Learning to grasp unknown objects in domestic environments

A dissertation submitted for the degree of
Doctor of Philosophy

By:

Anna Konrad

Under the supervision of:

Dr. Rudi Villing

Prof. John McDonald

Hamilton Institute
National University of Ireland Maynooth
Ollscoil na hÉireann, Má Nuad

December 2023

Abstract

For robots to become valuable assistants in real-world scenarios, they must effectively manipulate unknown objects in complex environments, such as homes with multiple objects on shelves or desks. A critical aspect of manipulating objects is grasping them with the robot's end-effector. The key challenge for grasping an unknown object lies in determining how to position and orientate the end-effector relative to the object. While prior research has predominantly focused on table-top scenarios with fixed camera-workspace transforms, in this thesis, we extend robotic grasping to domestic settings.

This thesis makes three principal contributions in this area: (i) extending a depth-image-based grasp quality prediction method to accommodate 6-DoF grasps and versatile camera poses, (ii) creating a depth-image-based grasp proposal method tailored for 6-DoF grasps and unknown camera-workspace transforms, and (iii) applying such a grasp proposal method in complex, domestic environments. For application in domestic environments, we develop a simulation pipeline specifically designed for training and evaluating grasp proposal models in domestic settings.

Quantitative experiments demonstrate competitive or superior performance of our approach when compared with widely used techniques across a number of challenging simulated experiments. Additionally, real-world experiments with a mobile manipulator demonstrate successful grasping of unknown objects positioned on tables and shelves in unknown poses. We make our code and pre-trained models available for fellow researchers, facilitating the direct application of our models when attempting to grasp unknown objects with mobile manipulators in domestic environments.

Zusammenfassung

Damit Roboter in Alltagssituationen als wertvolle Assistenten fungieren können, müssen sie unbekannte Objekte in komplexen Umgebungen, z. B. in Wohnungen mit Regalen oder Schreibtischen, effektiv manipulieren. Ein entscheidender Aspekt bei der Handhabung von Objekten ist das Greifen mit dem Endeffektor des Roboters. Die zentrale Herausforderung dabei besteht darin, die Position und Ausrichtung des Endeffektors relativ zum Objekt zu bestimmen. Während sich die bisherige Forschung vorwiegend auf Tischszenarien mit festen Kamera-Arbeitsraum-Transformationen konzentriert hat, wird in dieser Arbeit das Greifen mit Robotern auf häusliche Umgebungen ausgeweitet.

Die vorliegende Arbeit leistet drei konkrete Beiträge in diesem Bereich: (i) Erweiterung einer tiefenbildbasierten Methode zur Einschätzung der Griffqualität, um Greifpositionen mit 6 Freiheitsgraden und vielseitige Kamerapositionen zu berücksichtigen, (ii) Erstellung eines tiefenbildbasierten Modells, das Greifpositionen mit 6 Freiheitsgraden basierend auf unbekannten Kamera-Arbeitsraum-Transformationen vorschlägt, und (iii) Anwendung eines solchen Modells in komplexen, häuslichen Umgebungen. Für die Anwendung in häuslichen Umgebungen entwickeln wir eine Simulationsumgebung, die speziell für das Training und die Evaluierung von Modellen zum Vorschlag von Greifpositionen in häuslichen Umgebungen entwickelt wurde.

Quantitative Experimente zeigen in einer Reihe von anspruchsvollen simulierten Experimenten, dass unser Ansatz im Vergleich zu weit verbreiteten Lösungsansätzen konkurrenzfähig oder überlegen ist. Experimente mit einem realen mobilen Manipulator zeigen das erfolgreiche Greifen unbekannter Objekte, die auf Tischen und in Regalen positioniert sind. Unsere Software und die trainierten Modelle stehen anderen Forschern zur Verfügung und ermöglichen die direkte Anwendung unserer Modelle beim Versuch, unbekannte Objekte mit mobilen Manipulatoren in häuslichen Umgebungen zu greifen.

Declaration

I hereby declare that I have produced this manuscript without the prohibited assistance of any third parties and without making use of aids other than those specified.

The thesis work was conducted from September 2019 to December 2023 under the supervision of Dr. Rudi Villing and Prof. John McDonald in the Hamilton Institute, Maynooth University, Ireland.

Anna Konrad.

Maynooth, Ireland,

December 2023.

Sponsor

This work was supported by Grants from Science Foundation Ireland under Grant numbers 18/CRT/6049 and 16/RI/3399.

Publications

Chapter 3 has been published in the 2022 International Joint Conference on Neural Networks (IJCNN). Chapter 4 has been published in the 2023 International Conference on Advanced Robotics (ICAR). Chapter 5 is currently under preparation for submission to a journal.

Peer-reviewed conference articles:

- Konrad, A., McDonald J. & Villing, R. VGQ-CNN: Moving Beyond Fixed Cameras and Top-Grasps for Grasp Quality Prediction. 2022 International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 2022. https://doi.org/10.1109/IJCNN55064.2022.9892763.
- Konrad, A., McDonald J. & Villing, R. GP-net: Flexible Viewpoint Grasp Proposal. 2023 International Conference on Advanced Robotics (ICAR), Abu Dhabi, United Arab Emirates, 2023. https://doi.org/10.1109/ICAR58858.2023.10406781.

In preparation for submission:

- Konrad, A., McDonald J. & Villing, R. Grasping Objects in Domestic Environments.

Acknowledgements

First and foremost, I would like to thank my supervisors, Dr. Rudi Villing and Prof. John McDonald, for their guidance, encouragement and unwavering support throughout the last years. Every piece of feedback you provided has improved this work and motivated me to give my very best. I will never forget your calm optimism (and uplifting jokes) during the most difficult moments of this journey.

I would like to thank everyone in the Centre for Research Training in Foundations of Data Science for their great support and for making my PhD journey possible. In particular, I would like to thank Prof. David Malone, Prof. Ken Duffy, Janet Clifford and Joanna O'Grady for their assistance, the wonderful opportunities they provided and their continuous encouragement.

My PhD experience would not have been the same without all of the wonderful people at the Hamilton Institute, who provided emotional support, accompanied me on all the strolls and shared emergency chocolate with me. You are the absolute best! Rosemary and Kate, thank you for the chats and the stretching sessions - I miss them already. Thank you, Hazel, Sineád, Aoife, Dáire, Kevin, Andy, Mark, and Pete, for being the best adventure friends and introducing me to my favourite parts of Ireland.

Danke an Julia, Tobi, Beate, Karl und Steffen, für die endlosen Telefonate, das viele Zuhören, die Aufmunterungen und fürs immer-da-sein. Ihr seid die Besten!

Contents

List of Figures

List of Tables

Glossary

4-DoF grasp Grasp pose defined by a 3D position and a 1D orientation.

6-DoF grasp Grasp pose defined by a 3D position and a 3D orientation.

grasp synthesis Finding a grasp configuration for a given object that satisfies a set of criteria for a grasping task.

piled clutter Multiple objects lying on top of each other, for example, in a pile on a planar surface.

structured clutter Multiple objects which are spread out in a scene, such that they not neccesarily touch each other.

top-grasp A 4-DoF grasp where the grasp approach axis is aligned with gravity.

Acronyms

CNN Convolutional Neural Network

CR Clearance Rate

DoF Degrees of Freedom

EGAD Evolved Grasping Analysis Dataset

FCNN Fully Convolutional Neural Network

GP-net Grasp Proposal Network

GP-net+ Grasp Proposal Network Plus

GPD Grasp Pose Detection

GPU Graphics Processing Unit

GQ-CNN Grasp Quality Convolutional Neural Network

GSR Grasp Success Rate

IoU Intersection over Union

NLP Natural Language Processing

NMS Non-Maximum-Suppression

NN Neural Network

RL Reinforcement Learning

ROS Robot Operating System

TCP Tool Centre Point

TNR True Negative Rate

TPR True Positive Rate

TSDF Truncated Signed Distance Function

VAE Variational Autoencoder

VCR Visible Clearance Rate

VG-dset Versatile Grasp Dataset

VGN Volumetric Grasping Network

VGQ-CNN Versatile Grasp Quality Convolutional Neural Network

Nomenclature

Camera coordinates

θ	Camera azimuth (azimuthal) angle around table normal
d	Camera radial distance
φ	Camera inclination (elevation) angle
K	Intrinsic camera parameters

Domestic scene variables

$\mathcal{F}$	Set of furniture units
$lv(f)$	Volume of look-to-regions for furniture unit f
$wa(f)$	Area of object workspaces for furniture unit f

Geometric representations

(u, v)	Image coordinates
Ψ	Angle between grasp approach axis and camera principal ray
ω	Rotation of a grasp around its grasp axis (x-axis)
β	Angle between grasp approach axis and table normal
w	Absolute grasp width
κ	Maximum grasp distance to image centre

$\mathbf{p}_b^a$	Position of point b relative to the coordinate frame a
ν	Grasp width relative to maximum gripper width
$\mathbf{R}_b^a$	Rotation matrix of b relative to frame a
$\mathbf{T}_b^a$	Transformation matrix of b relative to frame a

Other symbols

$\mathcal{O}$	Set of object meshes
$\mathcal{S}(o)$	Set of stable poses for a given object
$\gamma_{\hat{x}}$	Threshold for applying NMS to predicted grasp metric $\hat{x}$
$\mathcal{G}(o)$	Set of ground-truth grasps for a given object
$\mathbb{1}_A(i)$	Indicator function, equals to 1 if $i \in A$

Quality metrics

Γ	Fitness function, an average of Grasp Success Rate and Clearance Rate
gc	Confidence of a grasp proposal
ϱ	Success of a grasp
$\hat{\varrho}$	Prediction of success for a grasp
ϵ_Q	Robust Ferrari-Canny metric
ϵ_{FC}	Robust Force Closure metric
δ	Threshold for robustness ϵ to define a successful grasp

CHAPTER 1

Introduction

Since the 1920s, science fiction has envisioned robot assistants capable of assisting humans in their daily lives [1]. These autonomous assistants navigate the world, achieving their goals by interacting with and manipulating objects in complex, unstructured environments. Researchers have pursued making such robots a reality since the invention of the early robots in the 1960s [2], achieving consistent progress in all areas required to create such platforms. However, despite the substantial progress made in specific contexts, e.g. robots assembling cars or waiting tables in restaurants, we have yet to arrive at general-purpose robotic platforms capable of manipulating any object in unstructured, real-world environments.

The difficulty in manipulating arbitrary objects in complex environments lies in the large variability and uncertainty of such tasks, where a successful grasp is dependent on properties such as an object's geometry, mass, surface properties, deformability, and pose. Furthermore, approaches must take account of an object's context, including the surrounding objects and other elements of the environment (e.g. support surfaces, furniture units, etc.). When using a mobile manipulator, a robot with both a mobile base and a robotic arm, the robot can grasp objects from different workspaces but does not have a fixed position with respect to those workspaces.

Use cases involving general-purpose mobile manipulators in real-world scenarios

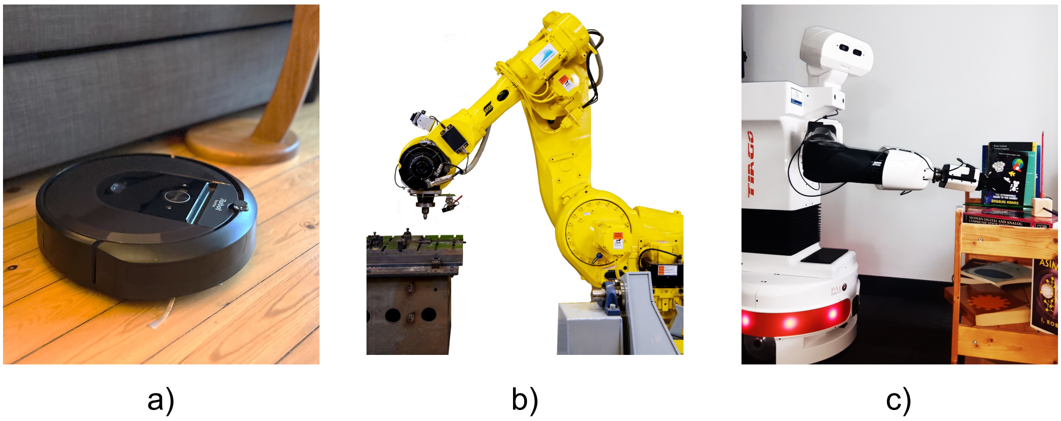


Figure 1.1: Example of a) a vacuum cleaning robot, b) an industrial robot and c) a PAL TIAGo mobile manipulator.

can, amongst others, be found in supermarkets, hospitals, care facilities or domestic environments. In domestic environments, robots can potentially assist people with limited mobility by, for example, fetching objects from a shelf, unloading groceries or picking up objects that have fallen to the ground. Introducing robots for these tasks could help those needing assistance to remain living independently and at home as long as possible. To perform such assistive tasks, robots must be able to both manipulate any object in complex, domestic environments and move between different workspaces in their surroundings.

Robots available for use in home environments today can often navigate in diverse and complex environments and thereby move between different workspaces. However, such robots usually do not have an arm or end-effector, preventing them from manipulating objects in their environment. Examples of such robots are vacuum cleaners (see Figure 1.1a) or autonomous lawnmowers. More recently, mobile robots have been deployed outside of home environments to serve customers and transport tablets between the kitchen and tables in restaurants. As such robots are usually not equipped with an arm and end-effector, they are unable to fill the role of general-purpose robotic manipulators mentioned above.

Manipulating objects, on the other hand, can already be achieved reliably in industrial settings with robotic arms subject to certain constraints. Such robots can be used to assemble structures, move products to or from a conveyor belt or pick objects from a bin. An example of an industrial robotic arm is shown in Figure 1.1 b). These robotic arms are normally fixed in place and cannot move in their surroundings. They are either programmed to repeat the same movements in a precise manner or use fixed sensors and computer vision systems to guide their movements. Furthermore, their surroundings and underlying programming are usually optimised to ensure reliable performance. They usually attempt to minimize collision hazards and include known object models, high-end tracking systems, pre-known workspace limits, and offline path planning.

General real-world scenarios, and in particular domestic environments, are not set up in such a controlled manner. They include a number of unknown objects, from simple to complex geometries and appearances, situated on a similarly diverse set of furniture units. They can include single objects or multiple objects in clutter placed in arbitrary relative poses within the environment. A mobile manipulator can move between different workspaces in the environment, for example, the kitchen counter and the bookshelf. This enables the robot to transport objects between those workspaces, for example, to store or fetch objects. For a robot to transport an object, it must first grasp the object successfully. In this context, grasping is defined as controlling all Degrees of Freedom (DoF) of an object with the end-effector of a robot [3]. In other words, the end-effector locks the object's pose in place such that the object does not move in relation to the end-effector.

The key focus of this thesis is grasping arbitrary and unknown objects in dynamic and complex environments, specifically in domestic settings. Figure 1.1 c) illustrates a mobile manipulator successfully grasping an object in such a domestic setting. In contrast to grasping with fixed manipulators in tabletop settings, mobile manipulators in domestic environments contend with a significantly expanded workspace beyond the robot's immediate surroundings. This expansion alters how scenes are perceived and influences what kind of grasp poses can be executed, increasing the overall complexity of the task. Given that recent research in robotic grasping has primarily focused on fixed robotic arms and tabletop scenarios, we describe the progress made in the following section.

1.1 The state of robotic grasping

Grasping objects with a robot usually involves three stages: (i) acquiring sensor readings to construct a scene representation, (ii) proposing grasps based on the scene representation, and (iii) path planning and execution of one grasp based on the grasp proposals.

