for improvement.

The Dexnet based grasp pipeline consists of two parts that are combined iteratively as shown by Figure 1: A uniform sampler suggesting possible grasp locations on the object and a neural network that has been trained on wrench resistance using the analytical model shown by Figure 2.

The sampled points are evaluated by the neural network after which the point with the highest score is returned to the uniform sampler to sample points around this point. This iterative process is repeated three times after which the grasp point with the highest score given by the neural network is selected as executable grasp.

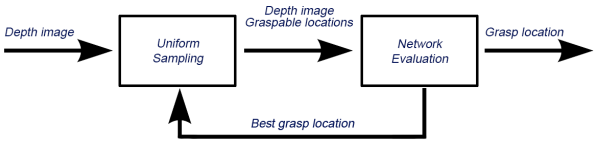


Fig. 1. Schematic of the iterative Dexnet based grasp pipeline.

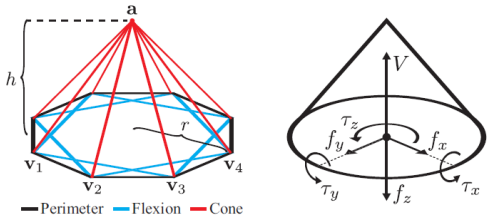


Fig. 2. The analytical model developed by Mahler *et al.* [1] to create the Dexnet 3 dataset from simulated objects. It determines both the forces and torques around the grasp point as well as a basic contact seal evaluation.

A. Setup

A 1280x1024 2D image and pixel aligned point cloud are recorded with an IDS NXT camera and a N35 Ensenso stereo camera respectively, both placed 1.5 meter above the workspace. The neural network used a NVIDIA GTX 1080 Ti video card to determine the graspability of the grasp locations. After which the best grasps were executed using an UR5 robotic arm equipped with a 20 mm ⌀double bellow silicon suction cup. A visualization of the experimental setup can be seen in Figure 3. Grasp attempts were marked as a success if the object was picked up and moved without being dropped.

During the evaluation of the Dexnet grasp pipeline single objects were placed in the workspace and detected by removing the background from the point cloud leaving a segmented point cloud containing only the object, which was then evaluated using the grasp pipeline.

B. Results

Forty-one different objects, ranging from basic to adversarial in shape as defined by Mahler *et all.* [1] were evaluated in three to five different orientations, depending on the shape of the object, leading to 177 grasp attempts of which 110 were successful. This success rate of 62% is lower than reported by Mahler *et al.* as only three of the evaluated objects could be deemed adversarial instead of basic or typical. Some of the errors were the result of grasp attempt on top of object surface structures, such as ridges, that weren't visible in the point cloud, but not all of them. On the contrary, a large amount of errors were caused by the generated grasp being located on top of edges or holes within the objects surface that were visible in the point cloud and should therefore have been evaded. A second observation was the existence of grasp attempts near

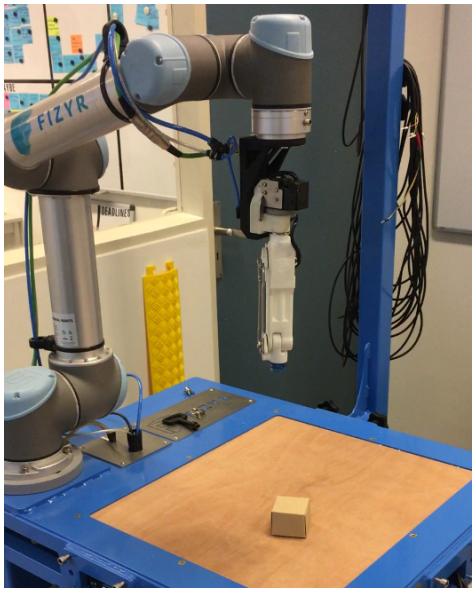


Fig. 3. The experimental setup used for all experiments. A UR 5 robotic arm equiped with Fizyrs swivel gripper, an end effector with two additional degrees of freedom to increase the arms range of motion and the ability to disconnect from the robotic arm to prevent potential damage in case a grasp point is wrong.

the edge of objects while those objects could easily be grasped closer to the center. This was a surprise due to the fact that the network was primarily trained on the forces and torques acting on the grasp location, suggesting it would be more likely to generate grasps near the center or mass.

C. Conclusion

The results from the evaluation suggest that the Dexnet based grasp pipeline has difficulty determining whether a suction seal between gripper and object surface could be achieved. Not too surprising, since the neural network has been predominantly trained on wrench resistance. This, combined with a potential bad initial sampling, is possibly also the cause of the grasp attempts near the edges of the objects. The training on wrench resistance does not matter if the few initial samples near the center of mass of an object are deemed unlikely to result in a successful pick due to noise in the point cloud, pushing the iterative process towards the edge of the object.

The observed issues lead to the decision to add another neural network in sequence with the existing one, resulting in the new grasp pipeline shown by Figure 4. This new neural network will determine which points on the objects surface are locally graspable, i.e. where on the objects surface can a seal be formed with the suction cup, before using its output as the input for the Dexnet based neural network. Then assuming that all these inputs are graspable it would follow that the only difference in the output of the Dexnet based network would be the result of potential forces and torques acting on the grasp points. So the network should be able to infer whether a grasp point is globally graspable, i.e. the suction seal won't be broken as a result of the forces caused by the location of the grasp point in relation to the objects center of mass.

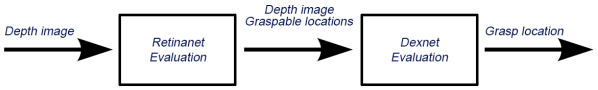


Fig. 4. Schematic of the newly proposed grasp pipeline combining a new local evaluator network with the existing Dexnet based neural network.

V. Local Graspability Network

Now that we decided how to enhance the Dexnet based grasp pipeline we will have to decide how to implement this new local graspability evaluation network. We want to detect areas in a depth image that correspond to the surface area of the used suction cup and determine whether the detected areas will lead to a locally successful grasp attempt, i.e. can suction be achieved. This means that we would only need to detect bounding boxes of potential grasp areas centered on the specific point the robot would be instructed to move towards during the grasp action. To determine what network architecture would likely give the best results we decided to use the AP_S metric as defined by the Common Objects in Context (COCO) dataset [10]. This metric is defined as the average precision for the detection of small objects, which are defined as detections with an area of $< 32^2$ pixels, which is what the area corresponding to a suction cup of 20mm ⌀would fall under for our camera setup. The network architecture Retinanet as presented by Lin *et al.* [3] was finally selected for the local graspability network as it had the best score for this metric with an average precision of $AP_S = 24.1$.

VI. Data Acquisition

When training a new neural network for a new purpose the most important thing is how to obtain the trainings data. That is because the data used to train a neural network defines its use. For our purpose we require data in the form of depth images, which can be obtained from point clouds, annotated with object location where a seal with a suction actuator can or can't be achieved. This data could be obtained in one of three ways: manual annotations, computer simulations and physical (robotic) simulations. The decision was made to use manual annotations for the following reasons:

- It is relatively easy to create a tool to obtain the annotations.
- It is faster, but less accurate than physical simulations and slower, but more accurate than computer simulations making it a balanced solution.
- It gives us the option to add extra information to the dataset making it possible to tune the dataset afterwards and possibly making it usable for both the global and local grasp evaluations.

A. Annotation tool

The annotation tool shown by Figure 5 was created as no tool capable of creating suction grasp annotations existed. It uniformly generates possible grasp locations on a given recording and shows these to the user sequentially. The grasp location is displayed in both a 2D color image as well as in a point cloud because some things that lead to a failed grasp can only be observed in only one of these. The user can then, for each proposed location, say whether it is a possible local and global valid grasp location, as well as specify additional information. See Appendix B for more information on the design of the annotation tool.

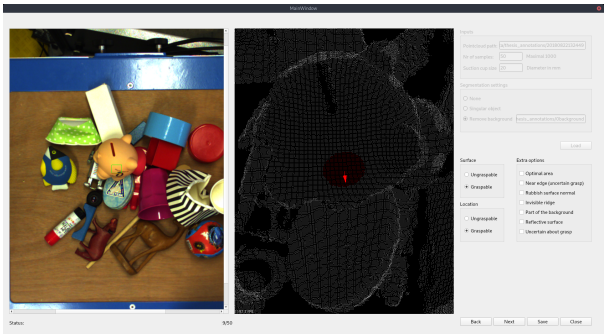


Fig. 5. The annotation tool designed to create the suction based grasp dataset showing a potential grasp point and the corresponding area where a suction cup would be placed in both 2D and 3D.

B. The neural network

The described annotation tool was used with a speed of approximately 400 annotation/hour to create a dataset of 20293 suction grasp annotations over the duration of two weeks. Fizyrs Keras implementation of RetinaNet [4] was used to obtain the trained neural network. Initially after training a low average precision of approximately 0.1 was observed suggesting the network potentially wouldn't perform very well as local grasp evaluator. However, a visual evaluation, as shown by Figure 6, showed this likely wasn't the case. Which meant the low precision could possibly be a result of the similarity between close object surface locations.

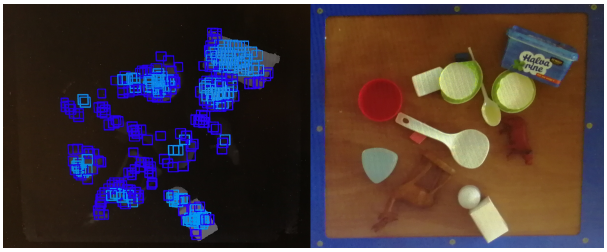


Fig. 6. Depth image superposed with detected possible grasp locations given by the trained neural network, with the lighter colored boxes corresponding to confidence scores above 0.5.

VII. Experiments

After obtaining the dataset and the trained neural network the newly proposed hybrid grasp pipeline could be assembled and evaluated to see how it compares to the original Dexnet based grasp pipeline. This evaluation was split into two parts: An initial experiment done to evaluate the correlation between the confidence scores q given by the local evaluator network and the likelihood of it resulting in a seal between suction cup and object surface. This experiment was used to make sure that the generated grasp areas would have a high likelihood of success in spite of the low average precision observed during the training of the network. The results were then used to determine which grasps to pass on to the global evaluation during the second experiment which was a comparison of the pick success rate between the original iterative Dexnet based grasp pipeline and the proposed hybrid grasp pipeline.

The setup used for the experiments is the same as the one used during the Dexnet grasp pipeline evaluation as described in IV-A.

A. Experiment 1.

During the first experiment between five and twenty objects were randomly placed in the workspace resulting in piles with objects on top of each other. The local evaluator network was used on the resulting clutter to generate possible grasp locations. From those generated possible grasp locations a single one was randomly chosen to be executed by the robotic actuator. Whether a suction seal could be formed at that object location was then marked down together with the corresponding confidence score q. A total of 605 such grasp attempts were executed of which the results are shown in Figure 7.

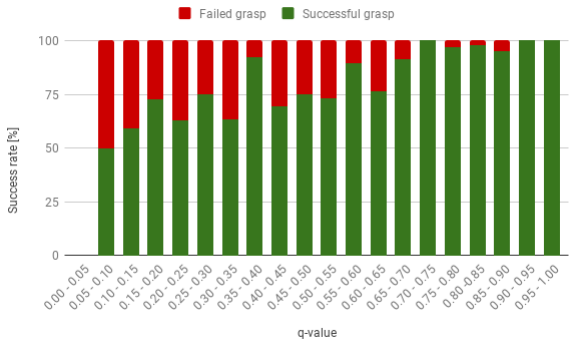


Fig. 7. Result from the experiment evaluating the correlation between the confidence score output by the neural network and the likelihood of a successful grasp.

