JAUME I UNIVERSITY

MASTER'S THESIS

Implementation and testing of point cloud based grasping algorithms for object picking

Author:
Nataliya NECHYPORENKO

Supervisors:
Dr. Antonio MORALES
Dr. Olivier KERMORGANT

A thesis submitted in fulfillment of the requirements for the degree of Master's Thesis

in the

Robotic Intelligence Lab
Department of Computer Science and Engineering

July 11, 2017

Declaration of Authorship

I, Nataliya NECHYPORENKO, declare that this thesis titled, "Implementation and testing of point cloud based grasping algorithms for object picking" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

Jaume I University

Abstract

Faculty Name
Department of Computer Science and Engineering

Master's Thesis

Implementation and testing of point cloud based grasping algorithms for object picking

by Nataliya NECHYPORENKO

The purpose of this study is to investigate the most effective methodologies for the grasping of items in an environment where success, robustness and time of the algorithmic computation and its implementation are a key constraint. The study originates from the Amazon Robotics Challenge 2017 (ARC'17) which addresses the problem of automating the picking process in online shopping warehouses. In a real warehouse environment the robot has to deal with restricted visibility and accessibility. The proposed solution to grasping was to retrieve a final position and orientation of the end effector given only sensory information without mesh reconstruction. Two grippers were used: a two finger gripper with a narrow opening width and a vacuum gripper. Antipodal Grasp Identification and Learning (AGILE) and Height Accumulated Features (HAF) methods were chosen for implementation on a two finger gripper due to their ease of applicability, same type of input, and reportedly high success rate. One major contribution of this work was the creation of the Centroid Normals Approach (CNA) method for the vacuum gripper that chooses the most central point cloud grasp location on the flattest part of the object. Since it does not include calculation of orientation, its computation time is faster than the other approaches. It was concluded that CNA should be used on as many objects as possible with both the vacuum gripper and the two finger gripper. A final scheme has been devised to pick up the maximum number of items by combining algorithms on the two different grippers, given the hardware restrictions, to cater to different objects in the challenge.

Acknowledgements

Thank you to my advisors and professors for their academic input and expertise. Also, a big thanks to my friends and colleagues in the lab. The final product for the challenge was a group effort which could not have been completed without mutual dedication, teamwork, and understanding. And of course, thank you to my family whose support and dedication could be felt across the seas.

Contents

List of Figures

List of Tables

List of Abbreviations

ARC'17	Amazon Robotics Challenge 2017
AGILE	Antipodal Grasp Identification and Learning
HAF	Heigh Accumulated Features
ICRA	International Conference on Robotics and Automation
RobIn Lab	Robotic Intelligence Laboratory
RoboCup	Robotic Soccer World Cup
UJI	Jaume I University
CNA	Central Normals Approach
IK	Inverse Kinematics
HOG	Histogram of Oriented Gradients
SAC_RANSAC	RANdom SAmple Consensus
SAC_LMEDS	Least Median of Squares
SAC_MSAC	M-Estimator SAmple Consensus
SAC_RRANSAC	Randomized RANSAC
SAC_RMSAC	Randomized MSAC
SAC_MLESAC	Maximum LikeLihood Estimation SAmple Consensus
SAC_PROSAC	PROgressive SAmple Consensus
OMP	Open Multi-Processing
HOG	Histogram of Gradients

Chapter 1

Introduction

The following chapter presents the background of the ARC'17 competition and its influence on the thesis as well as the approach taken by the UJI RobinLab team in order to tackle the challenge. Many of the decisions to be taken later in the selection of robotic strategies are based on the background of the competition and the restrictions posed by the solution of the team.

1.1 Amazon Robotics Challenge 2017

The Amazon Robotics aims to automate the task of customer order placement and delivery of its products. Amazon's automated warehouses successfully remove the walking and searching for the object but automated picking still remains a difficult challenge. The extraction of the product from a shelf and its packaging in an unstructured environment does not yet have a viable solution to Amazon's taste. In order to spur the advancement of these fundamental technologies, that in the end can be used at warehouse all over the world, Amazon Robotics organizes the Amazon Robotics Challenge. The first two challenges, formerly called the Amazon Picking Challenge, were held at the International Conference on Robotics and Automation (ICRA) 2015 and Robot Soccer World Cup (RoboCup) 2016. Amazon opened the applications for the competition in November 2016 asking for new and improved solutions from industries, universities, and private partnerships alike. The challenge tasks entrants to build their own robot hardware and software that can attempt simplified versions of the general task of picking and stowing items on shelves. The challenge combines object recognition, pose recognition, grasp planning, compliant manipulation, motion planning, task planning, task execution, and error detection and recovery. The challenge event will consist of three tasks; the pick task, the stow task, and the final round task.[1] The robots will be scored based on how many items are picked and stowed in a fixed amount of time. After the competition is complete, the teams share and disseminate their approach to improve future challenge results and industrial implementations.

1.1.1 Items

Figure 1.1 shows the items from the ARC'17 database. The items have been selected not only because of their common occurrence in the Amazon warehouses but also because of their varied form and composition. In terms of vision, the items that are difficult to recognize are the items that have no features or the ones that have generic features that can be applied to more than one object. The only direct features that the Avery binder and the face cloth have are their white color, which can potentially confuse the algorithm and mistake one object for the other. In terms of grasping, the difficulty lies in item dimensions, texture, and point cloud representation. Some items, such as the bath sponge can easily slip through the fingers of the gripper and others, like the mesh cup, lets through vacuum air pressure. Neither the point cloud nor the vision system gives information about the orientation of the object and the reality may not be what the algorithm perceives. These complications call of algorithms that are robust to changes in object orientation and the variety of shapes and textures that it takes on.

FIGURE 1.1: Known items in the ARC'17 dataset [1]

1.1.2 Pick Task

The competition set for the pick task will consist of 32 items that fit in two totes. The team members will stow by hand the competition set items into their storage system. Stowing all the items must take no more than 8 minutes, and all items must be stowed. After all the items are stowed, the judges may adjust the poses of some or all of the items. The judges will create an item location file with the contents of bins A through J, which they will provide to the team. Once the items are stowed, the team will be provided an order file that specifies 10 target items and three orders: a 5 unit order, a 3 unit order, and a 2 unit order. The Robot will then have 15 minutes to pick the three orders. Each order will have an associated cardboard box and target items for the order should be packed into the correct box for the order. The robot may also move non-target items between bins in the storage system if desired.

1.1.3 Stow task

The judges will provide one crowded tote, filled with 20 items in an unstructured jumble, and will place that tote in the position indicated by the team. An example of such tote can be seen in Figure 1.2. The judges will also provide the team an initial item location file. The robot will have 15 minutes to move all the items into the storage system. While the robot is allowed to move the tote, it must manipulate the items to get them out of the tote or into the storage system and may not dump the tote. At the end of the task, the robot will report the final location of each item (stow tote or which bin in the storage system) in the item location file.

FIGURE 1.2: Example totes for stow task

1.1.4 Final round task

The final round task will start with two totes of 16 items each, both of which will contain an even split of new and training items. The team members will stow the first tote by hand in 4 minutes following the same procedure outlined in the pick task. The judges will provide an item location file with the storage bin contents after the manual stow and the contents of the remaining stow tote. The robot will then have a total of 30 minutes for the remainder of the final round task, which can be divided between stow and pick however the team chooses. The robotic stow phase will involve putting away the remaining 16 items into the storage system. The robot will report the final location of each item (stow tote or which bin in the storage system) in the item location file; any errors may not be corrected. The team will then be provided three cardboard boxes and an order file that specifies 10 target items from the 32 in the competition set and three orders: a 5 unit order, a 3 unit order, and a 2 unit order. The robot will then pick the available target 8 of 11 items for each order into the associated cardboard box. Points will be awarded and subtracted using the same rubric as the pick and stow tasks.

1.2 UJI RobinLab enters the competition

The Robotic Intelligence Lab (RobinLab) located at Jaume I University (UJI) has been selected as a finalist for the 2017 competition. UJI's platform is based on Rethink Robotics Baxter. As opposed to some traditional industrial arms, a low-cost dual-arm robot is in line with ARC's spirit about manipulation in a warehouse.

The shelving design is based on a reliable industrial solution. Bins can smoothly slide on a system of free-rotating rollers that are actuated by an external mechanism attached to the robot system. The bins can, in this way, move in and out like drawers, and can be fully taken out of the storage system. Then, the mechanism also allows movement of the bin up or down to the desired height. The robot or the mechanism only has to slide the drawers without moving the shelf itself. The number of bins is optimized so that their overall surface is maximized to avoid cluttering as much as possible. The shelving system has been named Rrupert, and will be referred to by this name. In order to complete the challenge specifications, the hardware and

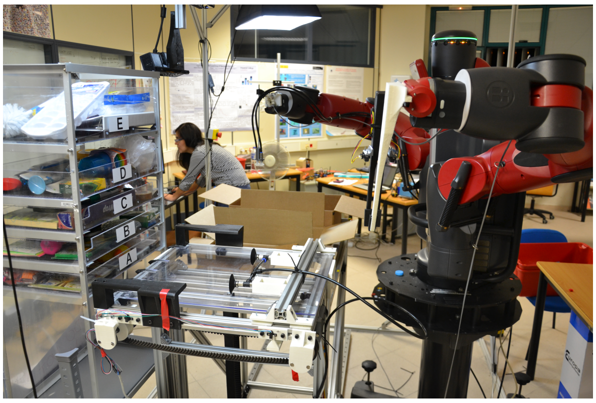


FIGURE 1.3: Robot, shelving, and storage system of the UJI RobinLab team

software components have been split among the members of the UJI RobinLab team. The team has been split into several Work Packages (WP) with each one responsible for a part of the competition, as shown in Table 1.1. Both the implementation and evaluation have been deemed important to the design and development process. Each team is essential to the functionality of the full system. The aim is to complete the pick and stow tasks during the ARC'17 with the most success and hence with the most accumulated points.

1.3 Thesis summary

The topic addressed in the following thesis will be the topic of grasping. The architectures of Figures 1.4 and 1.5 show where the task of grasping fits within the system. The grasping algorithm receives as input from the vision team, the object that has been identified and the location of the object through its approximate point cloud. The algorithm has to output the position and orientation of one or several grasps

TABLE 1.1: UJI RobinLab Work Packages

WP	Responsibility
1	Task Planning Architecture Integration
2	Object Recognition
3	**Grasping**
4	Motion Planning
5	Hardware Design
6	Performance Evaluation
7	Logistics
8	Management

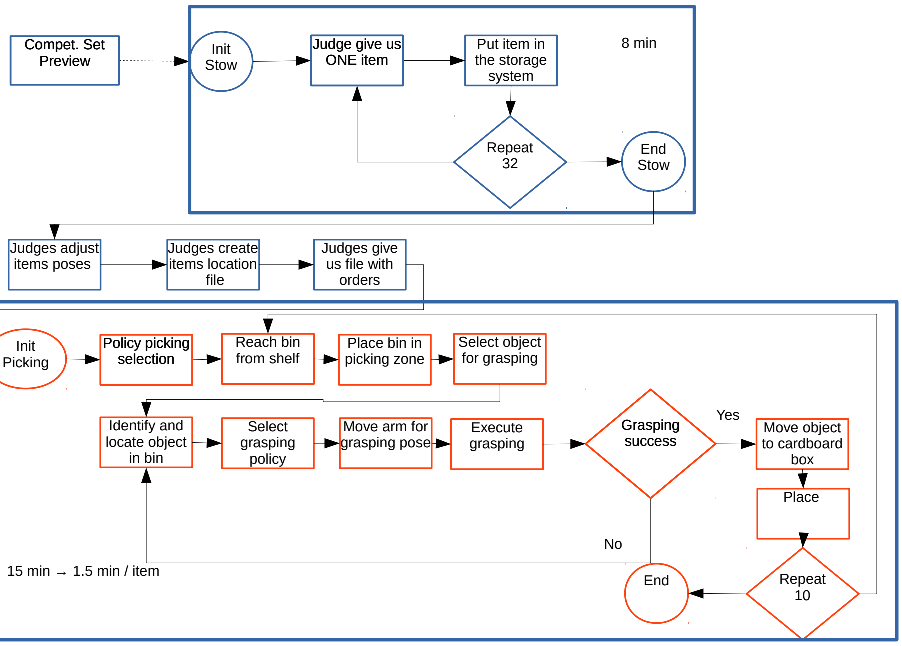


FIGURE 1.4: Pick task software architecture

that can be sent to the manipulation algorithm. The robot then has to move the arm such that the end effector ends up in the desired location to grasp the object. Two grippers have been mounted on the Baxter robot. The first gripper is a two finger gripper with a limited opening width, for which it has been named the Pincher. The second gripper is a vacuum gripper that uses air pressure to pick up objects. The goal of the following work is to implement software in order to make the most use of the grippers.

