HOCHSCHULE
RAVENSBURG-WEINGARTEN
UNIVERSITY
OF APPLIED SCIENCES

Master Mechatronics
Faculty for Electrical Engineering and Computer Science

Master-Thesis

Grasping Unknown Objects using Convolutional Neural Networks

for obtaining the academic degree
Master of Science Mechatronics

submitted by:

Pranav Krishna Prasad

February 13, 2020

1. Supervisor: Prof. Dr. rer. nat. Wolfgang Ertel
2. Supervisor: Prof. Dr. rer. nat. Stefan Elser

Abstract

Robotic grasping has been a prevailing problem ever since humans began creating robots to execute human-like tasks. The problems are usually due to the involvement of moving parts and sensors. Inaccuracy in sensor data usually leads to unexpected results. Researchers have used a variety of sensors for improving manipulation tasks in robots.
This thesis focuses specifically on grasping unknown objects using mobile service robots. An approach using convolutional neural networks to generate grasp points in a scene using RGBD sensor data is proposed. Two neural networks that perform grasp detection in a top down scenario are evaluated, enhanced and compared in a more general scenario. Experiments are performed in a simulated environment as well as the real world. The results are used to understand how difference in sensor data can affect grasping and enhancements are made to overcome these effects and to optimize the solution.
This thesis is an improvement on the works of Douglas Morrison, Peter Corke and Jürgen Leitner in their work Closing the Loop for Robotic Grasping: A Real-time, Generative Grasp Synthesis Approach [DPJ18] and Fu-Jen Chu, Ruinian Xu and Patricio A. Vela in their work Real-world Multi-object, Multi-grasp Detection [FJRP18].

Declaration of Originality

Hereby I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and all used resources are indicated in the list of references.

Signature

Place, Date

Contents

1 Introduction

1.1 Problem Statement

Service robots are developed and used in various indoor environments such as at home, in hospitals, offices and so on. In all these environments, grasping various objects is an extremely important task. While grasping objects is an easy and routine task for human beings, it is a challenging task for robots. Many researchers have used various techniques and on-board sensors to improve manipulation tasks in service robots and humanoid robots. The greatest problem with grasping and other manipulation tasks is that there are moving parts involved and these tasks greatly depend on sensor data. Sensor data usually contains noise and inaccuracies. To overcome this problem to the maximum extent possible, researchers turned to machine learning techniques which can detect patterns in sensor data to a great extent even with the noise. In this thesis, the Orbbec Astra 3D sensor is used and two convolutional neural networks are used to detect grasps and a few enhancements are made in order to optimize the result. These neural networks are evaluated and compared in a simulated environment as well as the real world.

1.2 Objective

The basic objective of this thesis is to use convolutional neural networks to detect and perform grasping using a service robot. Two neural networks originally created to detect grasps in top down scenarios are used and the performance and quality of both these networks are evaluated and compared in a more generic environment. Also as an enhancement, the surface normals of the grasp points detected by both the networks are extracted and used to create generic grasp orientations to improve the quality of the grasps. The performance is compared again after the enhancements are implemented. The evaluation and comparison is done on the robot's simulation in a table top scenario using various objects. Since the goal of the thesis is to grasp unknown objects, the objects for experimentation are selected at random. Finally both the networks are implemented on the real robot and the performance is compared. Since manipulation is a tedious task, fewer objects are used for experimentation on real robot.
To perform grasping using a robot the following important factors are necessary:

- Detection of ideal grasping poses (position and orientation) on the target object in a scene.
- Detection of other objects in a scene.
- The arm trajectory planning to reach the valid grasp position without collision.

1.3 Definitions

Robot State The defined configuration of all robot joints at a particular given time.

End Effector The free end of a kinematic chain. In this case the robot gripper.

DOF The degrees of freedom of a kinematic chain. In this case the degrees of freedom of the robot arm.

Frame A defined co-ordinate system.

Arm Pose The desired position and orientation of the robot end effector relative to a defined co-ordinate system. In this case the position of the end effector relative to the robot's base frame.

Inverse Kinematics The method used to calculate the configuration of all arm joints for a given end effector pose.

Grasp A set of contacts on the surface of the object, the purpose of which is to constrain the potential movements of the object in the event of external disturbances.

Transformations Mathematical calculations used to transform a vector or pose from one frame to another.

Quaternion is the quotient of two directed lines in a three-dimensional space or equivalently as the quotient of two vectors. It is used in robotics to define end-effector orientation in 3D space.

Depth Image A depth image is a representation of an image from a 3D sensor that only consists of depth data.

Point Cloud It is a set of 3D data points in space. It consists of one point for every pixel of the depth image.

RGB Image A RGB image is simply a regular 2D image as seen by the robot's camera. It consists of the colour data only.

Normal A 3D vector that is orthogonal to a point on a given plane.

Dataset A well defined set of information that usually consists of large amounts of data and corresponding labels that define an output. In robotics the dataset usually consists of sensor data and labeled outputs based on functionality.

Neural Networks have an input layer ,an output layer and multiple hidden layers. These networks are usually trained using a dataset to predict particular outputs for a given input.

Training The process in which a neural network is trained using a well defined dataset to learn patterns and labels. It is usually a one time process performed at the beginning during which the weights are optimized.

Validation It is the process of testing or proving the validity and accuracy of a neural network.

Cross Validation It is a type of validation where the dataset is split into a training set and test set. After training, the trained model is validated using the test set.

ROS The Robot Operating System is an open-source software platform used for communication and commanding the robot.

ROS Nodes These are software nodes that use ROS to communicate with the robot. Multiple ROS nodes can function simultaneously sending and receiving data to and from the robot.

1.4 Hardware

The goal of the thesis is to implement the functionality of grasping unknown objects on a service robot. For this reason, the Tiago robot made by PAL Robotics is used. The robot has a 7 DOF arm with the wrist having 3 DOF. This aspect of the wrist makes it possible to have a large variety of gripper orientations. Additionally the robot has a torso which is a linear joint and is capable of translational motion. This gives a greater region of reach-ability for the arm. The 3D sensor mounted in the robot's head is an Orbbec Astra RGB-D sensor. The camera provides a 640x480 pixel RGB image, a depth image and a 3D point cloud.

1.5 Software

This section briefly covers the various packages and software used in this thesis. The two most important software parts of this thesis are the convolutional neural network and ROS. The convolutional neural networks are trained using tensorflow and pytorch. Tensorflow and pytorch are platforms to build and train neural networks. Both these platforms have their own extensive libraries for different functionalities. ROS is used mostly for communication between different parts of the robot. In this case ROS Manipulation package MoveIt is used. This package is used just to communicate with the robot arm to reach particular grasp poses and grasp objects. Also the point cloud and images from the robot's 3D sensor is obtained using ROS. The CV Bridge package is used to convert images obtained as ROS messages into OpenCV compatible images. OpenCV is also slightly used in pre-processing the images to suit the needs of the neural network input layer and for post processing the output. The PCL library is used to obtain surface normals of the grasp points to calculate generic orientations.

1.6 Convolutional Neural Networks

Convolutional neural networks are a type of deep neural networks that are built for various applications to get a particular output for a given input. They are used in cases where algorithms do not work due to noise and complications in data. Convolutional Neural Networks are a regularized version of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer, but this does not mean the entire convolutional neural network is fully connected. These neural networks use a mathematical operation called convolution as indicated by the name. A convolutional neural network may have some number of fully connected layers and some number of convolutional layers (not fully connected). Every layer in a convolutional neural network should have the following attributes:

- Convolutional kernels defined by width and height.
- Number of input and output channels.
- Depth of the Convolutional filter (this must be equal to the number of channels of the input feature map).

Every neuron in a convolutional neural network has an input area for a fully connected layer, this area is the entire previous layer and for a convolutional layer, this area is smaller than the previous layer. The input are of a neuron is called it's receptive field. Neurons also are assigned a weight and bias. The weight and bias of a neuron are called filters and they represent a particular feature of the input. While training a neural network, the weights and biases are usually initialized to random default values which are the corrected during the training process.
The functioning of a convolutional neural network during the training process has two steps: Feed Forward and Back Propagation (Figure 1).

Feed Forward In this step the neural network initially makes some random predictions using some given default weights.

Back Propagation In this step the neural network calculates the error in the predictions made in the initial feed forward step using a defined loss function and corrects the weights. The weights are corrected by calculating the gradient of the loss function. Then the feed forward step is repeated using the corrected weights. This process is repeated to achieve minimum loss.