Proposing grasps, that is, how to position the end-effector to grasp an object, is at the core of robotic grasping research. Finding grasp configurations that satisfy a specific set of requirements for a given object is often called grasp synthesis [4, 5]. While this process might appear simple to people who exhibit this kind of dexterity subconsciously every day, developing algorithms to grasp objects with robots has been an active and challenging field of research for several decades [4, 6, 7]. The combination of problems in perception, dexterity, uncertainty, control and adaptability in robotic grasping makes it a very complex problem, the solution to which often requires different assumptions and simplifications.

Up to the year 2000, grasp synthesis has predominantly been solved by analytical approaches [4], where assumptions about the physical properties of objects, e.g. friction, mass and form, are used to calculate analytical metrics to judge grasp quality. Since this kind of detailed information is usually not available from the sensor data alone in real-world scenarios, analytical approaches have mostly been restricted to simulated environments [4].

In recent years, the primary approach for creating grasp proposals has switched to data-driven algorithms, which use grasp experience to build a model that may be used to infer a prediction of success for grasp proposals [4, 5, 7, 8]. In contrast to analytical approaches, data-driven algorithms usually do not require low-level physical details about the scene during inference, making it possible to use them in applications that directly employ sensor-based information from the scene.

When setting up the environment for generating the grasp experience, usually in simulation, several design choices and constraints have typically been used to simplify the problem. These simplifications were required initially to find workable and robust solutions to enable robotic grasping for unknown objects. Some of these simplifications include the camera placement [9, 10, 11], the DoF of potential grasp proposals [9, 10, 11, 12, 13, 14, 15], the rigidity of the objects [9, 10, 11, 12, 13, 14, 16, 17, 18, 19, 20], the number of objects presented [9, 10, 11], a limited workspace area [16, 17, 21] and the general surroundings of the objects [9, 11, 12, 16, 18].

There has been progress toward removing some of these simplifications. For example, several recent solutions are able to propose grasp poses for objects in clutter [16, 18, 20, 22, 23, 24, 25], to propose 6-DoF grasp grasp poses [16, 18, 20, 24], and to grasp non-rigid objects [26, 27].

However, to date, removing other simplifications, especially in regard to the camera placement, limited workspace areas, and the general surroundings of the objects, has not been fully addressed by the research community. Instead, many research projects have focused on tabletop scenarios with fixed manipulators, where the scene does not vary substantially in regard to the above simplifications. A central argument of this thesis is that moving towards robust general-purpose robotic grasping in complex, domestic environments will require solutions that do not rely on such simplifications.

1.2 Thesis scope

This thesis aims to develop grasping algorithms tailored for grasping objects in domestic environments. Such grasping algorithms have to address challenges arising from the inherent complexities of domestic scenes. We focus on three specific aspects of this problem: flexible camera placements, flexible workspace areas and flexible object surroundings.

- Flexible camera placements: Enabling a mobile manipulator to grasp objects in domestic environments requires a flexible camera placement since the pose of the mobile manipulator, its camera, and the object workspace is initially unknown. This changes how the object is viewed by the camera and, consequently, how it is depicted in the scene representation.
- Flexible workspaces: Objects in domestic environments are placed in an unknown pose that is not constrained to single, fixed workspaces. For example,

objects could be placed on the top of shelves, tables, the floor or the lowest drawer in a cupboard. Since the object poses are initially unknown, a robotic grasping solution should be able to propose grasp poses regardless of where the object appears in the scene representation. By adopting a simple yet effective scene representation in the form of a depth image, we utilise the advantage of proposing grasps for any pixel in the image efficiently. This, coupled with a 6-DoF grasp grasp representation, enables the proposal of grasps for flexible camera placements and workspace areas.

- Diverse object surroundings: Objects in domestic scenes are placed on different furniture units, not necessarily on tables but also on shelves, drawers or cupboards. This results in different immediate surroundings of the objects, with different furniture units introducing occlusions and a higher risk of collisions when attempting to grasp objects. To successfully grasp objects in such scenarios, corresponding training data has to be used for data-driven approaches.

The complexity of domestic environments necessitates a departure from conventional tabletop grasping approaches with a known or implicit camera-workspace transform.

1.3 Thesis contributions

The main contribution of this thesis is generating approaches for grasping unknown objects in domestic settings. This comprises the following individual contributions:

1. Versatile Grasp Quality Convolutional Neural Network (VGQ-CNN): A Convolutional Neural Network (CNN) model capable of predicting the quality of grasps under a wide range of camera poses. In contrast to other depth-image-based grasp quality prediction models, VGQ-CNN can predict the quality of 6-DoF grasp poses. Our results show that VGQ-CNN can be used with 6-DoF grasps and generalise to varying camera viewpoints while performing competitively to its baseline for top-grasps and overhead camera viewpoints.

2. Versatile Grasp Dataset (VG-dset): A versatile 6-DoF grasp quality dataset including 7.2 million grasps over an extended range of camera poses. The dataset can be used to train VGQ-CNN and other network architectures or be sub-sampled to modify the dataset distributions.

3. Grasp Proposal Network (GP-net): A Fully Convolutional Neural Network (FCNN) model to propose 6-DoF grasps for single objects with a parallel-jaw gripper for unknown camera-workspace transforms. In contrast to other grasp proposal models, it can operate over flexible camera viewpoints without having to define workspaces or run table segmentation to filter grasp poses. We show that GP-net performs better than widely used grasp proposal models in real-world experiments with a PAL TIAGo mobile manipulator.

4. A benchmark dataset and a simulation environment for robotic grasping in domestic environments. The dataset comprises depth images of domestic scenes with ground-truth grasp information. Objects in the domestic scenes are placed in clutter on all available workspaces of furniture units like shelves, tables and sideboards. The simulation environment can be used for training and evaluating grasping algorithms in domestic environments. It can be used with different scene representations and parallel-jaw grippers to enhance comparability and is publicly available.

5. Grasp Proposal Network Plus (GP-net+): A FCNN model to predict 6-DoF grasps for cluttered objects positioned on a diverse set of furniture units with a parallel-jaw gripper. The objects can be positioned on tables or shelves with an unknown camera-workspace transform. GP-net+ shows superior performance to three grasp proposal models in a set of simulated experiments. Furthermore, the results for a real-world experiment demonstrate how GP-net+ can be used in a real-world application and obtain good results with a PAL TIAGo mobile manipulator grasping objects placed in clutter on tables and shelves.

6. A Robot Operating System (ROS) package for deploying both our grasp proposal models, GP-net and GP-net+, on real robotic hardware. The package provides examples of how to combine our grasp proposal models with path

planning and collision-detection software. Further, our implementation for GP-net+ can be used for target-driven object grasping.

1.4 Thesis outline

The structure of the remainder of this thesis is as follows. We start by providing a review of related research in robotic grasping algorithms, detailing methods that have been proposed over the years and how they have been applied to robots in Chapter 2.

In Chapter 3, we develop a depth-image-based, 6-DoF grasp quality prediction model called VGQ-CNN. To do so, we extend a grasp quality prediction model, GQ-CNN [11], so that it can be used from versatile camera viewpoints (in contrast to fixed, overhead cameras) to predict the quality of 6-DoF rather than 4-DoF grasps. We evaluate VGQ-CNN on an object-wise test split focusing on the performance under variation of the camera viewpoint in relation to the object.

In the subsequent Chapter 4, we utilise our findings from VGQ-CNN to propose GP-net, a depth-image-based, fully-convolutional grasp proposal model that can be used from flexible camera viewpoints to propose 6-DoF grasps on single objects. In contrast to VGQ-CNN, GP-net represents a generative method that can be used in a one-shot manner for each pixel in an image to propose grasps with advantages in run-time and sampling efficiency. We demonstrate the performance of GP-net in simulation and real-world experiments using a PAL TIAGo mobile manipulator. To compare GP-net's performance with other algorithms, we repeat our real-world experiments with two widely used grasp proposal models and compare their performance with GP-net.

In Chapter 5, we present a simulation environment for grasping objects in clutter on a variety of furniture units in domestic environments. The core part of developing the simulation environment is creating reproducible steps for generating domestic scenes with objects in clutter placed on various furniture units like coffee tables, sideboards, shelves or drawers. The simulation environment can generate training data and can be used to evaluate different models for grasping unknown objects in such domestic scenes. We use the training data to train GP-net+, an improved version of GP-net, for proposing 6-DoF grasps for objects in clutter in domestic environments.

We demonstrate the performance of GP-net+ in simulation and real-world experiments with different stages of complexity. For the simulation experiments, we compare GP-net+ with two widely used grasp proposal models to set its performance in context. Furthermore, we show how GP-net+ can be coupled with a pre-trained object detection model to enable target-driven object grasping. Finally, in Chapter 6, we conclude the thesis and present several opportunities for future research.

CHAPTER 2

Literature Review

When grasping an unknown object with a robot, the process usually starts with the robot's end-effector in an unknown position in relation to the object. The end-effector is then moved in a non-linear path to a position linearly in front of a *grasp pose*. From there, the end-effector moves linearly toward the grasp pose along the approach axis of the grasp.

Once the end-effector has arrived at the grasp pose, the gripper is closed, for example, by moving the two plates of a parallel-jaw gripper linearly towards each other along the grasp axis until contact between the gripper and the object is established. This contact should be such that the contact wrenches, generated by the torques and forces at the gripper contacts, enable the gripper to control all DoF of the object. If that is the case, the object remains fixed in the gripper and can be lifted. Lifting the object completes the grasping process. An example of this process with a PAL TIAGo mobile manipulator grasping a mango is shown in Figure 2.1.

The area of robotic manipulation of objects goes beyond grasping objects, for example, with possibilities to use a manipulator for push-grasping objects [28], placing objects [29, 30] or interacting with humans during a hand-over [31, 32]. Eventually, such skills must be incorporated with a grasping pipeline to enable a useful deployment of robots in real-world scenarios. However, grasping objects is

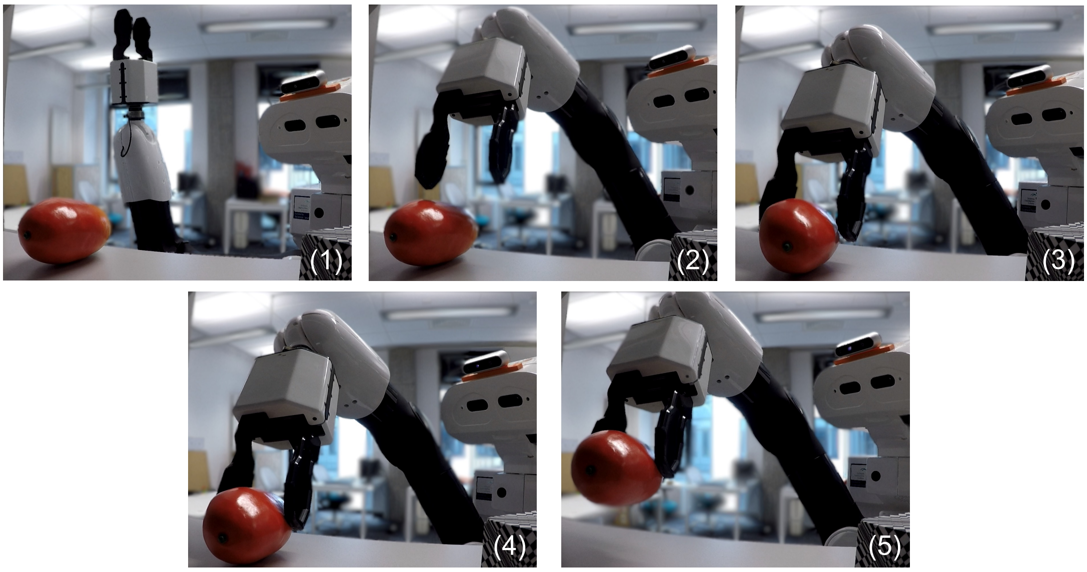


Figure 2.1: Grasping process with a robot as observed externally. The robot's end-effector is initially positioned in an unknown position to the target object (1). It then moves in a non-linear path to a position linearly in front of the grasp pose (2). Subsequently, it approaches the grasp pose linearly along the approach axis (3) and closes the gripper (4). Finally, the end-effector is lifted vertically upwards to lift the object from the surface (5).

a core part of many manipulation tasks with a robot. This makes robotic grasping an important skill for manipulating robots and the focus of this thesis. We briefly mention manipulation solutions exceeding robotic grasping throughout this review, but overall, we focus on robotic grasping with a parallel-jaw gripper.

Furthermore, research projects in recent years have used affordances [33, 34, 35], which describe the different actions or tasks that can be achieved with an object. Affordances can be used to choose the *right* grasp poses for those tasks. In contrast, most robotic grasping research in the last decades has focused on finding *any* suitable grasp for successfully lifting the object [5]. While affordances can still be incorporated into a developed training pipeline, we focus on proposing any suitable grasp in this work. In the following chapter, we present an overview of robotic grasping, including how grasp poses can be represented, proposed and executed, how methods have progressed over the years, how solutions are typically evaluated and how grasps have been proposed for mobile manipulators.

The remainder of this chapter is organised as follows: We start with an overview of robotic grasping and different variations in grasping pipelines in Section 2.1. Subsequently, we describe how grasping algorithms are typically evaluated and how different choices in setting up physical and simulated experiments can prevent direct comparison of algorithms in Section 2.2. Next, we briefly introduce the early research of robotic grasping and present several analytical grasp performance metrics in Section 2.3.

Grasping algorithms are often distinguished by the numbers of DoF their proposed grasps can vary. Exemplary methods for generating 4-DoF and 6-DoF grasp proposals are described and compared in Section 2.4 and Section 2.5, respectively. Focusing on grasping in domestic environments, we describe how grasps have been proposed for mobile manipulators in Section 2.7. We conclude this chapter in Section 2.8 with a summary of the field and describe how the remainder of this thesis will sequentially move grasping towards domestic scenarios with mobile manipulators.

2.1 An overview of robotic grasping

For grasping objects with a robot, the specific steps involved and how they are implemented depend heavily on the target application and the robots' surroundings. Figure 2.2 presents the key steps of an exemplary grasping pipeline, including several potential variations of individual steps. At the start of a grasping pipeline, information about the environment and surroundings of the robot needs to be gathered and processed to produce a scene representation.