The first two contributions deal with algorithms. A two finger method, as will be seen in the next chapter, has been widely researched and thus it remained to implement current work on the robot. AGILE and HAF grasping have been chosen since they both use a point cloud as an input and a grasp position and orientation as output. One contribution was the implementation of Antipodal Grasp Identification and Learning (AGILE) and Height Accumulated Features (HAF) grasping and quantitatively comparing their computation time and qualitatively comparing their robustness and success. Through simulation and arm approach, it was shown that AGILE was less robust and took on average 3s more than HAF grasping. HAF

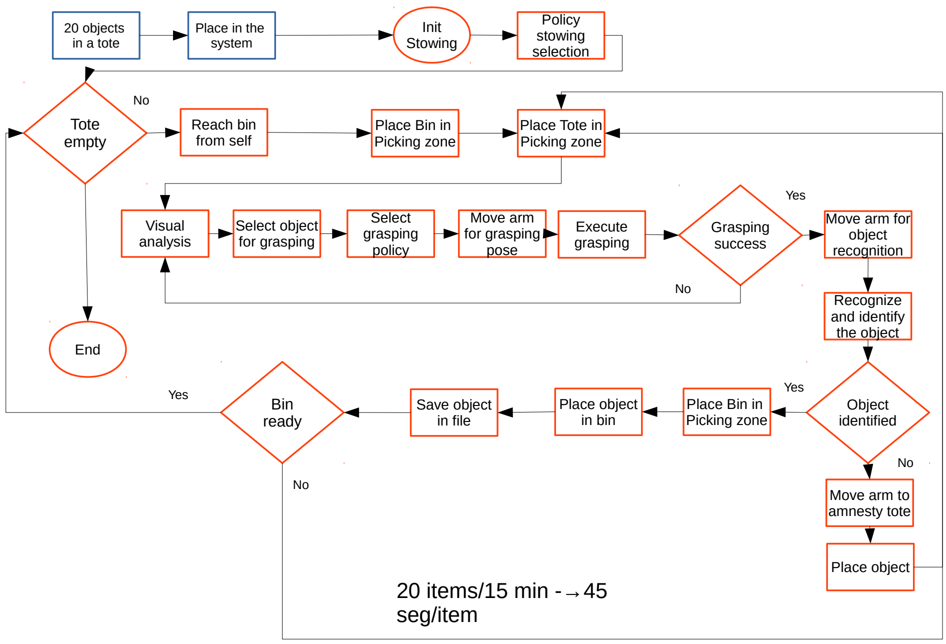


FIGURE 1.5: Stow task software architecture

grasping gave fewer grasp options with only vertical grasps but showed better robustness and computational speed. Table 1.2 shows the strengths and weaknesses of the methods in comparison with each other. Given the results, AGILE was eliminated from the system without full grasp testing. Once the HAF algorithm has been more thoroughly tested with the robot, it was also shown that many grasp positions and orientations are not feasible due to hardware restrictions of the robot arm and the robot gripper.

TABLE 1.2: Comparing AGILE and HAF

Criteria	AGILE	HAF
Time	-	+
Robustness	-	+
Success	+	+

The second major contribution was the creation of the Centroid Normals Approach(CNA) algorithm. It uses the point cloud and the major graspable component of the object in order to find the centroid and its normals in the flattest part of the point cloud of the object. Then grasps are rotated around the vertical z-axis such that the final grasp is most comfortable for the Baxter robot. This algorithm was used with the vacuum gripper for items with a nonporous texture and with the two finger gripper for soft items that can be "pinched."

The final contribution is to match the competition objects to the grasping techniques and grippers to maximize the number of points for the team. Table 4.7 has been put together after observation and quantitative analysis of successful trials of pick and stow tasks. The tests have been done in a real environment where the grasping algorithms have to perform from within the system integration in the final warehouse-like setting. Various charts expose problems that grasping may encounter when being fully integrated, adding another level to analysis of grasping success. But this table also exposes the analysis that one universal algorithm for one universal gripper has not been found and currently various algorithms for various grippers was the best solution. Demonstration videos are available online. [2]

1.4 Thesis outline

Given the introduction for the motivation of the thesis and its background information, the next chapter, Chapter 2, will present the most relevant research works targeted at solving grasping problems similar to those presented by ARC'17. After presenting the reader with the status of the current research, the justification for the selected algorithm will be given. Chapter 3, then explores chosen algorithms more in depth and presents the hardware that will be executing the algorithms. With the theoretical knowledge of the approaches, Chapter 4 leads into the experimental setup. The tests are presented in a systematic manner such that the given procedures can be replicated and confirmed. The experimentation aims to isolate the algorithms and compare their strength and weaknesses such that the best approach to the grasping problem of ARC'17 can be chosen. The results of such experimentation are presented concurrently with procedures. Chapter 5 leads the discussion of the results and the proposed solution to optimize the success of the grasping problem of the competition. Finally, Chapter 6 concludes the work and identifies further ideas to complement or extend the work that has been documented in the thesis.

Chapter 2

Robotic Grasping

The following chapter introduces the topic of robotic grasping. The first part defines grasping and the categorization of methodologies. The definitions have been included in order to establish a basis for the work presented in further chapters. The second part introduces the state of the art. The papers will be perceived in light of the restrictions posed by the competition. If the materials or the requirements of the grasping strategy presented in the publication is seen as incompatible then the strategy will not be considered for further exploration and implementation.

2.1 Terminology

The solution of grasping from a bin or a tote may seem specific however it has a broad range of applications. The solution to this problem can help as much in a warehouse as in agriculture picking fruit or in a hospital helping with medical equipment. As of now, robots have been successfully integrated into computer-controlled electromechanical devices into a wide variety of industrial environments. Routinely incorporated for mundane tasks of welding and painting car bodies on assembly lines, or stuffing printed circuit boards with IC components. But the future of robotics lies in its versatility and the abilities to be able to perform tasks such as inspecting and repairing structures in nuclear, undersea, and underground environments, and even picking oranges and harvesting grapes in agriculture. In order to perform such tasks, the robot needs to be able to adapt to any object in the environment or use the given manipulator in many ways. The fascination with manipulator adaptability has lead researchers to create anthropomorphic hands such as the Salisbury Hand (also known as the Stanford/JPL hand), the Utah/MIT hand, the NYU hand, and the research hand Styx.[17] However, the control of such hand, as well as the further need to investigate lightweight actuators has led many researchers to simplify the problem into a simple pinching system because," for most objects, there is typically a small region that a human (using a two-fingered pinch grasp) would choose to grasp it."[26]. So, as can be seen in Figure 2.1, the complexity of the robotic grasp can often be avoided because in order to hold an object enough force can be provided on either side of the object to constrain its motion. However, as will be explained later, the two finger grasp will not always ensure stability of the object and its ability to withstand external disturbances. The investigation of robotic grasping and robotic manipulation gives a great appreciation of the incredible power and subtlety of our own biological motor control systems.

A *grasp* is commonly defined as a set of contacts on the surface of the object, which purpose is to constrain the potential movements of the object in the event of external disturbances. *Grasp synthesis* refers to the problem of finding a grasp configuration that satisfies a set of criteria relevant for the gasping task.[4] The set of

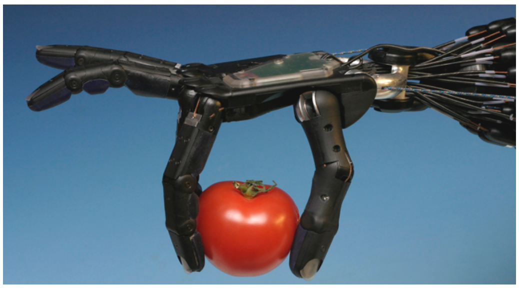


FIGURE 2.1: Two finger grasp [29]

contacts defining each grasp can be analyzed in order to test the grasp's ability to resist disturbances and its dexterity properties. As it is presented afterwards, the grasps that can be maintained for every possible disturbing load are known as *closure grasps*. A grasp on an object is a *force closure grasp* if and only if we can exert, through the set of contacts, arbitrary force and moment on this object. Equivalently, any motion of the object is resisted by a contact force, that is the object cannot break contact with the finger tips without some non-zero external work. [21] Form closure is related with the ability of constraining devices to prevent motions of the grasped object, relying only on unilateral, frictionless contact constraints. Having a form closure grasp is a necessary and sufficient condition for a force closure grasp. During a task execution, the grasping fingers must be controlled so that the grasp processes dexterity, equilibrium, stability and dynamic behavior.[27] These characteristics are an essential and baseline measurement of the grasp and need to be considered when evaluating different kinds of grasps.

Recent analysis and surveys have divided the methodology for constructing grasp synthesis into two categories: *analytic* and d*data-driven*.[4] Analytic refers to methods that construct force- closure grasps with a multi fingered robotic hand that are dexterous, in equilibrium, stable, and exhibit a certain dynamic behavior. This is a mathematical and geometric approach that often requires a 3D model of the object. Empirical or data-driven approaches rely on sampling grasp candidates for an object and ranking them according to a specific metric. This process is usually based on some existing grasp experience that can be a heuristic or is generated in simulation or on a real robot.

The work on data-driven grasp synthesis has been reviewed and the methodologies for sampling and ranking candidate grasps.The approaches have been divided into three groups based on whether they synthesize grasps for known, familiar, or unknown objects.[4] In the case of known objects, the approaches are based on object recognition and pose estimation. In the case of familiar objects, the techniques uses some form of a similarity matching to a set of previously encountered objects. Finally, for the approaches dealing with unknown objects, the core part is the extraction of specific features that are indicative of good grasps.

Figure 2.2 exemplifies a physical interpretation of a geometric grasp. The left-hand side shows the grasping as a combination of points p_1, p_2, p_3 and the right-hand side shows its relation to the center of mass. Knowing the configuration, it's possible to determine the quality and feasibility of the grasp. On the other hand, Figure 2.3 shows the data-driven approach with an analysis of the image as a set of features. Given these features, the output is a rectangle that gives the proper location of the grasp after a set of network training. In order to better understand the candidate algorithms that use either analytic or data-driven approach, it is important to perform a literature review which is the focus of the next section.

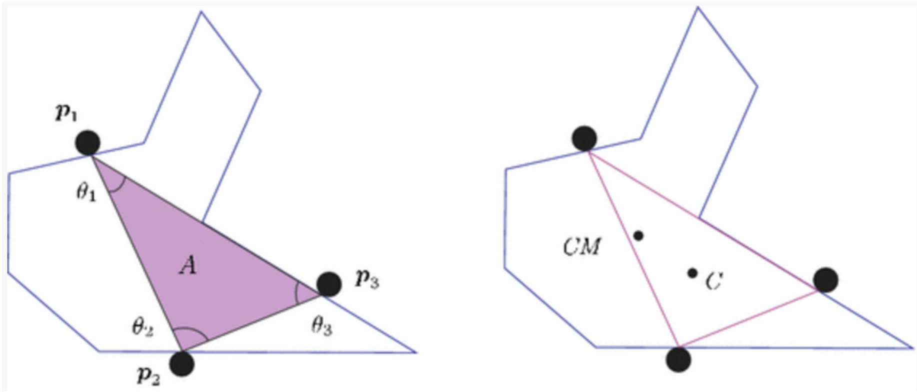


FIGURE 2.2: Analytic approach to grasp synthesis computation [24]

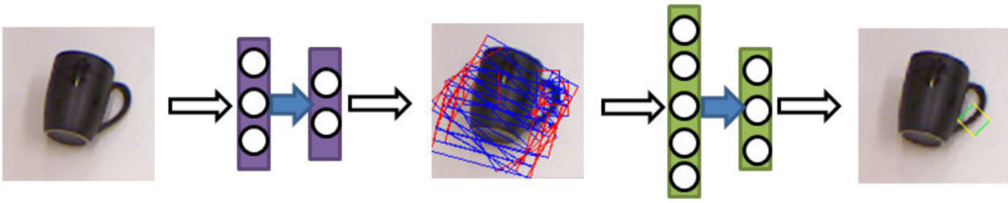


FIGURE 2.3: Data-driven approach to grasp synthesis computation[15]

2.2 State of the art

Robot autonomy requires adaptation to scenarios that have not been hard coded into behavioral decisions. In terms of grasping, this means that the robot needs to be able to synthesize a grasp of an object that has not been specifically added into its grasping database. When the programmer deals with information received from sensors and generates a set of grasps for consideration has been termed a *data-driven* approach.[4] The generation, evaluation, and selection of grasps can be done in various ways but the following review will explore research on the topic of robotic grasping such that the work is applicable to object picking from a tote or a bin in.