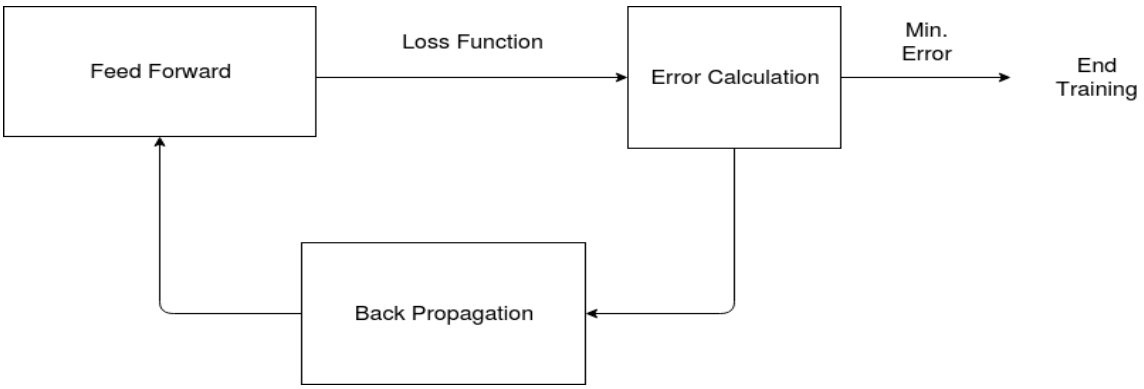

Figure 1: Neural Network training Flow Chart

Convolutional neural networks may also include local or global pooling layers to streamline the computation. Pooling layers reduce the dimensions of the data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. Local pooling acts on small clusters of neurons and global pooling acts on all the neurons of a convolutional layer.
Convolutional neural networks can be used in many applications that consists of a high amount of data. In robotics these networks are usually used to detect patterns in sensor data for particular functionalities such as object detection, grasp detection and so on. A dataset is used to train these networks. The dataset consists of recorded sensor data for a high number of instances and a variety of scenarios labeled with outputs. Once the training is finished, the trained model can be used to detect outputs from the same kind of input data present in the training data.

2 State of the Art

There are many approaches towards grasping unknown objects. In the classical approach, the manipulator is pre-programmed with the object position. But this method will only work in an environment where the process is repetitive. In industries using robot manipulators, the manipulation path is recorded as multiple way-points to reach a goal and this recorded motion is repeated continuously for a particular process. This is usually easy for industrial robot manipulators as the processes in the manufacturing industries are usually repeating cycles. On the other hand the situation is much different in the case of service robots where a single robot is used to perform multiple different tasks. In this case there is a decision making process that needs to be dealt with. Due to this reason, a generic motion planner has to be used such as the planners provided in ROS MoveIt.

2.1 Previous Work

Since this thesis covers grasping unknown objects, this section covers various approaches taken towards the grasping problem in recent years, specifically the approaches that involve machine learning techniques. The approaches are covered in a chronological order. Machine learning techniques have been used for robot manipulation tasks from as far back as the 1990s. But recent developments in machine learning and deep learning algorithms has improved the results of applying the same on robotic manipulation tasks. Jeannette Bohg and Danica Kragic in 2010 have used a vision based grasp predictor using supervised learning [JD10]. They have used several image examples as a training set to achieve this. In 2014, Pavol Bezak, Pavol Bozek and Yuri Nikitin trained a deep learning network that develops a hand-object contact model to achieve successful grasping [PPY14] . In 2015, Ian Lenz, et al., created a deep learning neural network for vision based robotic grasp detection [IAH15] . In 2017, Sergey Levine et al., used a deep learning technique trained with extensive data collection to learn hand-eye coordination [SPAD17]. There are many researchers using similar vision based learning for grasping objects. In 2018, Konstantinos Bousmalis et al., used simulation based training using a dataset of over 25,000 trials to achieve successful grasping [KAPe18]. Abhijit Makhal, et al., created a grasping method that relies on real-time superquadric (SQ) representation of partial view objects and incomplete object modelling, well suited for unknown symmetric objects in cluttered scenarios [AFA18]. Also Gary M. Bone et al., used a wrist mounted camera to capture images of target object from different angles to get a 3D vision based model to predict successful grasps [GAM18]. Philipp Schmidt et al., use a vision based neural network that predicts a single valid grasp from the depth image [PNMT18] (Figure 2).

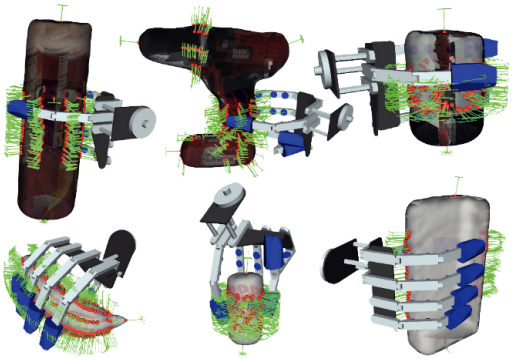

Figure 2: Sample grasps from a Vision based grasping neural network

Tianjian Chen1and and Matei Ciocarlie made an algorithm for grasping unknown objects using proprioception, the combination of joint position and torque sensing [TM18]. Igor Chernov and Wolfgang Ertel used neural networks to fit cuboids in the depth image of the target object creating generic grasping orientations [IW18] (Figure 3). On a similar note, Qujiang Leia, Guangming Chen et al., created a fast grasping algorithm that fits C sections on the target object to define grasps [QGJM18] (Figure 4).

Figure 3: Fitting cuboids on objects

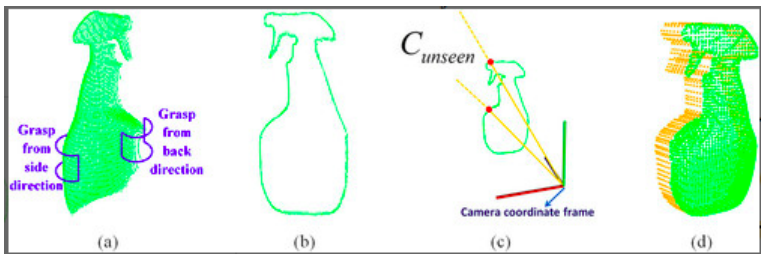

Figure 4: Fitting C sections on objects

In 2019, Sauvet Bruno, Lévesque François, Park SeungJae, et. al. created a three step algorithm to grasp unknown objects from a random pile [SLP+19]. The

three steps involve segmentation of the images, a decision algorithm and ranking according to a grasp robustness index.

2.2 Grasp Representation

Grasp representation is an important part of grasp detection. There are many different ways to represent grasps. The ideal way is to represent grasp poses as a 6 dimensional vector, 3 dimensions that represent a position in 3D space and 3 dimensions that represent 3 angles, such as the Euler angles, that define the orientation. The most recent works in grasping take approaches that represent grasps in lower dimensions such as a grasping point in 3D space (Figure 5), grasp rectangles (Figure 6) and so on. This lower dimensional representation is helpful in many ways. This makes it easier to label grasping datasets and decrease the requirements of the neural network output.

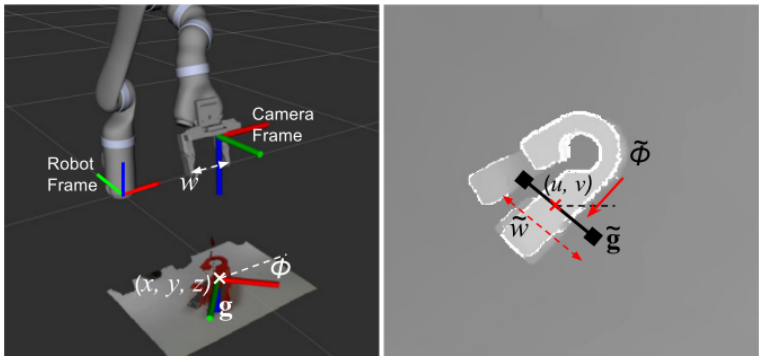

Figure 5: 3D Grasp point representation

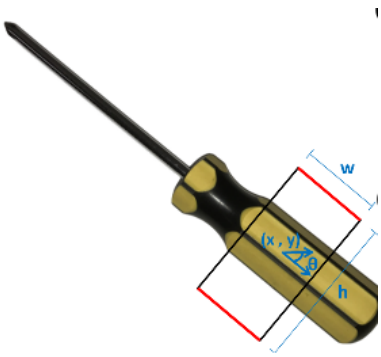

Figure 6: Grasp rectangle representation

3 Method

This section of the thesis will cover the two grasping neural networks (Generative Grasping CNN and RCNN Multi grasp), the two approaches taken to implement these networks on service robots and how these networks are evaluated and compared.

3.1 Grasping Neural Networks

Deep learning and neural networks are very useful in processing high amounts of sensor data. In vision based grasping systems, the sensor data would be the data received from the robot's 3D sensor. In this case, manually writing an algorithm extracting features and patterns in this data would be very complicated. Therefore using neural networks is a logical choice. Grasping neural networks are used to detect grasps on a given object and these networks are usually always vision based networks, i.e, these networks take an image or point cloud from the 3D sensor as an input. To this input the networks predict grasp points, grasp rectangles or other types of grasp representations as an output.

3.1.1 Cornell Grasping Dataset

The Cornell Grasping Dataset [CU09] is a grasping dataset created by the Cornell University, Robot Learning Laboratory. This dataset has been widely used to train grasping neural networks by researchers in the last decade.The dataset consists of 1035 images of 280 different objects. Each image is labelled with grasp rectangles as output. Every instance in the dataset has the RGB Image, Point Cloud, Depth image and the corresponding grasp rectangles. Figure 7 presents a set of sample objects from the Cornell Grasping Dataset.