The grasps with a confidence score of $q = 0.7$ and higher show an almost 100% success rate at suction seal formation and can therefore be send to a global grasp evaluator with a high probability of resulting in a successful pick action. But even confidence scores as low as $q = 0.45$ show a success rate of 75 % which suggest a successful grasp can still be generated even if the network finds no grasp locations with a

confidence score greater than 0.7. An 85 % success rate can also be observed for the q-values of $q = 0.35 - 0.40$, however, this is likely an outlier resulting from a limited amount of such grasps being attempted. Most of the observed grasps had q-values near the upper and lower limits, with the intervals between 0.25 and 0.65 having 20 or less observed attempts each.

B. Experiment 2.

With the results of the first experiment the proposed neural network was ready to act as a local grasp evaluator, allowing the assembly of the proposed hybrid grasp pipeline that can be compared with the original Dexnet based grasp pipeline. The comparison evaluation was done in cycles of 10 items which were placed in a random manner in the workspace as shown by Figure 8.

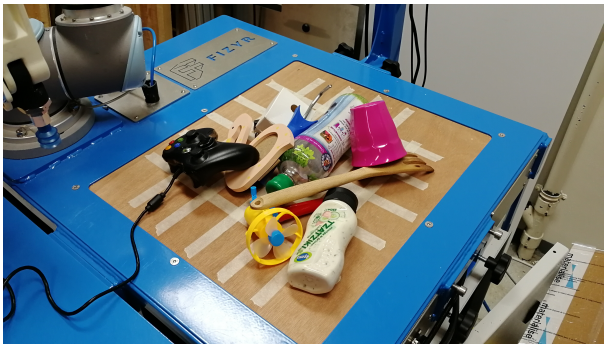


Fig. 8. Setup for the second experiment with a pile placed in a random manner.

Each cycle was repeated by recreating the placement of the objects for evaluation by both grasp pipelines. The objects were segmented manually because no automatic segmentation tool for the novel objects was available and it allowed us to enforce a picking order between evaluations, which improves the comparison between grasp pipelines. Additionally a limit of three pick attempts was specified for each object in case of failed grasps. This was done to reduce the likelihood of failures being caused by outside influences. The result of this experiment is shown in Table I including the amount of needed grasp attempts to achieve a successful pick.

	Dexnet pipeline	Hybrid pipeline
Successful pick	90.54 %	95.27 %
1st pick attempt	83.78 %	89.86 %
2nd pick attempt	4.73 %	3.38 %
3rd pick attempt	2.03 %	2.03 %
Failed to pick	9.46 %	4.73 %

TABLE I
EXPERIMENT 2 RESULTS.

The experiment contained both known and unknown objects for which grasps had to be generated as this more closely matches an industrial application. However, we're most interested in the performance on novel objects as that better describes the potential capability of the grasp pipelines. The performance of the grasp attempts on novel objects is shown in Table II.

	Dexnet pipeline	Hybrid pipeline
Successful pick	81.94 %	90.28 %
1st pick attempt	72.22 %	80.56 %
2nd pick attempt	6.94 %	5.56 %
3rd pick attempt	2.78 %	4.17 %
Failed to pick	18.06 %	9.72 %

TABLE II
EXPERIMENT 2 RESULTS FOR UNKNOWN OBJECTS.

From the results in Tables I and II we can conclude that the newly proposed grasp pipeline performs better than the original Dexnet based grasp pipeline. This is in addition to an observed decrease in calculation time. The original Dexnet grasp pipeline took an average of 1.56s to generate grasps during the experiment, of which 1.07s were taken up by the neural network. The proposed hybrid grasp pipeline only took 1.12s, of which 0.066s were used by the local evaluator network, 0.49s by the original Dexnet network and the leftover 0.564s for data handling and grasp pose calculation.

VIII. Discussion and Future Work

We evaluated the current state of the art for novel object bin picking for possible avenues of improvement. This lead us to propose to split grasp generation in a global and local component. The latter for which we presented a method by using a neural network with the RetinaNet architecture trained on a local graspability dataset created using a newly developed tool. We finally experimentally evaluated the resulting grasp pipeline showing a faster grasp generation with a higher pick success rate compared to the original Dexnet based grasp pipeline.

Our proposal for the split of grasp generation in a global and a local component leaves us with a lot of possible avenues for future research. The most obvious being the development of one such global evaluator to work in tandem with the proposed local evaluator as replacement for the Dexnet based neural network. But, even the proposed local evaluator gives us options to explore. The first option would be to simply increase the amount of data used during the training in order to improve the results. But, the input of the neural network itself could also be enhanced. Using the RGB image in addition to the depth image could possibly allow it to see issues that can't be observed using only the latter. Or alternatively the input of the neural network could be enhanced to accept the sparse xy-coordinates of a point cloud instead of an image aligned depth map so as to be less dependent on the used camera and setup.

IX. Acknowledgments

This research was performed at Fizyr, a computer vision company located in Delft, the Netherlands. We'd like to thank all our colleagues for their suggestions and assistance with the implementation and execution of this research.

REFERENCES

[1] J. Mahler, M. Matl, X. Liu, A. Li, D. Gealy and K. Goldberg, "Dex-Net 3.0: Computing Robust Robot Suction Grasp Targets in Point Clouds using a New Analytic Model and Deep Learning," 2017.

[2] D. Kanoulas, J. Lee, D. G. Caldwell and N. G. Tsagarakis, "Visual grasp affordance localization in point clouds using curved contact patches," International Journal of Humanoid Robotics, vol. 14, December 2016.

[3] T. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal Loss for Dense Object Detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 2, pp. 318-327, 1 February 2020.

[4] H. Gaiser, M. de Vries, V. Lacatusu, V. Carpani, A. Williamson, E. Liscio, … D. Dowling. (2018, October 17). fizyr/keras-retinanet: 0.5.0 (Version 0.5.0). Zenodo. http://doi.org/10.5281/zenodo.1464720

[5] A. J. Valencia, R. M. Idrovo, A. D. Sappa, D. P. Guingla and D. Ochoa, "A 3D vision based approach for optimal grasp of vacuum grippers," 2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM), Donostia-San Sebastian, 2017, pp. 1-6, doi: 10.1109/ECMSM.2017.7945886.

[6] Y. Domae, H. Okuda, Y. Taguchi, K. Sumi and T. Hirai, 11Fast graspability evaluation on single depth maps for bin picking with general grippers," 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, 2014, pp. 1997-2004, doi: 10.1109/ICRA.2014.6907124.

[7] M. Vona and D. Kanoulas, "Curved surface contact patches with quantified uncertainty," 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, 2011, pp. 1439-1446, doi: 10.1109/IROS.2011.6094990.

[8] A. Saxena, J. Driemeyer, A. Y. Ng, "Robotic Grasping of Novel Objects using Vision," I. J. Robotic Res.. 27. 157-173. 10.1177/0278364907087172. (2008).

[9] A. Zeng et al., "Robotic Pick-and-Place of Novel Objects in Clutter with Multi-Affordance Grasping and Cross-Domain Image Matching," 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, 2018, pp. 3750-3757, doi: 10.1109/ICRA.2018.8461044.

[10] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft COCO: common objects in context," CoRR, abs/1405.0312, 2014.

II

Appendices

A

Dexnet 3.0 evaluation

A.1. Reason for the Dexnet evaluation

Only a limited amount of research into suction cup grasp planning exists, with a significant percentage being developed for applications such as the amazon picking challenge. Most are focused on speed achieving decent performance with calculation times of 200 miliseconds, but the difficulty of the to be grasped objects is relatively low. Allowing for the use of surface maps mostly based on flat surface areas. The development of the Dexnet based grasp pipeline of Mahler et all [3] is actually quite different. Instead of focusing on speed they focus on the functionality for difficult objects. They proposed an object designation of basic, typical and adversarial objects which had the following definitions:

- *Basic* objects are the most basic primitive shapes like cubes and cylinders.
- *Typical* objects are general objects with varied geometry, but containing multiple graspable areas.
- *Adversarial* objects are the most difficult to grasp, having limited graspable surface locations and very irregular geometry.

Their Dexnet based grasping policy was able to achieve a 98% success rate for basic objects and a 58% success rate for adversarial objects which could be pushed to an astonishing 81% at the cost of simpler objects.

For industrial applications the success rate of the grasp algorithm can be said to be more important than speed as the necessary speed can be achieved in alternative ways. For example by using multiple bin-pick stations simultaneously. Additionally the possible system speed is limited by the slowest member in the total pipeline anyway. Bin-pick speeds of 200 ms are pointless in cases where the acquisition of a new bin takes 500ms as that would leave the robotic arm waiting for 300 ms. And in the case of a failed grasp time is lost by needing multiple grasp attempts or the help of a human operator. Therefore the decision was made to focus on grasp success rate by continuing the work on Dexnet by evaluating its issues and limitations and proposing a solution to improve on these.

A.2. Dexnet based grasp pipeline

In order to accurately determine the issues and propose useful improvements the Dexnet based grasp pipeline has to be understood. The grasp pipeline consists of two separate interacting parts: a uniform sampler that suggests possible grasp locations by taking random samples over an objects surface and a deep convolutional neural network, trained on the Dexnet 3 dataset, that evaluates the proposed grasp locations. The best scoring of the proposed grasp locations is then returned to the sampler to iteratively find the optimal grasp location by sampling possible grasp locations around this point.

The most important part in this pipeline responsible for the high reported success rate is the neural network shown by Figure A.1. For each grasp location a new input is created from the original depth image by taking a window, corresponding to approximately 10 cm, centered on the proposed location. This image is then fed to the network together with a calculated surface normal to get a grasp score that can be compared with other possible grasps.

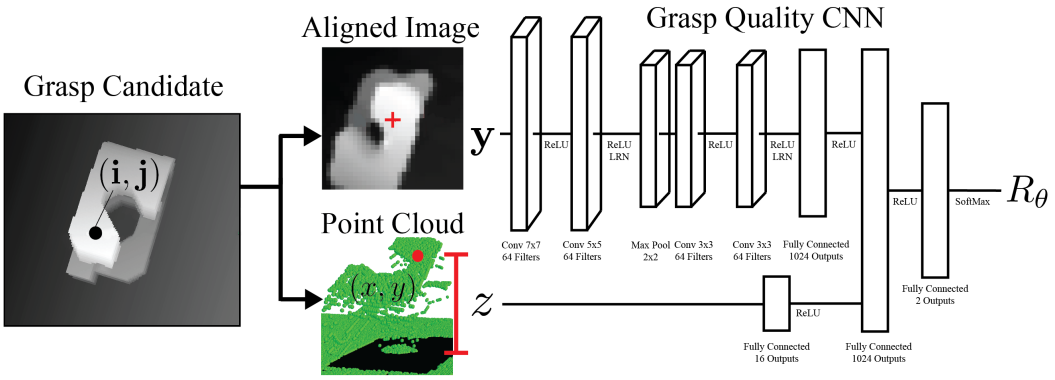


Figure A.1: The neural network used by the Dexnet based grasp pipeline.

This neural network has been trained on the Dexnet 3 dataset, which contains 2.8 million example grasp locations and grasp robustness labels. These labels have been obtained by applying the compliant suction contact model, shown by Figure A.2, to 1500 3D object models. Because the mass and properties of the 3D object models was known this contact model could be used to determine whether the suction cup could resist external wrenches caused by forces like gravity and acceleration. Unfortunately the usability of this dataset is limited to the type of network Mahler et all used, because it consists solely of those windowed images containing a single grasp point instead of the total images annotated with all the possible grasps.