While navigating through the environment, a robot encounters different objects which the robot need not identify in order to perform a grasp in the same way that a human need not know what brand is the aluminum can in order to pick it up, all the human needs to know is how to pick up a can. Therefore handling object data for object identification purposes must be different from handling data for grasping purposes, especially since objects that are in the same category and perform the same task can be grasped similarly thus simplifying the problem and generalizing robot behavior that is suitable for autonomy. Strict category analysis had been performed using data analysis.[9] Furthermore, the may need to know the task for which the object will be used for example a mug can be used for pouring, which requires dexterity and stability hence forcing the final selected grasp to naturally be the one with the highest rank. Task-based grasping has been separately studied in context of Bayesian networks for encoding the probabilistic relations among various task-relevant variables.[28] The synthesis of category and task has been performed based on 2D and 3D data from low-level features.[18] [14] Instead of relying on sensor data points the proposition is to synthesize grasps based on semantics, which are stable grasps that are functionally suitable for specific object manipulation tasks that use overall object features.[8] Most recently, The role of robustness and adaptability inside semantic grasping has been addressed by noting that shape variations inside the category and transferring semantic grasps between objects needs to be addressed by adding probabilistic framework to use online sensory information for grasp planning, which allows to refine object's pose based on tactile feedback and modifying the grasp accordingly.[22]

A simple object analysis based on shape has been proposed in order to pick an object no matter the orientation. One study uses shape primitives like spheres, cones and boxes to approximate object shape and used the simulation environment GraspIt but not on a real life robot application.[20] Saxena et al. proposed supervised learning with local patch-based image and depth features for grasping novel objects in cluttered environments.[26] This path led towards exploring the idea of grasping as a method for finding graspable features in the incoming data. Using the heights of objects as features has in particular been a topic of theoretical study [25] [11] and complete implementation on various robots.[12] [30] Features of geometry have been explored by ten Pas et al. and shows a unique combination of analytical understanding and data-driven applications.[23] It has been inspired by the autonomous checkout robot but is supplemented with an SVM learning mechanism.[16] Another approach also uses features but rather than those of the 3D sensor, relies on supervised deep learning of 2D RGB images.[15] This method requires a large dataset and large amount fo training hours. The greater the network the better the result however the slower the timing during online execution.

2.3 Methodologies for implementation

Given the previously demonstrated literature review, it is then necessary to select the given research material for application. The object categorization idea and task analysis uses information that is not required for the challenge as the robot will not need to use the object but rather only lift and place. The methodologies focusing on specialized picking such as tool use were discarded.

The next main topic was grasping based on calculation of grasp points on a 3D mesh object. It was an inspiring concept since these methodologies use directly the sensor data either from RGB-D images or a point cloud and then fit the viewed object with a real model. In the Amazon Fulfillment Center, the objects vary greatly in orientation, size and shape. Loading a whole database full of these objects would be a grand task plus does not grantee success in a clutter. Implementations focusing on neural networks and 3D object matching were also discarded.

Given this analysis, it is natural to move in a direction of a paper that was able to demonstrate results in the scenario of many object placed in unpredictable locations as it happens in a dense clutter. The work presented by Saxena et al. focused on a collection of local visual features. The author later supplemented the idea of object features to grasps with machine learning. Then Fischinger and Vincze develop another idea of features through height-maps where they report indicates a 92% single object grasp success rate while taking only 2-3s of time.[30] The application to Baxter shows especially successful results, which is important given that Baxter is a platform with a simple gripper and only 1cm precision. The research done on the method shows robustness, repeatability, speed, and ease of access since the work is an open source code on Github. If this project can be implemented, tuned, and improved then the success rate would be high enough for presentation during the competition. Thus HAF grasping as presented by Fischinger et al. has been chosen for implantation and demonstration. One of the disadvantages of the work is that it has been mainly tested with vertical only grasps. This approach does not often work if the object is inside the tote with multiple orientations. Hence it would be useful to supplement and compare the algorithm with another one that can provide a wide range of grasp orientations.

The approach that needs to be provided with the same input information but possibly hundreds of combinations of grasps is one that uses geometry representative features.[13] The output of the function is also geared towards a two finger gripper and outputs a position and orientation of the gripper. So, the second chosen algorithm for testing will be the AGILE grasping algorithm presented by ten Pas et al.[23]

Finally, the mechanism for a vacuum gripper in combination with vision has been previously tested in warehouse and competition environments. The results of this approach have appeared in previous ARC competitions and have been briefly described or presented with complete practical and theoretical analysis.[7] [10] [31] However, the algorithm will be prepared from scratch in order to match it the current system and gripper requirements. The three algorithms will be thoroughly explained in the upcoming chapter of the report. These algorithms will be used to maximize the picking and stowing ability of the Baxter robot for the competition.

Chapter 3

Methodology

3.1 Software

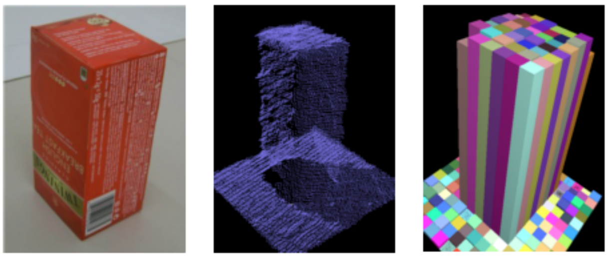
The following section provides theoretical knowledge for the algorithms that have been chosen for implementation. From a bird's eye perspective, the separate algorithms have their individual environments in which each one can exhibit its strength. HAF grasping takes into account the height of the objects and is used with a top grasp thereby reducing the dimensionality of the object. AGILE grasping explores the geometry of the whole object in order to find handle-like sections to exploit for grasping and thus it's often aiming at side grasps. The suction algorithm is computationally less heavy and thus faster and more capable of grasping very flat objects like books for which a two-finger gripper with limited opening width cannot be used.

3.1.1 HAF

As the name suggests, the HAF algorithm utilizes the heights of surface points, gathered from the point cloud data, relative to their neighbors in order to learn how to grasp the objects. The authors stress three important advantages of the algorithm; *segmentation independent*, *integrated path planning*, and use of *known depth regions*.[11] [30] The first point stresses the fact that the objects do not need to be separate from each other in order to be picked. The height of the features supersedes the need to know where and on what objects these features are located. Once the features are located then the second point stresses the idea that the heights take into account the surfaces near them so the final grasp will naturally be collision free. The final point takes into account previously explored methods and states that object reconstruction in cluttered scenes is unreliable and the HAF grasping method avoid this by only using given data from the sensors without estimation.

First, it's important to address the idea behind the method. Let's assume that there is a pile of objects inside a box that is standing in front of the robot. The robot sees this pile of objects with a Kinect 2.0 camera and has the mission to unstack the items. If the robot has the HAF grasping algorithm installed then the pile of objects that a human would see would transform into a grid of heights. This method exploits the idea that the highest protruding part of a scene is the most graspable part of a protruding object. The peak height that has much lower heights surrounding it allows the gripper to go down without collisions until the next highest region. By picking out peaks from a mountain range and grasping, the robot is able to recursively unload the box item by item.

Now for a more in-depth analysis of the algorithm, the acronym will be explained in terms of the letters it contains. In order to facilitate understanding, the order will be H, F, and then A.



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

FIGURE 3.1: Height processing [11]

The term *height* refers to the measure of the perpendicular distance from the table plane to the points on the top surface of the object. The input point cloud is first discretized and the height grid H now contains a 1x1cm^2 cell that saves the highest z-valued points with corresponding x and y values. [11]

In computer vision and image processing, a feature is a piece of information which is relevant for solving the computational task related to a certain application. It is equivalent to the feature used in machine learning and pattern recognition, though image processing has a very sophisticated collection of features. Features may be specific structures in the image such as points, edges or objects. Features may also be the result of a general neighborhood operation or feature detection applied to the image. A specific image feature, defined in terms of a specific structure in the image data, can often be represented in different ways. HAF features are defined similarly to Haar Basis functions. Figure 3.2 shows example HAF features with 2 overlapping, 3 disjunct, and 4 overlapping regions. The first defines two overlapping rectangular regions. R_1 contains all the cells of red assuming to interpolate underneath green as well as the red cells. R_2 contains solely the green. All height

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

FIGURE 3.2: HAF feature examples [11]

grid values of each region R_i, on a height grid H, are summed up. The sums r_i are individually weighted by w_i and then summed up. The regions and weights are dependent on the HAF feature that are defined by an SVM classification. A feature value f_i is defined as the weighted sum of all regions. The j^th HAF value f_i is calculated as:

$$f_j = \sum_{i=1}^{nrRegions_j} w_{i,j} \cdot r_{i,j} \tag{3.1}$$

$$r_{i,j} = \sum_{k,l \in \mathbb{N}: H(k,l) \in R_{i,j}} H(k,l) \tag{3.2}$$

The HAF vector f contains the sequence of HAF values.

$$\boldsymbol{f} = (f_1, f_2, ..., f_{nrFeatures}) \tag{3.3}$$

The paper claims to have tested 71,000 features (70,000 of which are automatically generated) and finally selected 300 to 325 with an F-score selection. [30]

The height grid is modified for computational efficiency. In each location (x,y) of AH contains above and to the left of (x,y) in the grid. Using height accumulated rectangular regions, each region sum can be computed with four or fewer array references.

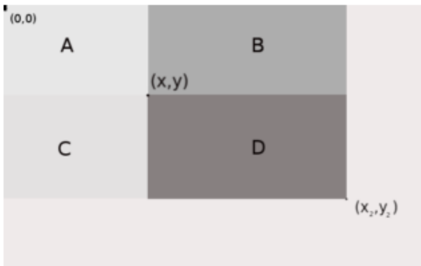


FIGURE 3.3: Accumulation grid [11]

$$AH(x,y) = \sum_{x' \leq x, y' \leq y} H(x',y') \tag{3.4}$$

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

$$AH(x,y) = \sum_{x' \leq x, y' \leq y} H(x',y') \tag{3.5}$$

After implementation, the visualization on Figure 3.4 shows the calculations in process. Bigger gray rectangle: indicates the area where heights can be used for grasp

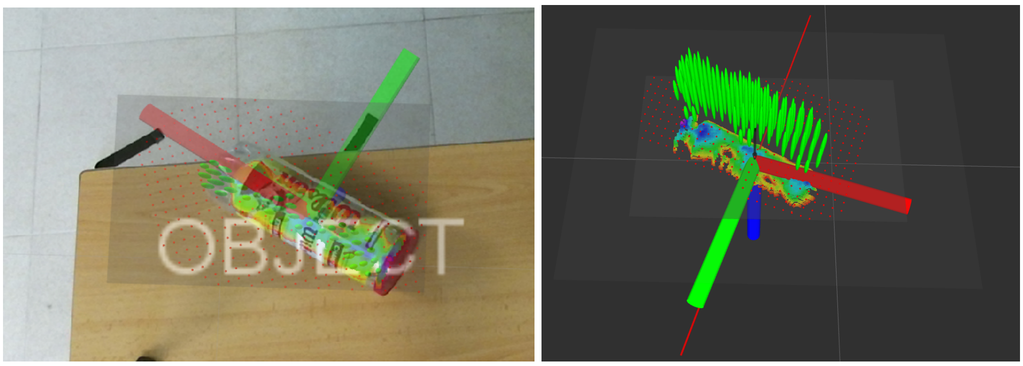


FIGURE 3.4: (left) HAF visualization on tennis ball container. view from the Kinect 2.0. (right) view from Baxter.

calculation. Inner gray rectangle: defines the area which is searched for potential grasps (grasp centers). Long red line: indicates the closing direction (for a two finger gripper) Red/green spots: indicate the positions where grasps are really tested for the current gripper roll (ignoring points where no calculation is needed, e.g. no data there) Green bars: indicate identified potential grasps available. The height of the bars indicates the grasp evaluation score (the higher the better). Finally, the green, red and blue frame represent the final grasp hypothesis chosen by the algorithm and indicates the final position as to where the end effector should go.