The **scene representation** provides a model of the robot's workspace and can be used to propose grasp poses. In some instances, information about the scene is already available, for example, in industrial scenarios where a fixed robotic manipulator grasps objects with a known shape in a fixed, pre-determined pose. In such settings, neither sensing of the workspace to compute a scene representation nor grasp planning is required, and instead, the robot can execute a pre-computed path. Robotic grasping can also be achieved in scenarios with more variability and less initial knowledge, for example, with unknown objects in unknown poses

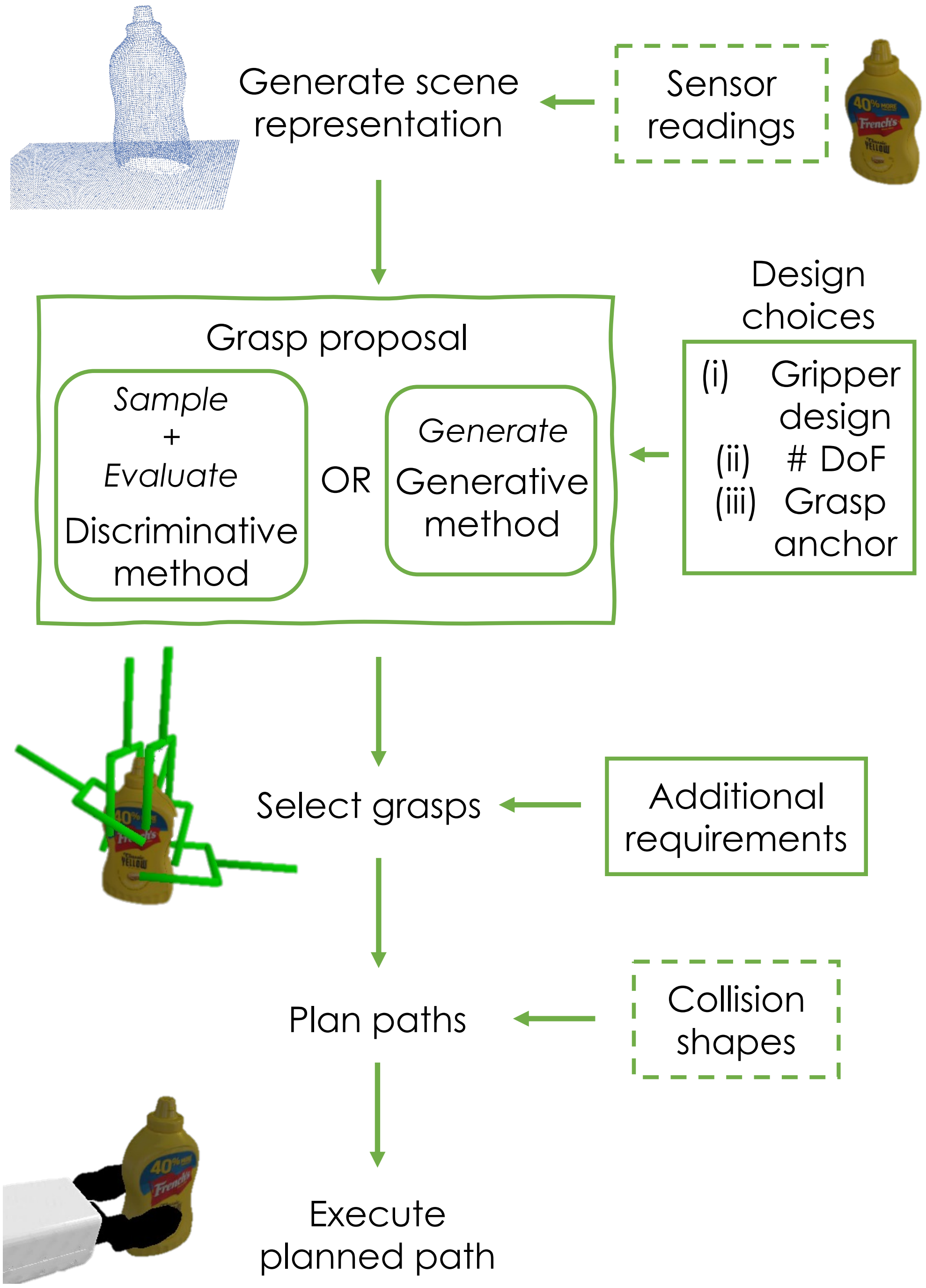


Figure 2.2: Exemplary pipeline for grasping objects with a robot. It acquires sensor readings and constructs a scene representation, proposes and selects grasps, plans paths, and executes the planned paths to grasp the object. Choices by the grasping system developer are indicated with solid rectangles, while information input about the environment is indicated with dashed rectangles.

in any tabletop or domestic scenario. In such cases, sensors are used to gather information about the surroundings and construct a scene representation to be used as a basis for grasp proposal.

The most widely used sensors to gather information about the robots' surroundings for robotic grasping are active sensors such as RGB-D cameras [11, 12, 16, 18, 20, 24, 36, 37, 38]. Active sensors can use a variety of representations to model the 3D structure of the world, for example point clouds [20, 38, 39, 40], depth images [11, 12, 23, 41, 42, 43], RGB-D images [23, 25, 44] and voxel grids [15, 16, 45]. Combining sensor readings from multiple viewpoints can be used to reduce uncertainty and increase the quality of the scene representations such as point clouds [46] or voxel grids [16, 17] if the relative camera poses are known [5]. All of these representations have been used in grasping pipelines, often exhibiting different strengths and weaknesses, with no one representation proving optimal to date [5].

While RGB-D cameras were the most widely used sensors up to 2022 [5], alternative input modalities like tactile [47, 48, 49], force-torque [50, 51] or acoustic [52] sensors can also be used in the grasping process. For example, tactile sensors have been used to detect object slippage [47], reconstruct object geometry [48] or reduce uncertainty in object pose estimates [49]. Researchers have also investigated approaches that combine different sensor modalities using sensor fusion, for example, combining visual and tactile sensor data for reconstructing object surfaces [53].

Once sensor readings have been processed, the resulting scene representation can be used to propose potential grasp poses. Several **design choices** in the generation of grasp poses depend on the robotic hardware used and the requirements the final pipeline should meet. These choices include (i) the gripper design, (ii) the number of DoF the grasp poses can vary in, and (iii) the grasp anchor.

The *gripper design* describes the end-effector used on the robot. It can range from complex designs with human-like hands [54, 55] to soft end-effectors [56], magnetic grippers [57, 58], pneumatic suction-cup designs [59, 60] and parallel-jaw grippers [11, 16]. An example of different gripper designs is shown in Figure 2.3. Each gripper design comes with different advantages and disadvantages. For exam-

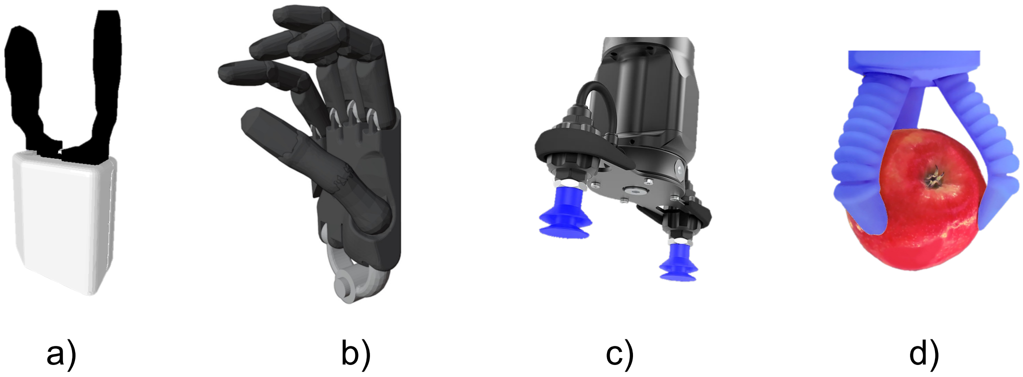


Figure 2.3: Different gripper designs with a) a parallel jaw gripper, b) a human-like hand, c) a pneumatic suction cup and d) a soft end-effector.

ple, soft robotic grippers can adapt seamlessly to various object shapes but often struggle with robustness, speed and control [56]. Human-like robotic hands have major advantages in terms of dexterous manipulation but currently still struggle with reliability, complexity and cost [55]. In this work, we focus on parallel-jaw grippers, which consist of two parallel gripper plates with a single DoF. The advantages of parallel-jaw grippers lie in their long lifetime and low complexity [61], making them the most commonly used grippers in robotics grasping research up to 2022 [5].

A full 6-DoF grasp pose $g^b \in (\mathbf{t}^b, \mathbf{R}^b, w)$ for a parallel-jaw gripper can be defined with a 3D translation $\mathbf{t}^b \in (x, y, z)$, a 3D rotation $\mathbf{R}^b$ and the distance between the gripper plates when executing the grasp on the object, i.e. the width w of the grasp. This grasp pose is always given in relation to a coordinate base frame, for example, the robot base b, which we denote with g^b.

In order to reduce the complexity of the grasp pose representation, researchers often reduce the grasp poses' *number of DoF* from 6-DoF grasp, a 3D translation and 3D orientation, to 4-DoF grasp with a 3D translation and a 1D orientation [5, 8]. The approach vector of 4-DoF grasps is usually aligned with an arbitrary vector, where the one orientational DoF rotates around that arbitrary vector. Especially in tabletop scenarios, this approach vector is often aligned with gravity, resulting in a top-grasp [9, 10, 11, 13, 23].

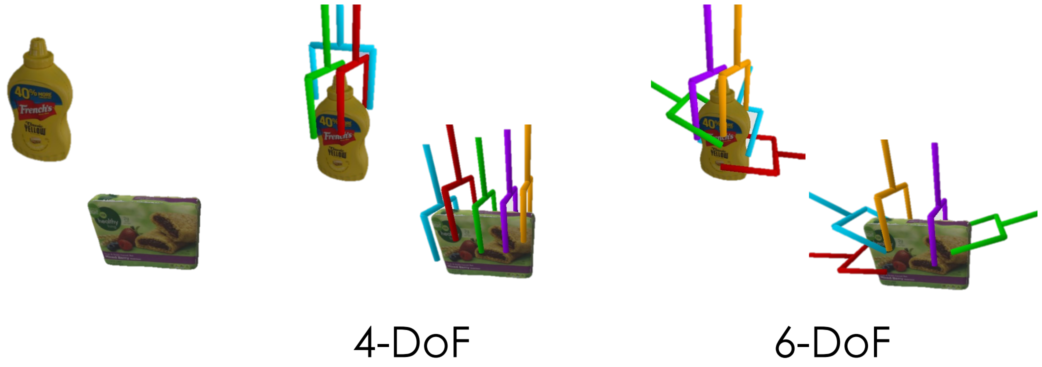


Figure 2.4: Example of 4-DoF and 6-DoF grasp poses on a bottle and a food packaging box.

If the approach vector of a 4-DoF grasp is aligned with the camera principal ray, the grasp x-axis between the gripper plates is perpendicular to that camera principal ray, causing the grasp x-axis to be parallel to the image plane. In such cases, the grasps can be presented as rectangles in the image plane [9, 10, 13, 14, 23, 62], which often specify a 2D position, 1D rotation and width of the grasps. The depth of the grasp in relation to the camera origin is then often inferred with a set delta to the object surface [10, 23] or given as an explicit variable [11]. Examples of 4-DoF and 6-DoF grasp poses outlining their differences in variability are shown in Figure 2.4.

Analytical methods usually analyse grasps based on each contact point between gripper and object [6, 28]. When working with data-driven approaches based on grasp experience, grasps are commonly described with a 6-DoF grasp pose relative to a single *grasp anchor* point in the gripper reference frame. Parallel-jaw grippers have a so-called Tool Centre Point (TCP), which lies in the centre between the two gripper plates. This TCP seems to be the most commonly used grasp-anchor, with usage in [9, 10, 11, 12, 13, 14, 15, 16, 21, 23, 37, 38, 45, 63, 64].

Alternatively, the grasp pose can be anchored at the contact point of one of the gripper plates, as proposed by Contact-GraspNet [20]. By anchoring the grasp pose to the visual grasp contact points, the learning process and pose accuracy can be improved [17, 20]. The original authors argue this is due to the reduced dimensionality and immediate connection to the observed geometry in the scene representation [20]. There is an open question about how a TCP and a visible grasp contact compare as grasp anchors and their quantitative differences on the performance of grasp proposal algorithms.

Once all design choices have been considered, grasps can be proposed. This process can be divided into discriminative and generative methods [7]. Discriminative methods propose grasps in two stages by sampling grasp candidates and subsequently evaluating the quality of each grasp individually, e.g. [10, 11, 13, 14, 15, 17, 19, 38, 40, 64]. The sampling of grasp candidates in discriminative methods is usually based on random distributions, often taking into account the object geometry [14, 17, 24, 38, 65].

In contrast to discriminative methods, generative methods do not evaluate the quality of sampled poses *individually* but directly propose grasp poses [9, 12, 16, 20, 23, 62, 63, 66, 67]. They can propose single [9, 66] or several grasp poses [9, 68] for a given scene representation. When proposing more than a single grasp, the proposals are usually coupled with a quality measure for grasp selection [12, 16, 20, 23]. Many generative methods propose dense outputs with grasps for every point in the represented scene, for example, every pixel for a depth image [12, 23, 44, 63, 67] or every voxel in a voxel grid [16, 17, 69].

An overview of the difference between discriminative and generative methods within grasp proposal (often also called grasp synthesis) is shown in Figure 2.5. We note alternative names exist for these methods, likely since the division was only introduced recently with some of the first generative methods described in 2015 [9]. Discriminative methods are also called sampling [5] or hypothesize-and-test [8] methods, while generative methods are also called regression [5] methods in different reviews. Overall, the terminology for these different approaches has not been settled as of 2023.

After grasps have been proposed, appropriate grasps must be selected, paths planned and finally **executed by the robot**. The selection of grasps is often based on the predicted quality or success of grasp proposals. However, additional requirements might be taken into account for a specific setup. For example, grasp

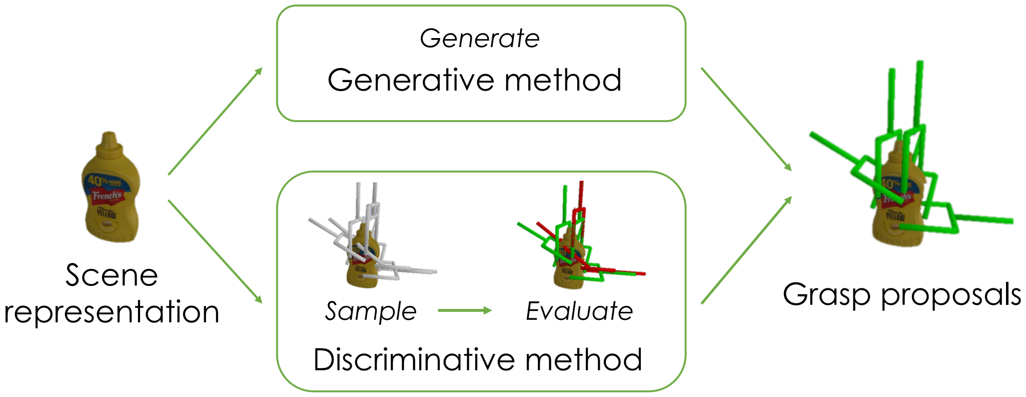


Figure 2.5: Example of discriminative and generative grasp proposal. While discriminative methods first sample grasp candidates and then evaluate them individually, generative methods directly regress to good grasp proposals.

proposals can be filtered for grasping specific objects from a cluttered scene by excluding all grasp proposals for other objects [20]. Such target-driven object grasping can also be achieved by masking or indexing the scene representation before grasps are proposed, as demonstrated in [38, 70]. While both methods lead to several grasp proposals on the targeted objects, filtering grasp proposals ensures geometric cues of the immediate object surroundings are used for the grasp proposals, reducing the risk for collisions [20]. After grasps have been selected, paths for reaching the grasp poses with the end-effector must be planned.

In most cases, path planning for a grasp requires collision checking to ensure safe execution with a real robot. For known environments, collision objects can be added manually to the scene, for example, in a scenario with a table-mounted robotic arm where the transformation $\mathbf{T}_t^b$ from the robot base b to the table collision shape t is known. In scenarios where such transformations and collision shapes are not known in advance or are not fixed, solutions like OctoMap [71] for mapping sensor readings to an occupancy grid can be used and integrated into the planning scene of a path planner, for example, using MoveIt! [72] with the ROS.

The proposed grasps represent the pose of the anchored gripper-point when the gripper is closed for grasping the object. When executing the grasp, it is common to plan a joint path (where the end-effector can move in any way) to a point at a short distance in the negative grasp z-direction from the final grasp pose.

The subsequent grasp approach then involves following along a linear path in the positive grasp z-direction until the final grasp pose is reached with the end-effector. Once the gripper is closed, the object is usually lifted by moving the end-effector for a certain distance upwards [21, 22, 38, 73, 74]. This completes the grasping pipeline. In the following section, we discuss how the performance of grasp proposal algorithms can be evaluated and benchmarked.