Figure 7: Cornell Grasping Dataset sample objects

3.1.2 Generative Grasping CNN

The Generative Grasping CNN [DPJ18] is a 6 layer CNN that takes a 300x300 pixel depth image as an input layer. Once a depth image is provided to the network, it performs a pixel-wise grasp detection using features extracted from the depth image. A grasp map (Figure 9) is provided as an output in which every pixel consists of grasp quality, grasp angle and gripper width parameters. M_θ in equation 1 is a simple mathematical representation of the Generative Grasping CNN and g in equation 2 is the grasp representation.

$$M_\theta(I) = (Q_\theta, \Phi_\theta, W_\theta) \tag{1}$$

M - Neural Network, θ - Weights, Q_θ - Grasp Quality, Φ_θ - Grasp Angle and W_θ - Gripper width.

$$g = (s, \phi, w, q) \tag{2}$$

s - Grasp Pose, ϕ - Grasp Angle, w - Gripper Width and q - Grasp Quality

The architecture (Figure 8) of the neural network is a simple 6 layer convolutional architecture. The network follows a simple feed forward and back propagation method as described in section 1.6. It has an L2 loss function (Figure 3) which is used during training for back propagation (refer 1.6). Once a saturation level has been reached, where no more loss reduction can be achieved, then the training is finished. Once the training is done, the trained model can be used to predict grasps in a new scene.

$$\theta = argmin_\theta L(G_T, M_\theta(I_T)) \tag{3}$$

θ - Weights, G - Predicted Grasps, M - Neural Network and I - Depth Image.

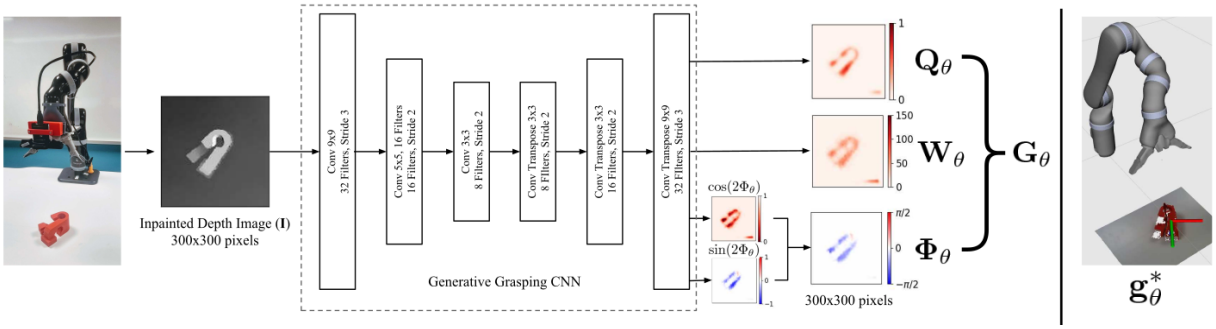

Figure 8: Architecture of the Generative Grasping CNN

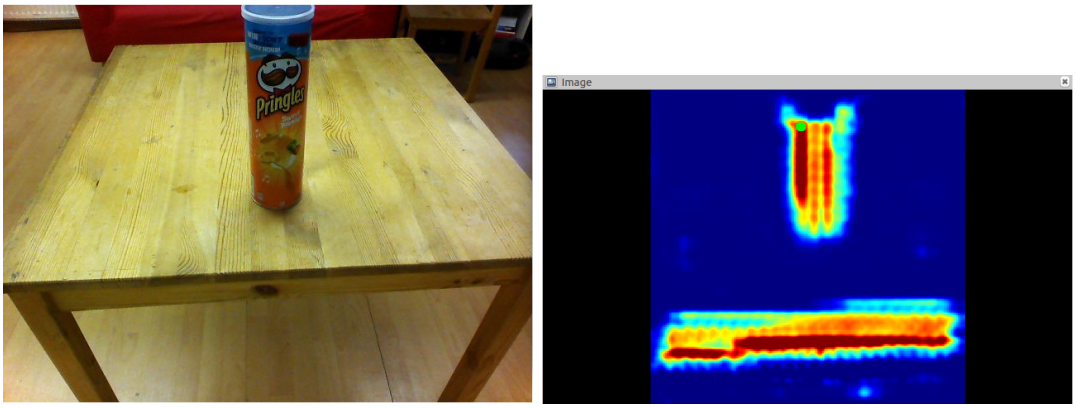

Figure 9: Grasp map of a scene generated by the Generative Grasping CNN

3.1.3 RCNN Multi Grasp Network

The RCNN Multi Grasp network [FJRP18] is a network that is derived from the object detection network, Faster-RCNN [SKRJ15]. The faster-RCNN network takes an input image and gives bounding boxes around detected objects. The RCNN Multi Grasp network is implemented to be trained on a dataset with grasp rectangles instead of bounding boxes. Therefore, the trained model gives grasp rectangles as output when an input RGB image is fed (Figure 11). Although the predictions are made using only 2D data, 3D data is used to create grasps in 3D space based on the predictions. Figure 10 shows the architecture of the RCNN Multi Grasp network.

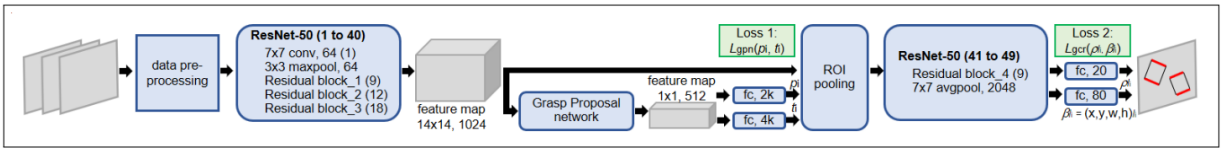

Figure 10: Architecture of the RCNN Multi Grasp

Equation 4 is the grasp representation used by this network. This network uses ResNet-50 [KXSJ16] for feature extraction and then consists of other layers for grasp proposal and loss reduction. The defined loss function (5) is used during back propagation (refer 1.6) to minimize errors in prediction. The loss function is only used during the training process to reach a saturation point in minimizing prediction errors. Once training is done, the trained model can be used for predicting grasps in a new environment.

$$g = x, y, \theta, w, h \tag{4}$$

$$L_{gpn}((p_i, t_i)^{\mathrm{I}}{}_{i=0}) = \sum_i L_{gp_cls}(p_i, p_i^*) + \lambda \sum_i p_i^* L_{gp_reg}(t_i, t_i^*) \tag{5}$$

Figure 11: Grasp predictions by the RCNN Multi Grasp

3.2 Robot Implementation

3.2.1 ROS Implementation

The initial stage of the thesis is to create ROS wrappers for the two neural networks. The ROS wrappers are ROS nodes that receive data from the robot sensors, use the neural networks to make grasp predictions and return the predictions as command signals to the robot hardware. The nodes also perform pre-processing and post-processing of the data. The computation of grasps and conversion of predicted grasps into information acceptable to the robot is done here. Separate ROS nodes are created for both approaches taken in this thesis.

3.2.2 ROS Manipulation

ROS manipulation framework MoveIt [DISN14] is used in this thesis. MoveIt is a manipulation framework with various functionalities such as collision avoidance, inverse kinematic planners and most importantly it enables communication with the robot arm. After the grasping neural networks predict grasps, MoveIt is used to command the robot to execute a valid grasp.

3.2.3 Object detection - Region Of Interest

Object detection plays an important role in grasping objects. The Object detection framework used here is YOLO object detection neural network [Mar18]. Object detection is necessary for the following reasons:

- Adding an object to the robot's planning scene. This is necessary to avoid collision with the object and to be able to grasp the object.

- Eliminating the unnecessary grasps. Object detection provides the robot with a region of interest (Figure 12) and all the grasps predicted outside this region of interest can be eliminated.

- Object detection is also necessary to make the decision of target object in a scene with multiple objects.

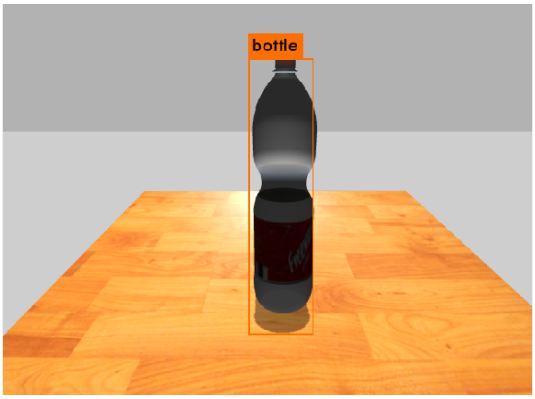

Figure 12: Object detection with YOLO

3.3 Enhancements

This section covers the enhanced implementation approaches used in this thesis. Enhancements are necessary because the two neural networks used in this thesis are originally created for only top-down scenario. The enhancements enable the implementation on a mobile service robot to perform grasping tasks in the 3D world.