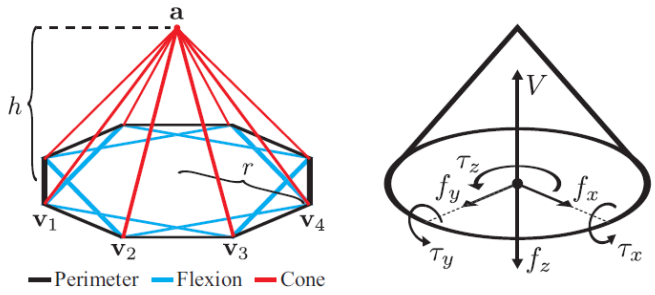


Figure A.2: Compliant suction contact model used to create the Dexnet 3 dataset.

A.3. Implemention of Dexnet

The hardware and software pipeline we have available is different from the one used by Mahler et all and therefore has to be adapted so our and their code can work in tandem. Luckily Mahler et all published the code of their grasp quality convolutional neural network (GQCNN) as a robotic operating system (ROS) package. Which is great, as this meant only the grasp planner code needed to be changed to interact with the GQCNN package. This interaction was achieved by having the grasp planner act as a ROS service client with the GQCNN acting as ROS service server. The ros service request used a color image, a depth image, the camera info and the bounding box and gave a grasp success score and a geometry_msg/pose as response.

Almost all of request objects are directly available only the depth image has to be created from the point cloud. This is easily done since the point cloud is aligned to the color image, meaning each point in the point cloud corresponds to a pixel in the color image. An Opencv Mat structure is initialized with the same dimensions as the color image after which each value is set to the height (z) of the corresponding point in the point cloud. In case of NaN values in the point cloud the corresponding pixel in the depth image is set to zero.

During test evaluations after the creation of the depth map one issue became apparent: our camera setup gave us the point cloud in the workspace frame while the GQCNN expected the origin to be in the camera frame. Meaning the point cloud ($P_{workspace}$) had to be transformed from the workspace frame to the camera frame. With the known transformation matrix ($[T_{workspace}]$), obtained during the calibration of the camera,

this was easily done by using the following equation for each point in the point cloud:

$$P_{camera,i} = [T_{workspace}]^{-1} P_{workspace,i} \tag{A.1}$$

Additionally a different distance value between the camera and the workspace had to be passed on to the GQCNN grasp policy. However, an input parameter for this value already existed making for an easy adaptation.

This was not the only needed adaption as a result from the difference in camera setup. The Ensenso N35 gave a higher quality image than the camera used by Mahler et all. This meant that a different window size had to be determined for creating the centered images created for each grasp point. By measuring the corresponding size of a 10cm long object in the image a value of 192 pixels was chosen to be used. No significant impact on the grasp success rate is expected from this change as even the images using original window size were scaled down when passed on to the neural network.

A.4. Experimental evaluation

As we're doing a general evaluation the experiment is rather simple. A single object is placed in the workspace as this allows for segmentation by background removal. A point cloud is recorded using an Ensenso N35 stereo camera which is converted into a depth image after the segmentation. The depth image is then passed on to the dexnet based grasp policy, whose neural network is running on a NVIDIA GTX 1080 Ti graphics card, to determine a grasp location. This grasp is finally executed using a UR-5 robotic arm with Fizyrs swivel actuator as end-effector. Grasps are determined to be successful if a local seal can be achieved and the object can be lifted and moved without being dropped. Additionally details pertaining to the chosen grasp location and possible observed issues are notated for evaluation. Multiple objects of different sizes and grasping difficulties are evaluated in different orientations. This is done to adequately determine the situations the original dexnet based grasp policy can and can't deal with.

A.4.1. Results

In total 177 grasps were attempted with an average confidence score of $q = 0.607$. Of these grasps 110 were successful, with an average confidence score of $q_{success} = 0.645$, and 67 were failed grasps, with a confidence score of $q_{failure} = 0.546$. This means we observed a success rate of 62.15%, which would be higher than expected if all evaluated objects were adversarial. However, this wasn't the case as the items used during the evaluation were of varying difficulties.

A.4.2. Issues

During the implementation and evaluation of the Dexnet based grasp policy we came across a number of issues. These could generally be separated into two types: general issues resulting from the way the grasp policy was designed and experimental observations of unexpected or suboptimal grasp locations.

The general issues with the grasping policy are as follows:

- As mentioned earlier, the current state of the Dexnet based grasp policy makes use of the point cloud in camera frame instead of in workspace frame. This inherently hinders the usability as modifications have to be made to either use an identical setup or make adaptations to fit a new camera setup, as have been done for the evaluation. Which increases the chances of unforeseen errors.

- The grasp policy will only ever generate a single grasp location due to it's iterative nature, even when multiple possible grasp locations exist. This means that the object can't be picked up when this grasp can't be executed due to inability of the robotic actuator to reach the grasp as a result of possible collisions or physical limitations in reaching the grasp location.

- The iterative nature of the grasp policy has the possibility of finding a locally optimal grasp pose. Unless an incredibly high amount of points is evaluated during the initial sampling a few noisy point can steer the algorithm away from the best grasp locations.

- The combination of the grasp policy's iterative process results in high calculation times. Especially because of the sequential classification of singular grasp points using a neural network instead of using it to determine all possible grasp locations in the search space.

Changing or optimizing these issues will likely improve the usability of the grasp policy but for improvement of the grasp success rate we need to look at the issues observed during the experimental observation. These were as follows:

- The first observed issue could be considered to be more of a general issue but it is added here as it came forward during the observation. Some of the observed surface normal used by the Dexnet based neural network were highly incorrect. Pointing almost perfectly vertical instead of being perpendicular to the angled object surface. This is likely the result of a surface normal calculation based on a limited amount of points, making it susceptible to noise combined with a preference for grasps with a small angle between the surface normal and the vertical axis.
- The suspected preference for grasps with vertical surface normals was also observed to be a cause for worse local grasp locations. Grasps were generated in very difficult and limited surface locations with a vertical surface normal, while significantly better flat grasp areas, albeit with a more angled surface normal, existed.
- A lot of grasps were generated away from the objects center of mass even when grasping there would have been doable. This made it almost impossible to pick up the bigger evaluated objects, while it could result in grasps hanging over the edge for smaller objects. This could partially, at least for the larger objects, be the result of the 10cm window not covering the total object.
- The grasp policy seemed to prefer completely flat surfaces over curved surfaces with which a seal with the suction cup could easily be achieved. This lead to to grasp locations close to the edge of the object while better grasp locations existed near the center of the object where the surface was curved.
- Grasps were generated on ungraspable locations such as on edges, hanging over the side and over holes. Sometimes these still succeeded due to the suction cups ability to slightly adapt, but that is still suboptimal and could easily lead to failures for heavier objects.
- Grasp attempts were made on thin vertical surface ridges sticking out of the surface. This was likely due to these surface structures only being visible in the color image and not in the point cloud making it impossible for the grasp policy to take them into account.
- Weird grasps with suboptimal position and orientation were generated in case of holes in the object or larger areas of noise in the point cloud.
- There seemed to be a high inconsistency between the grasp score q given by the network and the likelihood of success the grasp. A correlation of high q scores for easily graspable surface locations was expected and vice versa for difficult or impossible to grasp ones. However, a decent amount of easily graspable flat surfaces would have scores of $q = 0.2 - 0.5$, while some terrible grasp locations would have scores of $q = 0.8$ and above

A.5. Conclusion

The Dexnet based grasp policy performed significantly worse than expected during the evaluation with a success rate of 62.15 %. And while this success rate might increase a bit with some minor tweaks, it suffers from a couple of major issues. The most important being that grasp attempts are made on object surface locations were a seal formation with the suction cup is obviously impossible. This could be the result of the GQCNN being trained mainly on wrench resistance of the suction cup and not on seal formation. Which leads us to the proposal of a new grasp policy to be researched.

We suggest to split up the grasp policy into two parts: a global and a local evaluator. The local evaluator determines possible grasp locations where a seal can be formed between the suction cup and the objects surface. These proposed possible grasp locations would then be passed on to the global evaluator which determines which of the proposed object locations will most likely lead to a stable grasp depending on the shape and size of the object. The latter could be achieved with the original GQCNN as it is trained on wrench resistance, which is theoretically the only differing property the network will pick up on if all its inputs can (easily) achieve a seal with the suction cup. This leaves the development of such a local evaluator which is the main part of this research as a method has to be determined and implemented.

In line with the current trend of research into grasp generation the decision was made to use of a neural network for the local grasp generation. Specifically the Retinanet architecture, which hasn't been used for

grasp generation yet, as a literature review showed it had the highest reported result mean average precision for small detections for the COCO benchmark. These small detections had the definition of detections of up to 32^2 pixels which grasps locations for a suction cup with a diameter of 20 mm would fall under. The use of such a neural network for the local evaluation will already have 2 improvements compared to the original grasp policy. It will remove the problem of falling into a local optimum as the complete input image is evaluated and not a random set of points and it will likely be significantly faster due to the removal of the iterative part of the original grasp policy. The neural network itself will be trained to determine bounding boxes around graspable object locations, which can then be evaluated using the GQCNN to determine the best one. The choice for the use of bounding boxes instead of more precise masks is rather simple as that improvement wouldn't add anything to the detection. This is because the detection of precise masks is done by first detecting the corresponding bounding box and then detecting which parts would form the exact grasp area. Therefore, a network trained to find a bounding box containing a circular grasp point should already learn the features needed to make the distinction between graspable and ungrapspable object locations.

B

Data acquisition

The most important thing for the use of deep learning is the training data used to train the neural network. It defines the functionality of the network by giving examples that the network attempts to emulate. Training data containing annotations of cats and dogs will result in a network that recognizes these in images, so for the functionality of detecting suction grasps a dataset of images containing such grasps is needed. For the training data we have two requirements for what it must contain:

1. We require annotations of bounding boxes corresponding to object locations with the designation whether they're graspable or ungraspable. This will allow the network to learn to detect graspable object locations.

2. Each image that is annotated must contain multiple annotations of possible grasp locations. This will train the network to determine multiple possible grasp locations for each scenario, which would allow for the selection of alternatives if one of the grasp locations wouldn't be accessible due to constraints.

B.1. Types of data annotation

Data needed to train a neural network for robotic grasping can generally be obtained in three ways: by robotic simulation, by computer simulation or by manual annotation.

Robotic simulation is done by using a physical robotical setup to attempt random grasp locations of which the result is stored as a successful or failed grasp. This has the advantage that the obtained grasp annotations are as accurate as they can be. A successful grasp at that location will always be a successful grasp assuming all related circumstances are the same. However, this advantage is the only one compared to the other methods of obtaining training data. Due to the use of a physical robot that has to move in a safe manner it is by far the slowest method of obtaining data. Additionally due to the result of the end-effector coming into contact with the workspace it is changed meaning only a single data point can be created for each image or state.

Obtaining grasp data with computer simulation is done by creating a analytical model of the interaction between object surface and suction cup end-effector. Using this model the result of a pick attempt at any object surface location can then be determined. This has the advantage that a large amount of data in a relatively short amount of time. However, the accuracy of the data is entirely dependent on the accuracy of the model and the time put into developing and tuning it. Any systematic errors present into the model would just be propagated into the dataset.

The final method for generating data works by having a person use a tool to manually annotate whether an object surface location is graspable or not. It is faster at generating data than the physical simulation, but slower than a computer simulation allowing for a decently sized dataset in a reasonable time frame. Additionally, while it is slightly less accurate than the physical simulation, it is currently still better than the models used for computer simulation. Finally due to the human oversight it is possible to add additional information to each data point, which would be very difficult to obtain otherwise.