3.1.2 AGILE

AGILE grasping is an algorithm that uses a point cloud to predict the presence of geometric conditions that are indicative of good grasps on an object.[23] First, geometry is used to reduce the size of the sample space by applying the condition that for a grasp to exist the hand must be collision free and part of the object surface must be contained between two fingers. Then, the remaining grasps are classified using machine learning for which geometry is used in order to automatically label the training set. Specifically, the antipodal condition is used in order to in order to label a set of grasp hypothesis in arbitrary point clouds of ordinary objects. A pair of point contacts with friction is *antipodal* if and only if the line connecting the contact points lies inside both friction cones.[21] This is a sufficient condition because if an antipodal grasp exists then the robot can hold the object by applying sufficiently large forces along the line connecting the two contact points. A *friction cone* describes the space of normal and frictional forces that a point contact with friction can apply to the contacted surface. Grasp geometry is quantified by certain parameters. The reason that this algorithm is easy to implement is that these parameters are easy to tune depending on the dimensions of the two finger gripper. The gripper is specified by the parameters $\theta = (\theta_l, \theta_w, \theta_d, \theta_t)$ which respectively stand for gripper length, width, the distance between two fingers, and the thickness of fingers, Fig. 3.5a. The closing region is the volumetric region swept out by the fingers when they close. The hand $h \in H$ is a parallel jaw gripper comprised of two parallel fingers each modeled as a rectangular prism that moves parallel to a common plane. $\hat{a}(h)$ is a unit vector orthogonal to this plane. And if $r(h) \in R(h)$ is an arbitrary reference point inside the closing region then the *closing plane*, $C(h)$, is the subset of the plane that intersects $r(h)$ and is orthogonal to $\hat{a}(h)$ and is contained within $R(h)$:

$$C(h) = p \in R(h) | (p - r(h))^T \hat{a}(h) = 0 \tag{3.6}$$

The *cutting plane*, Fig. 3.5b, will be the plane orthogonal to the minimum principal curvature at a point on the surface of the object, and passing through p which is a point where the calculations have determined to have a frame aligned with the surface normal on the object. It is important to find many points p heuristically after fitting the maximum number of points to a quadratic surface for which the author uses Taubin's method. Having the previously setup definitions it is possible to define the first set of sampling of hands that the author collects given a point cloud.

1. The body of the hand is not in collision with the point cloud.

2. The hand closing plane contains p.

3. The closing plane of the hand is parallel to the cutting plane at p.

The method, as mentioned before relies on features. The second part of AGILE grasping, classification of hand hypothesis does just this with a feature descriptor of a hand hypothesis as seen on Figure 3.6. A feature descriptor is a representation

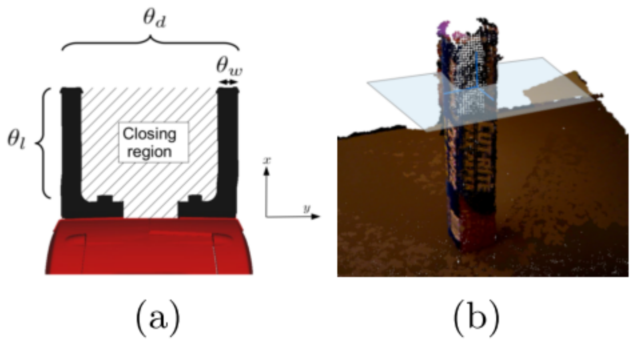


FIGURE 3.5: (a) hand geometry. (b) cutting plane geometry. [23]

of an image or an image patch that simplifies the image by extracting useful information and throwing away extraneous information. In the Histogram of Gradients (HOG) feature descriptor, the distribution (histograms) of directions of gradients (oriented gradients) are used as features. Gradients (x and y derivatives) of an image are useful because the magnitude of gradients is large around edges and corners (regions of abrupt intensity changes) and edges and corners pack in a lot more information about object shape than flat regions. [19] Given an example point cloud, once the three constraints mentioned above are satisfied, feature descriptors are extracted and included in the training dataset. In this case, the labeled dataset is created with many samples and is much faster than creating a hand labeled dataset.

The results of the implementation can be seen on Figure 4.3c,d.

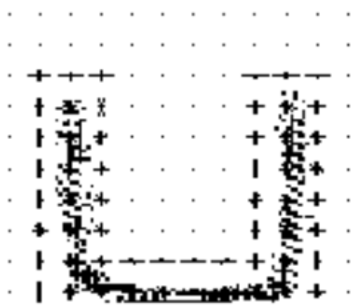


FIGURE 3.6: HOG feature representation of a hand hypothesis[23]

3.1.3 CNA

The idea behind Centroid Normals Approach (CNA) comes from the observation that most objects in the ARC'17 dataset are symmetric and have a large central surface that is most suitable for creating an air sealed grasp. The logic flow is presented on Figure 3.7. The main idea is to receive a point cloud and downsample it using a voxel grid. Then extract a cylinder or a plane, depending on the most prominent object shape. Finally, using the extracted shape, find grasps located in the center of it using surface normals and euler to quaternion rotations. The VoxelGrid class creates a 3D voxel grid. A voxel represents a value on a regular grid in three-dimensional space and a voxel grid is a division of 3D space into voxels to fully represent the given space. In this case, the given space is covered by the input point cloud data.

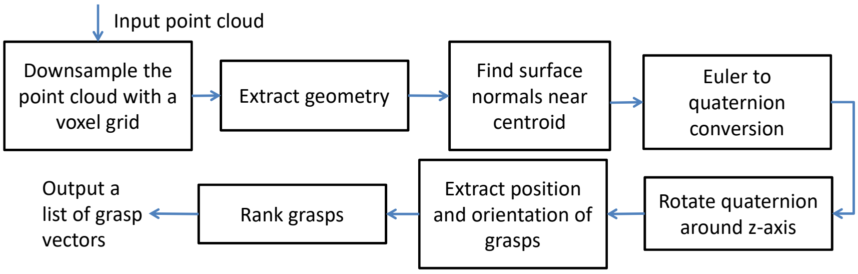


FIGURE 3.7: CNA logic flow

Then, in each voxel (i.e., 3D box), all the points present will be approximated (i.e., downsampled) with their centroid. This approach is a bit slower than approximating them with the center of the voxel, but it represents the underlying surface more accurately.

The PCL library contains the following options:

- SAC_RANSAC - RANdom SAmple Consensus
- SAC_LMEDS - Least Median of Squares
- SAC_MSAC - M-Estimator SAmple Consensus
- SAC_RRANSAC - Randomized RANSAC
- SAC_RMSAC - Randomized MSAC
- SAC_MLESAC - Maximum LikeLihood Estimation SAmple Consensus
- SAC_PROSAC - PROgressive SAmple Consensus

The abbreviation of "RANdom SAmple Consensus" is RANSAC, and it is an iterative method that is used to estimate parameters of a mathematical model from a set of data containing outliers. The RANSAC algorithm assumes that all of the data we are looking at is comprised of both inliers and outliers. Inliers can be explained by a model with a particular set of parameter values, while outliers do not fit that model in any circumstance. Another necessary assumption is that a procedure which can optimally estimate the parameters of the chosen model from the data is available. MSAC is an extension of RANSAC. It adopts the same sampling strategy to generate putative solutions but chooses the solution to maximize the likelihood rather than just the number of inliers. The randomized methods achieve computational efficiency by evaluating only a fraction of data points for models contaminated with outliers. The PROSAC algorithm exploits the ordering structure of the set of tentative correspondences, assuming that the ordering by similarity computed on local descriptors is better than random. A thorough comparison based on accuracy, computing, time and robustness has been done between RANSAC and its descendants as well as other consensus models by Choi et al. and Chum et al. [5] [6] The challenge places an important role in robustness because the incoming point cloud is often saturated with noise and other elements as well as speed since the team gets bonus points for completing the test on time. The results presented by the authors show that for good time and robustness PROSAC performs best, hence it was chosen in the methodology.

Once the relevant shape has been extracted which would be planar for a book and cylindrical for the tennis ball container, the algorithm finds the centroid or the 3D average of all the points fed into it. Then it calculates the surface normals closest to the centroid and stores. The closeness is achieved by the L^2 norm. This gives the ordinary distance from the origin to the point x, a consequence of the Pythagorean theorem.

$$\|\mathbf{x}\|_2 := \sqrt{x_1^2 + ... + x_n^2}. \tag{3.7}$$

The distance is thresholded by the maximum possible distance from the centroid. Which means the distance d_{L_2} of vector x_i, y_i, z_i from the centroid x_c, y_c, z_c is:

$$\|\mathbf{d_{L_2}}\| := \sqrt{|x_i - x_c|^2 + |y_i - y_c|^2 + |z_i - z_c|^2} \tag{3.8}$$

Then position of the normals is directly put in as the desired resulting positions of the end effector. The Euler orientations of the normals are converted to quaternions according to the following strategy. Any rotation matrix can be given as a composition of rotations about three axes,R_x, R_y, R_z with a rotation angle θ, and thus can be represent a 3×3 matrix operating on a vector.

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) \\ 0 & sin(\theta) & cos(\theta) \end{bmatrix} \tag{3.9}$$

$$R_y(\theta) = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{3.10}$$

$$R_z(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.11}$$

For the pick task the idea is to have the robot appraoch as vertical as possible. However, the stow task is different since the objects might be tightly pressed against the wall. For the stow task the bin has been divided into South (W_1), North (W_2), East (W_3), West (W_4) and Planar grasp orientations which originate from the sections of the tote as seen on Figure 3.8. The idea behind this classifications comes from the results of the normals. The normals provide a vector that has the z axis normal to

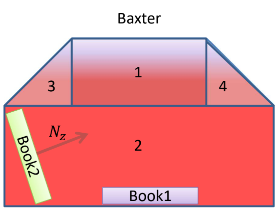


FIGURE 3.8: Labeling of the tote

the surface however the x and y axis are random based on the computation and reconstruction of the surface. The orientation of the original Baxter gripper and most other arms does not matter however it does matter in the case of the relocated gripper that the RobinLab team constructed. (These difficulties will be addressed more closely in the Hardware section)The gripper cannot rotate 360 degrees and the position of the grasp influences heavily the position of the gripper. Four basis frames have been established, going out about 45° from the surface of each wall. And from the basis frames the robot has an option of +/- 30°. In this case the IK always has an option of the arm orientation and can discard certain grasps solely because the gripper will have an unfeasible compatibility with the Baxter arm. This is especially useful, as seen on Figure 3.8, when Book2 is very close to the wall. The way to find out which wall is being addressed has been done through statistical analysis of the components of the quaternion, $q = q_w + q_x\mathbf{i} + q_y\mathbf{j} + q_z\mathbf{k}$. Knowing the location and threshold C of how tilted the grasp needs to be to transfer from Planar to a Wall position, the following will classification roughly estimates the orientation of the normal within the bin:

$$\begin{aligned} W_1 &:= q_w, q_z > C \\ W_2 &:= q_w, q_z < -C \\ W_3 &:= q_w > C, q_z \approx 0 \\ W_4 &:= q_w < -C, q_z \approx 0 \end{aligned}$$

The quaternion rotation about the z axis has been performed according to the following principle: Let $\overrightarrow{u}$ be a unit vector (the rotation axis),α the angular rotation, then $q = cos\frac{\alpha}{2} + \overrightarrow{u}sin\frac{\alpha}{2}$. It can be shown that:

$$\overrightarrow{v'} = q\overrightarrow{v}q^{-1} = (cos\frac{\alpha}{2} + \overrightarrow{u}sin\frac{\alpha}{2})\overrightarrow{v}(cos\frac{\alpha}{2} - \overrightarrow{u}sin\frac{\alpha}{2}) \tag{3.12}$$

Two rotation quaternions can be combined into one equivalent quaternion by the relation:

$$q' = q_2 q_1 \tag{3.13}$$

In which q' corresponds to the rotation q_1 followed by the rotation q_2. Another role is the location within the tote. The tilted grasp would be sent to the gripper only if it's tilt is not too close to the walls. This appears in the algorithm as a simple threshold.

Finally, a scoring algorithm has been devised in order to sort the grasps with the first one in the list being the one that IK will try to solve and have the Baxter reach and the last one being the last option to reach. The object is most likely located at the center of the point cloud and the surface to be grasped which is most likely to avoid all other collisions will be the top one. The grasp score S_i is calculated as follows. Knowing the pose of all the grasps, first find the average of all the x and y points, then subtract the x_i and y_i values of the current grasp from the averages, x_{av} and $y_{av}$to find how far away it is from the center. $x_{dist} = |x_{av} - x_i|$ and consequently $y_{dist} = |y_{av} - y_i|$. The further the distance, the more of an outlier the grasp is so the score will be higher and the grasp will less likely to be selected first. The next in consideration is the z which will be subtracted from the previous score. The higher the z, the more of a subtraction and hence the lower the score and the more likely it is that the grasp will be selected first. The w values are scalars that can adjust how much each score should matter. In the case applied for the ARC'17 system, $w_x = w_y = .7$ and $w_z = .3$.

$$S_i = (w_x, w_y) \cdot (x_{dist}, y_{dist}) - (w_z) \cdot (z_i) \tag{3.14}$$

The right-hand side of Figure 3.9 shows the input point cloud of the mesh cup. Note that the top is the part of the cup that is most likely to be grasped hence the algorithm is assigned to extract the planar part of the object. The right-hand side shows the same point cloud but with the violet dots showing the post-processed point cloud, the yellow lines showing the surface normals, the green circle shows the x and y position of the centroid of the point cloud, and the purple central surface normals around the centroid are the CNA vectors which have the relevant z-axis that will be used for processing of the outputted grasps. The frame on the right-hand side shows one of the final grasp frames for the end effector to reach in order to grasp the object.