3.3.1 Approach 1 - Distinguishing Grasps based on objects

This approach is a direct implementation that makes a grasp decision based on object dimensions. The method uses thresholds on object dimensions to decide the grasp type (front grasp / top-down grasp). Then the orientation of the grasp is defined based on this decision and the grasp angle predicted by the neural network. This approach is not a generic solution but works better for small and irregular objects. Figure 13 represents the decision making process.

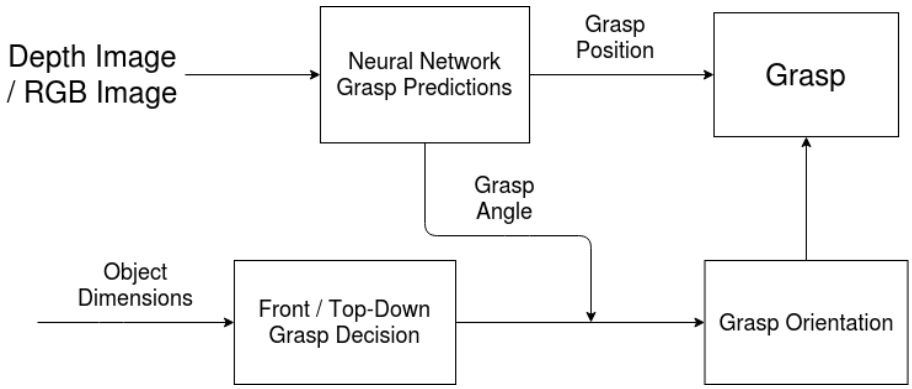

Figure 13: Decision process in Approach 1

3.3.2 Approach 2 - Using Surface normals to create grasps

This approach is a generic approach which extracts the surface normals (Figure 14) to determine the grasp orientation. Once the neural network predicts the grasp points in a scene, the algorithm extracts the surface normals of these grasp points to calculate the orientations.

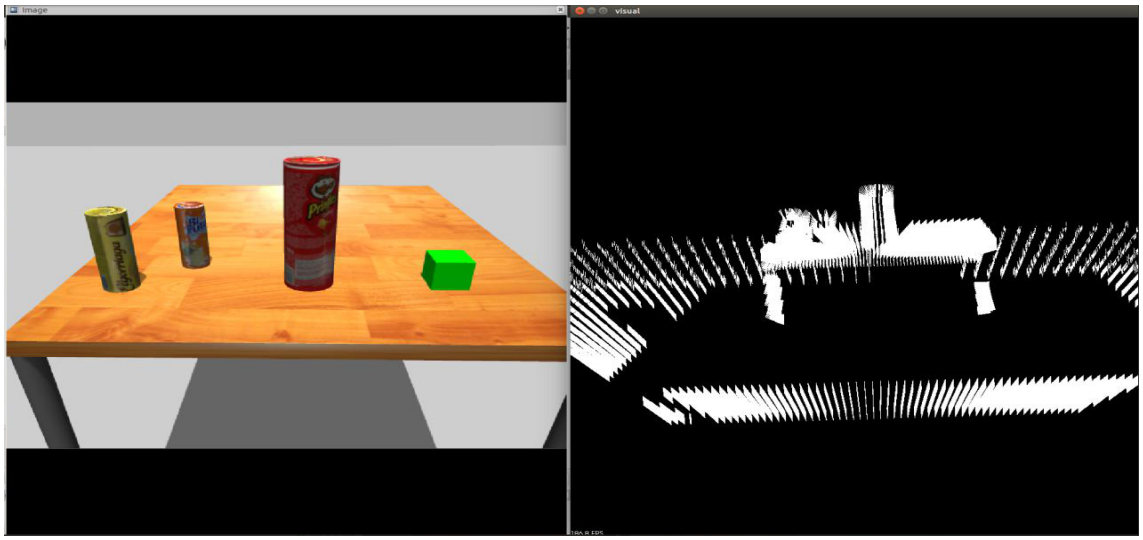

Figure 14: Surface normals of a scene extracted from the point cloud

Let x_n, y_n, z_n be the extracted surface normal. A 4D matrix M (refer 6) using the surface normal and two vectors orthogonal to the surface normal are created.

$$M = \begin{bmatrix} x_n & \sqrt{1+\frac{x_n^2}{z_n^2}} & 0 & 0 \\ y_n & 0 & \sqrt{1+\frac{y_n^2}{z_n^2}} & 0 \\ z_n & (\sqrt{1+\frac{x_n^2}{z_n^2}})(\frac{-x_n}{z_n}) & (\sqrt{1+\frac{y_n^2}{z_n^2}})(\frac{-y_n}{z_n}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Next the matrix is multiplied by a rotation matrix R (refer 7), which represents rotation about the surface normal for an angle α. α being the grasp angle provided by the neural network corresponding to the grasp point. Then the rotated matrix M_R (refer 8) is converted into a quaternion Q_M (refer 9).
Let t = 1 - $\cos\alpha$,

$$R = \begin{bmatrix} t * x_n^2 + \cos\alpha & t * x_n * y_n - z_n * \sin\alpha & t * x_n * z_n + y_n * \sin\alpha & 0 \\ t * x_n * y_n + z_n * \sin\alpha & t * y_n^2 + \cos\alpha & t * y_n * z_n - x_n * \sin\alpha & 0 \\ t * x_n * z_n - y_n * \sin\alpha & t * y_n * z_n + x_n * \sin\alpha & t * z_n^2 + \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

$$M_R = R * M \tag{8}$$

$$Q_M = \begin{bmatrix} (M_R[2,1] - M_R[1,2])/2 * \sqrt{(1 + M_R[0,0] + M_R[1,1] + M_R[2,2])} \\ (M_R[0,2] - M_R[2,0])/2 * \sqrt{(1 + M_R[0,0] + M_R[1,1] + M_R[2,2])} \\ (M_R[2,1] - M_R[1,2])/2 * \sqrt{(1 + M_R[0,0] + M_R[1,1] + M_R[2,2])} \\ \sqrt{(1 + M_R[0,0] + M_R[1,1] + M_R[2,2])}/2 \end{bmatrix} \tag{9}$$

Next a quaternion Q_{AA} is calculated using only the surface normal and angle α (refer 10). The two calculated quaternions are multiplied to get the final gripper orientation Q_g (refer 11).

$$Q_{AA} = normalize \begin{bmatrix} x_n \\ y_n \\ z_n \\ \cos 2\alpha \end{bmatrix} \tag{10}$$

$$Q_g = Q_M * Q_{AA} \tag{11}$$

3.4 Control Flow Chart

This section consists of detailed flow diagrams of both the approaches implemented in this thesis. The flow charts consist of all the important processes in the implementation. Figure 15 represents approach 1 which distinguishes grasps based on object dimensions and Figure 16 represents approach 2 which creates generic grasps using surface normals of the predicted grasp points.