Looking at the three different methods of obtaining grasp data we've made the decision to use manual annotation for the following reasons: First off the neural network architecture that we're using, Retinanet, requires multiple annotated data points per image. This means the use of robot simulation, which only gives a single data point per instance, is not usable for our use case. Additionally the speed of obtaining the data

points would have been to slow. While we don't need an insane amount of data for this initial evaluation we still need more than what could be obtained this way in a reasonable time frame. The use of computer simulated data would be best if the results are accurate enough, due to its high speed. However, it is difficult and time intensive to get this desired accuracy and a suboptimal implementation would leave a lot of room for errors that can be propagated into the trained neural network. So it is still suboptimal in comparison to the manual annotation method which allows for a high accuracy without the implementation issues. And while the manual method is slower it is fast enough for generating the amount of data we expect to need for this research. Additionally the human oversight allows us to easily fix issues related to errors in the segmentation and add additional information that might not have been observable otherwise, such as thin surface ridges that are invisible in the point cloud. For this reason the decision was made to use manual data annotation.

B.2. Tool design

In order to create the grasp data manually a tool is needed so a person can input information corresponding to possible grasp locations. Unfortunately at the time of this research no such tool exists for suction based grasping and therefore one has to be developed. In order to acquire the required data the following functionality is required from this suction grasp annotation tool:

1. A way to select and load a data instance and set the relevant parameters.
2. A way to segment the point cloud belonging to the data instance.
3. A way to visualize possible grasp locations so the user can determine whether it is graspable or not.
4. An option to select whether the suggested grasp location is graspable or not.
5. Additional options to give extra information about the suggested grasp location.

The tool with the required functionality was created using the Qt Library resulting in the interface shown in Figure B.1. Further details on the implementation of each of the tools requirements are given in the following subsections.

B.2.1. Initialization

The first thing that the tool needs to do is load the data that's supposed to be annotated and set all the related parameters. All the needed data instances of (piles of) objects are stored in a folder containing both a color image and a point cloud with a standardized file names. Therefore the tool was decided to take the path of such a folder from which both the color image and point cloud can then be loaded. After the image and point cloud are loaded two more parameters need to be set. The first determining the amount of data points or possible grasp locations will be generated for the data instance. Less grasp points are required if only a single or a small amount of objects is present in the data instance and vice versa for a larger amount of objects. Additionally the size of the suction cup that will be used as end effector has to be specified as the graspability of an object surface location differs depending on the this size. After all a smaller suction cup can achieve suction at a spot that a larger one might not. The corresponding part of the tools interface related to this initialization is highlighted as nr. 1 in Figure B.1.

B.2.2. Segmentation

Three options were implemented for segmenting the point cloud, which can be seen highlighted as nr. 2 in Figure B.1. The first option is to simply not apply a segmentation to the point cloud. This option is unlikely to be used a lot, but was left as an option due to it's ease of implementation.

The second segmentation option was for when only data on a single object was desired. It works by opening a new window containing the color image, which is aligned to the point cloud, on which the outline of the desired object can be drawn. This outline is then used to create a mask which allows for the removal of all points in the point cloud outside of the selected area resulting in a segmented point cloud containing only the desired object.

Finally the third and likely most used segmentation option was to remove the background, so data points will only be generated on the objects. This option compares the point cloud of the data instance with a second point cloud of just the used workspace without objects placed on it. By removing the points that lie within a small distance from the point cloud only the 3D data pertaining to the objects remains in the point cloud.

B.2.3. Visualization

After the segmentation is applied possible grasp locations have to be generated before they can be shown to the user. This is done by first uniformly sub-sampling the segmented point clouds to get possible grasp points. These generated points represent where the center of the suction cup would be placed during a robotic grasp action. However, just the single point isn't representative of the entire contact area, as the suction seal is dependent on its entire contact area. Therefore a kd-tree is used to get all the points withing the radius of the specified suction cup size to get an approximate grasp area. This object surface area can then be visualized, combined with the addition of the surface normal at the location of the sampled grasp point, for the user to determine whether a grasp there would achieve suction.

The generated grasp areas are visualized sequentially using the point cloud library (pcl) [4], as shown in Figure B.1 highlighted by the nr. 3. This open source library has a tool for visualizing point clouds and has the option to highlight points by changing their color. The viewer is automatically centered each time a new point is to be evaluated by the user when the last one is confirmed. Additionally it allows for the implementation of the ability to zoom in and move the point cloud around for a closer look when it isn't immediately obvious whether suction could be achieved.

Unfortunately not everything is perfectly visible in the point cloud and some things can even be difficult to see in it. Additionally the surface of a graspable object location can be noisy in the point cloud and some small thin obstruction that would lead to a failed grasp have the chance of not being visible in it. Therefore the choice was made to have an additional way of visualizing the grasp area. Because the point cloud is aligned to the color image each point in the one corresponds to a point in the other with the same index. This allows us to determine the bounding box parameters for the grasp area and visualize them on top of the color image as shown in Figure B.1 highlighted by the nr. 4. Giving this additional visualization to the user of the tool increases the accuracy in determining whether a specific surface location will lead to a successful grasp or not. It also allows for the observation of additional data such as whether a hole in the point cloud corresponds to an actual hole in the object or if it's noise resulting from the recording.

B.2.4. Graspability

With the possible grasp areas visualized it becomes possible for the user of the tool to see and determine whether each specific one will succeed in a successful grasp. As a successful grasp can be divided into two subproblems of local and global graspability this has also been done in the suction grasp data annotation tool, as highlight by nr. 5 in Figure B.1.

- The surface tag corresponds to the local graspability of a proposed grasp point. So the tool user has to answer the question whether a successful seal can be formed between the imaginary suction cup and the proposed grasp region. If the answer is yes the user marks it as graspable and if not as ungraspable.

- The location tag corresponds to the global graspability of a proposed grasp point. So the tool user has to answer the question whether the torques acting on the contact point resulting from the grasp location in relation to the objects center of mass will cause the grasp to fail or succeed. If the grasp would succeed the user marks it as graspable and if not as ungraspable.

Now, since the arm movements required by the tool user to select options and click on buttons to get the next grasp point take up a large part of the annotation time, an improvement in speed could be achieved by reducing these. The first of such improvements was to implement the option to use hotkeys to cycle through the generated potential grasp points, removing the need to click buttons in the tools' interface. The second of the improvements was to set the initial graspability value of the local and global parameters so the least amount of changes were necessary, giving the tool user the task of changing wrong pre-set values. This was easily done for the location parameter since the size of the objects used in the dataset were relatively small, which meant most of the generated grasp locations would be globally graspable. So the location option would be already correct most often if it was set to graspable by default. The surface option was comparatively more difficult to set a default value for as the surfaces of the objects were ever changing. Therefore the decision was made to use a tool used by Fizyr with a decent accuracy at determining the local graspability to set the initial graspability value. While this is less accurate than having a human determine graspability it is accurate enough to reduce the amount of needed interactions allowing for an increase in annotation speed.

B.2.5. Additional options

The main advantage of using manual annotations to generate the grasp data, beside it's high accuracy, is that it allows for the addition of extra information pertaining each annotated grasp location. The additional options for this that were added to the annotation tool can be divided into two types: options giving additional information, allowing for possible extra use cases of the data, and options notifying the annotations possibly contain a problem, allowing for filtering of the data. These options are selected in the block highlighted by the nr. 6 in Figure B.1.

Informative options There are three options for extra information pertaining to the annotation: optimal area, near edge and invisible ridge.

Optimal area is selected when the current possible grasp point lies on or near the center of mass of the object. This could potentially be used for training a network to select an optimal grasp from multiple possible grasps.

Near edge is selected when the current possible grasp point lies close to the edge of an object or additionally if it is near a hole in the surface. These grasp locations tend to be more risky in case of small errors in the recording or execution, because they can lead to failures. This parameter can be used to either penalize risky locations or set them to ungraspable entirely.

Invisible ridge is selected when the current possibly grasp point will lead to a failed grasp because it lies on a small ridge that's not visible when looking solely at the point cloud.

Notifying options There are four options notifying the user of possible errors or issues that they might want to filter out of the dataset: rubbish surface normal, part of background, reflective surface and uncertain about grasp.

Rubbish surface normal is selected when the calculated surface normal for the current possible grasp point is not correct in relation to the object surface. This is important for when the surface normals are used when training the neural network to determine graspability, because then these data points would have to be fixed or removed.

Part of the background is selected when the current possible grasp point is generated on the background workspace instead of an object. This is a potential result of an incorrect segmentation and should be removed from the dataset.

Reflective surface is selected when the point cloud at the point of the current grasp point is very noisy. This can be caused by things like as a reflective surface. Often times this makes it very difficult or impossible to infer graspability solely using the point cloud, due to the erratic spread of the points. However, using the 2D image the correct graspability can be inferred. This option allows such difficult points to be filtered in order to not potentially hurt the performance of the neural network.

Uncertain about grasp is selected when the annotator isn't 100 percent certain whether they know the correct answer to the question whether the possible grasp location is graspable or not. So this parameter potentially allows someone else to potentially check the decision or to simply set all these to ungraspable to play it safe.

B.3. Annotation

While annotating an image the annotations are saved to a newly created csv*(comma separated value)* file and when the image finished the csv file is appended to a file containing all annotations for all images with each line in this file corresponding to an annotation. The created suction grasp annotation tool was used for approximately two weeks resulting in 20293 annotations over 51 images.