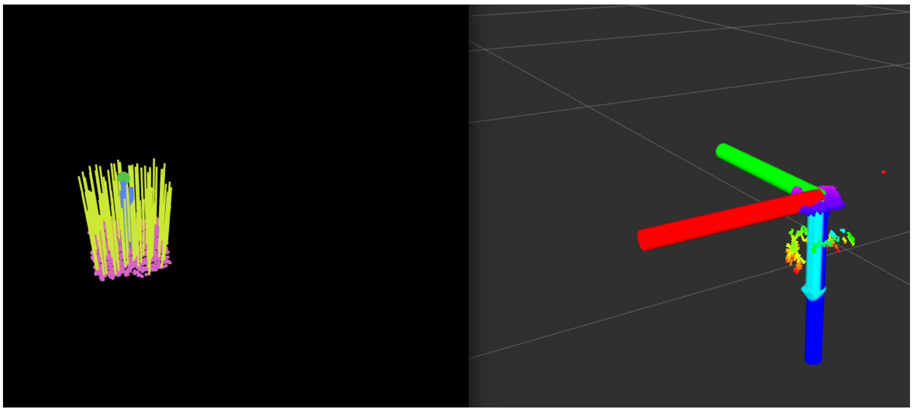


FIGURE 3.9: CNA algorithm visualization

3.2 Hardware

3.2.1 Grippers

The two finger gripper is nothing but two straight plastic fingers that open to a certain low distance. The two-finger gripper is called the Pincher. Figure 3.10 show the suction gripper used during the competition and Table 3.1 defines the dimensions.

TABLE 3.1: Two finger grippper dimensions

Property	Dimension (cm)
Finger length	35
Finger depth	2.5
Finger thickness	1
Max inside opening width	3.5
Min inside opening width	0

Figure 3.10 show the vacuum gripper used during the competition and Table 3.2 defines the dimensions. Vacuum grippers use a suction cup, also known as a sucker, to create a negative fluid pressure of air to adhere to nonporous surfaces, creating a partial vacuum. Suction cups are peripheral traits of some animals such as octopuses and squids and have been reproduced artificially for numerous purposes. This type of grippers will provide good handling if the objects are smooth, flat, and clean. It has only one surface for gripping the objects. Most importantly, it is not best suitable for handling the objects with holes.

3.2.2 Hardware restrictions

Figure 3.11 shows the names with which the joints of the Baxter robot are associated. Each joint has a separate range of motion whether bent or twist. Joints S1, E1, and W1

FIGURE 3.10: (left) Pincher gripper. (right) vacuum gripper.

TABLE 3.2: Vacuum grippper dimensions

Property	Dimension (cm)
Length	30
Inner diameter of hole	2
Outer diameter of cup	3.5

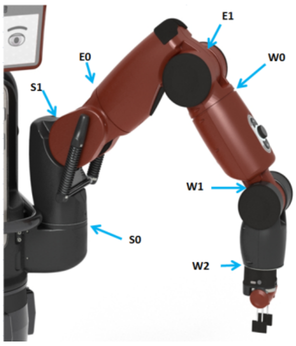


FIGURE 3.11: Baxter joint description

are considered to have bend motion and are restricted by the width of the robot arm. They can move the robot arm larger distances across the workspace but do not have a wide range of twist around its axis. Joints S0, E0, W0, and W2 and considered twist joints and have a much wider range of motion around the axis even though they cannot move an object far along the workspace. When the grippers were mounted onto the Baxter robot such that they are perpendicular with respect to the wrist, the twist of the gripper was moved from joint W2 to joint W1. The restriction of twist means that the angular roll along the z-axis of the gripper is also restricted. Table 3.3 summarizes the angular range in each joint. Note that the angular motion has been reduced by 140.5 degrees. Within those degrees, the robot will not be able to move and the IK will not find a solution. On Figure 3.13 the R_0 represents the zero degrees of roll of the W0 joint, R_i represents the current roll, and finally R_{lim_1} and R_{lim_2} represent the two limits of the W1 motion. Between these two limits the W1 joint will never find an IK solution. Note the difference between the limit differences between Figure 3.13 and Figure 3.12. Although this decision was made in order to increase the number of possible vertical grasps where the z-axis is pointing directly downwards, the sacrifice was the limitation on the gripper.

TABLE 3.3: Joint range comparison

Joint	lim_1	lim_2	Range
W1	120°	270°	210°
W2	175.25°	184.75°	350.5°

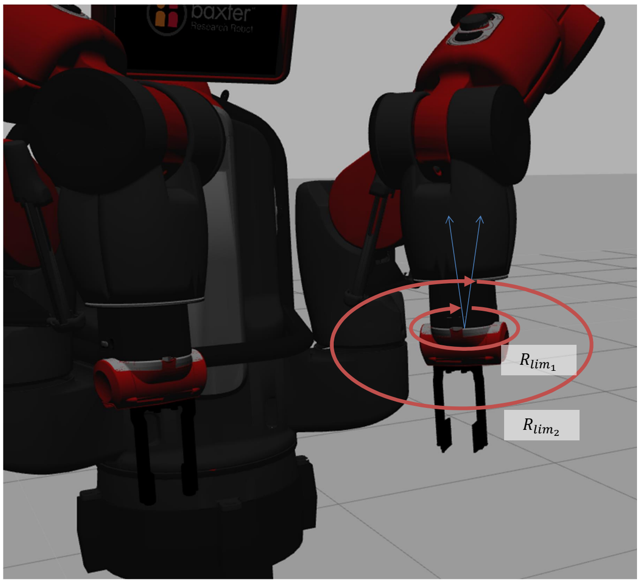


FIGURE 3.12: Baxter W2 limits

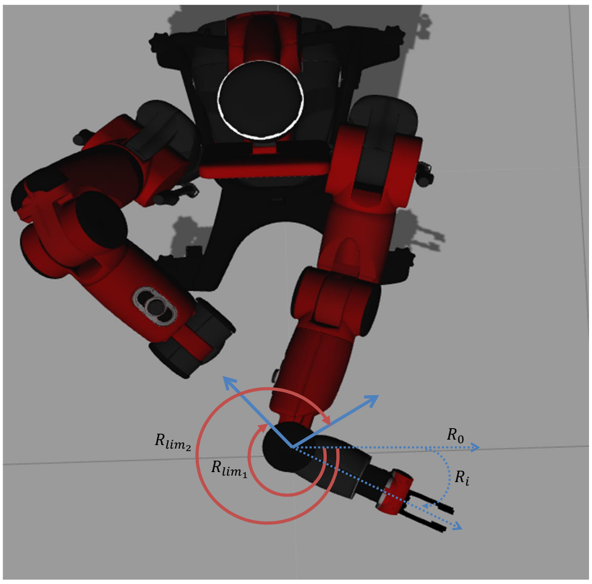


FIGURE 3.13: Baxter W1 limits

The position of the gripper on the robot arm poses two great challenges with regards to grasping:

1. If the orientation of the object to be grasped requires the roll of the arm to be

within the impossible limits of the W1 joint then the object cannot be grasped although the grasp vector can be deemed to be successful.

2. The final position and orientation of the grasp determines greatly the position of the arm and its configuration. If the grasp is feasible but the x-axis of the robot arm points away from the robot then the IK will struggle to find a suitable solution because there could be a collision with the arm. Since IK does not try to find the proper motion to avoid collisions, the grasps must be carefully selected in order to maximize as much as possible a suitable solution. The grasps should try to put the robot arms similar to that when it is int he "tuck-arms" position.

3.2.3 Software compensation

In order to compensate for the limited angle roll of the wrist, the software for the output of the final position and orientation of the robot has been transformed for HAF grasping such that the roll angle around the z-axis places the wrist in a suitable orientation. For this, first, a rolls is extracted from the quaternion then it is modified. The following methodology converts a quaternion, $q = q_w + q_x\mathbf{i} + q_y\mathbf{j} + q_z\mathbf{k}$,into axis-angle representation.

$$(a_z, a_y, a_z) = \frac{q_x, q_y, q_z}{\sqrt{q_x^2 + q_y^2 + q_z^2}} \tag{3.15}$$

$$\theta = a\tan 2(\sqrt{q_x^2 + q_y^2 + q_z^2}, q_w) \tag{3.16}$$

Algorithm 1: Robot wrist roll compensation

Result: Roll is always within workspace of robot

if $|\frac{\pi}{2} - \theta| < \frac{\pi}{9}$ **then**

  $\theta = \theta - \frac{\pi}{2}$

else if $\theta < \frac{\pi}{2}$ **then**

  $\theta = \theta + \frac{\pi}{2}$

else

  $\theta = \theta - \frac{\pi}{2}$

Once the roll of θ is performed using the following algorithm, it's possible to convert it back to quaternion using the R equations specified in Section 3.1.3. This does not guarantee that the IK will be found via the computer, this only guarantees that the position exists for the Baxter robot. A human guided motion to such position will be possible. In future testing, it will be shown whether this holds true within the computational system as well. For AGILE grasping since the options on the object are numerous and usually surround the object itself, the impossible grasps will be filtered out naturally when doing the IK for motion planning.

Chapter 4

Experimentation Procedures & Results

4.1 Experimentation procedures

The following section presents the experimental setup for testing the grasping algorithms as well as the results. During the pick and stow tasks, the grasping algorithms receive a single point cloud of an object to be manipulated. There is no need to test objects in a dense clutter because the input will be the isolated object that has been identified by the vision pipeline in the preceding step of the system architecture. The algorithm for grasping has to be able to receive a point cloud of a single object and propose a suitable grasp vector. During the stow task, the tote contains 20 objects in the mixed jumble. The vision pipeline, in this case, does not guarantee the segmentation of a single object and can include parts of other objects. This noisy data requires the algorithms to be robust. The grasp must also be calculated quickly and successfully which will increase the chances of victory for the team. Knowing which algorithm performs best under which circumstance is key to devising a final structure that can pick up the maximum amount of objects.

4.1.1 Preliminary object to gripper matching

Before applying the testing of algorithms it is important to know which gripper works well with which object. In a perfect scenario, the best grasp algorithm will output a similar result to what a human would choose. Hence, it is important to first do a controlled test with a human guiding the gripper to know that grasping a given object with a given gripper is feasible. If this object shows low success in the future with a certain algorithm then the experimenters will know that the flaw is in the algorithm not in the gripper. The item can be considered to be graspable by the two-finger gripper if at least one of its dimensions can fit within the gripper given the gripper's minimum and maximum opening width.

The procedure for finding out whether the object can be grasped by the vacuum consists of:

1. Turn on the suction pump
2. Place your hand on the wrist of the Baxter robot and approach the item with the vacuum pump
3. Attempt to lift the vacuum gripper along with the item
4. If the item can be lifted along most locations on the flat part of the item then the item is graspable by suction

5. If the object falls down then the object cannot be grasped by the suction gripper

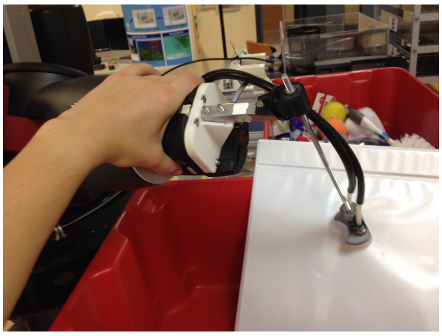


FIGURE 4.1: Preliminary classification demonstration

The procedure for finding out if the object can be grasped by the Baxter gripper consists of:

1. Open the fingers of the Baxter gripper using a ROS command
2. Place your hand on the wrist of the Baxter robot and approach the item with the fingers of the gripper attempting to envelop the object as best as possible. This is a gross assumption of the perfection of the algorithm but it is only to prove that the dimensions of the object are feasible given the dimensions of the gripper.
3. Close the fingers of the Baxter gripper using a ROS command
4. Attempt to lift the two-finger gripper along with the item
5. If the item can be lifted along most locations on the flat part of the item then the item is graspable by the gripper
6. If the object falls down then the object cannot be grasped by the two-finger gripper

The matching procedure allows the task planning team to know which arm to use for object manipulation. If the object can be lifted using vacuum pressure on most sides of the object then the right arm will be used with suction. If the object is best grasped with the gripper then left arm will be used to manipulate the object.

Table 4.1 shows the results of the above specified procedures with official names of the items.