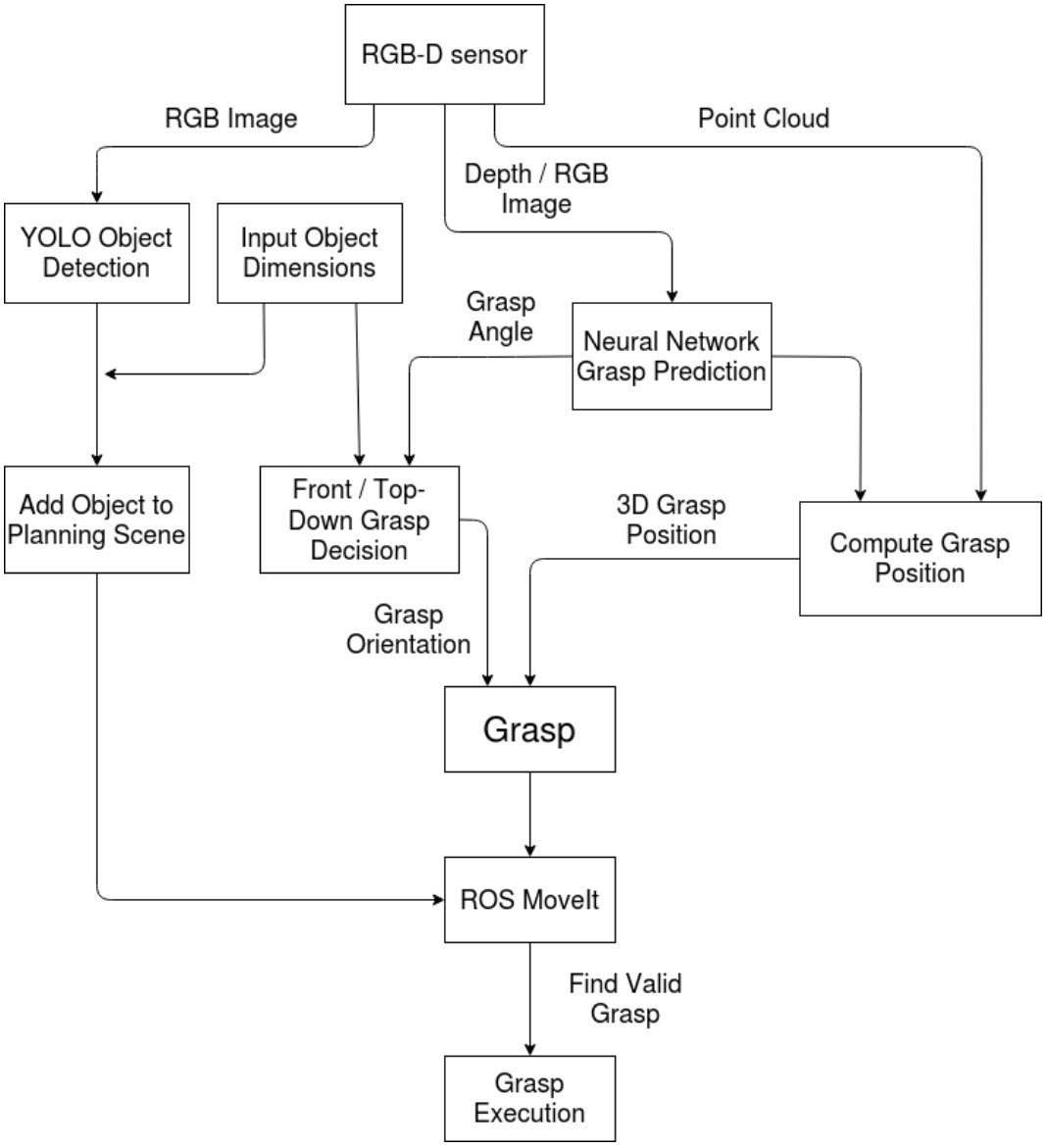

Figure 15: Flowchart representation of Approach 1

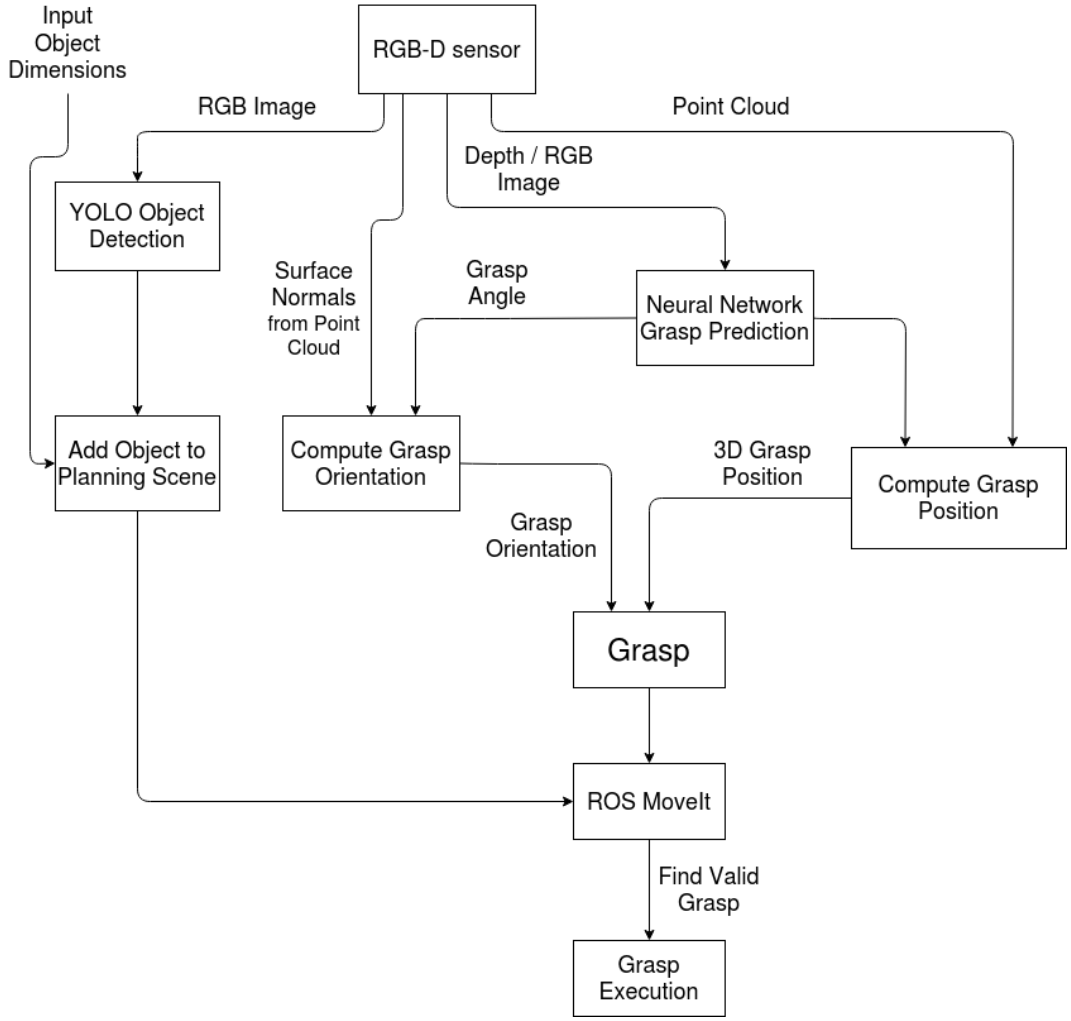

Figure 16: Flowchart representation of Approach 2

4 Experiments and Results

4.1 Evaluation and Comparison Method

This section covers the evaluation process, metrics, bench-marking and experimentation. Initially, the first task is to create ROS wrappers and implement the two grasp detection neural networks on the robot simulation. Then initial testing is performed on a few random objects to debug the implementation. The next step is to define a set of objects to perform experiments on, in this case the a subset of the RoboCup@Home German Open [Rob19] object set. The object set used for experiments does not overlap with the objects in the training set. Simulated models of the RoboCup object set are created to run experiments on ROS Gazebo. The initial evaluation involve experiments using approach 1 (refer 3.3.1) performed on both the neural networks. The experiments are performed on 21 objects, 5 trials per object per network, giving a total of 210 trials. For all experiments, the object position, robot position and success/failure are noted. The results of the experiments performed using approach 1 are used as a benchmark to compare the results of approach 2 (refer 3.3.2). The same experiments are performed using approach 2 and the results are compared.

Evaluation Metric The evaluation metric used to evaluate the pipelines created using the two neural networks is Force Closure (Figure 17) [SDTe11]. Force Closure is achieved when an object is present in between the parallel grippers restricting the complete closure of the gripper causing the grippers to exert force on the object. In this thesis, a grasp is considered as successful when Force Closure is achieved.

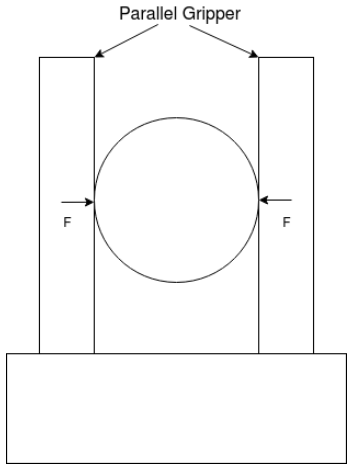

Figure 17: Force Closure

Results and conclusions are drawn based on comparing the two neural networks in various cases. The neural networks are compared in the following cases:

- Overall success rates of the two neural networks based on force closure.
- Success rates of the neural networks in approach 1 and approach 2.
- Performance of the neural networks on particular groups of objects such as cylindrical, cuboid and so on.
- Finally the performance of the neural networks on the real robot.

The real robot experiments are performed only using the second approach which is the most enhanced version of the implementation. The object set used for real robot experiments is a subset of the objects used in the simulated experiments and a few extra difficult objects (Figure 19). Real world experiments are performed on 15 objects, 7 trials per object per network, giving a total of 210 trials.

4.2 Experiments

The experimental setup for all the experiments performed in this thesis is a table-top scene. The goal is to grasp objects present on a table. The same setup is used in simulated experiments and real world experiments. The objects are created in the simulated environment (see Figure 18) and the experiments are performed.

Figure 18: Simulated Objects from the RoboCup@Home German Open object set

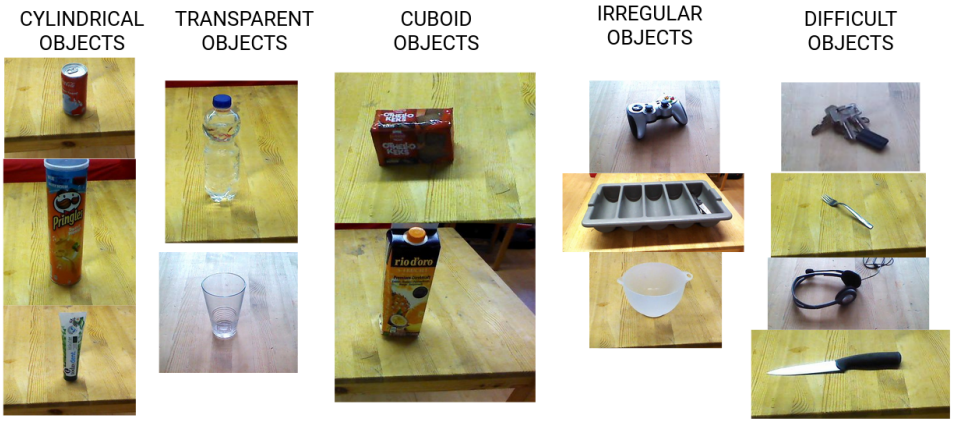

Figure 19: Objects used for real world experiments

The positions of the objects in the experiments are random but are the same for all objects (refer Table 1 , 2 and Figure 20). A specific set of object positions are not defined in most grasping research work because random object positions are more challenging and object positions cannot be defined in a generic method for all environments.