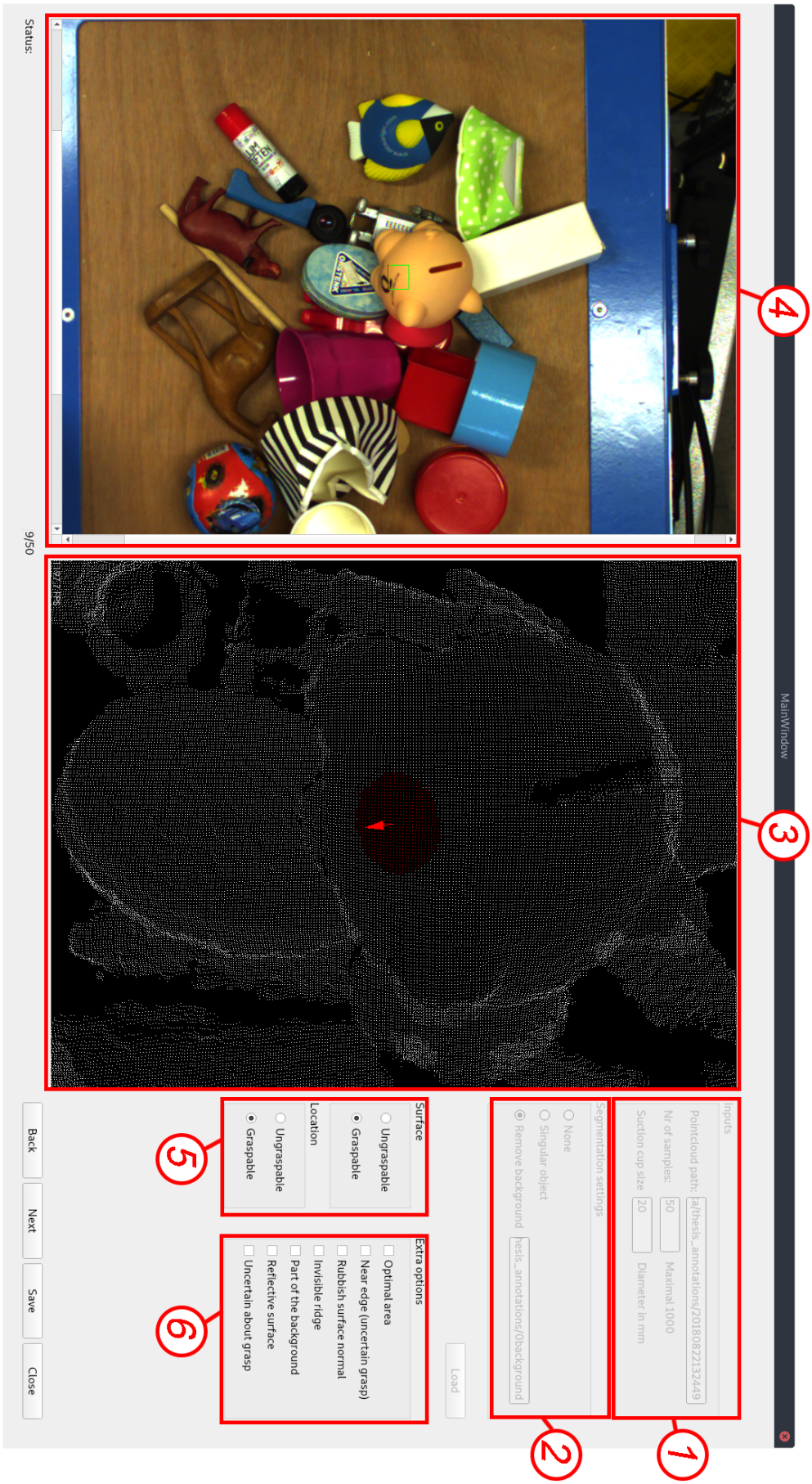


Figure B.1: The Designed annotation tool 1) Initialization settings 2) Segmentations options 3) 3D viewer 4) 2D viewer 5) Graspability selection 6) Additional options

C

Local grasp evaluator

With the obtained data we can now train a neural network to function as local grasp evaluator i.e. can a suction seal be achieved. For the network architecture the choice was made to use the Retinanet architecture proposed by Lin *et all* in their "Focal Loss for Dense Object Detection" paper [2]. This choice was made because the reported average precision for small detections AP_s, which are defined as detections smaller than 32^2 pixels [1], were the highest reported with a value of 21.8. This metric was used to select a neural network architecture because it is the detection size that suction seal detections would fall under.

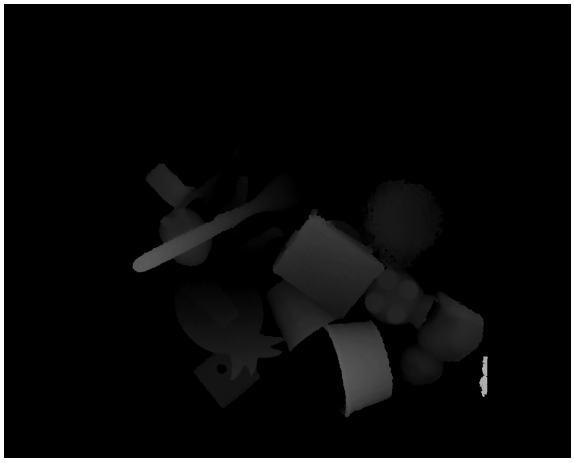
C.1. Data preparation

The neural network can't take a point cloud as input even if it is aligned to a 2D image. Therefore the first step to prepare the data is to transform the point cloud to a depth image. This is easily done by initializing an empty image of the same size as the recorded 2D image using opencv and replacing each value with the z-value of each corresponding point in the point cloud with the same index.

The quality of the data obtained from the 3D camera is good, but it does contain some missing data points. These empty ,or not a number (NaN), points could simply be set to the value of the workspace, but that would leave erroneous points in object surfaces. Such inconsistency could decrease the quality of the results when using the neural network. So to improve the accuracy of the local grasp detection we would like to remove these small data errors, without changing the originally recorded data. To do this two morpological filters where used. The first filter was used to find the likely value of the missing data points by creating a new image where each point was calculated as the average of the points around them. These values can't immediately be used to replace the missing data points though as a decent amount of such points are the result of actual holes existing in the objects and replacing them would ruin the data. So a second filter is used to detect which points can be replaced. First an *isnan()* function is used to create a copy of the image where each valid point gains the value of 1 and each empty point gains the value of zero. Applying the morphological filter then tells us the amount of valid points each point is surrounded with. Points fully surrounded, and thus likely recording errors, having a value of 8 and points having no surrounding valid points, and thus likely belonging to a correctly recorded hole, having a value of 0.

Combining the results of the application of these filters then allows us to replace the missing data points with their interpolated values without ruining the data by only replacing those with at least 7 valid data points surrounding them. A example of the results from the application of these filters is given by Figure C.1.

(a) Dexnets iterative grasp pipeline.



(b) The hybrid grasp pipeline combining Retinanet with Dexnet.

Figure C.1: Schematic overview of the evaluated grasp pipelines.

C.2. Initial results

After the data is prepared it is split up in a training and validation set containing 80% and 20% of the data respectively. The training set was then used to train the neural network for approximately 24 hours on a NVIDIA GTX 1080 Ti graphics card. The trained network was then evaluated on its performance on the validation set giving the results shown in Table C.1.

ungraspable AP	0.0272
graspable AP	0.0348
mean AP	0.0310

Table C.1: Evaluation result of the initial training.

Now we expected a lower average precision value compared to the one reported in the original Retinanet paper [2] due to the nature of the data. Unlike object detection in color images no hard boundaries exist between possible grasp areas as they occur on a constant surface. So if you have a large graspable surface area multiple overlapping grasp areas exist. Making for a higher chance of the network detecting a graspable or ungraspable point near but not perfectly centered on an originally annotated one, resulting in a lower intersection of union (IOU) value and thus a lower precision. Additionally the dataset was fairly small for the amount of possible variance so a small over-fitting and thus lower detection success rate on the validation dataset makes sense. However, even with these reasons the results are still way lower than expected and it would likely be beneficial to look into a way to improve the detection success rate.

C.3. Data augmentation

Another possible cause of the very low average precision, which was suggested by a colleague at Fizyr, was that it could be a result of the annotations in the training dataset not covering the entire image. After all, the network learns to discern regions of interest, which are then separated into graspable and ungraspable detections, and background based on the annotations. An object surface location that should be detected as graspable or ungraspable could instead end up being classified as background resulting in no detection due to it not being present in the dataset. The similarity between such a non annotated area and an annotated one could increase the networks uncertainty and thus reduce the detection accuracy. This issue was likely a problem resulting from decisions made during the creation of the dataset. In order to add more different images for an increase in object variance the total number of annotations for each image might have been set too low resulting in the presence non annotated areas in each image. Unfortunately this couldn't be remedied by simply adding more annotations to each image in the dataset due to time constraints, so another solution had to be found.

The solution that was decided on was to slightly increase the size of all annotated bounding boxes outward

from the center by a fixed amount. Since all bounding boxes in the dataset are centered on single grasp points the assumption was made that this wouldn't significantly change the detection. Because, even if the outer size of the bounding box increases, resulting in less non annotation space, the area were the suction cup would be placed would be the same. The only possible issue, which is mostly due to the small dataset, would be that this size increase could potentially decrease the variance in objects the network will be able to handle by learning some of the surface structures surrounding the grasp area. However, a limited increase in size shouldn't significantly decrease the performance on novel objects, while likely resulting in a much better performance.

C.4. Final selection

A couple different annotation size augmentations were applied to the dataset and evaluated by training the neural network before a final one was selected. Each increase seemed to improve the precision values when evaluated with the validation set, but had the downside of decreasing the applicability of the network as local grasp evaluator. Because, while the center of an annotation would still determine it's graspability, larger annotation sizes possibly increase the chance of the network learning to take into account surrounding surface structures, which is undesired. Therefore the network using the dataset of an augmentation size of 10 pixels was selected to function as local grasp evaluator as that version showed a significant increase in average precision without increasing the annotation size too much. Training with this dataset for approximately 24 hours or 35 epochs had the results shown by table C.2.

ungraspable AP	0.0988
graspable AP	0.0990
mean AP	0.0989

Table C.2: The final selected network after augmentation.

Now the observed average precision is still quite low, but like mentioned before a lower value isn't entirely surprising, because of the type of data we try to detect. Though in order to conclude whether a experimental evaluation would be meaningful with the obtained network a visual evaluation was done. A new recording was fed to the trained network to see if the output could result in a correct local grasp evaluation. As shown by Figure C.2 this was indeed the case allowing us to move forward with an experimental evaluation of the obtained grasp evaluator.

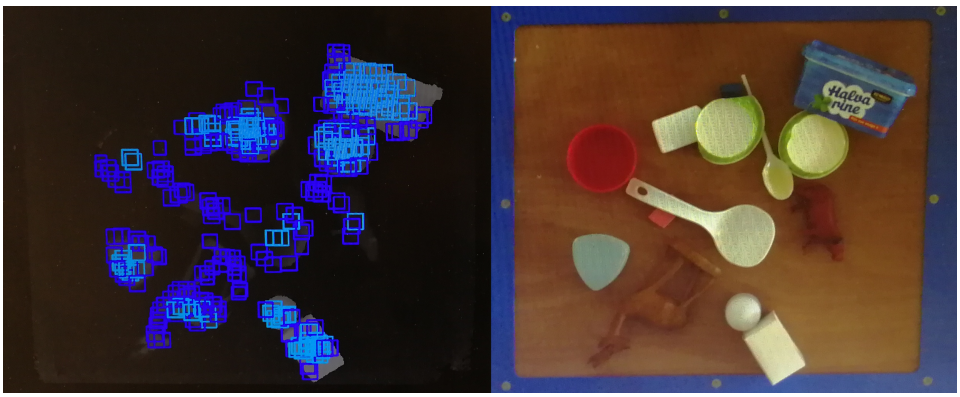


Figure C.2: Bounding boxes of graspable locations as determined by the trained Retinanet network. The light blue bounding boxes are those with a high confidence score.

D

Experiment 1: Retinanet Grasp evaluation

D.1. Experiment goal

With the neural network trained on the annotations it is now almost ready to be used to determine graspable locations on unknown objects. It outputs possibly graspable regions together with corresponding q-values. However, the outputs of the network, in the shape of confidence scores (q), first need to be evaluated. Specifically what the correlation between the confidence score of an outputted location and the likelihood of this object location resulting in a successful grasp action. The best way to find this correlation is to evaluate it experimentally.

D.2. Experimental design

The goal of this experiment is to find the correlation between the confidence score output by the neural network and the likelihood of this location resulting in a successful grasp action. For this an entire grasp pipeline is needed from creation of the depth image to execution of the robotic grasp. An Ensenso camera is used to create a RGB image and a 2D picture aligned 3D pointcloud. This means that the point cloud contains a 3D data point for each pixel in the 2D image. The created pointcloud is calibrated to place the origin at the base of the UR-5 robotic arm, which places the workspace at $z-axis = 0$. A background removal segmentation is applied to the pointcloud by comparing it with a pointcloud taken of only the background and removing all the points within a few millimeters. Finally the pointcloud is transformed into a depth map by taking the z-values and corresponding indices, so it can be used by the neural network. Using the depth map as input for the trained Retinanet neural network gets a number of possibly grasp locations together with a confidence score (q) as output.To get the best grasp locations the highest scores would be used, but in order to evaluate the correlations between confidence score (q) and likelihood of a successful grasp one of the output grasp locations is randomly selected. This grasp location is then passed on to be executed by the robotic arm. The experiment is done with an UR-5 arm combined with a suction actuator designed by Fizyr. This actuator has two properties that make it excellent for experimental evaluation: it contains a swivel joint at the end of the actuator to reduce the needed approach space and it has the ability to disconnect from the robot arm if the grasp fails and the robot end effector keeps moving forward into the to be grasped object in order to not damage it. The grasp motion is done by moving the end-effector in line with the surface normal of the object at the grasp location after which it slowly approaches until one of the following scenarios occurs:

- A successful suction grasp is detected with the use of a pressure sensor.
- The end-effector has disconnected from the robot.
- The end-effector has moved passed the grasp location by either a wrong detection or pushing the object aside.