2.2 Performance of grasping algorithms

Judging the performance of grasping algorithms and benchmarking them against each other can prove difficult due to the variety of problems studied in the field. Differences in hardware, object sets, and setups often prevent direct comparison of performance metrics between studies. This difficulty in comparing several approaches is one of the reasons why there are no clear state-of-the-art robotic grasping algorithms to date. To set the scene for how grasping algorithms are evaluated and what is needed to compare them, we describe the process and potential differences in evaluation methods in the following sections.

2.2.1 Evaluation and performance metrics

Depending on the application and design choices of grasping algorithms, they can be evaluated in different ways. For discriminative methods, one can evaluate how good a quality prediction algorithm is at evaluating the quality of grasp candidates, presented as (i) in Figure 2.6. For both discriminative and generative methods, one can evaluate the end-product of the approach by checking the quality of the final grasp proposals, presented as (ii) in Figure 2.6.

Evaluating the performance of grasp quality predictions is usually done with a dataset, where a test set with grasp candidates and their corresponding ground-truth qualities is available. This evaluation method typically yields some kind of accuracy, describing the ratio of the ground-truth grasps properly classified by the algorithm. Examples of such evaluation techniques are presented in [11, 75] and our VGQ-CNN in Chapter 3.

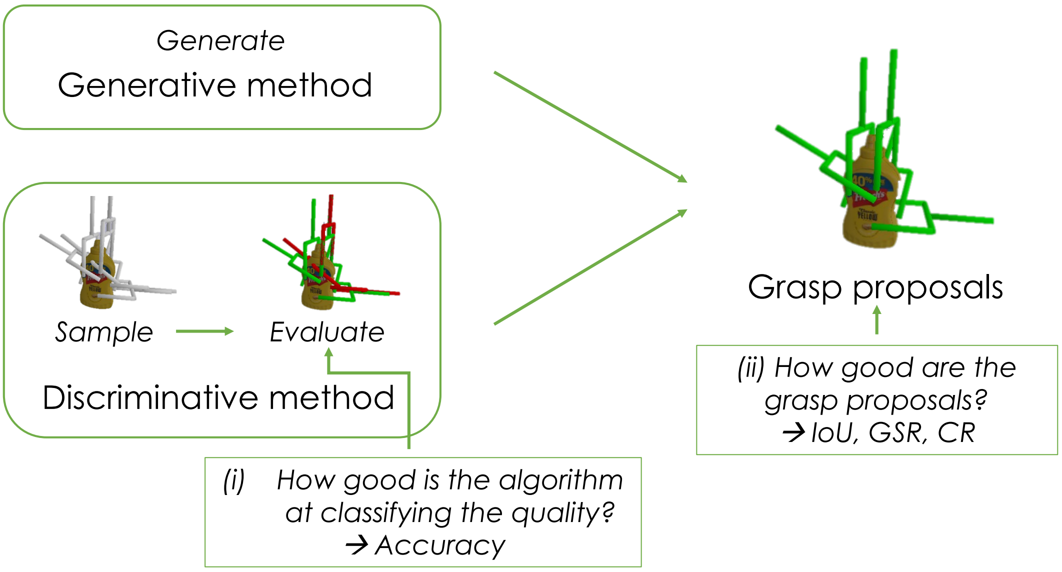


Figure 2.6: Example of possible performance evaluations in grasping pipelines.

Evaluating how good grasp proposals are can either be done *offline* by running inference on a dataset or *online* by interactively testing grasp proposals in simulation or real-world experiments.

When evaluating grasp proposals *offline* on a dataset, they are compared to ground-truth information in the dataset. This type of evaluation can only be used effectively if the ground-truth information covers all possible grasps that the algorithm can propose. Only 4-DoF grasp proposal algorithms seem to have been evaluated directly on datasets [9, 23, 62, 67], likely because it is computationally expensive to pre-compute ground-truth information for all possible 6-DoF grasps in the evaluation set.

For 4-DoF grasp proposal datasets [76, 77], the grasp approach axis is usually aligned with the camera principal ray and the grasps are presented as oriented rectangles in RGB or RGB-D images. Algorithms trained on such datasets are evaluated by how well their predicted grasp proposal rectangles match the ground-truth rectangles. There are dedicated metrics for how well the rectangles match the ground truth, for example, using the Intersection over Union (IoU) [12, 62, 77] or specifically introduced rectangle [9, 23, 69] and point metrics [69, 78].

When evaluating grasp proposals *online* using simulation and real-world experiments, grasp proposals are interactively executed in the scene and the grasp outcome is observed. Discriminative methods and, specifically, grasp quality prediction algorithms need to be coupled with a grasp sampling algorithm to be tested in simulation or real-world experiments [10, 11, 22]. The most commonly used metric in both simulated and real-world experiments is the grasp success:

$$Grasp\ Success = \frac{\#\ Successful\ Grasps}{\#\ Attempted\ Grasps} \tag{2.1}$$

Grasp success can be expressed as a rate or a percentage. The definition of what makes a grasp "successful" can vary. Commonly used approaches include the object being lifted a certain distance above the surface [45, 70], staying in the gripper for a certain amount of time [79], or staying in the gripper under the influence of perturbations [11, 18, 20, 38].

A high grasp success does not necessarily correspond to a "good" grasp proposal algorithm since it does not include a measure of how many of the graspable objects in the scene can successfully be grasped. For example, the grasp success is high if a grasp proposal algorithm is conservative and proposes successful grasps but only for a few graspable objects. This is similar to the concept of precision, a common evaluation metric used in machine learning for binary decision problems [80].

A high precision can be achieved with a conservative prediction algorithm only classifying very few actual positive data points as such, thereby reducing the risk of false positives. Since this is only one part of the performance, there is a second metric called recall. Recall would be low in the case described above since many actual positive examples are missed and predicted as negative, resulting in a high number of false negatives. There is a similar concept to recall to identify such cases in grasping with the so-called clearance or completion of an algorithm [5]. It measures the ability of an algorithm to propose (successful) grasps for any object in the scene:

$$Clearance = \frac{\#\ Objects\ Cleared\ fromScene}{\#\ Objects\ in\ Scene} \tag{2.2}$$

Clearance can also be expressed as a rate or a percentage. There can be a single object [11, 15, 34, 38, 42] or multiple objects [16, 18, 20, 22, 35, 36, 39, 65, 81, 82] in the scene to be cleared. Scenes that include multiple objects are often described as cluttered, where one can differentiate between piled clutter and structured clutter [5]. Piled clutter includes objects which are dropped on top of each other, often found in bin-picking scenarios [16, 18, 22, 35, 36, 65]. Structured clutter, on the other hand, consists of objects that are spread out in a scene and do not necessarily touch each other [20, 39, 81, 82].

Apart from its success in grasping or clearing objects, an algorithm's run-time can be crucial in real-world applications. Timing is often reported using the mean absolute computation time it takes to propose grasps [12, 15, 16, 18, 20], or the Mean Picks Per Hour (MPPH) [60, 63] which describes the average number of successful grasps executed by a robot in an hour. Note that all timing metrics also depend on the hardware being used, which needs to be reported by researchers in order to generate reproducible results or compare the run-time of different algorithms.

2.2.2 Benchmarking algorithms

With many projects evaluating grasp success and clearance on simulated and real-world experiments, it seems natural to compare their metrics and choose "the best-performing method" as the state-of-the-art. Unfortunately, lack of consistency in the experiment design in terms of (i) object sets, (ii) scene setup, (iii) perception systems, and (iv) robotic hardware substantially influence the performance metrics such that the results are often not comparable to each other. In the following, we explain why these choices influence the experiments and what efforts are being undertaken to make results more comparable.

The difficulty of a grasping experiment is influenced by the *object sets* used [5]. Object sets can differ in many aspects, for example, in object geometry, including shape and size, the mass and density of objects and their surface material, including friction, rigidity and reflectivity. All of these aspects can influence the outcome of grasp experiments. There are some projects specifically tailored to certain properties of objects, e.g. grasping deformable [26, 27, 83] or transparent objects [84]. However, most of the mentioned object properties have not yet been considered in robotic grasping research. One aspect that is commonly described for object sets is their geometric shape.

The shapes of objects can vary significantly, ranging from simple shapes like cylinders and cubes [16, 44] to household objects [16, 76, 85, 86, 87] and irregular or random shapes [74, 88]. The choice of object sets determines not only the difficulty of experiments but also their reproducibility. Some methods are tested on a set of household objects [10, 14, 16, 23]. Household object sets can provide a useful performance measure for scenarios encountered in domestic environments but make experiments difficult to reproduce unless the object set has been carefully specified.

Various object sets have been introduced to make experiments more reproducible. For example, the YCB object set [85] comes with 3D object models for simulation experiments and the option to purchase the object set for real-world experiments. It has been used with the 3D object models in simulation [16, 40, 45] and is also used for real-world experiments with the real objects, usually featuring a subset of the full object set [18, 25, 40, 45, 81]. Example objects from the YCB object set are depicted in Figure 2.7 a). While the type of objects in household-based object sets like YCB [85], BigBIRD [86] or KIT [87] capture many common items in household scenarios, they often lack variety in aspects like object geometry, grasp difficulty or object deformity [5]. Only if challenging and balanced object sets are used for training and evaluating grasping algorithms can they be ensured to be robust to such.

To test grasping algorithms on objects of varying complexity and difficulty, the Evolved Grasping Analysis Dataset (EGAD) [74] has been introduced. The test set of the dataset consists of 49 objects with 7 different levels of shape complexity and grasp difficulty. Here, shape complexity refers to a measure of morphological complexity [89, 90] and grasp difficulty to the 75^{th} percentile of the robust Ferrari-

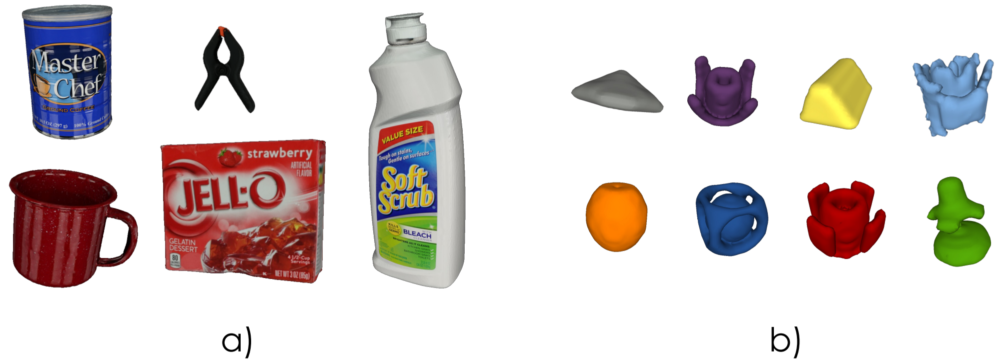


Figure 2.7: Example of objects in a) the YCB object set and b) the EGAD evaluation set.

Canny metric described in Section 2.3 [91]. Examples of some of the test objects in EGAD are depicted in Figure 2.7 b).

EGAD has been used in various research projects to evaluate the performance of grasping algorithms [81, 92, 93, 94]. An advantage of EGAD is the standardised approach for generating 3D printed object models, providing consistency and adjustability for different gripper designs. While providing a good basis for evaluating different object shapes, the EGAD objects do not hold any semantic meaning [5]. Therefore, EGAD experiments often accompany additional experiments using a set of household objects.

Additionally to which objects are used, how these objects are set up in the *scene* affects the difficulty of the experiments. As mentioned in Section 2.2.1, objects can be placed individually (with no other graspable objects in the scene) or with multiple objects in clutter. Some algorithms are tested on different setups, for example, with piled clutter and structured clutter, which show differences in performance of up to 20% in their Grasp Success Rate (GSR) and Clearance Rate (CR) [16, 21]. More complex setups, including non-planar support surfaces or complex furniture units like shelves, are likely to reduce the performance of grasping algorithms further since they can reduce direct visibility and increase the chances of collisions.

Besides the choice of objects used, the *sensors* for gathering the scene representation influence the experiments significantly. The scene representation for robotic

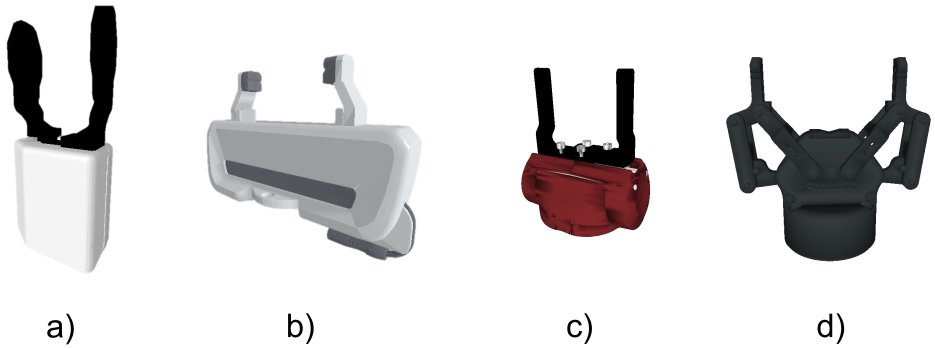


Figure 2.8: Example of parallel-jaw grippers with a) a PAL [98] b) a Panda [99], c) a Baxter [100] and d) a Robotiq 2F-85 gripper [101].

grasping algorithms usually comprises sensor readings from a single or several depth sensors. Depth sensors suffer from depth sensor noise [95, 96], minimum and maximum measurable depth ranges, and depth resolution, which varies between different sensors. Furthermore, the transformation between the sensors and the robotic arm needs to be established with hand-eye calibration to map grasp poses from the sensor readings to the robot base and execute the grasps. Inaccuracies in the transformation between the sensor and the arm cause misalignments in the planned and achieved grasp pose, possibly influencing the grasping outcome.

Finally, the *robotic hardware* can influence the outcome of real-world experiments. Setups can differ both in the robotic manipulator and end-effector used. Having a robotic manipulator with more DoF will increase the reachability of the robot while increasing the difficulty of planning paths using the inverse kinematics [97].

Even within the category of parallel-jaw grippers, available solutions differ in the size of their gripper plates, maximum gripper width, maximum grasping force, collision shape and the friction of their gripper plates. Examples of different designs are shown in Figure 2.8 with the Franka Emika Panda gripper [99], Robotiq 2 finger gripper [101], Baxter gripper [100] and the PAL gripper [98]. These differences can affect performance and lead to a different outcome for the same grasp proposal. The most popular choices for robotic manipulators as of 2022 according to [5] are the Franka Emika Panda arm [102] and the Franka Emika Panda gripper [99].

In general, the grasp success of different algorithms can only be directly compared if tested on the same object set in the same setup using the same perception system and robotic hardware. While we mention the advantages and disadvantages of different methods and algorithms in the following sections, we refrain from doing so in terms of absolute grasp success due to those issues. This is similar to comparative reviews and surveys in the field [4, 5, 7, 8].

2.3 Analytical approaches for grasp proposal

The first solutions for grasp proposal used discriminative, analytical approaches, where knowledge about the physical properties of objects is used to calculate analytical metrics and thereby judge grasp quality [4]. These analytical metrics are a way to judge the gripper-object behaviour of a grasp under the influence of external forces. Ultimately, they try to estimate if a gripper can control an object's DoF sufficiently when grasping the object at that specific grasp pose.