TRIAL	OBJECT POSITION
TRIAL 1	x: 0.7 y: 0.0 z: 0.85
TRIAL 2	x: 0.7 y: 0.07 z: 0.85
TRIAL 3	x: 0.65 y: -0.07 z: 0.85
TRIAL 4	x: 0.8 y: -0.1 z: 0.85
TRIAL 5	x: 0.75 y: 0.1 z: 0.85

Table 1: Object Positions for Simulated Experiments in Gazebo world frame

TRIAL	OBJECT POSITION
TRIAL 1	Position: x: 0.673 y: -0.02 z: 0.87
TRIAL 2	Position: x: 0.723 y: -0.13 z: 0.87
TRIAL 3	Position: x: 0.702 y: 0.072 z: 0.87
TRIAL 4	Position: x: 0.697 y: 0.031 z: 0.87
TRIAL 5	Position: x: 0.692 y: 0.127 z: 0.87
TRIAL 6	Position: x: 0.774 y: -0.08 z: 0.87
TRIAL 7	Position: x: 0.757 y: 0.102 z: 0.87

Table 2: Object Positions for Real World Experiments in robot footprint frame

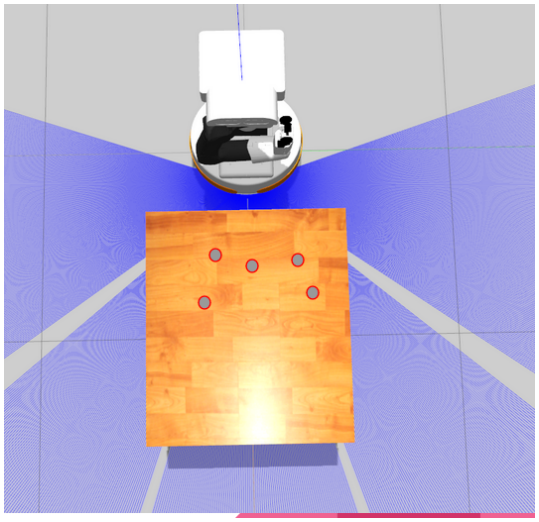

Figure 20: Object Positions in Simulation

Totally 616 experiments are performed in the following six different cases:

CASES	Environment	GGCNN	RCNN	Objects	Trials
Approach 1	Simulated	Table 4	Table 5	21	210
Approach 2	Simulated	Table 6	Table 7	21	210
	Real World	Table 8	Table 9	14	196

Table 3: Experimental cases and trials

[1] [2]

4.3 Results

This section consists of results based on object groups and comparison of both the neural networks in the two different approaches. Further the two approaches are also compared with each other and conclusions are drawn based on these comparisons. The objects used in the simulated and real world experiments are classified into groups in order to make comparisons and conclusions. The simulated objects are divided into 4 groups: Cylindrical objects, Cuboid objects, Irregular objects and Spherical objects (refer Table 10. The real world objects are divided into 5 groups: Cylindrical objects, Cuboid objects, Irregular objects, Transparent objects and Difficult objects (refer Table 10). The group difficult objects contains objects that are small and difficult to grasp. The conclusion also contains a proposal on how to further improve the results.

[1] GGCNN - Generative Grasping CNN
[2] RCNN - RCNN Multi Grasp

Figures 21, 22 and 23 represent the comparative results based on grouped objects for all the six experimental cases.

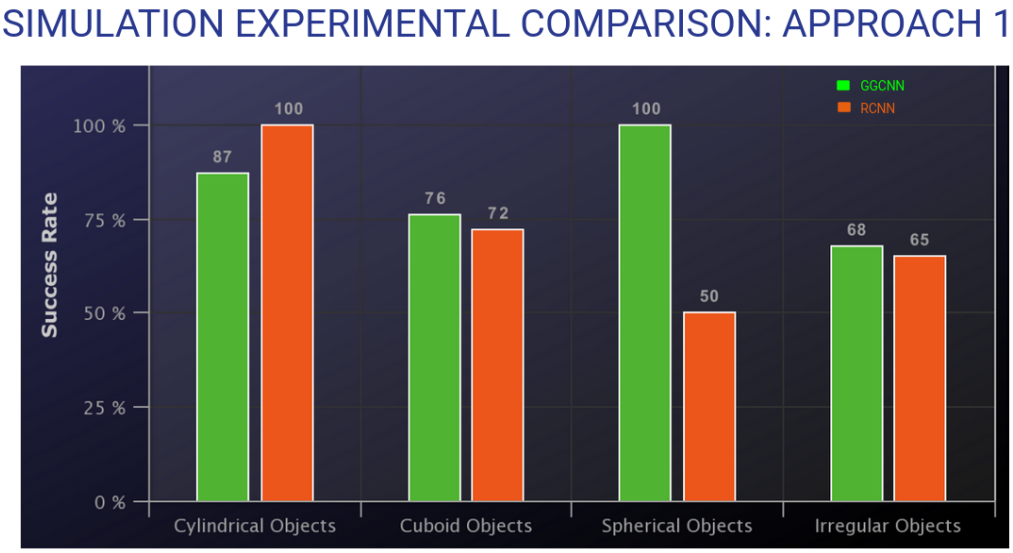

Figure 21: Success rates based on grouped objects for Approach 1

Figure 22: Success rates based on grouped objects for Approach 2

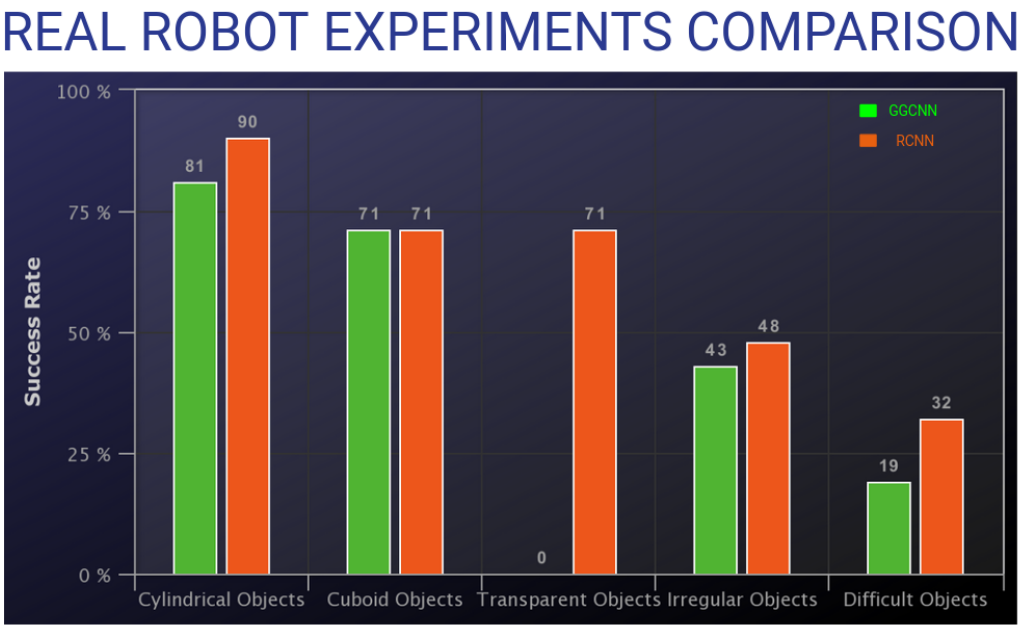

Figure 23: Success rates based on grouped objects for Real Robot Experiments

4.3.1 Approach 1 vs Approach 2

The first approach taken in this thesis decides grasp type based on object dimensions (refer section 3.3.1). Although this approach is not generic, this approach is a good solution to grasp small and irregular objects because it is easier to grasp small objects in a straight top-down manner. Moreover, since Approach 2 uses surface normals, irregular objects will have surface normals that will lead to bad gripper orientations (Figure 24). The results based on grouped objects in Figures 21 and 22 show clearly that both neural networks perform better for irregular objects in approach 1 than in approach 2.