The result of the grasp action, together with the generation speed an confidence score (q), is then logged after which a new random grasp is attempted. Additionally the pile of items in the workspace is replaced multiple times.

D.3. Results

This pipeline was evaluated for a few days resulting in a total of 665 grasp attempts using an 20mm suction cup. Some of these had to be thrown out due to them being either unreachable by the gripper or because a piece of the workspace was selected due to a failure with the segmentation. This left 605 grasp attempts to be evaluated. Notably, the grasps that were attempted on the workspace had a q-value close to zero, from which can be inferred that the network didn't like these grasp locations despite being flat perfectly graspable locations. The neural network took an average of 71.13 ms to run and the total pipeline from detection to selection of the random grasp location took on average 234.83 ms.

q-value	Total grasps	Successful grasps	Failed grasps
0.00 - 0.05	0	0	0
0.05 - 0.10	94	47	47
0.10 - 0.15	39	23	16
0.15 - 0.20	33	24	9
0.20 - 0.25	27	17	10
0.25 - 0.30	20	15	5
0.30 - 0.35	19	12	7
0.35 - 0.40	13	12	1
0.40 - 0.45	13	9	4
0.45 - 0.50	12	9	3
0.50 - 0.55	15	11	4
0.55 - 0.60	19	17	2
0.60 - 0.65	17	13	4
0.65 - 0.70	23	21	2
0.70 - 0.75	22	22	0
0.75 - 0.80	32	31	1
0.80 -0.85	48	47	1
0.85 - 0.90	42	40	2
0.90 - 0.95	52	52	0
0.95 - 1.00	65	65	0

Table D.1: The total number of successful and failed grasp attempt corresponding to the q-value output by the network.

However, while all successful grasps in Table D.1 achieved a high enough suction force to lift up the object, not all were perfect. During the experiment two different successful outcomes were observed: optimal grasps that easily achieved suction and suboptimal grasps that, while achieving enough suction force, often had some air leakage. Most of the time this was a result of grasping near the edge of an object or near an extrusion within the object, causing the suction cup to curve sharply. This division in successfulness has been recorded in Table D.2.

q-value	Total successful grasps	Optimal grasps	Suboptimal grasps
0.00 - 0.05	0	0	0
0.05 - 0.10	47	34	13
0.10 - 0.15	23	18	5
0.15 - 0.20	24	19	5
0.20 - 0.25	17	17	0
0.25 - 0.30	15	14	1
0.30 - 0.35	12	11	1
0.35 - 0.40	12	11	1
0.40 - 0.45	9	9	0
0.45 - 0.50	9	9	0
0.50 - 0.55	11	10	1
0.55 - 0.60	17	16	1
0.60 - 0.65	13	13	0
0.65 - 0.70	21	21	0
0.70 - 0.75	22	21	1
0.75 - 0.80	31	31	0
0.80 -0.85	47	47	0
0.85 - 0.90	40	40	0
0.90 - 0.95	52	51	1
0.95 - 1.00	65	65	0

Table D.2: The total number of successful and failed grasp attempt corresponding to the q-value output by the network.

In order to use the number of grasps to evaluate the correlation between q-value and likelihood of a successful grasp a more useful way of displaying the data is to use percentages of the total grasps. Doing this for both table D.1 and D.2 gives us table D.3.

q-value	Total successful grasps (%)	Optimal grasps (%)	Suboptimal grasps (%)	Failed grasps (%)
0.00 - 0.05	0.00	0.00	0.00	0.00
0.05 - 0.10	50.00	36.17	13.83	50.00
0.10 - 0.15	58.97	46.15	12.82	41.03
0.15 - 0.20	72.73	57.58	15.15	27.27
0.20 - 0.25	62.96	62.96	0.00	37.04
0.25 - 0.30	75.00	70.00	5.00	25.00
0.30 - 0.35	63.16	57.89	5.26	36.84
0.35 - 0.40	92.31	84.62	7.69	7.69
0.40 - 0.45	69.23	69.23	0.00	30.77
0.45 - 0.50	75.00	75.00	0.00	25.00
0.50 - 0.55	73.33	66.67	6.67	26.67
0.55 - 0.60	89.47	84.21	5.26	10.53
0.60 - 0.65	76.47	76.47	0.00	23.53
0.65 - 0.70	91.30	91.30	0.00	8.70
0.70 - 0.75	100.00	95.45	4.55	0.00
0.75 - 0.80	96.88	96.88	0.00	3.13
0.80 - 0.85	97.92	97.92	0.00	2.08
0.85 - 0.90	95.24	95.24	0.00	4.76
0.90 - 0.95	100.00	98.08	1.92	0.00
0.95 - 1.00	100.00	100.00	0.00	0.00

Table D.3: The total number of successful and failed grasp attempt corresponding to the q-value output by the network.

A visual representation of Table D.3 can be in Figure D.1.

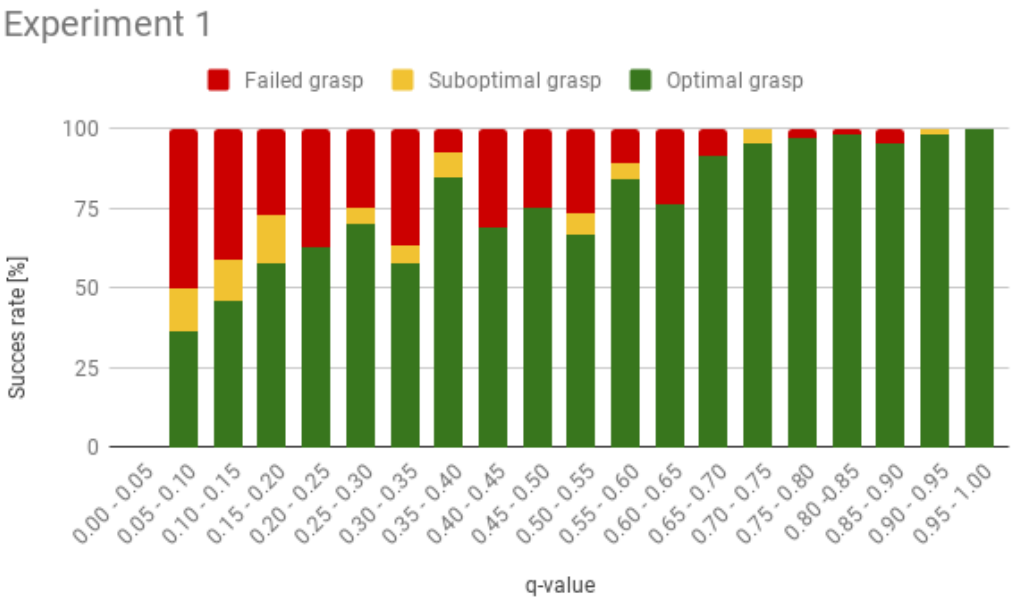


Figure D.1: Result from the experiment evaluating the correlation between the confidence score output by the neural network and the likelihood of a successful grasp.

Due to the relatively low amount of evaluated grasps the transition between success rates isn't very smooth, however, a curve approaching 100% success rate for higher q-values can be vaguely observed. So it might not be too far out there to assume that the shape of the general correlation between the network outputs q and the grasp success rate takes the shape of a logarithmic curve starting around 50% and approaching 100%. This result is very promising as outputs with a q-value as low as 0.7 still had a grasp success rate of 95% and up, while q-values around 0.5 still had a 70% grasp success rate. Additionally even the lower q-values had a grasp success rate approaching 50%. So with Retinanet outputting multiple grasp locations at the same time there is a very high chance that the output of the network contains multiple graspable locations, even for difficult objects.

Some additional observations were made during the experiment that show a good depth image quality is very important for the network. On the lower end of confidence scores it was noticed that some of the successful grasps likely had low confidence scores due to noisy pointclouds. Possibly due to reflections or occlusions good grasping locations were obfuscated causing the network to be less confident about a successful grasp. On the higher side of the confidence scores failures were sometimes the result of things that couldn't be observed in the depth image. Thin extrusions on the object surface made it so the suction gripped couldn't achieve a seal on a surface that seemed flat in the depth image. So while it might decrease the grasp success rate for the lower confidence scores better images will likely improve the robustness of the network overall.

E

Experiment 2: Pipeline comparison

E.1. Experiment goal

With the neural network trained on the annotations and a completed evaluation of the correlation between the outputted confidence score and the likelihood of a successful grasp the complete pipeline can now be evaluated. The main goal is to see whether the replacement of Dexnets iterative pipeline with an initial evaluation for determining locally graspable object locations using Retinanet will decrease computation time without increasing the amount of erroneous grasp attempts.

E.2. Experiment design

The goal of this experiment is to compare how well the newly suggested grasp pipeline compares to the original Dexnet pipeline. To do this both pipelines need to be used to determine the grasp locations of the same objects in the same position and orientation in the same order. Similar to experiment 1 random objects are put in a box, which is shaken and turned upside down over the workspace in order to get a random orientation. However, there are two differences: A grid made out of tape is added to the workspace, as shown in Figure E.1, in order to place the objects in the same location and orientation when evaluating each of the grasp pipelines. And, the number of objects is limited to 10 for each cycle, so it remains possible to assemble the item piles in the same way. This is then repeated for multiple cycles.

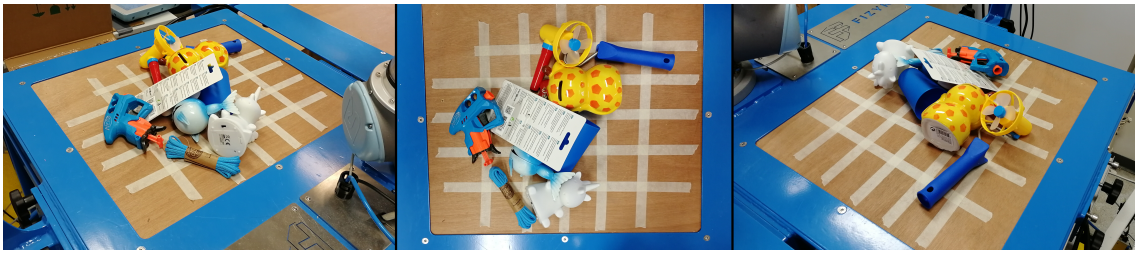


Figure E.1: The setup used during experiment 2, with a pile of objects. Photographed from multiple angles to ensure the objects can be replaced accurately between experiments.

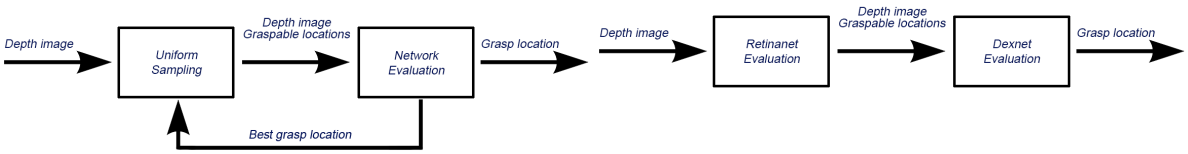
An Ensenso camera is used to create a RGB image and a 2D picture aligned 3D pointcloud. This means that the point cloud contains a 3D data point for each pixel in the 2D image. The created pointcloud is calibrated to place the origin at the base of the UR-5 robotic arm, which places the workspace at $z-axis = 0$.