The analytical metrics are usually calculated using the forces acting on the body. A general force acting on a rigid body consists of a linear and an angular component, a force $\mathbf{F} \in \mathbf{R}^3$ and a moment $\boldsymbol{\tau} \in \mathbf{R}^3$, respectively. When represented in a vector, these two components are called a wrench $\boldsymbol{\varsigma}$ [28, 103]

$$\boldsymbol{\varsigma} = \begin{pmatrix} \mathbf{F} \\ \boldsymbol{\tau} \end{pmatrix} \tag{2.3}$$

Probably the most useful and widely-known metrics for the quality of grasps are form closure and force closure, which have been used since 1876 to distinguish contacts in machine design [28].

Form closure describes grasps where it is impossible to move the object relative to the gripper, given that the gripper links are locked and fixed in space. Force closure, on the other hand, is defined as a grasp that can resist any external wrench $\boldsymbol{\varsigma}$ applied to an object, assuming an arbitrary amount of force can be applied at the grasp's contacts [28]. In other words, a grasp in force closure "squeezes" an object to keep it in place, while a form closure grasp prevents movement solely

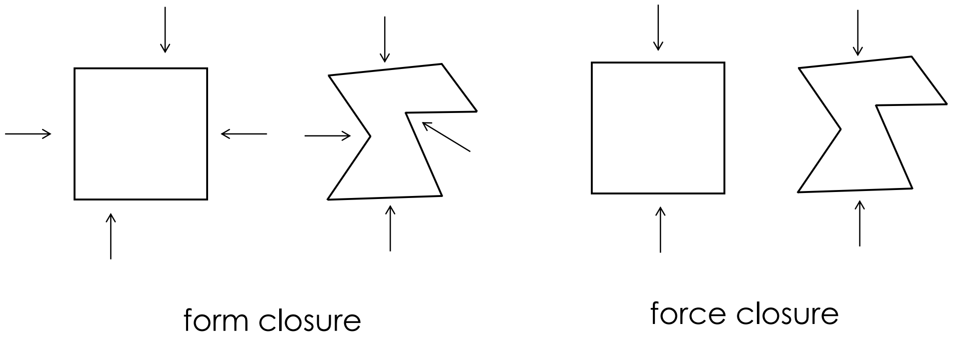


Figure 2.9: Example of form closure and force closure grasps in 2D, based on a visualisation in [104, 105].

through the positioning of the locked gripper fingers [28, 105]. In consequence, any form closure grasps also exhibit force closure.

Even though there exist exact, non-linear force closure tests, e.g. [106], approximate tests, e.g. [104], are often used to calculate force closure in an efficient manner [28, 88]. Examples of several grasps with form closure and force closure in 2D are depicted in Figure 2.9.

In an attempt to extend analytical metrics beyond binary categories to further distinguish successful grasps between optimal and less-optimal grasps, quantitative closure tests have been introduced [105]. One of them, the Ferrari-Canny metric [107], sometimes also called the epsilon quality [108], is defined using the grasp wrench space and presented as a continuous metric, where a higher value corresponds to a more robust grasp. It uses a local grasp quality measure $LQ(\boldsymbol{\varsigma})$, which describes how efficiently a wrench $\boldsymbol{\varsigma}$ can be resisted at a given contact. The Ferrari-Canny metric Q is subsequently calculated by:

$$\begin{aligned} LQ(\boldsymbol{\varsigma}) &= \max_{\mathbf{gf} \in \boldsymbol{\varsigma} A} \frac{\parallel \boldsymbol{\varsigma} \parallel}{\parallel \mathbf{gf} \parallel} \\ Q &= \min_{\boldsymbol{\varsigma}} LQ(\boldsymbol{\varsigma}) \end{aligned} \tag{2.4}$$

Here, $\mathbf{gf}$ is the generalised force vector, a vector combining all finger forces $\mathbf{f}_i^{\perp}$ applied at a given contact, A is the set of all couples $(\boldsymbol{\varsigma}, \mathbf{gf})$ where the generalised force vector $\mathbf{gf}$ can resist wrench $\boldsymbol{\varsigma}$, and subsequently $\boldsymbol{\varsigma}A$ is the set of generalised forces $\mathbf{gf}$ that can resist wrench $\boldsymbol{\varsigma}$ [107]. In other words, the Ferrari-Canny metric describes the "largest perturbation wrench $\boldsymbol{\varsigma}$ that the grasp can resist in any direction" [109] to control all DoF of an object with a gripper [107].

Analytical metrics like form closure, force closure or the Ferrari-Canny metric can be computed if physical properties like the shape and friction of an object are known. Given knowledge or assumptions about these properties, analytical metrics provide mathematical guarantees for the grasp parameters like grasp pose and initial finger configuration [4]. In contrast to simulated grasp success using physics engines, they usually do not consider the explicit gripper design and do not check for collisions with the object or environment. Furthermore, many of them, like the Ferrari-Canny metric, force closure, and form closure, can be computed for any gripper configuration, e.g. for parallel-jaw, three-fingered grippers or human-like hands.

However, analytical metrics are heavily dependent on the accuracy of the physical properties, with their applicability decreasing in the presence of uncertainty in sensor measurements and actuation. In an attempt to strengthen analytical metrics under such circumstances, robust versions of analytical metrics can be used by calculating their probability of success under the influence of grasp perturbations [88, 108, 110]. The perturbations show the variability of the analytical metric over a range of possible gripper poses [108] and can additionally include variations in object poses and friction values [88, 110]. Perturbations can be sampled using Monte Carlo sampling [88, 108] and used in large-scale supervised learning to estimate grasp robustness [110].

In recent years, analytical grasp metrics have mostly been used to generate ground-truth information in simulated training datasets [11, 24, 74] and for applications which require grasp proposals for universal gripper configurations [111, 112]. As an alternative to using purely analytical grasp metrics, many data-driven methods in recent years have used simulated or real-world grasp experience recording grasp success or failure as ground truth for training grasp proposal algorithms [16, 20, 38, 82]

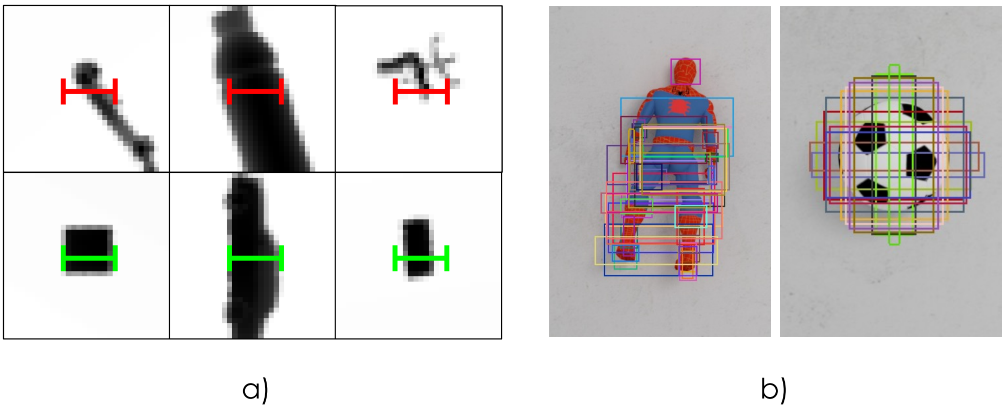


Figure 2.10: Example of images with a) a single, centred and horizontally aligned grasp in DexNet2.0 [11] and b) multiple grasp rectangles in the Jacquard dataset [77].

2.4 4-DoF grasp proposal

When moving from purely analytical to data-driven approaches for grasp proposal, most initial methods focused on producing 4-DoF grasp proposals to simplify the problem and the search space [9, 10, 11, 12, 13, 14, 15, 23, 63, 64]. 4-DoF grasps refer to grasp poses with a 3D position and a 1D rotation, usually around the approach vector of the grasp, as shown in Figure 2.4. The approach vector of those grasps is often aligned with the camera principal ray, resulting in planar grasps in the image plane, and/or aligned with gravity, which are commonly referred to as top-grasps. In other words, 4-DoF grasps limit the variability and complexity of grasp poses by locking 2 of the DoF in place.

Although the representation of 4-DoF grasps shows minor variations across different approaches, they are often represented as rectangles in the image plane, e.g. as shown in Figure 2.10 b). There are several datasets available with ground-truth information on 4-DoF grasp samples, for example, the Cornell dataset [76], the Jacquard dataset [77] and the DexNet2.0 dataset [11]. Ground-truth information is either manually annotated [76], based on simulated grasp experience [77] or based on analytical metrics calculated in simulation [11]. Those datasets provide a basis to train data-driven methods for 4-DoF grasp proposal.

2.4.1 Discriminative 4-DoF grasp proposal

There are a variety of discriminative methods that sample and subsequently classify 4-DoF grasps [10, 11, 22, 59, 113]. The sampling of candidates in such methods during inference is often realised by uniform random sampling from distributions based on the scene representation [11, 113] or using specifically trained Neural Networks (NNs) to sample grasp candidates [10]. Grasp candidates are then usually encoded in an input for a (Convolutional) Neural Network specifically trained to estimate the quality of the proposed grasp candidates. This process can also be run iteratively to improve the quality of grasp proposals [11].

One of the seminal approaches in this field is the Grasp Quality Convolutional Neural Network (GQ-CNN) [11, 60], which represents grasp candidates in a depth image and evaluates their quality using a CNN. The CNN consists of two parallel streams, one for the depth image and one for the distance z between the grasp TCP and the camera. The grasp position is centred in the depth image, and the grasp x-axis, along which the gripper plates close, is aligned to lie horizontally in the image, see Figure 2.10 a). This results in an aligned depth image, which reduces the number of explicit variables to define a grasp. The full DoF for a given grasp are thereby given by the aligned depth image and the distance z.

GQ-CNN is trained on the DexNet2.0 [11] dataset, containing synthetic images with a robust version of the Ferrari-Canny metric (see Equation 2.4) [88, 107, 110] as ground-truth information. In subsequent work, GQ-CNN has been extended to GQ-CNN2.1 to work in cluttered environments [22]. GQ-CNN2.1 is based on a GQ-CNN2.0 model, which is fine-tuned on a small dataset with cluttered scenes and simulated grasp experience generated in PyBullet [114]. In further extensions, GQ-CNN has been adapted to work for suction-cup grippers [59] and embedded in a partially observable Markov decision process with an ambidextrous robot with both a parallel-jaw and a suction-cup gripper [60].

GQ-CNN is one of the most popular approaches among the discriminative 4-DoF grasp proposal methods with a widely used open-access codebase and a ROS node that can be used for application on a real robot. Similar to most approaches for discriminative 4-DoF grasp proposal, GQ-CNN's usage is limited to tabletop scenarios with fixed workspaces in relation to an overhead camera, as this setup matches the scenes in the training data. A major disadvantage in GQ-CNN and all discriminative methods is the run-time since the individual evaluation of all grasp samples is computationally expensive [8].

2.4.2 Generative 4-DoF grasp proposal

Rather than sampling grasp candidates and evaluating their quality individually, one can also directly regress to several "good" grasp proposals using a generative grasp proposal method to improve run-time. One of the first generative methods for 4-DoF grasp proposal was proposed in 2015 by Redmon and Angelova [9], with an option to propose a single grasp or multiple grasps for a given scene. The model consists of a CNN with fully connected layers trained on the Cornell dataset [76] by randomly choosing a ground-truth positive grasp for each training image. When proposing a single grasp for a scene, it is assumed there is at least one successful grasp per scene. When proposing several grasps, which the authors call the MultiGrasp model, it divides the image into a grid and proposes a grasp coupled with a likelihood of success for each cell.

The concept for the MultiGrasp model in [9] is tightly coupled with the concept of region proposal networks often used in object detection algorithms [115]. Algorithms proposed after these initial generative methods with single and multiple grasp proposals per scene [9, 66] moved to outputting a grasp proposal for each pixel in the input image coupled with a quality prediction, more related to a segmentation than an object detection problem.

These later generative methods usually consist of FCNNs or transformer networks and output dense tensors [12, 23, 63, 67, 116]. The dense output tensors have the same size as the input image and apply a pixel-wise representation for suitable grasps based on the input image. At least two different approaches with FCNNs have been used for generative grasp proposal, with differences in the structure of the output tensors. One of them uses multiple 2D tensors, which include a quality estimation coupled with an angle and width for each pixel and anchor the grasps at the TCP [12, 23, 116].

The alternative to using multiple 2D tensors consists of 3D [44, 63] or 4D [63] grasp quality heatmaps where the size of the third and fourth dimensions define quantified values for the rotation [44, 63] and height [63] of grasps in so-called “structured bins”. Each cell in the resulting heatmap defines the quality of a TCP-anchored grasp. Due to the quantitisation, the position of the cell in the heatmap fully defines the position, rotation [44, 63] and height [63] of the grasp.

The quality estimates in the 2D tensors and multi-dimensional heatmaps are used to select the best grasp during inference. The grasp proposals are constructed from the angle and width output tensors at that pixel or inferred from the cell’s position in the multi-dimensional heatmap. One of the major advantages of such methods is the run-time during inference, as many grasps are proposed with a single forward pass. Some methods in this category, like the Generative Grasping Convolutional Neural Network (GG-CNN), are specifically developed to enable closed-loop control [12].

Similar to discriminative approaches for 4-DoF grasp proposals, the presented generative approaches are limited to tabletop scenarios with mostly fixed workspaces in relation to the camera pose, limiting their application in more complex scenarios. Comparing the IoU of the different methods on an object-wise test split of the 4-DoF Cornell grasping dataset shows a higher performance for generative methods than discriminative methods in [8]. This indicates that the advantages of generative methods in terms of run-time do not come at the cost of grasp performance.

In general, 4-DoF grasping approaches are less complex than 6-DoF grasping approaches due to the reduced flexibility and complexity of their grasp representation. While 6-DoF grasp proposals provide more flexibility for the grasp pose, this might not be necessary for every scenario [5]. For example, tasks such as bin-picking [60] in pre-defined and well-reachable spaces might not require the added flexibility of 6-DoF grasp proposals. In such a case, solutions can benefit from the advantages in run-time and efficiency of 4-DoF grasp proposal methods.

However, these advantages come at the cost of reduced flexibility and, thereby, reachability of the grasp poses. While specific scenarios might not suffer from this reduction, more complex scenarios with objects in clutter or in constrained environments require 6-DoF grasps in order to grasp objects [7]. This is especially apparent when moving into domestic environments where objects can be placed on different furniture units at different heights. Objects placed very high or low in the scene require different approach directions to enable collision-free grasps, particularly when considering the robot end-effector's reachability. Being able to propose 6-DoF grasps for objects independent of the object's pose and surroundings without limiting the DoF of the grasp proposals is a key factor for enabling successful robotic grasping in such scenarios.

2.5 6-DoF grasp proposal

In light of the advantages in flexibility and reachability, robotic grasping research has progressed towards 6-DoF grasp proposal in recent years. 6-DoF grasp poses are defined by a 3D position and a 3D orientation in space, see Figure 2.4. In contrast to 4-DoF grasps, none of the 6 DoFs is restricted, resulting in increased complexity for the grasp representation and an increased search space for the grasps. In the same manner as 4-DoF grasp proposal, 6-DoF grasp proposal methods can be distinguished into discriminative and generative methods.

2.5.1 Discriminative 6-DoF grasp proposal

There are several examples for 6-DoF discriminative methods [24, 38, 40, 82, 117] which sample 6-DoF grasp poses and subsequently evaluate their quality individually. Grasp candidates can, for example, be sampled from the scene representation by sampling points and estimating their orientation based on surface normals [15, 24, 40, 45, 117] or generated by trained model architectures like Variational Autoencoders (VAEs) based on a probability density function over the latent space [38, 82]. The grasp candidates are then encoded in some input form corresponding to the scene representation and evaluated by grasp quality prediction models, for example, using PointNet-like architectures [38, 40, 82], CNNs [24, 117] or analytical grasp quality metrics based on shape-completed meshes [15, 45].