4.1.2 Preliminary implementation of algorithms

For object manipulation the algorithm options are either AGILE, HAF, or CNA. The AGILE and HAF grasping are used with the same gripper and produce similar results. In order to understand the potential of the grasping algorithms, it is first important to implement them and obtain results to see whether the algorithm is capable in dealing with the point cloud presented by the vision pipeline. The goal of this testing is to see how the algorithm behaves and fits once it is integrated into the whole system. The work for this requires understanding the topics, messages, parameters, and the inputs and outputs. Then rewriting all of the code into a library format as required by the system and in this way, the integration work package needs only to call a single function within a service call in order to launch the whole algorithm.

TABLE 4.1: ARC'17 items and classification

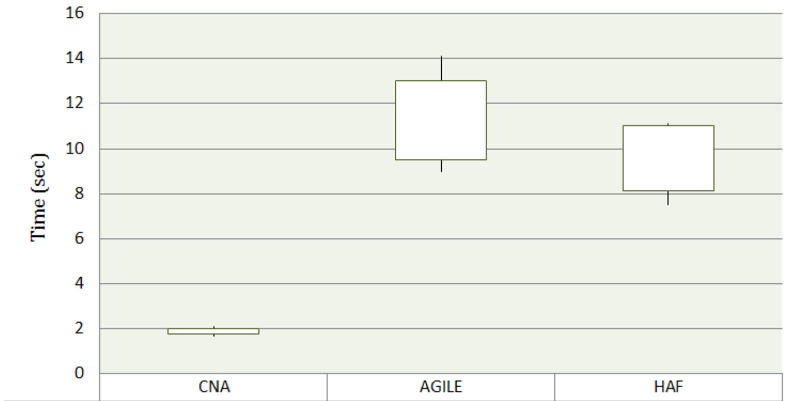
#	Name	Official Name	Pincher	Vacuum
1	Avery 1" Binder - White	avery_binder		x
2	Bag of Balloons	balloons	x	
3	Buns Bees Baby Wipes	burts_bees_baby_wipes		x
4	Clorox Toilet Brush	toilet_brush	x	
5	Colgate Toothbrushes	colgate_toothbrush_4pk		x
6	Crayola Crayons	crayons		x
7	Dr. Teal's Epsom Salts	epsom_salts		
8	DVD Robots	robots_dvd		x
9	Elmers Glue Sticks	glue_sticks		x
10	Expo Eraser	expo_eraser	x	x
11	Fiskar Scissors	fiskars_scissors		
12	Green Composition Book	composition_book		x
13	Hanes White Socks	hanes_socks		
14	Irish Spring	irish_spring_soap		x
15	Johnson & Johnson Paper Tape	band_aid_tape		
16	Kleenex Cool Touch Tissues	tissue_box		x
17	Knit Gloves Black	black_fashion_gloves	x	
18	Laugh Out Loud Jokes For Kids	laugh_out_loud_jokes		x
19	Mesh Pencil Cup	mesh_cup		
20	Mini Marbles Clear Lustre	marbles	x	
21	Neoprene Weight - Pink	hand_weight	x	
22	Plastic Wine Glasses	plastic_wine_glass		x
23	Poland Springs Water Bottle	poland_spring_water		
24	Reynolds Pie Pans	pie_plates		x
25	Reynolds Wrap 85 Sq. Ft.	reynolds_wrap		x
26	Robots Everywhere	robots_everywhere		x
27	Scotch Cloth Duct Tape	duct_tape	x	x
28	Scotch Sponges	scotch_sponges		x
29	Speed Stick 2 Pack	speed_stick		x
30	Spiral Index Cards	hinged_ruled_index_cards		x
31	Sterilite Ice Cube Tray	ice_cube_tray		
32	Table Cover (In Bag)	table_cloth	x	
33	Target Brand Measuring Spoons	measuring_spoons	x	
34	The Bathery Delicate Bath Sponge	bath_sponge	x	
35	Ticonderoga Pencils	ticonderoga_pencils		x
36	TomCat Mousetraps	mouse_traps		x
37	White Face Cloth	white_facecloth	x	
38	Wilson Tennis Balls	tennis_ball_container		x
39	Windex Spray Bottle	windex		x
40	Flashlights	flashlight		x
			11/40	25/40

It is important to see the output of the final obtained approach vector as well as the time taken to perform the calculation of the given vector or vectors. By seeing how each algorithm behaves, it is possible to not only classify the objects as was described in the previous section, but also add on as to which algorithm to use for which object. Using the Open Multi-Processing library (OMP) the time before and after each the function call has been calculated using the simple commands *start = omp_get_wtime()* and *end = omp_get_wtime()*. Then compared the differences between the algorithms. The calls were recorded 10 times for 10 different objects. Below, on Table 4.2 are the ,main function calls that have been timed for each algorithm: Figure

TABLE 4.2: Main function calls

Algorithm	Function call
AGILE	*getAGILEgrasps()*
HAF	*getHAFgrasps()*
CMA	*getGraspsWRot()*

4.2 shows the distribution of time for each algorithm after integrating it into the system, given that the input point cloud presented to each algorithm is approximately the same. With this information further steps have been taken to simplify the grasping optimization. Figure 4.3 shows the qualitative analysis of robustness.



	CNA	AGILE	HAF
1st quartile	1.75	9.5	8.11
Max	2.1	14.1	11.1
Min	1.65	9	7.5
3rd quartile	1.99	13	11

FIGURE 4.2: Time each algorithm takes to calculate a set of grasps

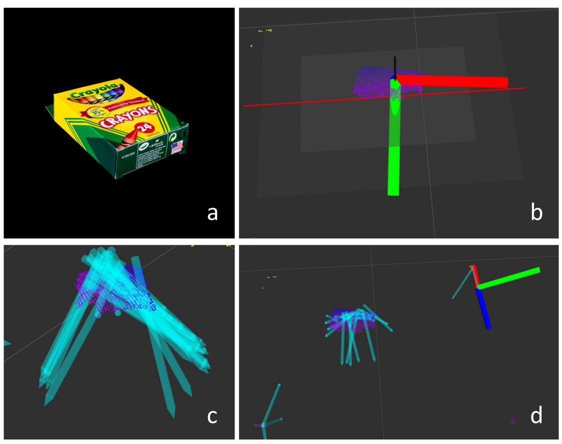


FIGURE 4.3: Robustness comparison of AGILE and HAF

4.1.3 Testing objects in isolation

Testing objects in isolation is relevant for the ARC'17 challenge since the point cloud given from the vision pipeline is a point cloud of a single object. The point cloud will be fed to the algorithms with the main function calls seen on Table 4.2. The set of tunable parameters and code modifications in each algorithm can be read inside the *libraries/grasp* folder of the *RobinLab-ARC2017 SourceForge* repository.[3] The evaluation is done based on 10 grasping attempts, each of which will be scored on a binary 0/1 system to mark whether the object has been grasped and lifted or not. The setting of grasping is exactly as would be during the competition and in the warehouse. If the point cloud or hardware is not perfect then the algorithm has to be able to deal with this. The scenario is not fully controlled but rather as real life as possible The successful grasps will be added and the $\frac{success}{num.attempt} \times 100\%$ will be calculated. For demonstration videos, please visit the references website. [2] The experiment ran according to the following rules:

1. All objects, originating from the ARC'17 competition, can be grasped by the gripper in their respective category. The data for this is provided by Table 4.1.
2. The point cloud is as provided by the vision pipeline. No changes to make the object clearer than what vision sees it. The testing is done with full integration of the whole system architecture.
3. The environment is exactly as the system would predict to be in a real-world setting. A failure of gripper arm orientation or positioning is considered a failure for grasping.

Tables 4.3, 4.4 and 4.5 show the testing results as a measure of percentile of grasping success. Figures 4.6, 4.7, 4.8 shows the reasons behind failure, to give an idea as to what are the scenarios where the algorithms are weak. Table 4.6 shows which object have been blacklisted from the grasping scenario. The reasons are mostly due to hardware restrictions for example lack of actuator power or vacuum pressure. Tables 4.4 and 4.5 show the number of items included in each strategy and the success per strategy. The success rate can be analyzed in terms of the success of the algorithm as well as the percent chance of being able to pick up the object if it is assigned for the pick and stow tasks.

4.1.4 Object to algorithm classification

Next, given a set of repeated testing and analysis of success, a Table 4.7 has been created to show which algorithm will be used for which object. This table maximizes the success of the object picking. Number 1 stands for the fact that the first try will be attempted with the algorithm in that column, and number 2 stands for the second try and this algorithm has the second priority. This has been done because an object like the duct tape can be picked up with different grippers depending on its orientation. Unfortunately, the object orientation is not included in the scheme of the algorithms and hence the side is unknown. The solution to this is to blindly try picking it up more than once.

TABLE 4.3: Objects in Isolation: CNA with Vacuum

Object	% Success
1. colgate_toothbrush_4pk	60
2. composition_book	100
3. crayons	100
4. expo_eraser	100
5. glue_sticks	70
6. hinged_ruled_index_cards	100
7. irish_spring_soap	100
8. laugh_out_loud_jokes	100
9. reynolds_wrap	100
10. robots_dvd	100
11. robots_everywhere	100
12. scotch_sponges	100
13. speed_stick	100
14. tennis_ball_container	100
15. ticonderoga_pencils	70
16. tissue_box	100
17. flashlight	100
18. windex	100
19. mouse_traps	90
20. ice_cube_tray	100
21. burts_bees_baby_wipes	100
22. pie_plates	50
23. plastic_wine_glass	70
24. avery_binder	100
Average Success	92.08

TABLE 4.4: Objects in Isolation: CNA with Pincher

Object	% Success
1. balloons	100
2. black_fashion_gloves	90
3. table_cloth	70
4. white_facecloth	100
5. marbles	10
6. bath_sponge	80
Average Success	75.00

TABLE 4.5: Objects in Isolation: HAF with Pincher

Object	% Success
1. hand_weight	10
2. measuring_spoons	10
3. toilet_brush	30
4. duct_tape	50
Average Success	25.00

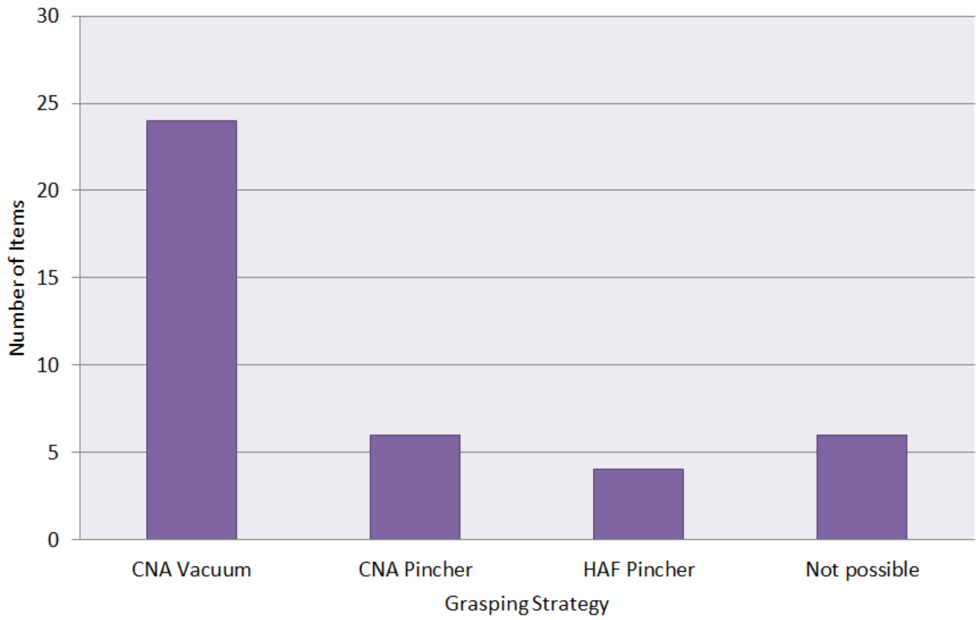


FIGURE 4.4: Number of items per grasping strategy

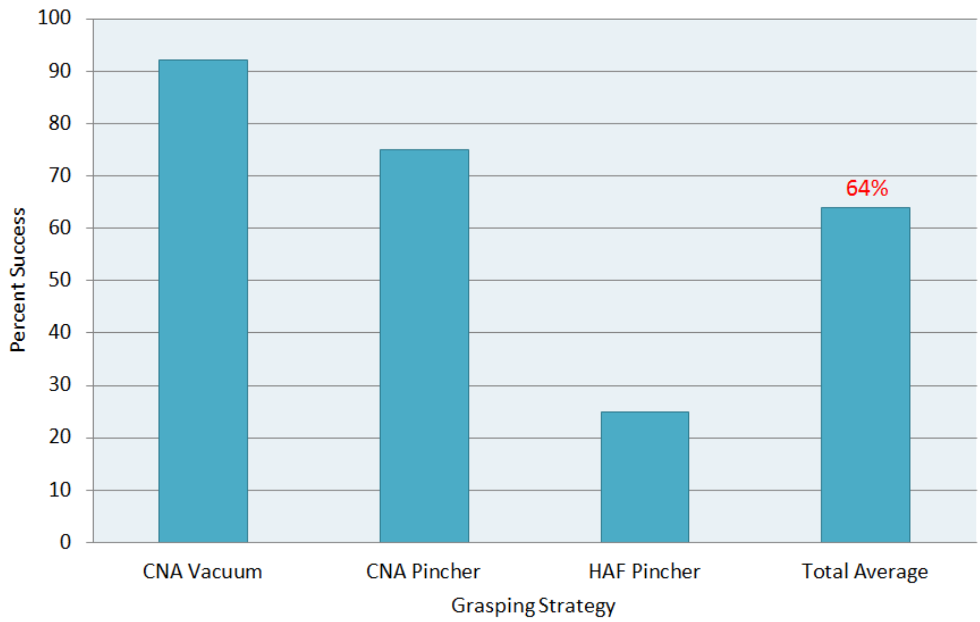


FIGURE 4.5: Percent success per grasping strategy

TABLE 4.6: Objects with no grasping scenario

Object	Reason
1. band_aid_tape	Unavailable item
2. epsom_salts	Too heavy
3. mesh_cup	Too large for Pincher, unfeasible texture for vacuum
4. poland_spring_water	Too large for Pincher, unfeasible texture for vacuum
5. fiskars_scissors	Small and flat, too small for vacuum
6. hanes_socks	Too heavy