After the image is taken the objects need to be segmented. Preferably this would be done automatically, but in this case that isn't possible. A trained network that can segment the used objects with the required accurately wasn't available at the time of the experiment. Additionally due to time constraints and the fact that the segmentation falls outside the scope of this project a manual segmentation was implemented. This has the additional advantage that the person doing the segmentation can decide which object is attempted to be grasped, allowing for the same object grasp order between both grasp pipelines' picking cycle. The segmentation was done by drawing an objects outline on top of the 2D image, which could then be used as mask for the point cloud to get the segmented point cloud. Finally a depth image, of the same size as the

2D image, was created by taking the z-values of the segmented point cloud for each of the images' indices, resulting in an image like Figure E.2.

Figure E.2: Example of a segmented depth image.

After the segmentation was done the grasp pipeline was called to determine the best grasp location for the suction gripper. Depending on the cycle either the original Dexnet pipeline, shown by Figure E.3a, or the hybrid pipeline combining Retinanet with Dexnet, shown by Figure E.3b, was selected.



(a) The original Dexnet based iterative grasp pipeline.

(b) The hybrid grasp pipeline combining the new Retinanet local evaluator with the Dexnet based neural network to act as global evaluator.

Figure E.3: Schematic overview of the evaluated grasp pipelines.

The Dexnet based grasp pipeline works as described in Appendix A and shown by Figure E.3a. A segmented depth image is uniformly sampled for a number of potential grasp locations. These locations are then evaluated by Dexnet based neural network to determine which potential grasp location is the best. The best location is then passed back to the uniform sampler to sample new points around it. This iterative process is then repeated a number times to close in on the global optimal grasp location.

The hybrid pipeline works by replacing the iterative sampling part with a neural network, using the RetinaNet architecture, that has been trained to find locally optimal grasp locations: points in the object that will lead to a successful seal with the suction cup. These points are then passed on to the Dexnet based neural network to select which of these grasp locations has the highest chance at a successful grasp action. The hybrid pipeline was implemented by first taking the potential grasp location with the highest confidence scores outputted by the Retinanet network. These potential grasp locations were then added as an extra input when calling the Dexnet based grasp pipeline function. If this additional input contained possible grasp locations they were immediately evaluated using the Dexnet based neural network leading to a final grasp, while circumventing the iterative sampling parts. However, if this extra input was empty the original iterative pipeline was used, making it possible to easily switch between them.

After a grasp location was determined it was passed on to the UR5 robot arm to attempt to pick up the object. Fizyrs swivel gripper was used with a suction cup of 20mm. The objects needed to achieve local suction and be lifted from the workspace to be defined as a successful grasp. Grasp attempts were done for each object until either they were picked up successfully or three attempts had failed, whichever came first. These extra attempts were done to ensure erroneous grasps actually showed a failure of the grasp pipeline and weren't a result of outside issues, such as a bad point cloud. The limit of three was picked as it gave a couple attempts without wasting a large amount of time, before the object was removed by hand. After each attempt a new image was taken and grasp attempted until all objects had been removed from the workspace.

E.3. Results

Over a couple of weeks 15 cycles of the experiment were completed, for 150 evaluated object grasps. Unfortunately two objects had to removed from the evaluation as they couldn't be picked up. One object, because its weight was too high and another that was placed in such an unstable pose that the slightest disturbance made it move leaving it ungraspable with the current image. This leaves 148 object grasps to be evaluated in this section.

The first 12 cycles were placed randomly as described at the start of section E.2. The final three piles, however, were comprised of the most difficult to grasp objects available placed in their most difficult to grasp orientation in order to test the limits of the grasp pipeline. The former will be described as the random setup, while the latter will be called the adversarial setup.

E.3.1. Total evaluation

The first thing that can be compared between the grasp pipelines is their calculation time. The original Dexnet pipeline takes on average 1.559 seconds to go from segmented point cloud to executable grasp, of which 1.070 seconds are used to iteratively determine a graspable location. While the hybrid pipeline only takes 1.117 seconds on average to do the same thing. The Retinanet evaluation takes 0.066 seconds on average to determine locally graspable locations and reduces the Dexnet evaluation to only 0.485 seconds.

The average confidence score of the chosen grasp location given by the Dexnet network turned out to be significantly lower for the hybrid pipeline compared to the original pipeline: 0.3327 for the hybrid pipeline and 0.6155 for the original one. However, this is to be expected as the original pipeline searches for a local optimum while the hybrid pipeline doesn't. And this higher confidence score didn't seem to necessarily correlate with a higher pick success rate as is shown by Tables E.1 and E.2.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	134	124	7	3	14
Hybrid pipeline	141	133	5	3	7

Table E.1: The number of picked objects for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	90.54 %	83.78 %	4.73 %	2.03 %	9.46 %
Hybrid pipeline	95.27 %	89.86 %	3.38 %	2.03 %	4.73 %

Table E.2: The success rate of picked objects for each pipeline and the amount of attempts it took to succeed.

Random setup evaluation As mentioned at the start of this section not all objects were placed in the same way. Only the 118 objects placed during the first 12 cycles were placed completely randomly. This placement can have an effect on the grasp success rate and is therefore also evaluated separately from the last cycles. The pick success rate is shown by Tables E.3 and E.4.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	111	102	7	2	7
Hybrid pipeline	114	110	2	2	4

Table E.3: The number of picked objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

Adversarial setup evaluation As mentioned at the start of this section not all objects were placed in the same way. The 30 objects placed during the last three cycles were selected and placed in specific poses to evaluate how well the grasp pipelines handled objects that had limited places they could be grasped successfully. The pick success rate is shown by Tables E.5 and E.6.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	94.06	86.44 %	5.93 %	1.69 %	5.93 %
Hybrid pipeline	96.60	93.22 %	1.69 %	1.69 %	3.39 %

Table E.4: The success rate of picked objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	23	22	0	1	7
Hybrid pipeline	27	23	3	1	3

Table E.5: The number of picked objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	76.66 %	73.33 %	0.00 %	3.33 %	23.33 %
Hybrid pipeline	90.00 %	76.67 %	10.00 %	3.33 %	10.00 %

Table E.6: The success rate of picked objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

E.3.2. Known object evaluation

Now we're especially interested in how the grasp pipelines perform on novel objects. However, not all objects used during the experiment were unknown to the Retinanet network. And, while the training dataset is small, likely containing only a single instance of the evaluated object in a different pose, it is best to control for this. So, to display the results in more detail the results have been split for known and unknown objects. The results of the 76 pick attempts on known objects can be found in Tables E.7 and E.8. As expected the average confidence score given by the Retinanet network was higher for the pick attempts on known objects compared to all objects, but not much: $\bar{q}_{known} = 0.7562$ and $\bar{q}_{all} = 0.7042$.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	75	72	2	1	1
Hybrid pipeline	76	75	1	0	0

Table E.7: The number of picked known objects for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	98.69 %	94.74 %	2.63 %	1.32 %	1.32 %
Hybrid pipeline	100.00 %	98.68 %	1.32 %	0.00 %	0.00 %

Table E.8: The success rate of picked known objects for each pipeline and the amount of attempts it took to succeed.

Random setup evaluation The initial 12 cycles contained 64 pick attempts on known objects. The pick success rate for these is shown by Tables E.9 and E.10.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	64	61	2	1	1
Hybrid pipeline	64	63	1	0	0

Table E.9: The number of picked known objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	100.00 %	95.31 %	3.13 %	1.56 %	0.00 %
Hybrid pipeline	100.00 %	98.44 %	1.56 %	0.00 %	0.00 %

Table E.10: The success rate of picked known objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

Adversarial setup evaluation The final three cycles contained 12 pick attempts on known objects. The pick success rate for these is shown by Tables E.9 and E.10.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	11	11	0	0	1
Hybrid pipeline	12	12	0	0	0

Table E.11: The number of picked known objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	91.67 %	91.67 %	0.00 %	0.00 %	8.33 %
Hybrid pipeline	100.00 %	100.00 %	0.00 %	0.00 %	0.00 %

Table E.12: The success rate of picked known objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

E.3.3. Unknown object evaluation

The performance on novel objects is the most important for this report. During the experiment 72 pick attempts were made on unknown objects. The results of these attempts are shown by Tables E.13 and E.14. As expected the average confidence score given by the Retinanet network was lower for the pick attempts on unknown objects compared to all objects, but not much: $\bar{q}_{unknown} = 0.6625$ and $\bar{q}_{all} = 0.7042$.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	59	52	5	2	13
Hybrid pipeline	65	58	4	3	7

Table E.13: The number of picked unknown objects for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	81.94 %	72.22 %	6.94 %	2.78 %	18.06 %
Hybrid pipeline	90.28 %	80.56 %	5.56 %	4.17 %	9.72 %

Table E.14: The success rate of picked unknown objects for each pipeline and the amount of attempts it took to succeed.

Random setup evaluation The initial 12 cycles contained 54 pick attempts on unknown objects. The pick success rate for these is shown by E.15 and E.16.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	47	41	5	1	7
Hybrid pipeline	50	47	1	2	4

Table E.15: The number of picked unknown objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	87.04 %	75.93 %	9.26 %	1.85 %	12.96 %
Hybrid pipeline	92.59 %	87.04 %	1.85 %	3.70 %	7.41 %

Table E.16: The success rate of picked unknown objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

Adversarial setup evaluation The final three cycles contained 18 pick attempts on unknown objects. The pick success rate for these is shown by Tables E.17 and E.18.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	12	11	0	1	6
Hybrid pipeline	15	11	3	1	3

Table E.17: The number of picked unknown objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	66.67 %	61.11 %	0.00 %	5.56 %	33.33 %
Hybrid pipeline	83.34 %	61.11 %	16.67 %	5.56 %	16.67 %

Table E.18: The success rate of picked unknown objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

E.4. Conclusion

Looking at the results presented above we can confidently say the goal of this thesis has been successful. The newly proposed hybrid grasp pipeline, combing a predetermination of locally graspable locations using Retinanet as input for the Dexnet evaluation, decreased the calculation time without a decrement in pick success rate. On the contrary, the new grasp pipeline actually outperforms the old one for every evaluation. Not only for those where the used objects were partly known to the pipeline, but also for the most difficult novel objects. As few as 50 training images resulted in a pick success rate of over 90% for novel objects and over 80% for the most difficult scenarios, which could possibly be increased with more training data.

Interestingly a closer inspection of the objects that the hybrid pipeline failed or had difficulty with showed a large overlap with those the original pipeline failed. After the reduction in failed picks only two objects caused issues for the hybrid grasp pipeline and not for the original pipeline. This similarity could be a result of the overlapping objects being the most difficult to determine a grasping location for. However, another possible reason might be due to the use of Dexnet as final grasp selector in the hybrid pipeline. So to figure out whether the selection of these erroneous pick locations is a result of Dexnet, another experiment is needed. An experiment where Dexnet is replaced by something else as decisive tool for selecting the best of the locally graspable object locations proposed by the Retinanet network. This experiment has been done and can be found in Appendix F.