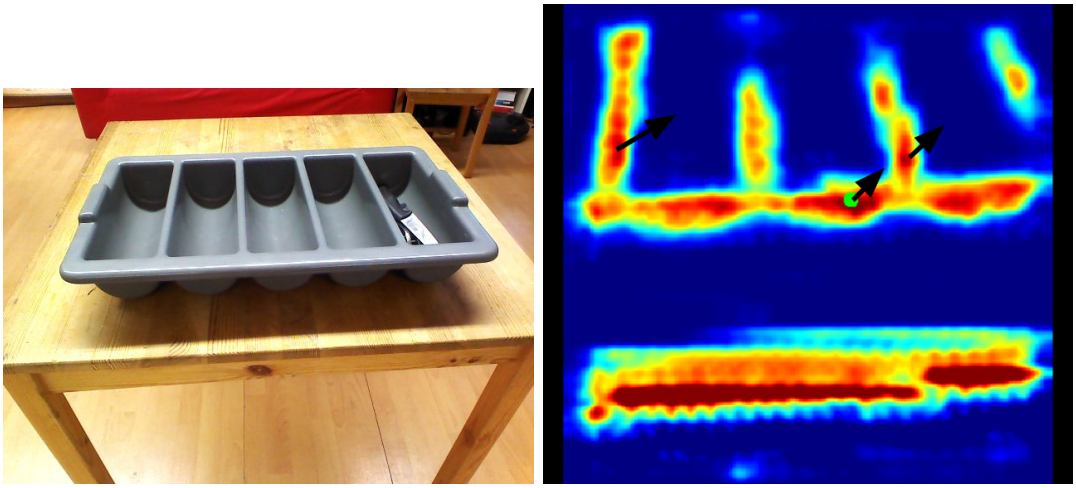

Figure 24: Bad Surface Normals at grasp points on irregular object

The major problem with approach 1 occurs when detecting grasps for large objects. Since approach 1 only uses standard front grasp and top-down grasp orientations, when grasps are detected with an offset from the center of large objects, grasping fails. This problem is resolved in the second approach by using surface normals to create orientations that make the gripper align with the surface normal. The surface normal approach only solves this problem for cylindrical and spherical objects not for cuboid objects. It can be seen from the results based on grouped objects in Figures 21 and 22 that the performance on cylindrical and spherical objects is better in approach 2 than approach 1.

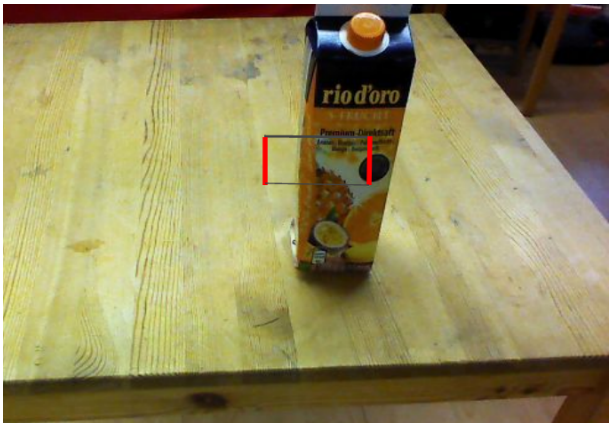

Figure 25: Example corner grasp that leads to failure

4.3.2 Generative Grasping CNN vs RCNN Multi Grasp

The major difference between the Generative Grasping CNN (refer 3.1.2) and the RCNN Multi Grasp (refer 3.1.3) is that the former predicts grasps from depth images and the later uses RGB images. This section will cover how the inputs of the neural networks affect the grasping performance in specific cases.
The Generative Grasping CNN is incapable of predicting grasps for transparent objects while the RCNN Multi Grasp is capable (Figure 26). This is because transparent objects are not properly visible in the depth image and point cloud produced by the RGB-D sensor but transparent objects are clearly visible in RGB images.
Predicting grasps for small objects also poses a similar problem since small objects are not properly visible on the depth image and this also greatly depend on lighting conditions. But when it comes to RGB images, small objects are clear and predictions mostly are independent of lighting. As a result the RCNN Multi Grasp performs better on small objects than the Generative Grasping CNN (Figure 27).
It is very clear from Figures 21, 22 and 23 that the RCNN Multi Grasp network performs better, in terms of grasp detection, than the Generative Grasping CNN in most cases. The only major advantage of using depth image is that the grasp poses are in 3D.

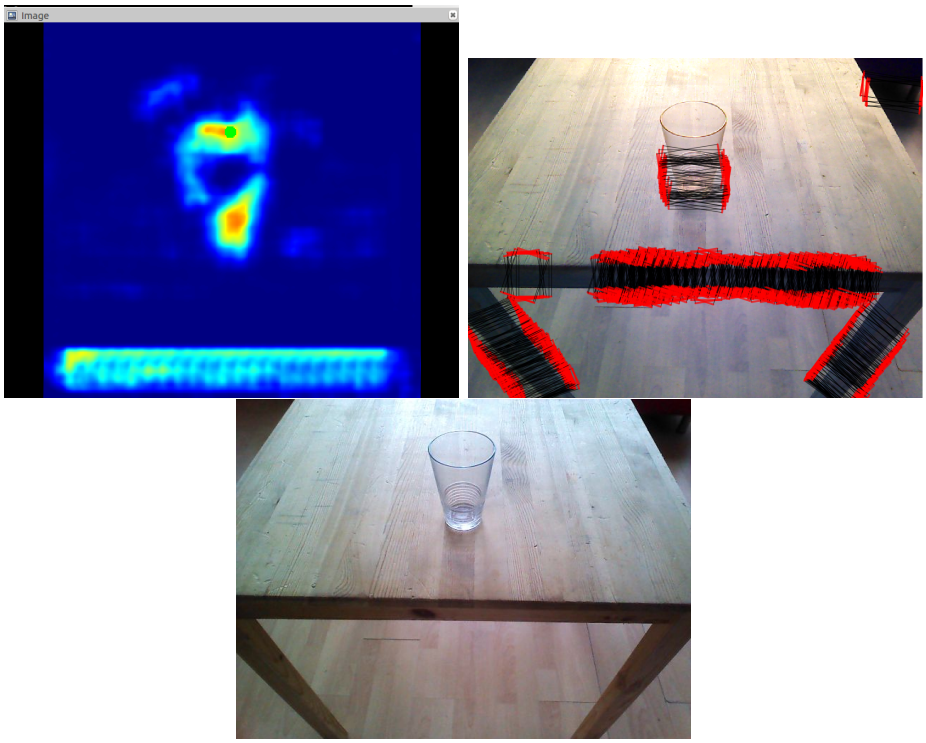

Figure 26: Grasp predictions by the two neural networks for a transparent object

Figure 27: Grasp predictions by the two neural networks for a small object

5 Conclusion and Future Work

The conclusion of this thesis is drawn from the experimental results and comparisons as explained in section 4.3. The initial conclusion is that the RCNN Multi Grasp which uses RGB images to predict grasps outperforms the Generative Grasping CNN in most cases. This is specifically noticeable in the cases of transparent, small and irregular objects.

The results based on grouped objects show categorically the specific cases in which the two approaches thrive. Both the approaches have positives and negatives. Based on this it would be possible to improve the results further by combining the two approaches. In this new approach, the algorithm would initially distinguish the grasp type based on the object dimensions, then if the grasp type is front/side grasp the surface normal approach will be used and for small objects that need top-down grasps the first approach is used. Implementing this new approach would solve most of the problems that exist while individually implementing the two approaches.

The Tiago robot's arm to camera calibration was not optimal during the period of experimentation. Performing a calibration to optimize this would lead to better results. The Tiago robot also has self-collision problems. If these problems are solved the performance would improve substantially. Further the neural networks were trained on the Cornell Grasping Dataset which is a comparatively small dataset. Using larger datasets, that consist of more objects and more instances per object, to train the neural networks would also improve the results further.

6 Appendix

GENERATIVE GRASPING CNN EXPERIMENTS - APPROACH 1					
OBJECT	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
Basket	X		X	X	
Bowl					
Can Food	X	X	X	X	X
Candy Bar	X	X	X	X	X
Cereal box		X	X		X
Cup	X	X	X	X	X
Cola Tin	X	X	X	X	X
Cola Bottle	X	X	X		X
Energy Drink		X	X	X	X
Fork	X	X	X	X	X
Juice Box		X	X	X	
Knife	X	X	X	X	X
Lemon	X	X	X	X	X
Milk Box	X			X	X
Orange	X	X	X	X	X
Plate					
Pringles	X	X		X	
Shower Gel	X		X	X	X
Soap	X	X	X	X	X
Spoon	X	X	X	X	X
Toothpaste	X	X	X	X	X
Overall Success Rate = 77%					

Table 4: Generative Grasping CNN Experiments - Approach 1 (X denotes a successful grasp)

RCNN MULTI GRASP EXPERIMENTS - APPROACH 1					
OBJECT	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
Basket		X	X		X
Bowl					
Can Food	X	X	X	X	X
Candy Bar	X	X	X		X
Cereal box	X			X	
Cup	X	X	X	X	X
Cola Tin	X	X	X	X	X
Cola Bottle	X	X	X	X	X
Energy Drink	X	X	X	X	X
Fork	X	X	X	X	X
Juice Box	X	X		X	
Knife	X		X	X	X
Lemon	X	X	X	X	X
Milk Box	X	X		X	X
Orange					
Plate					
Pringles	X	X	X	X	X
Shower Gel	X	X	X	X	X
Soap	X	X	X	X	X
Spoon	X	X	X	X	
Toothpaste	X	X	X	X	X
Overall Success Rate = 75%					