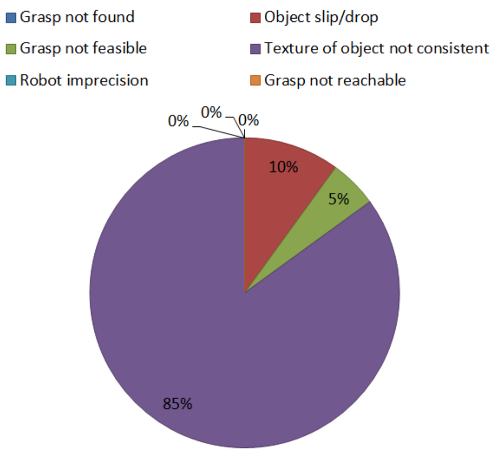


FIGURE 4.6: Reasons for failure of CNA with vacuum

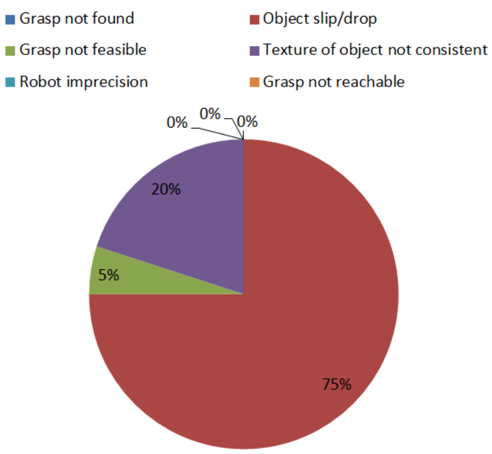


FIGURE 4.7: Reasons for failure of CNA with Pincher

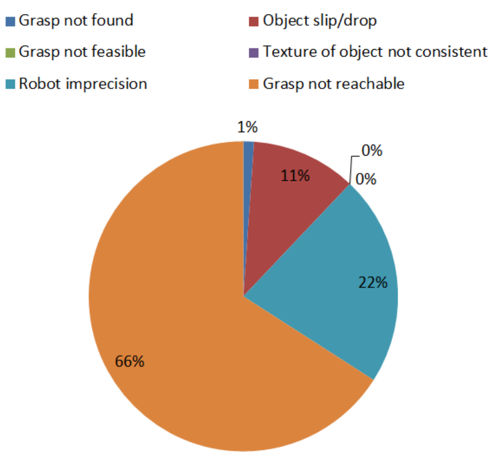


FIGURE 4.8: Reasons for failure of HAF with Pincher

FIGURE 4.9: CNA with vacuum successful grasp of the reynold's wrap, tennis ball container and irish spring soap

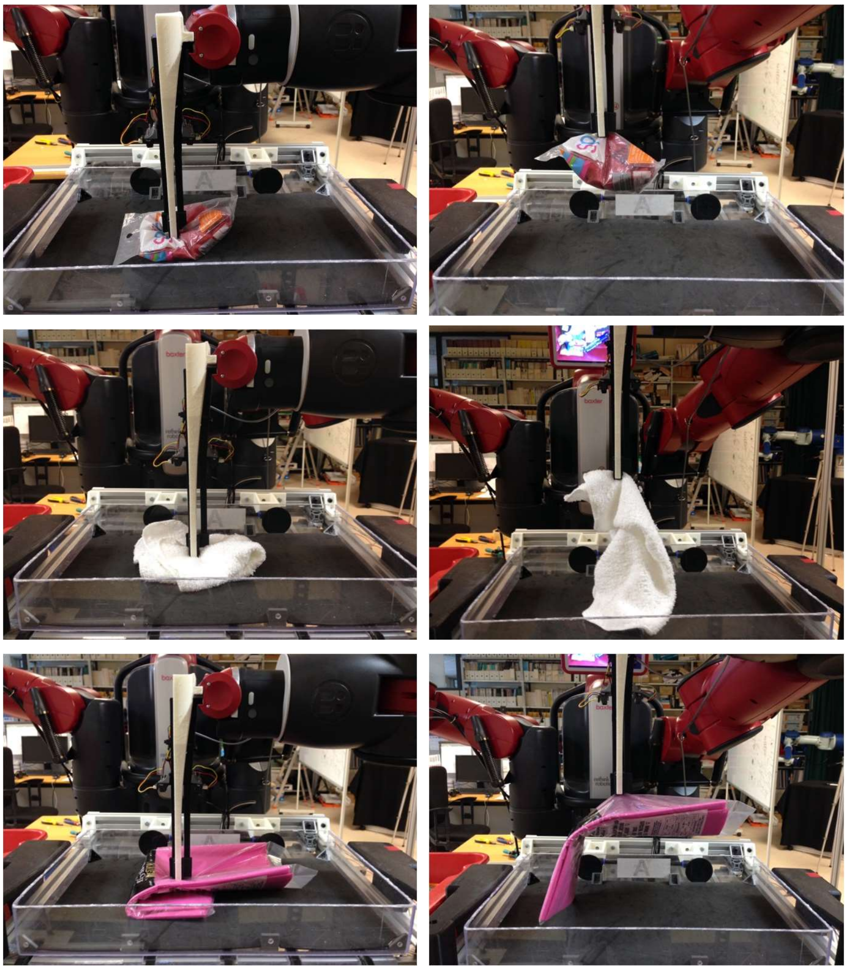


FIGURE 4.10: CNA with Pincher successful grasp of balloons, face cloth, and table cloth

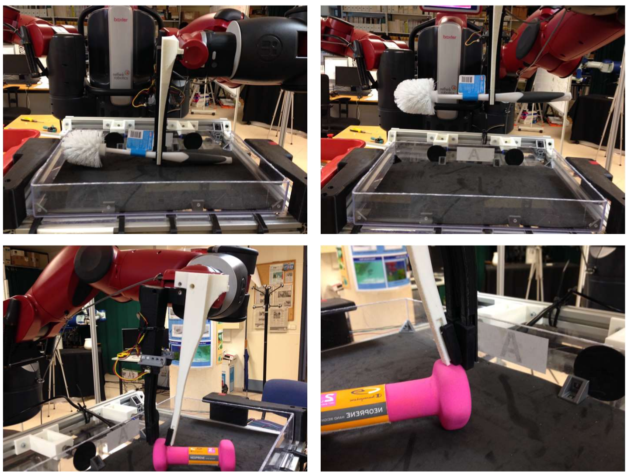


FIGURE 4.11: HAF with Pincher successful grasp of the toilet brush and false positive of heights with hand weight

TABLE 4.7: Optimization table for ARC'17 items

#	Official Name	CNA with vacuum	CNA with Pincher	HAF with Pincher
1	avery_binder	1		
2	balloons		1	
3	burts_bees_baby_wipes	1		
4	toilet_brush		2	1
5	colgate_toothbrush_4pk	1		
6	crayons	1		
7	epsom_salts			
8	robots_dvd	1		
9	glue_sticks	1		
10	expo_eraser	1		
11	fiskars_scissors	1		
12	composition_book	1		
13	hanes_socks			
14	irish_spring_soap	1		
15	band_aid_tape			
16	tissue_box	1		
17	black_fashion_gloves		1	
18	laugh_out_loud_jokes	1		
19	mesh_cup			
20	marbles	2	1	
21	hand_weight		2	1
22	plastic_wine_glass	1		
23	poland_spring_water	1		
24	pie_plates	1		2
25	reynolds_wrap	1		
26	robots_everywhere	1		
27	duct_tape	2		1
28	scotch_sponges	1		
29	speed_stick	1		
30	hinged_ruled_index_cards	1		
31	ice_cube_tray	1		
32	table_cloth		1	
33	measuring_spoons	2		1
34	bath_sponge		1	
35	ticonderoga_pencils	1		
36	mouse_traps	1		
37	white_facecloth		1	
38	tennis_ball_container	1		
39	windex	1		
40	flashlight	1		
		29/40	8/40	5/40

Chapter 5

Discussions

The following section explains the tabular and graphical results presented in Chapter named Experimental Procedures & Results.

5.1 Elimianting AGILE

On Figure 4.2 it can be seen that the AGILE grasping algorithm takes up to 14 seconds to calculate, HAF takes up to 11 while Suction Normals takes up to 2 seconds. This is a huge drawback to computing height and geometrical features since the time-per-object increases greatly. This time increase is not desired during the competition since the pick and the stow tasks have a time limit. In order to minimize time and thus maximize the points, it has been decided to limit the number of objects that will be used with the HAF or AGILE grasping algorithms. If an object has an option of being picked with CNA then this algorithm has the priority.

After a few preliminary testings with time, it has been determined that having both AGILE and HAF grasping to be applied to the same gripper increases complexity and adds a redundancy. These algorithms both should be able to accomplish the same task and it is unnecessary to keep both of them in the system. From the first step, it is known that AGILE takes more time. The second important step is determining the robustness and success of the algorithms. With the point cloud of the crayons box,4.3a, HAF grasping computed a downward facing grasp vector as seen on Figure 4.3b. The grasp is unique with limited variations of twist around any other axis but it is stable and the result is repeatable. By choosing the vertical approach rather than the one at an angle, the robot is less likely to collide with the bin and with the tote. HAF grasping does not present the final list with many grasp possibilities but the one that is presented is the highest scored grasp that has been selected by the algorithm. In this case, the grasps with low scores will not appear in the output and hence will not be executed. This saves time for not executing grasps that are likely to fail, shows robustness since the grasps with low scores often result in a polluted point cloud and are eliminated, and shows safety since the arm is unlikely to do do a large angle tilt away from the axis as this is controlled.

AGILE grasping, as can be seen on Figure 4.3c, presents the result with many possibilities. The possibilities are often at an angle away from the vertical z-axis since the search looks for handle-like part of the object. Unfortunately, the gripper often cannot pick up an object from the sideways position since there are no feasible non-friction fingers to be able to do so. AGILE grasping also takes into considerations the many parts of the point cloud that are not relevant, as can be seen on Figure 4.3d. Although initially, the sorting algorithm will take the best grasps into account, the scenario can be such that all of the good grasps are impossible and the robot will resort to the bad ones in the list. These grasps, although unlikely, still exist and can potentially cost the team time and points if executed and can cause a collision and if executed improperly because the Baxter arm orientation behaves unpredictably. Hence it was concluded for further testing to eliminate the AGILE grasping from the system architecture as the lack of robustness reduces success and increases the possibility of whole system failure.

5.2 Maximizing use of CNA

From Table 4.1 it can be seen that 25 out of 40 items can be lifted by the vacuum gripper. In this case, we already know that if they have easy symmetry and a large part to be used with suction, then maximizing the use of CNA algorithm and hence reducing time will be an easy task. As can be seen, there was a maximization of the use of the CNA algorithm, first of all, because it is successful, second of all, because it is fast, and third of all because it is not restricted by the workspace of the robot arm.

After an extensive amount of testing, the final optimized table has been created in order to maximize success for grasping with various algorithms and two grippers. Table 4.7 shows which objects will be grasped with what algorithm and with what grippers. Table 4.6 lists the objects for which the grasping does not work even in a perfect scenario and hence will be excluded from further analysis of success. Note that the CNA approach has been used with the Pincher gripper as well. The CNA algorithm works because it takes the center of the object given the point cloud and exploits consistent texture quality of objects like the gloves. These deformable objects can be grasped in the same way that a book is grasped. There is no difference in computation and the hardware of the Pincher adapts well to the object. Note that some objects can be picked up with both grippers depending on which way the orientation is or which part of the object is visible. In this case, the robot can be programmed to try twice with different algorithms. With this exploitation, it can be seen on Figure 4.4 that over 75% of the objects will be picked up using the CNA algorithm, whether with vacuum or with Pincher.