F

Experiment 3: Alternative pipeline

F.1. Experiment goal

As described in the conclusion of appendix E some picking errors might be the result of using Dexnet to determine the best global grasp location. Therefore additional experiments are needed to fully evaluate the functionality of the trained Retinanet network as local grasp evaluator by using a different global evaluation method.

F.2. Experiment design

The setup for this experiment is identical to the experiment described in Appendix E so as to accurately compare the pick results. This means that everything is done in the same way; the random and adversarial object piles are assembled to be identical to those in Appendix E; an Ensenso camera is used to create a point cloud and depth image; the segmentation is done by manually creating a mask; and the selected grasp location is send to an UR5 robot arm with Fizyrs swivel gripper as end effector to be executed. The only difference is that a new grasp pipeline, as shown by Figure F.1, is used.

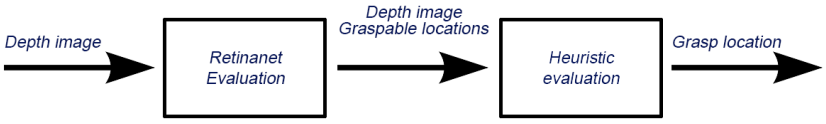


Figure F.1: Schematic overview of the alternative grasp pipeline that uses heuristics to evaluate the locally graspable object locations proposed by the Retinanet network.

The newly proposed grasp pipeline uses the same predetermination of locally graspable object locations as the hybrid pipeline, i.e., the trained Retinanet network. Only the global evaluation needed to be replaced. This was preferably done with a different neural network trained on optimal global object grasp locations, but this was unfortunately not possible due to time required to do so. Therefore the decision was made to use a heuristic rule to determine which of the locally viable grasps was most likely to succeed in a successful pick. This heuristic rule was to use the proposed grasp closest to the objects center of mass as that would lead to the smallest torque around the gripper. However, this is tricky to do as the density and weight distribution of novel object is unknown. Therefore the assumption is made that the whole object consists of the same material so its center of mass lies in its center of geometry. This assumption leads to the following grasp pipeline: first the local graspability is evaluated using Retinanet; then the grasps with the highest confidence scores are passed on to the heuristic evaluation; and finally the grasp location closest to the objects center of geometry, calculated as the mean of all the points in the segmented point cloud, is selected to be executed.

F.3. Results

The setup of the experiment was identical to the one described in Appendix E: 15 cycles containing 148 objects were completed with the same objects placed in the same position and orientation to make the results of the comparison more conclusive. The cycles were again divided as described in Appendix E with the first 12 cycles making up the random setup and the latter three the adversarial setup.

F.3.1. Total evaluation

The newly proposed and evaluated grasp pipeline is significantly faster than both the original dexnet and the new hybrid grasp pipeline with an average duration of 0.226 seconds. However, this is expected as multiple network evaluations are replaced by a simple heuristic selection. What we're most interested in is how the performance changes by removing the Dexnet evaluation as that will give a better picture of the capabilities of the trained Retinanet network to determine the local graspability. The results of all the cycles are shown by Tables F.1 and F.2 together with the results from experiment included for an easier comparison. Interestingly the heuristic based grasp pipeline only fails at picking up three objects and therefore outperforms the hybrid grasp pipeline. However, this is in line with the higher average confidence score for the executed grasps of $\bar{q}_{heuristic} = 0.8694$ compared to $\bar{q}_{hybrid} = 0.7042$

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	134	124	7	3	14
Hybrid pipeline	141	133	5	3	7
Retinanet pipeline	145	139	5	1	3

Table F.1: The number of picked objects for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	90.54 %	83.78 %	4.73 %	2.03 %	9.46 %
Hybrid pipeline	95.27 %	89.86 %	3.38 %	2.03 %	4.73 %
Retinanet pipeline	97.98 %	93.92 %	3.38 %	0.68 %	2.03 %

Table F.2: The success rate of picked objects for each pipeline and the amount of attempts it took to succeed.

Random setup evaluation As mentioned at the start of this section not all objects were placed in the same way. Only the 118 objects placed during the first 12 cycles were placed completely randomly. This placement can have an effect on the grasp success rate and is therefore also evaluated separately from the last cycles. The pick success rate is shown by Tables F.3 and F.4.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	111	102	7	2	7
Hybrid pipeline	114	110	2	2	4
Retinanet pipeline	117	113	4	0	1

Table F.3: The number of picked objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	94.06 %	86.44 %	5.93 %	1.69 %	5.93 %
Hybrid pipeline	96.60 %	93.22 %	1.69 %	1.69 %	3.39 %
Retinanet pipeline	99.15 %	95.76 %	3.39 %	0.00 %	0.85 %

Table F.4: The success rate of picked objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

Adversarial setup evaluation As mentioned at the start of this section not all objects were placed in the same way. The 30 objects placed during the last three cycles were selected and placed in specific poses to evaluate how well the grasp pipelines handled objects that had limited places they could be grasped successfully. The pick success rate is shown by Tables F.5 and F.6.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	23	22	0	1	7
Hybrid pipeline	27	23	3	1	3
Retinanet pipeline	28	26	1	1	2

Table F.5: The number of picked objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	76.66 %	73.33 %	0.00 %	3.33 %	23.33 %
Hybrid pipeline	90.00 %	76.67 %	10.00 %	3.33 %	10.00 %
Retinanet pipeline	93.33 %	86.67 %	3.33 %	3.33 %	6.67 %

Table F.6: The success rate of picked objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

F.3.2. Known object evaluation

Now we're especially interested in how the grasp pipelines perform on novel objects. Unfortunately not all objects used in the experiment were unknown to the Retinanet network. And, while the training dataset is small, likely containing only a single instance of the evaluated object in a different pose, it is best to control for this. So, to display the results in more detail the results have been split for known and unknown objects. The results of the 76 pick attempts on known objects can be found in Tables F.7 and F.8. As expected the average confidence score given by the Retinanet network was higher for the pick attempts on known objects compared to all objects, but not much: $\bar{q}_{known} = 0.9131$ and $\bar{q}_{all} = 0.8695$.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	75	72	2	1	1
Hybrid pipeline	76	75	1	0	0
Retinanet pipeline	76	75	1	0	0

Table F.7: The number of picked known objects for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	98.69 %	94.74 %	2.63 %	1.32 %	1.32 %
Hybrid pipeline	100.00 %	98.68 %	1.32 %	0.00 %	0.00 %
Retinanet pipeline	100.00 %	98.68 %	1.32 %	0.00 %	0.00 %

Table F.8: The success rate of picked known objects for each pipeline and the amount of attempts it took to succeed.

Random setup evaluation The initial 12 cycles contained 64 pick attempts on known objects. The pick success rate for these is shown by Tables F.9 and F.10.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	64	61	2	1	1
Hybrid pipeline	64	63	1	0	0
Retinanet pipeline	64	63	1	0	0

Table F.9: The number of picked known objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	100.00 %	95.31 %	3.13 %	1.56 %	0.00 %
Hybrid pipeline	100.00 %	98.44 %	1.56 %	0.00 %	0.00 %
Retinanet pipeline	100.00 %	98.44 %	1.56 %	0.00 %	0.00 %

Table F.10: The success rate of picked known objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

Adversarial setup evaluation The final three cycles contained 12 pick attempts on known objects. The pick success rate for these is shown by Tables F.11 and F.12.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	11	11	0	0	1
Hybrid pipeline	12	12	0	0	0
Retinanet pipeline	12	12	0	0	0

Table F.11: The number of picked known objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	91.67 %	91.67 %	0.00 %	0.00 %	8.33 %
Hybrid pipeline	100.00 %	100.00 %	0.00 %	0.00 %	0.00 %
Retinanet pipeline	100.00 %	100.00 %	0.00 %	0.00 %	0.00 %

Table F.12: The success rate of picked known objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

F.3.3. Unknown object evaluation

The performance on novel objects is the most important for this report. During the experiment 72 pick attempts were made on unknown objects. The results of these attempts are shown by Tables F.13 and F.14. As expected the average confidence score given by the Retinanet network was lower for the pick attempts on unknown objects compared to all objects, but not much: $\bar{q}_{unknown} = 0.8314$ and $\bar{q}_{all} = 0.8695$.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	59	52	5	2	13
Hybrid pipeline	65	58	4	3	7
Retinanet pipeline	69	64	4	1	3

Table F.13: The number of picked unknown objects for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	81.94 %	72.22 %	6.94 %	2.78 %	18.06 %
Hybrid pipeline	90.28 %	80.56 %	5.56 %	4.17 %	9.72 %
Retinanet pipeline	95.84 %	88.89 %	5.56 %	1.39 %	4.17 %

Table F.14: The success rate of picked unknown objects for each pipeline and the amount of attempts it took to succeed.

Random setup evaluation The initial 12 cycles contained 54 pick attempts on unknown objects. The pick success rate for these is shown by F.15 and F.16.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	47	41	5	1	7
Hybrid pipeline	50	47	1	2	4
Retinanet pipeline	53	50	3	0	1

Table F.15: The number of picked unknown objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	87.04 %	75.93 %	9.26 %	1.85 %	12.96 %
Hybrid pipeline	92.59 %	87.04 %	1.85 %	3.70 %	7.41 %
Retinanet pipeline	98.15 %	92.59 %	5.56 %	0.00 %	1.85 %

Table F.16: The success rate of picked unknown objects, placed randomly, for each pipeline and the amount of attempts it took to succeed.

Adversarial setup evaluation The final three cycles contained 18 pick attempts on unknown objects. The pick success rate for these is shown by Tables F.17 and F.18.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	12	11	0	1	6
Hybrid pipeline	15	11	3	1	3
Retinanet pipeline	16	14	1	1	2

Table F.17: The number of picked unknown objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

	Successful picks	1st attempt	2nd attempt	3rd attempt	Failed picks
Dexnet pipeline	66.67 %	61.11 %	0.00 %	5.56 %	33.33 %
Hybrid pipeline	83.34 %	61.11 %	16.67 %	5.56 %	16.67 %
Retinanet pipeline	88.90 %	77.78 %	5.56 %	5.56 %	11.11 %

Table F.18: The success rate of picked unknown objects, placed in a adversarial way, for each pipeline and the amount of attempts it took to succeed.

F.4. Conclusion

Looking at the experimental results presented above the new pipeline with a heuristic to select a grasp location from the proposed locally graspable locations performed even better than the hybrid pipeline, achieving an almost 96% pick success rate for novel objects. Confirming the suspicion mentioned in the conclusion of Appendix E that the results of the Retinanet network as local evaluator could be even better than that experiment suggested.

Still improvements can be made for better and more consistent results or improved usability. For example color data from the input image can be used in addition to the depth image. This would make it likely that small surface structures such as ridges or different materials that aren't visible in the point cloud can be noticed by the network, reducing errors. Or large pieces of the surface that have the same color and material could be observed which have a large chance of being favorable grasp areas.

An alternative possible improvement would be to make it so a point cloud was directly used as input instead of using a dept map. This would make the resulting network likely instantly usable on a new setup, without issues resulting from the use of a different camera, due to the use of xyz-coordinates.

Bibliography

[1] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL `http://arxiv.org/abs/1405.0312`.

[2] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *CoRR*, abs/1708.02002, 2017. URL `http://arxiv.org/abs/1708.02002`.

[3] Jeffrey Mahler, Matthew Matl, Xinyu Liu, Albert Li, David V. Gealy, and Ken Goldberg. Dex-net 3.0: Computing robust robot suction grasp targets in point clouds using a new analytic model and deep learning. *CoRR*, abs/1709.06670, 2017. URL `http://arxiv.org/abs/1709.06670`.

[4] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.