One of the most widely used and referenced methods in this area is the Grasp Pose Detection (GPD) algorithm [24]. Grasps are sampled uniformly over a region of interest in the input point cloud and subsequently encoded in a multi-channel image showing different projections of the grasp. A CNN is then used to predict grasp quality based on the multi-channel image. The BigBird [86] object meshes are used for training, with force closure used as the ground-truth metric for sampled grasp poses. GPD is often used to compare against other algorithms [16, 17, 21], possibly in part due to the availability of a reference implementation in ROS since 2017.

Similar to the discriminative 4-DoF methods described in Section 2.4.1, a major disadvantage of discriminative 6-DoF grasp proposal methods is their low computational effiency [5, 8] and consequently slow run-time. This disadvantage is further increased when moving from 4-DoF to 6-DoF, since the search space for potential grasp poses increases, potentially leading to more grasp samples which need to be evaluated. As of a survey from 2022, none of these methods have demonstrated real-time capabilities [5].

2.5.2 Generative 6-DoF grasp proposal

There exist many different approaches for generative 6-DoF grasp proposal. Those approaches can be further distinguished by the scene representation used for the model input. Among the possible scene representations are point clouds [20, 70, 118, 119], depth images [18, 37, 120], and voxelised grids [16, 17, 21]. As of 2023, there is no consensus as to the effect the scene representation has on grasp performance and if any single representation is superior for this kind of application. However, the typical architectures that are used differ between scene representations, which we detail in the following.

When presenting the scene in the form of a point cloud, the models typically use a 3D CNN such as PointNet++ [20, 38, 70, 118], a transformer network [119] or a graph network [121] for their model architecture. The point clouds for generative methods are usually subsampled to increase efficiency [5], which introduces a question on how to process the point cloud for optimal results. Overall, the methods seem to perform well in terms of grasp success [8]. While some approaches demonstrate operation in real-time, these are usually demonstrated on high-end

Graphics Processing Units (GPUs) [20], which have limited applicability to mobile platforms.

It is also possible to create a voxelised scene of a 3D workspace, for example, using a Truncated Signed Distance Function (TSDF), and to subsequently use a 3D CNN to propose grasps [16, 17, 21]. Since the 3D convolutions are computationally expensive [8], the workspace size and resolution must be carefully picked to enable fast run times while retaining a high grasp success. Furthermore, for application of any of these algorithms, the pose of the 3D workspace in relation to the camera has to be known in advance, which poses a problem in complex and unstructured environments.

One example of a method using a 3D CNN is the Volumetric Grasping Network (VGN) [16]. It scans a workspace along a pre-defined trajectory using a wrist-mounted depth camera. The scene information is integrated into a TSDF, which is input to the model in the form of a voxelised grid. The model is trained on synthetic data from 3D object meshes and simulated grasp attempts using PyBullet [114]. Using multiple viewpoints for constructing these representations serves to increase grasp performance by decreasing the scene representation's uncertainty and noise. However, scanning a workspace around a graspable object to construct such a representation might not always be feasible or desirable due to the need for appropriate equipment, time and robot motions.

Similar to the 4-DoF grasp proposal methods presented in Section 2.4, the scene in 6-DoF grasp proposal methods can also be represented with depth images, where a FCNN is used to propose grasps [18, 120]. A general advantage of using FCNNs in grasping is their simple design and the computational efficiency of the 2D convolutional layers. The approaches usually propose 4-DoF grasps based on multiple virtual camera viewpoints to create 6-DoF grasps [18, 120]. While those predicted grasps have their approach axis aligned with the virtual camera ray and only vary in 4-DoF, the versatile placement of the virtual camera varies the grasp proposals in 6-DoF. However, the variability in the grasp poses can only approach the flexibility of "real" 6-DoF grasp poses when a sufficiently large number of virtual camera placements is assumed. This, in turn, would reduce the computational efficiency of such approaches.

Overall, it is likely that robotic grasping research will continue towards 6-DoF compared to 4-DoF grasp proposal methods. Generating such 6-DoF grasp proposals is necessary when applying robotic grasping solutions to complex environments, for example, in domestic scenarios. Objects in such scenes are placed in unknown poses and can be surrounded by different objects or furniture units, increasing the risk of collisions. If a grasp proposal method is limited to 4-DoF grasps, crowded immediate surroundings of the object can lead to a substantial decrease in clearance when all proposed grasps result in collisions. 6-DoF grasp poses, on the other hand, result in an increased search space and flexibility of the end-effector's pose. This can increase clearance by including previously unreachable grasp poses, making ungraspable objects graspable.

When moving towards the usage of general-purpose robots in real-world scenarios, the environments are becoming more complex and flexible. The resulting increase in search space of corresponding grasp proposal methods causes computational efficiency to be of even greater importance. As such, computational efficiency is essential when selecting a grasp proposal algorithm for real-world applications. In general, generative methods seem to become dominant and take over from discriminative methods due to their computational efficiency [5, 8].

2.6 Reinforcement Learning in robotic grasping

While most of the data-driven methods in robotic grasping are trained using the generative and discriminative methods described above [5], one can also use a Reinforcement Learning (RL) approach and learn a policy to grasp objects with 4-DoF or 6-DoF grasp poses. Examples of such methods have been trained in simulation [43] and real-world experiments [73, 122]. An advantage presented in many RL approaches is their ability to be used with closed-loop control, enabling them to perform dynamic grasping for changing environments and potentially learning a sequence of actions to achieve a grasping goal, e.g. with push-to-grasp policies [123].

A review in 2019 identified issues in applying RL approaches to real-world problems, including sample inefficiency, the balance between exploration and exploitation, the generalisation and reproducibility of the algorithms, and how to efficiently learn from demonstration [124, 125]. More recently, there have been improvements made in those aspects, for example, with advances in the sample efficiency [122, 126, 127] or learning from human demonstration [36, 128]. An example of an improvement for the sample efficiency of RL approaches is presented with [122], which uses equivariant models to reduce training time substantially. Their presented method learns *online* on real robotic hardware in as few as 1.5 hours. This enables learning different, specific tasks like grasping opaque or transparent objects. While the experiments demonstrated with these new methods are based on objects in settings with fixed manipulators and fixed cameras, further efforts will likely extend their applicability to more complex scenarios.

An issue with some RL algorithms lies in their setup, where agents learn actions to follow a trajectory for grasping objects rather than proposing grasp poses. This means those algorithms usually can only execute a single grasp per scene with no choice between different grasp proposals, for example [36, 37, 43, 73]. This, in turn, makes it difficult to incorporate further constraints in the overall pipeline [5]. For example, if a single grasp for a scene is learned by a RL approach, target-driven grasping for a single object in a cluttered environment can not be achieved easily.

Overall, RL approaches are usually more challenging to implement than supervised generative or discriminative grasp proposal approaches [8]. However, they have received more attention in recent years, progressing the field and providing new solutions to problems. They will likely continue to do so, enabling solutions to the abovementioned problems and potential solutions to robotic grasping.

2.7 Mobile manipulation

Mobile manipulators combine a base's mobility with an arm's dexterity, significantly increasing the robot's workspace compared to fixed manipulators. Domestic environments usually have multiple workspaces, where objects are placed in different rooms and on different furniture units. Here, mobile manipulators can transport objects or provide a service at different workspaces in the environment.

An example of a mobile manipulator can be seen with the PAL TIAGo robot in Figure 1.1 c).

The design of mobile manipulators varies significantly depending on their purpose. While many research projects are based on customised combinations of a mobile base and a manipulator [129, 130, 131, 132, 133], there also exist complete mobile robots that provide hardware platforms e.g. the Toyota Human Support Robot [134] or the PAL TIAGo mobile manipulator [98]. Wheeled mobile bases are the most popular choice for mobile manipulators, but specific applications and tasks might require legged robots, aerial platforms or underwater robots [3].

The work on analytical, 4-DoF, and 6-DoF grasp proposal has mainly focused on robotic arms mounted on planar surfaces, with 66 out of 75 robotic platforms in a survey on data-driven grasping from 2022 using fixed robotic arms [5]. The perception sensors for the research projects on robotic arms are either in a fixed position to the workspace [11, 20, 63, 73] or attached to the wrist of the robotic arm [12, 16, 17, 36, 43]. Data-driven methods developed for application on fixed robotic arms likely have limited applicability to mobile manipulators, however. The principal issue is that methods for fixed robotic arms have not encountered sufficient variation in their workspace surroundings and the camera-workspace transformation during training.

Despite this limitation, data-driven grasp proposal methods can be used with a mobile manipulator in combination with workarounds, assuming the robot's surroundings are appropriate. The workarounds can be used to position the robot and camera in relation to the workspace such that the viewed scene is similar to the training data samples. For example, one can use 3D object detection to identify the pose of a specific object on a table, move the mobile manipulator to a suitable position and use a data-driven method like GPD [19], VGN [16] or GQ-CNN [60] for proposing grasps.

While such workarounds are possible, complete multi-purpose systems described for mobile manipulators rarely use the developed data-driven grasp proposal methods. Instead, they tend to utilise hand-crafted approaches based on point clouds [132, 133, 135, 136], analytical grasp metrics like force closure using reconstructed surface patches [137, 138], offline grasp proposal based on CAD models [136, 139] or use specialised grippers for the target objects in the desired application [140].

Grasping with mobile manipulators can also be achieved by applying Reinforcement Learning approaches [131, 141, 142]. This direction of research is relatively new, where methods are typically demonstrated on specific tasks [141] and a limited number of relatively simple objects [131], or where they may circumvent the need for grasp pose detection by employing landmarks within the scene [142].

There exist few projects to this date which apply data-driven 4-DoF and 6-DoF grasp proposal methods specifically to mobile manipulators, for example, DexNet-MM [143] and GarbageNet [129]. These projects are focused on grasping objects from the floor. When objects are positioned on the floor, the relative pose of the camera to the object and the surroundings of the objects include little variability, enabling the application of simple workarounds for data-driven approaches. DexNet-MM [143] uses the DexNet4.0 [60] 4-DoF grasp planner with adapted parameters for decluttering the floor, where the camera on the mobile manipulator is positioned overhead the graspable objects. GarbageNet, on the other hand, applies the more flexible 6-DoF grasp proposal method GPD on a segmented point cloud. While the initial viewpoints for GarbageNet are more flexible than those of DexNet-MM, it is limited to picking garbage off the floor and not designed for more complex environments [129].

As of 2023, there is little work for grasping unknown objects with mobile manipulators in complex environments exceeding tabletop scenarios. More particularly, data-driven grasp proposal algorithms are rarely applied to mobile manipulators, likely because the simplifications used during the training of the algorithms do not represent the environments encountered by such robots. Solutions for grasping objects with mobile manipulators are often tailored specifically for a single use case and thereby far from a robust general-purpose system that could be applied in complex, domestic environments.

2.8 Conclusion

In this chapter, we presented an overview of grasping objects with a robotic manipulator, emphasising different solutions to grasp proposal. Data-driven grasp proposal has surpassed purely analytical approaches in recent years. Such data-driven grasp proposal methods can generally only be expected to work on the kind of input data and scenarios they have been trained on. For most research projects in robotic grasping, the training scenarios focus on tabletop scenarios, often with a fixed robotic arm and a fixed overhead camera close to a pre-defined workspace [5].

This reduces scene complexity based on several assumptions and simplifications; for example, objects are always placed on planar surfaces and in a certain relation to the camera. These simplifications were necessary to progress in the field under the influence of the otherwise large search space and variability that comes with real-world conditions. However, with the field advancing, more complex scenarios need to be considered to move robotic grasping towards robust application in the real world and specifically in domestic environments.

Such domestic environments are dynamic, complex and not defined in advance, with various surroundings where objects should be grasped from different poses like the floor, tables or shelves. The scenarios place broader requirements on proposed grasps, requiring diverse, high-quality grasps to increase the probability of finding collision-free, reachable grasp poses among them [5]. Since these scenarios are more complex than the tabletop settings, they must be reflected in the training and evaluation of grasp proposal algorithms to enable application in domestic environments.

The next steps are, therefore, to extend robotic grasping to such applications by removing assumptions and simplifications placed on the training pipelines. This can ultimately lead to general-purpose solutions that can be applied reliably in complex, domestic environments. It is important to test resulting algorithms in those complex scenarios, such that they are evaluated under the influence of the additional constraints posed by those scenarios.

In this thesis, we approach the goal of grasping unknown objects in domestic environments in several steps. We initially focus on the variability of camera viewpoints and how a discriminative grasp proposal method with a simple scene representation can generalise to different camera viewpoints and 6-DoF grasps. Next, we convert this approach to a generative grasp proposal model that can be used for 6-DoF grasp proposal of single objects on a planar surface. Finally, we show how a grasp proposal model can be applied to domestic scenarios by introducing a reproducible pipeline for generating domestic scenes with ground-truth information about grasps in simulation. We use this simulation environment to train and evaluate a grasp proposal method to grasp objects on shelves and tables.

CHAPTER 3

Versatile Camera Viewpoints and 6-DoF Grasps for Grasp Quality Prediction

In Chapter 2, we reviewed how grasp proposal for 4-DoF and 6-DoF grasps has been approached in recent years and how intrinsic limitations in these approaches have restricted their transferability to real-world mobile robotics applications. Conventionally, and to simplify the search space, grasp proposals for unknown objects are usually generated with limited DoF and from fixed viewpoints in regard to a workspace. One widely studied use-case with these limitations is that of proposing 4-DoF top-grasps on planar surfaces from a fixed, overhead camera viewpoint [9, 10, 11, 63, 66, 113]. The resulting models are usually used with fixed manipulators, for example, moving single objects within the workspace or picking objects from a bin. Due to the simplified grasp representation and the scenarios encountered in the training data, methods developed for this use-case are not transferable to scenarios with flexible viewpoints and 6-DoF grasps.

This chapter shows how a depth-image based, 4-DoF grasp quality prediction model for fixed camera viewpoints can be extended to 6-DoF grasps from versatile and unknown viewpoints. The difference in variability for relative grasp,

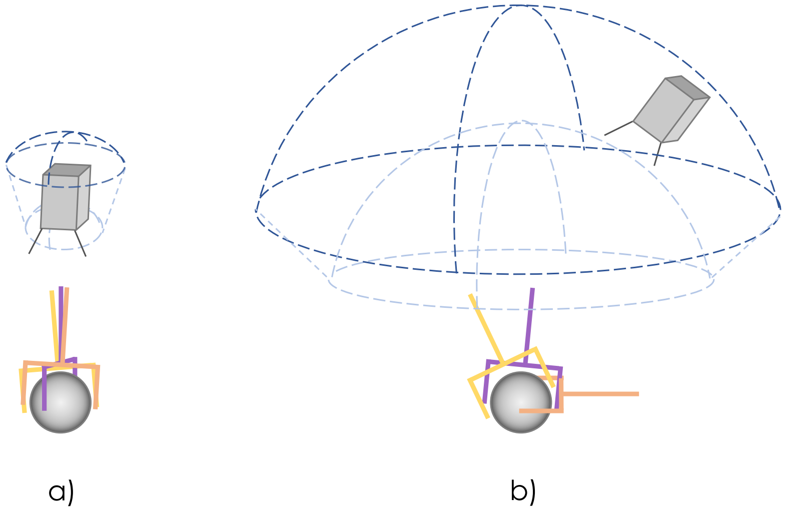


Figure 3.1: Visualisation of the differences in relative grasp, camera and object pose variety for a) 4-DoF grasps and overhead camera poses and b) 6-DoF grasps with flexible viewpoints. The sampling boundaries for the camera placement are visualised with the dashed, blue lines.

camera and object poses of the two different models is visualised in Figure 3.1. We see this extension as the first step when moving from fixed workspaces with stationary cameras and 4-DoF grasps to the application on mobile manipulators. Furthermore, this chapter can be seen as a proof-of-concept for Chapter 4, where we evolve this depth-image based, 6-DoF and versatile viewpoint approach to a grasp proposal method.