5.3 Failure analysis

Table 4.3 shows that not only is CNA with vacuum is most used but it is also the most reliable approach. So if it is deemed that a vacuum is most useful for the certain object then there is over 90% chance that is will be successfully picked up. The set of pie charts gives important information as to what happened during testing with the unsuccessful grasps. The pie charts document the reasons for failure. The most notable reasons, relevant to grasping, that were seen during testing are listed below.

- Grasp not found: The algorithm does not output a grasp based on the input point cloud.
- Grasp not feasible: The grasp can be executed but is not successful because it does not allow the robot to lift the object.
- Robot imprecision: The grasp is feasible, and can be executed however the robot misses the location of grasp due to calibration error or actuation error.
- Object slip/drop: Object is grasped but then slips between the fingers or escapes from the vacuum pressure.

- Texture of object not consistent: The gasp from the point cloud looks potentially successful however a label or a bump in the object gets in the way of lifting the object.

- Grasp not reachable: The restrictions to robot's wrist motion deem the grasp not executable because the robot cannot reach the potential grasp as it is outside of the workspace or leads to a collision with itself.

In the case of CNA with vacuum, the main reason, as seen on Figure 4.6 is the inconsistency of object texture. For example, the pie plates item has the back side that is perfectly flat and has 100% chance of being picked up however testing was done on all sides and the other side is curved and air pressure slips between the holes. On the other side, the pie plates item has been picked up 0/5 times hence the total success can only be 50%. Figure 4.7 shows the reason behind the failure of CNA with Pincher. The biggest reason is that the object slips. The material in an object like the table cloth is very thin hence drops unexpectedly. Another major reason is that the texture is inconsistent. An object like the black fashion gloves has a label and if one of the fingers touches this label then it glides across the label without grabbing any of the material. Without the grasp of the material, there can be no grasp of the object. Note that in the above case the CNA algorithm is successful despite the hardware restrictions. This is due to the preassigned angular rolls around the z-axis of the robot wrist. This technique in CNA allows for the configuration to always be comfortable for Baxter and the IK to always find a solution. Hence the strength of this algorithm is its independence of the orientation of the wrist. Figure 4.8 shows the reasons for the failure of the HAF algorithm with Pincher. This combination shows a low success rate of 25%. This reason behind the low success rate can be attributed to the hardware restriction specified in the Methodology chapter. A grasp can always be calculated, even if it is not the best, but it is rarely executed. As can be seen on Figure 5.1 the fully vertical grasp which would allow the robot the most strength and collision avoidance, is not feasible in the further half of the bin. Even this configuration would not be allowed as this forces the arm to go into a singular configuration. Furthermore, certain orientations restrict the motion, on the leftmost image of Figure 5.2 the hand weight has an orientation that is not reachable from the comfortable position of Baxter. The corresponding right image shows the opposite arm motion with an 180-degree roll around the z-axis. Although this is technically reachable by Baxter, the motion of the arm first moves the shoulder and hence forces the robot into collision. This motion is then not executed either.

FIGURE 5.1: The distance to the bin results in many unreachable configurations

Another reason, especially relevant for the hand weight item, is robot imprecision. The Pincher opening allows about a millimeter of clearance with the center

FIGURE 5.2: (left) Maximum angle restriction. (right) Constitutes a collision due to shoulder motion.

of the hand weight and the robot has 1cm accuracy. Hence robot simply misses the correct grasp location.

Note that some of the failure reasons mentioned above do not deal directly with grasping. Some deal with the problem that the robot may have while attempting to execute the grasp. The grasp itself may look good in a simulation, but may fail in real life. That's why the conditions for testing have been as realistic as possible. It is very important to evaluate a grasp in this perspective because grasping criteria often only give the success with regards to the object not to the scene. Here, the whole scene is provided because it is believed that this grasping aids in robot autonomy and a given grasp needs to allow the robot to be autonomous. If the robot autonomy fails due to grasping then the grasp fails as well.

Despite the setback described above, the total success rate as shown on Figure 4.5 shows a complete 64% success of object grasping.

Chapter 6

Conclusions

The following section summarizes the achievements of the presented work and investigates possibilities for improvements.

6.1 Summary

The UJI RobinLab team took on the ARC'17 in order to automate the warehouse environment. One of the necessary accomplishments for the robot is to be able to grasp an object, which requires a software algorithm. The thesis proposes a solution for the object grasping and using two algorithms that take a point cloud as an input outputs a position and orientation of the gripper given the hardware specifications and restrictions. Two algorithms for the two finger gripper have been compared and the most suitable one has been selected for further testing due to the analysis of time, robustness, and success. The chosen algorithm uses height features of point cloud in order to find a final grasping position and orientation. A vacuum gripper approach has been created based on the normals estimated based on the surfaces of the point cloud of an object. This approach has been used for objects that are generally symmetric and have a surface that can be extracted from the point cloud in order to be used for picking them up using vacuum pressure. This algorithm proved to be fast and hence has also been implemented with the gripper on deformable objects such as a towel for which the orientation of the gripper's x-axis does not play a role. Finally, an optimized table has been devised to maximize the manipulation of an object by customizing the algorithms to each specific object.

6.2 Future Work

The limitations of the work posed by the limitation of the gripper led to insufficient analysis on the differences between AGILE and HAF grasping. A gripper with at least 15cm of opening width would be a hardware component that would allow testing based on the success of lift rather than qualitative analysis in a simulation. With this gripper it would be useful to implement and test not only AGILE, HAF, but also other grasping algorithms for the purpose of testing current applications. Another testing to be done is object picking from a dense clutter. The input would be a large cluttered scene, and the output would be a list of grasp vectors. The purpose would be to have the robot take out objects from a bin one by one without having to rely on vision to segment the point cloud. This would allow for a strong quantitative comparison of the algorithms such as AGILE and HAF since they were created for the purposes of a dense clutter. More testing would be done with a different robot as well, one that has the ability to perform any kind of rotation of the wrist. This would allow variety of approaches to be tested.

Another limitation lies in autonomy. Industries aim to create robots that act without supervision in home and work environments. In this case, the success of an algorithm lies in the recognition of its use and then successful results in each demonstration. Once a robot approaches a scene with multiple objects, the scene itself should give certain features with which a robot understands what algorithm would be most useful. For the competition, the results of this top-level analysis have been simulated with a simple excel sheet. Each object has its personal attributes, like a passport. It gives details of the object shape, the algorithm with which the object should be picked, and the parameters associated with the algorithm. To make such an excel sheet with all the objects in the totality of the Amazon task would prove to be an arduous task. Making the excel sheet an autonomous part would be more a logical next step. The proposed work includes the design of an algorithm that takes in already available image data used for vision processing and Point Cloud data in order to define which of the three algorithms, mainly HAF, AGILE, or CNA, would be used for a particular object either in the cluttered or single object scene. In the first stages of the creation of the algorithm, the output would create a dataset similar to the excel sheet with a success or failure associated with it. The second step of the creation of the algorithm would be to have the robot autonomously learn to improve its choices in the future. After a set of trials, the success rate would be similar to the one demonstrated without human analysis. Given the readily available machine learning classification methods, such an output would be implemented through a set of supervised learning. The algorithm would lead to more robot autonomy in terms of grasping and manipulation.

6.3 Conclusion

An important lesson from the given report is that given a problem, one can put together a solution given current advances in technology. Often research focuses on select few specifications but the real world scenario is full of unexpected encounters. There is often no guarantee as to how the environment will look in front of a robot and most likely that it will not resemble a perfect laboratory scenario. With this is mind, it has been quite important to learn what solutions are available and how can they complement each other given a scenario such as the ARC'17. In this case the solution consisted of applications of software to different hardware and the reuse of innovative applications of code. Perhaps the robotics community cannot yet replicate the intricacy of the human brain however through various modes of application the solution can be brought to the best possible outcome. It is also important to understand the advantages and disadvantages of previous work and carry on the work for the advancement of the knowledge of robotics and science as a whole.

Bibliography

[1] https://www.amazonrobotics.com/roboticschallenge. [Online]. 2017.

[2] https://vimeo.com/grasps. [Online]. 2017.

[3] https://sourceforge.net/p/robinlab-arc2017/robinlab-arc2017/ci/develop/tree/. [Online]. 2017.

[4] Jeannette Bohg et al. "Data-Driven Grasp Synthesis—A Survey". In: *IEEE Transactions on Robotics* 30.2 (2013), pp. 289–309. ISSN: 15523098.

[5] Sunglok Choi and Wonpil Yu. "Performance Evaluation of RANSAC Family". In: (2009), pp. 1–12.

[6] Ondrej Chum and Jiri Matas. "Matching with PROSAC " Progressive Sample Consensus". In: *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 220–226. ISBN: 0-7695-2372-2.

[7] Nikolaus Correll et al. "Lessons from the Amazon Picking Challenge". In: 6.1 (2007), pp. 1–14.

[8] Hao Dang and Peter K. Allen. "Semantic grasping: Planning task-specific stable robotic grasps". In: *Autonomous Robots* 37.3 (2014), pp. 301–316. ISSN: 09295593.

[9] O. Duchenne, A. Joulin, and J. Ponce. "A graph-matching kernel for object categorization". In: *2011 International Conference on Computer Vision*. 2011, pp. 1792–1799.

[10] Clemens Eppner and H Sebastian. "Lessons from the Amazon Picking Challenge : Four Aspects of Building Robotic Systems". In: ().

[11] David Fischinger and Markus Vincze. "Empty the Basket - A Shape Based Learning Approach for Grasping Piles of Unknown Objects". In: ().

[12] David Fischinger and Markus Vincze. "Learning Grasps for Unknown Objects in Cluttered Scenes". In: (2013), pp. 609–616.

[13] Marcus Gualtieri et al. "High precision grasp pose detection in dense clutter *". In: (2016).

[14] K. Huebner, S. Ruthotto, and D. Kragic. "Minimum volume bounding box decomposition for shape approximation in robot grasping". In: *2008 IEEE International Conference on Robotics and Automation*. 2008, pp. 1628–1633.

[15] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. "Efficient Grasping from RGBD Images : Learning using a new Rectangle Representation". In: ().

[16] Ellen Klingbeil et al. "Grasping with Application to an Autonomous Checkout Robot". In: (2011), pp. 2837–2844.

[17] B Le, Cognitive Systems Monographs, and Springer International. "Robot Grasping Foundations 2.1". In: (), pp. 15–32.

[18] M. Madry, D. Song, and D. Kragic. “From object categories to grasp transfer using probabilistic reasoning”. In: *2012 IEEE International Conference on Robotics and Automation*. 2012, pp. 1716–1723.

[19] Satya Mallick. *Histogram of Oriented Gradients*. [Online]. 2016.

[20] Andrew T Miller et al. “Automatic Grasp Planning Using Shape Primitives”. In: (), pp. 1–7.

[21] Van-duc Nguyen. “Constructing Force-Closure”. In: (1986), pp. 1368–1373.

[22] Ekaterina Nikandrova and Ville Kyrki. “Category-based task specific grasping”. In: *Robotics and Autonomous Systems* 70 (2015), pp. 25–35. ISSN: 09218890.

[23] Robert Platt. “Using geometry to detect grasps in 3d point cloud”. In: (2015).

[24] Deepak Rao et al. “Grasping novel objects with depth segmentation”. In: *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings* (2010), pp. 2578–2585. ISSN: 2153-0858.

[25] Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. “Robotic Grasping of Novel Objects using Vision”. In: (2000).

[26] Ashutosh Saxena, Lawson L S Wong, and Andrew Y Ng. “Learning Grasp Strategies with Partial Shape Information”. In: (2008).

[27] K. B. Shimoga. “Robot Grasp Synthesis Algorithms: A Survey”. In: *Int. J. Rob. Res.* 15.3 (June 1996), pp. 230–266. ISSN: 0278-3649.

[28] Dan Song et al. “Task-Based Robot Grasp Planning Using Probabilistic Inference”. In: *IEEE Transactions on Robotics* 31.3 (2015), pp. 546–561. ISSN: 15523098.

[29] *Teaching Robots How To Manipulate Objects By Having Them Watch YouTube Videos*. [Online]. 2015.

[30] Astrid Weiss et al. “Learning grasps with topographic features”. In: May 2015 (2016).

[31] Kuan-ting Yu et al. “A Summary of Team MIT ′ s Approach to the Amazon Picking Challenge 2015”. In: (2015).