Table 5: RCNN Multi Grasp Experiments - Approach 1 (X denotes a successful grasp)

GENERATIVE GRASPING CNN EXPERIMENTS - APPROACH 2					
OBJECT	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
Basket	X		X		X
Bowl					
Can Food	X	X	X	X	X
Candy Bar	X	X	X	X	X
Cereal box		X	X		X
Cup	X	X	X	X	X
Cola Tin	X	X	X	X	X
Cola Bottle	X	X	X	X	X
Energy Drink	X	X	X	X	X
Fork	X	X			X
Juice Box	X	X	X	X	X
Knife	X		X	X	
Lemon	X	X	X	X	X
Milk Box	X	X	X		
Orange	X	X	X	X	X
Plate					
Pringles	X	X	X	X	X
Shower Gel	X	X	X	X	X
Soap	X	X	X	X	X
Spoon			X	X	
Toothpaste	X	X	X	X	X
Overall Success Rate = 78%					

Table 6: Generative Grasping CNN Experiments - Approach 2 (X denotes a successful grasp)

RCNN MULTI GRASP EXPERIMENTS - APPROACH 2					
OBJECT	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
Basket		X	X		X
Bowl					
Can Food	X	X	X	X	X
Candy Bar	X	X	X	X	
Cereal box	X	X		X	X
Cup	X	X	X	X	X
Cola Tin	X	X	X	X	X
Cola Bottle	X	X	X	X	X
Energy Drink	X	X	X	X	X
Fork	X		X	X	X
Juice Box	X	X	X	X	X
Knife		X	X	X	X
Lemon	X	X	X	X	X
Milk Box	X	X	X	X	X
Orange	X	X	X	X	X
Plate					
Pringles	X	X	X	X	X
Shower Gel	X	X	X	X	X
Soap	X		X	X	
Spoon	X	X			X
Toothpaste	X	X	X	X	X
Overall Success Rate = 80.9%					

Table 7: RCNN Multi Grasp Experiments - Approach 2 (X denotes a successful grasp)

GENERATIVE GRASPING CNN REAL ROBOT EXPERIMENTS							
OBJECT	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5	TRIAL 6	TRIAL 7
Pringles	X	X		X	X	X	X
Water Bottle							
Small Cola	X	X	X	X	X	X	
Fork							
Knife							
Biscuit Pack	X		X	X		X	X
Toothpaste	X	X	X			X	X
Bowl				X			
Juice Box		X	X	X		X	X
Headphones	X			X	X		
Controller	X	X		X			X
Keys						X	
Glass							
Tray	X				X	X	X
Overall Success Rate = 40%							

Table 8: Generative Grasping CNN - Real Robot Experiments (X denotes a successful grasp)

RCNN MULTI GRASP REAL ROBOT EXPERIMENTS							
OBJECT	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5	TRIAL 6	TRIAL 7
Pringles	X	X	X	X		X	X
Water Bottle	X	X	X	X		X	X
Small Cola	X		X	X	X	X	X
Fork							
Knife			X				
Biscuit Pack		X	X	X		X	X
Toothpaste	X	X	X	X	X	X	X
Bowl			X		X		
Juice Box	X			X	X	X	X
Headphones	X		X	X	X		X
Controller		X		X	X	X	
Keys		X				X	
Glass		X		X	X		X
Tray		X	X		X	X	
Overall Success Rate = 58%							

Table 9: RCNN Multi Grasp - Real Robot Experiments (X denotes a successful grasp)

Simulated Objects Grouped into Classes				
Cylindrical Objects	Cuboid Objects	Irregular Objects	Spherical Objects	
Can Food	Candy Bar	Basket	Lemon	
Cola Bottle	Cereal Box	Bowl	Orange	
Cola Tin	Juice Box	Fork		
Cup	Milk Box	Knife		
Energy Drink	Soap	Plate		
Pringles		Shower Gel		
		Spoon		
		Toothpaste		
Real World Objects Grouped into Classes				
Cylindrical Objects	Cuboid Objects	Irregular Objects	Transparent Objects	Difficult Objects
Pringles	Juice Box	Controller	Water Bottle	Headphones
Cola Tin	Biscuit Pack	Bowl	Glass	Keys
Toothpaste		Tray		Knife
				Fork

Table 10: Objects Grouped into Classes

References

[AFA18] Abhijit, Makhal ; Frederico, Thomas ; Alba, Perez G.: Grasping Unknown Objects in Clutter by Superquadric Representation. In: *2018 Second IEEE International Conference on Robotic Computing (IRC)* (2018)

[CU09] Cornell University, Robot Learning L.: *Cornell Grasping Dataset*. http://pr.cs.cornell.edu/grasping/rect_data/data.php. Version: 2009

[DISN14] David, Coleman ; Ioan, Sucan ; Sachin, Chitta ; Nikolaus, Correll: Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study. In: *arXiv preprint arXiv:1404.3785* (2014)

[DPJ18] Douglas, Morrison ; Peter, Corke ; Jürgen, Leitner: Closing the Loop for Robotic Grasping: A Real-time, Generative Grasp Synthesis Approach. In: *Robotics: Science and Systems (RSS), 2018* (2018)

[FJRP18] Fu-Jen, Chu ; Ruinian, Xu ; Patricio, A. V.: Real-world Multiobject, Multi-grasp Detection. In: *IEEE Robotics and Automation Letters* (2018)

[GAM18] Gary, M. B. ; Andrew, Lambert ; Mark, Edwards: Automated modeling and robotic grasping of unknown three-dimensional objects. In: *IEEE International Conference* (2018)

[IAH15] Ian, Lenz ; Ashutosh, Saxena ; Honglak, Lee: Deep learning for detecting robotic grasps. In: *The International Journal of Robotics Research, 2015* (2015)

[IW18] Igor, Chernov ; Wolfgang, Ertel: Generating Optimal Gripper Orientation for Robotic Grasping of Unknown Objects using Neural Network. In: *Federated AI for Robotics Workshop (FAIR), IJCAI-ECAI-18, Stockholm* (2018)

[JD10] Jeannette, Bohg ; Danica, Kragic: Learning grasping points with shape context. In: *Robotics and Autonomous Systems, Volume 58, Issue 4, 30 April 2010, Pages 362-377.* (2010)

[KAPe18] Konstantinos, Bousmalis ; Alex, Irpan ; Paul, Wohlhart ; et., al.: Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping. In: *IEEE International Conference* (2018)

[KXSJ16] Kaiming, He ; Xiangyu, Zhang ; Shaoqing, Ren ; Jian, Sun: Deep Residual Learning for Image Recognition. In: *2016 IEEE International Conference* (2016)

[Mar18] Marko, Bjelonic: *YOLO ROS: Real-Time Object Detection for ROS.* https://github.com/leggedrobotics/darknet_ros, 2018

[PNMT18] Philipp, Schmidt ; Nikolaus, Vahrenkamp ; Mirko, Wachter ; Tamim, Asfour: Grasping of Unknown Objects using Deep Convolutional Neural Networks based on Depth Images. In: *IEEE International Conference on Robotics and Automation (ICRA), May 21-25, 2018, Brisbane, Australia: Proceedings* (2018)

[PPY14] Pavol, Bezak ; Pavol, Bozek ; Yuri, Nikitin: Advanced Robotic Grasping Systems using Deep Learning. In: *Modelling of Mechanical and Mechatronic Systems MMaMS 2014* (2014)

[QGJM18] Qujiang, Leia ; Guangming, Chen ; Jonathan, Meijer ; Martijn, Wisse: A novel algorithm for fast grasping of unknown objects using C-shape configuration. In: *AIP Advances,Volume 8, Issue 2 10.1063/1.5006570* (2018)

[Rob19] RoboCup@Home: *Robocup@Home German Open 2019.* https://github.com/RoboCupAtHome/GermanOpen2019. Version: 2019

[SDTe11] S, Ulbrich ; D, Kappler ; T, Asfour ; et., al.: The OpenGRASP Benchmarking Suite: An Environment for the Comparative Analysis of Grasping and Dexterous Manipulation. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2011)

[SKRJ15] Shaoqing, Ren ; Kaiming, He ; Ross, Girshick ; Jian, Sun: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence. 39. 10.1109/TPAMI.2016.2577031* (2015)

[SLP+19] Sauvet, Bruno ; Lévesque, François ; Park, SeungJae ; Cardou, Philippe ; Gosselin, Clément: Model-Based Grasping of Unknown Objects from a Random Pile. In: *Robotics. 8. 79. 10.3390/robotics8030079* (2019)

[SPAD17] Sergey, Levine ; Peter, Pastor ; Alex, Krizhevsky ; Deirdre, Quillen: Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. In: *The International Journal of Robotics Research, 2017* (2017)

[TM18] Tianjian, Chen1and ; Matei, Ciocarlie: Proprioception-Based Grasping for Unknown Objects Using a Series-Elastic-Actuated Gripper. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018)