We based our VGQ-CNN on GQ-CNN by Mahler et al. [11]. GQ-CNN is a model for grasp quality evaluation of 4-DoF top-grasps from a fixed overhead camera. In contrast, VGQ-CNN can evaluate 6-DoF grasp proposals of objects on planar surfaces viewed from a wide range of camera poses above the object. We introduced the VG-dset to include these variations in camera poses and grasp orientations, significantly exceeding the range available in commonly used depth-image-based datasets such as [10, 11, 77].

In summary, the contributions of the work presented in this chapter are:

- VG-dset: A versatile 6-DoF grasp quality dataset including 7.2 million grasps over an extended range of camera poses.

- VGQ-CNN: A model capable of predicting the quality of 6-DoF grasps under a wide range of camera poses.
- Fast-VGQ-CNN: An alternative model structure to VGQ-CNN to speed up inference run-time.

3.1 Related work

We based VGQ-CNN on the GQ-CNN first introduced by Mahler et al. in 2017 [11]. GQ-CNN can be categorised as a discriminative, 4-DoF method (see Section 2.4). It predicts the quality of individual grasp samples presented in an aligned depth image. The ground-truth grasps are sampled using an antipodal grasp sampling technique and are annotated with the robust ferrari-canny metric.

One part of the work presented with GQ-CNN is the construction of a synthetic, depth-image-based dataset to train GQ-CNN, called the Dexterity Network (DexNet2.0). The depth images include objects from the 3Dnet [144] and the KIT [87] mesh datasets in stable poses on a planar surface as rendered from an overhead camera viewpoint. The 32×32 px images are aligned with ground-truth grasp poses, where the TCP of the grasp is centred, and the grasp x-axis between the gripper plates is aligned to be horizontal in the image. The ground-truth quality of the grasps is binary and based both on the robust ferrari-canny metric [110] and the condition of it being a collision-free grasp.

In total, DexNet2.0 consists of 6.7 million images of ground-truth grasp poses, including almost 1,500 different 3D object models. DexNet2.0 is used to train GQ-CNN, a CNN to predict the quality of grasp samples. It consists of two parallel input streams, an image- and a pose stream, merged in the head end of the model and finally produces a prediction of grasp success $\hat{\varrho}$, see Figure 3.2. The input to the image stream is an aligned depth image with a single grasp centred and oriented to lie horizontally in the image. The input to the pose stream is a single float value z describing the distance between the camera origin and the TCP of the grasp. This means GQ-CNN evaluates just a single grasp at a time

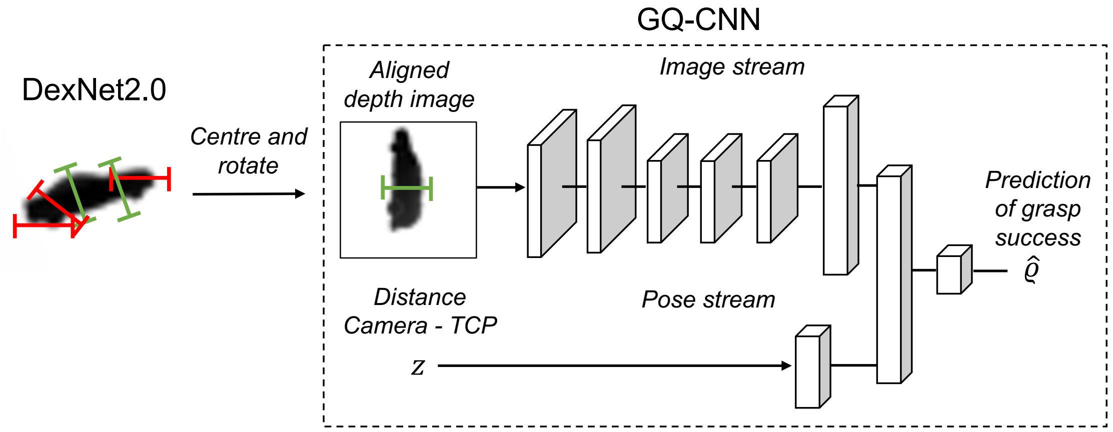


Figure 3.2: Visualisation of the GQ-CNN [11] architecture and training pipeline. GQ-CNN consists of a CNN with two parallel streams, one for the aligned depth image and one for the distance z between the camera and grasp TCP.

and requires an appropriately translated and rotated depth image for evaluating each grasp.

GQ-CNN and its extensions have been widely used and referenced in the literature [15, 145, 146, 147] and a reference implementation has been made available to use with the Robot Operating System (ROS) and MoveIt! [148]. The extensions of GQ-CNN have mostly focused on objects in clutter [22] and alternative gripper mechanisms like suction-cup grippers [59, 60]. As such, these models maintain the 4-DoF grasp representation and are limited to top-grasps.

Another extension of GQ-CNN has been presented with a FCNN [63], which works as a generative approach for 4-DoF grasp proposal by predicting the grasp quality for a range of height and orientation bins for each pixel in the input depth image. The bins are pre-defined and encompass the discrete range of all possible grasp proposals in the workspace. Although this approach improves run-time, the necessary parameterisation to allow for 6-DoF grasps would significantly increase the size of the output tensor as currently formulated. For example, the output tensor would increase from 4-DoF to 6-DoF, and, if using a similar quantisation as in [63] with 16 bins for each variable, the represented grasps for a *single* pixel in the output tensor would increase from $16^2 = 256$ for 4-DoF to $16^4 = 65,536$ with 6-DoF. Hence, we use GQ-CNN as the basis for our extension to 6-DoF from

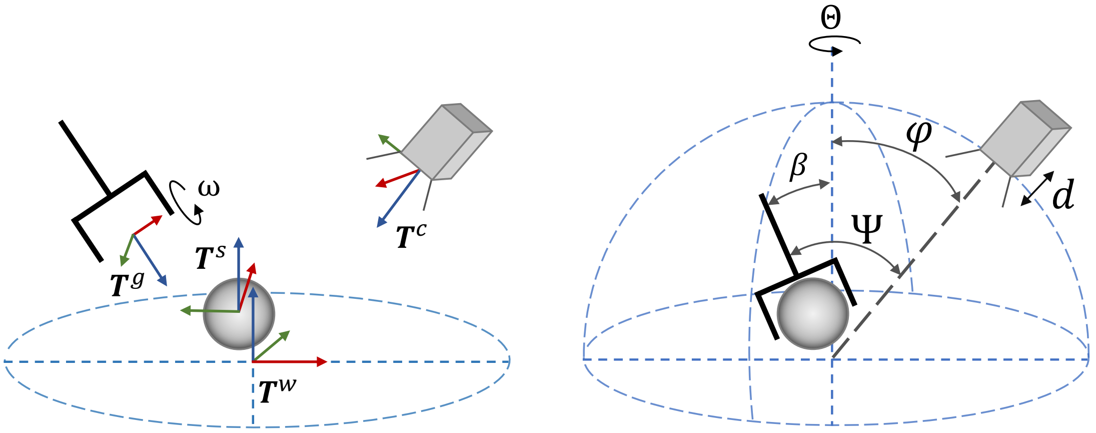


Figure 3.3: Visualisation of the coordinate frames $\mathbf{T}$ for the workspace wo, stable pose s, camera c and grasp g, as well as the angle ω for rotating the grasp g around its contact points. Furthermore, we show the spherical coordinates of the camera c relative to the workspace $\mathbf{T}^{\text{wo}}$ with the radial distance d, the inclination angle φ and the azimuth angle θ. Relative orientations of the grasp approach axis are denoted with the angle to the table normal β and the angle to the camera principal ray Ψ.

flexible viewpoints as opposed to any follow-up models proposed by the original authors.

3.2 Problem formulation

We consider the problem of predicting grasp qualities for 6-DoF grasp poses with a parallel jaw gripper as observed from a wide range of camera perspectives. The environment is limited to a single object placed on a planar surface, with objects in clutter not being considered in this chapter. The goal is to train a model that can predict the qualities of sampled grasp poses based on depth images from a wide range of camera poses without having to retrain the model for each new camera viewpoint. In addition, the model should be able to evaluate 6-DoF grasp poses where the gripper approach axis is not necessarily aligned with a pre-specified fixed axis as in 4-DoF grasps. Such a model caters to situations that involve changing the camera pose with regard to the workspace between setups whilst also supporting mobile manipulators, where the relationship between camera, workspace and object can change when performing different tasks in an environment.

Figure 3.3 shows a visualisation of the setup. It consists of a workspace w, with its coordinate frame origin $\mathbf{T}^{wo}$ sitting on top of a planar surface. The objects are placed in the workspace in predefined, stable resting poses s [149] with $s \in SE(3)$. Their coordinate frame $\mathbf{T}^{s}$ is positioned in close proximity to the origin of workspace coordinate frame $\mathbf{T}^{\text{wo}}$. The camera position is defined in spherical coordinates relative to $\mathbf{T}^{\text{wo}}$, with the radial distance to the workspace origin d, the inclination angle φ relative to the workspace z-axis and the azimuth angle θ rotating around the workspace z-axis. The camera principal ray points towards the origin of $\mathbf{T}^{\text{wo}}$, and the camera x-axis is parallel to the x-y plane of the workspace, i.e. the table surface. This orients the camera frame, $\mathbf{T}^{c}$, such that the camera views the table horizontally. The gripper frame, $\mathbf{T}^{g}$, is defined such that it rests at the TCP of the parallel jaw gripper. The x-axis of $\mathbf{T}^{g}$ lies between the contact points of the parallel jaw gripper, and the z-axis denotes the direction of the final, linear approach of the grasp point from a pre-grasp position as shown in Figure 2.1 (2) to (3).

We define the camera pose in spherical coordinates from the workspace origin $\mathbf{T}^{\text{wo}}$ with $d \in [0.4m, 1.1m]$, $\theta \in [0°, 360°]$ and $\varphi \in [0°, 70°]$, which corresponds to a volume of roughly $2.1m^3$. The limits were chosen in reference to the typical operating volume of available mobile manipulators, e.g. PAL TIAGo robot [98] or Toyota HSR [134] when grasping objects placed on a table. Configurations where the camera is below or level with the planar surface, for example, when grasping objects from a top shelf, and those with objects placed close to the edge of the surface are excluded.

Since the quality and usability of a grasp are influenced not only by the gripper pose in relation to the table and the object but also by the visibility of the graspable surface from the camera, we define a set of relative angles to parameterise the space of unique camera-object-gripper configurations. These angles are visualised in Figure 3.3. We denote the angle between the gripper z-axis and the table normal as β, and the angle between the gripper z-axis and the camera principal ray as Ψ. Rotating the grasp around the grasp x-axis is denoted by ω and does not alter the position of the contact points.

Dataset	φ_{max}	d_{min}	d_{max}	β_{max}
VG-dset	$70°$	$0.4m$	$1.1m$	$90°$
DexNet2.0	$5.7°$	$0.65m$	$0.75m$	$5°$

Table 3.1: Parameter overview for VG-dset and DexNet2.0. The parameters include the spherical camera coordinate limits with inclination angle φ and radial distance to the workspace origin d, as well as the angle between the gripper z-axis and the table normal, β.

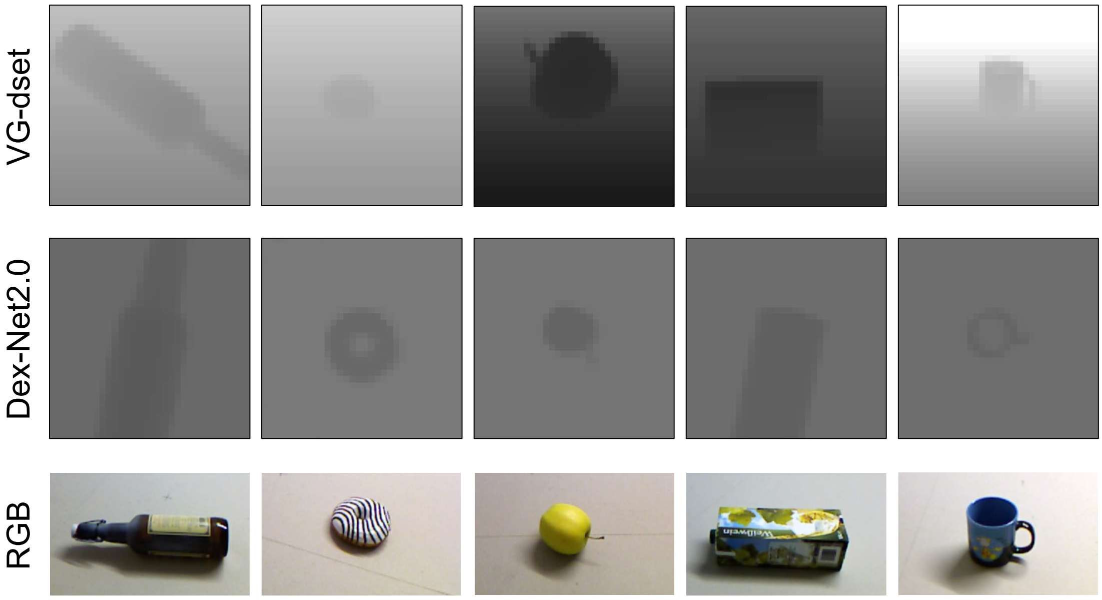


Figure 3.4: Example images with possible variations of camera viewpoints in VG-dset (upper), DexNet2.0 (middle) and an RGB representation of the corresponding object (lower). Each column shows objects in the same 3D pose in all rows, with a bottle, doughnut, apple, carton and mug appearing from left to right. The same normalisation boundaries for visualisation were applied to all depth images with $[0.4m; 1.3m]$.

3.3 Dataset

We created a new 6-DoF grasp quality dataset called VG-dset to satisfy the specifications in our problem formulation in Section 3.2. VG-dset exceeds the sampling ranges for multiple parameters in DexNet2.0 substantially, as shown by the parameter ranges used for the two datasets in Table 3.1 and the example images in Figure 3.4. We based VG-dset on the pre-sampled antipodal grasps $g \in \mathcal{G}(o)$

included in DexNet2.0 [11], where the object meshes $o \in \mathcal{O}$ were taken from the KIT [87] and 3DNet [144] mesh datasets.

Dataset preparation for VG-dset consisted of dataset rendering and sampling. Datapoints were created throughout the dataset rendering process by rendering images and storing associated grasp poses and quality values. A total of 131 million grasps were stored in the process, providing a baseline that could be sampled for different purposes. The dataset sampling process was necessary since certain minimal ratios, e.g. between ground-truth negative and ground-truth positive grasps, had to be satisfied to ensure successful training of VGQ-CNN. The process could also be used to allow for different dataset compositions, for example, excluding certain camera or grasp configurations. While the dataset generation process could be adjusted to generate a single, balanced dataset to be used for training, the time-consuming process of rendering the data and checking the grasps would have to be repeated each time a different dataset composition was desired.

3.3.1 Dataset rendering

We based our dataset rendering algorithm on the DexNet2.0 implementation by Mahler et al. [11] available on Github[1] with key changes in the camera poses, grasp alignment and grasp representation. Each object mesh $o \in \mathcal{O}$ has an average of 9 stable resting poses $s \in \mathcal{S}(o)$. When rendering the dataset, we repeated the steps detailed in Algorithm 3.1 for each stable resting pose of each of the 1492 object meshes.

We sampled the camera coordinates by uniformly sampling the radial distance d between the camera and the workspace origin, the azimuth (or azimuthal) angle θ and the inclination (or polar) angle φ individually. We used d, θ and φ to define the translation $\mathbf{p}_c^{\mathrm{wo}}$ of the camera origin in workspace coordinates and constructed the camera transform $\mathbf{T}_c^{\mathrm{wo}}$ to position the camera at $\mathbf{p}_c^{\mathrm{wo}}$ and view the workspace origin centred and upright (see Figure 3.3). We rendered and saved a depth image based on the camera transform, as can be seen in Algorithm 3.1, lines 2-4.

[1] `https://github.com/BerkeleyAutomation/dex-net`

Algorithm 3.1 Generating VG-dset

```
for $1:n$ do
    Sample $d, \theta$ and $\varphi$ for camera origin $\mathbf{p}_c^{\text{wo}}$ from uniform distributions
    Set camera transform $\mathbf{T}_c^{\text{wo}}$ to camera origin $\mathbf{p}_c^{\text{wo}}$ pointing towards $(0,0,0)$
    Render and save depth image $\mathcal{I}$ from $\mathbf{T}_c^{\text{wo}}$
    for each grasp $g \in \mathcal{G}(o)$ do
                                        # Apply random gripper rotation around grasp x-axis
        $\mathbf{R}_g = \mathbf{R}_g \cdot \mathbf{R}_X(\omega)$, $\omega \sim \mathcal{U}(0, 2\pi)$
        if $g^{\text{wo}}.z$ is pointing toward positive wo.$z$ then
                                        # Grasp coming in from under the planar surface
            $\mathbf{R}_g = \mathbf{R}_g \cdot \mathbf{R}_X(\omega)$, $\omega = \pi$
        end if
        if linear approach of $g$ collides or $\epsilon_Q(g) < \delta$ then
            Set success of grasp $\varrho(g) = 0$                      # Negative grasp
        else
            Set success of grasp $\varrho(g) = 1$                      # Positive grasp
        end if
        $u(g), v(g) = \text{Project}(\mathbf{T}_g^c, K)$              # Project grasp into image
        $z(g^c) = \|p_g^c\|$                          # Calculate grasp distance to camera
        Save $u(g), v(g)$, $z(g^c)$, $\mathbf{r}(g^c)$ and $\varrho(g)$
    end for
end for
```

Moving away from top-grasps and allowing for a variety of grasp orientations represents a challenge to both the grasp preparation and representation. For stationary cameras and top-grasps, some sense of alignment is usually assumed between the grasp orientation and the camera position. In DexNet2.0 [11], this was realised by rotating the gripper around ω to minimise β and discarding grasps with $\beta \geq 5°$. By projecting the gripper TCP and gripper x-axis into the image plane and rotating/cropping the image accordingly, Mahler et al. [11] reduced the grasp representation to an aligned image and the distance between the grasp T^g and the camera T^c. This is similar to the reduced grasp representations in [9, 12, 14], all defined as positions and orientations in the 2-D image plane.

Since VG-dset should include 6-DoF grasps, we aimed to include grasps with the same grasp x-axis and varying approach angles. For example, a top-grasp rotated by 90° around ω would end up grasping the object parallel to the table. Ideally, a model trained on VG-dset should be able to predict the quality of both grasps and thereby select all good grasps for the path planning system of a robot to choose from. We augmented the data with a variety of grasp orientations in VG-dset by applying a random rotation around ω to the grasps with each new camera pose. We flipped grasps approaching from under the table (i.e. with $\beta > 90°$) by rotating them by a further 180° around ω, since they would likely cause a collision with the table, see lines 6-9 in Algorithm 3.1.

We generated this variety in grasp orientations to provide flexibility to various sampling schemes (see Section 3.3.2) and to simplify constraining the final grasp orientation. Note that rotating the gripper around ω does not change the robust Ferrari-Canny metric $\epsilon_Q(g)$ [110] of a given grasp, and therefore does not influence the grasp success $\varrho(g)$ aside from collisions checked after the final grasp orientation has been decided. As in DexNet2.0, we set the robustness threshold $\delta_Q = 0.002$.

Similar to DexNet2.0 [11], we then proceeded to collision checking by setting the grasp success for grasps colliding with the object or table to $\varrho(g) = 0$ and thereby marking the grasp as ground-truth negative, see lines 10-14 in Algorithm 3.1. After collision checking, the gripper TCP was projected into the image plane to calculate its image coordinates (u, v).

We rendered $n = 100$ images $\mathcal{I}$ for each stable resting pose $s \in \mathcal{S}(o)$ for each of the 1492 object meshes $o \in \mathcal{O}$, while $n = 50$ for the DexNet2.0 dataset generation. Each image $\mathcal{I}$ contained a single object o in one of an average of 9 stable resting poses s, and up to 100 ground-truth grasps g. This resulted in a total of 130.8 million aligned images, i.e., images with a single grasp centred and oriented to lie horizontally in the image, which we refer to as grasping data points in the following.

Since our setup included a wider range of camera poses and grasp orientations, increasing n and sampling more images $\mathcal{I}$ per stable pose s allowed us to cover the increased parameter space in the resulting images. We then sub-sampled the grasping data points to ensure specific dataset balances, reducing the number of grasping data points in the dataset to a comparable size of DexNet2.0.

3.3.2 Dataset sampling

The 130.8 million grasping data points from the dataset rendering stage needed to be sub-sampled to generate VG-dset, a dataset for training VGQ-CNN. This sub-sampling process of the grasping data points had three purposes:

1. Filtering grasp and/or camera configurations for training VGQ-CNN, e.g. removing grasps with $\Psi \geq 90°$.
2. Adjusting the positivity rate $pr = \frac{\#ground\ truth\ positive\ grasps}{\#all\ grasps}$ over the angle between the table normal and the grasp approach β, since it is naturally decreasing for increasing β which affects the model performance.
3. Balancing the number of grasps across the camera and grasp configurations since it affects the model performance.

We found the positivity rate pr (2.) and the number of grasps (3.) across the camera and grasp configurations substantially influenced model performance. For example, training a model on the native distribution in the rendered dataset, as shown in Figure 3.5 a), led to a high error rate when trying to predict ground-truth positive grasps due to the overall low positivity rate pr. Furthermore, there were very few examples of top-grasps available, causing a drop in performance when using the model to predict them. We decided to use a sub-sampling process to balance the dataset and achieve consistent performance. We discuss the possibility of using a weighted loss function rather than sub-sampling the dataset in Section 3.7.

The sub-sampling process was not needed in DexNet2.0 [11] since their dataset parameters only included a limited range for the parameters like Ψ and β (see Table 3.1 and Figure 3.1) in which the grasping points were naturally balanced for the configurations mentioned above. The large size of the rendered dataset enabled differing dataset compositions to be sampled and their effect on the model performance to be tested. We started by removing grasps with $\Psi \geq 90°$ from the dataset. This maximum value was set so that all remaining grasps approach the final grasp pose in the direction of the camera principal ray. This selection criterion

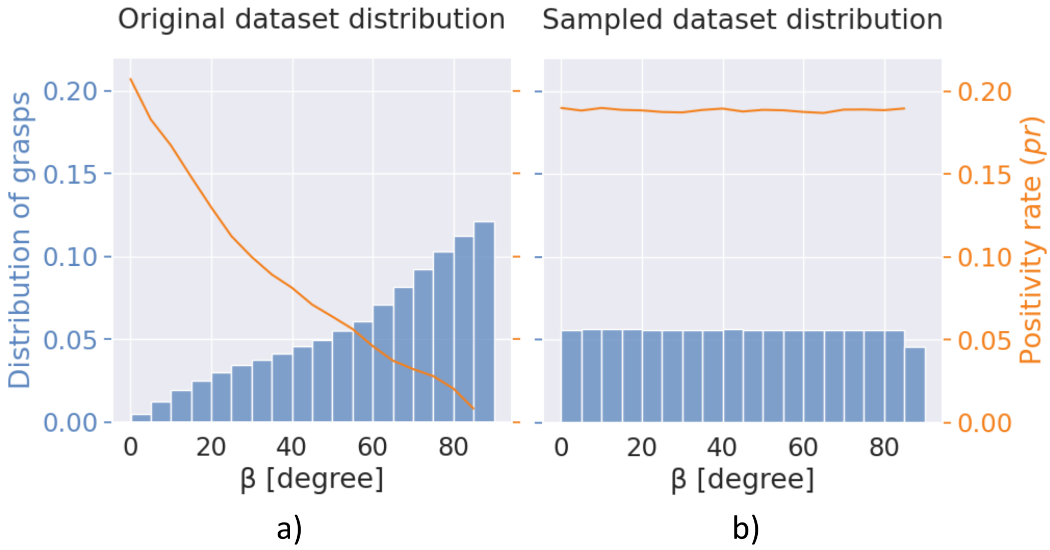


Figure 3.5: Positivity rate pr and grasp distribution over β a) before and b) after sampling.

excluded grasps with a negative approach axis pointing towards the camera origin, as they are more likely to be occluded by the object.

Both the positivity rate pr and the total number of grasps across the camera and grasp configurations affect model performance. The native distribution in the rendered dataset can be seen in Figure 3.5 a). The positivity rate pr declined with increasing β, since grasps with a higher β tend to collide with the table more often. The native, rendered dataset had a varying positivity rate with an overall pr of $pr = 0.06$, while the positivity rate for DexNet2.0 is $pr_{DexNet} = 0.19$.

In the rendered dataset, the number of grasps also increased with increasing β, due to the random sampling of the gripper rotation (around ω) when setting the grasp orientation in Algorithm 3.1, lines 6-9. When the native distribution of the dataset was used for training, our tests yielded models that suffered in performance due to this imbalance, for example, classifying all grasps with a high β as negative due to an insufficient balance of ground-truth positive examples.

For the reasons above, we adjusted the positivity rate pr by sub-sampling negative/positive grasps based on the sampling rate $sample\_rate_x$. The goal of adjusting the positivity rate was to match VG-dset's *pr* to the positivity rate in DexNet2.0 (with $pr_{DexNet} = 0.19$). Since *pr* is dependent on β, we calculated the sampling rate for sub-sampling negative grasps as:

$$sample\_rate_{neg}(\Delta\beta) = \frac{\frac{100}{19} \times pr(\Delta\beta) - pr(\Delta\beta)}{1 - pr(\Delta\beta)}$$

where, $\Delta\beta$ was varied in steps of 5°. The sub-sampling rate for positive grasps was set to $sample\_rate_{pos} = \frac{1}{sample\_rate_{neg}}$. Subsampling rates of $sample\_rate_x > 1$ were set to 1, as we did not introduce duplicates or render new images during the dataset sampling stage. We then randomly skipped positive and negative grasps based on their sampling rate for the given β.

Additionally to the imbalance of the positivity rate, the number of grasping data points for different camera-grasp configurations influenced the network's performance. For example, substantially more grasping data points had a large angle β between the table normal and the grasp approach axis due to the 3D representation of the sampled ground-truth grasps. However, this led to an imbalanced distribution and comparably few examples of top-grasps. This made it difficult for a model trained on such an imbalanced dataset to provide good predictions for top-grasps. To counter that imbalance in grasp distribution over the camera-grasp configurations, we sub-sampled the remaining grasps after adjusting the positivity rate *pr* to have a uniform sample size over φ and β. The resulting dataset VG-dset had balanced positivity rates and distribution of grasps as shown in Figure 3.5 b).

The process of dataset sampling substantially reduced the size of our rendered dataset from 130.8 million to 7.2 million grasping datapoints. However, the 7.2 million grasping datapoints are still more than those available in the DexNet2.0 dataset with 6.7 million [11]. In general, any dataset should include enough samples to prevent a supervised model from overfitting the presented samples during training. To assess the risk of overfitting when using different dataset sizes, we conducted an ablation study in Section 3.5.3 by training models on datasets with a different number of training grasps. We found that training VGQ-CNN on datasets with 1 million or more grasping data points showed a constant performance. In comparison, training on a dataset with fewer than 1 million grasping data points showed signs of overfitting. Hence, we concluded that the 7.2 million grasping

	Total	Train	Validation	Test
# Images	1.3m	1.05m	0.12m	0.13m
# Grasps	7.2m	5.82m	0.67m	0.72m

Table 3.2: Number of images and grasps in the train, validation and test split of VG-dset.

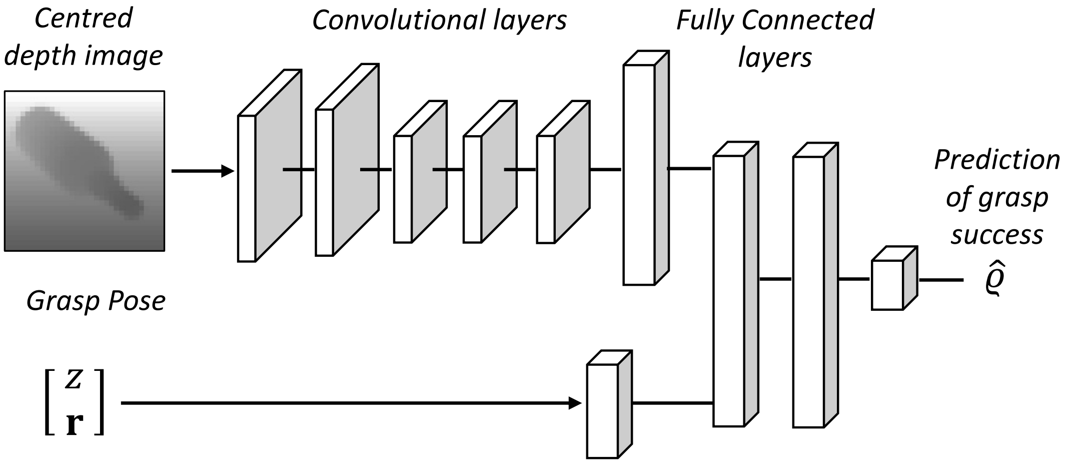


Figure 3.6: Architecture of VGQ-CNN. The model consists of an image stream (upper) and a pose stream (lower) combined in a merge stream to predict grasp success. The input image is a 32 × 32 pixel depth image, and the pose is given by the distance z between the camera and the gripper TCP, and the orientation quaternion, $\mathbf{r} \in R^4$, where $|\mathbf{r}| = 1$. It outputs a prediction of grasp success $\hat{\varrho} \in [0, 1]$.

data points in VG-dset were sufficient for training VGQ-CNN and could even be sub-sampled further if necessary.

We divided both DexNet2.0 and VG-dset into an object-wise training, validation and test split of 80-10-10, using the same objects for the test sets in both DexNet2.0 and VG-dset. This allowed us to compare performance between VGQ-CNN and GQ-CNN on the same test objects in Section 3.5.1. An overview of the split composition with the number of images and grasps can be seen in Table 3.2.

3.4 Model architecture

We used VG-dset to train VGQ-CNN. The architecture of VGQ-CNN, a CNN with two parallel streams, is shown in Figure 3.6. To represent 6-DoF grasp poses for VGQ-CNN, we made two key adjustments compared to other image-based 4-DoF grasp quality and grasp proposal models: encoding the grasp orientation as a quaternion and providing the grasp orientation explicitly as an input to the model.

In DexNet2.0 [11], as well as in other image-based 4-DoF grasp predictors [9, 10, 12, 23, 77], the TCP of each grasp proposal is projected into the image to be represented in the 2-D image plane, for example, as an oriented rectangle. In these approaches, the grasp orientation is constrained to some arbitrary axis like the camera principal ray and not explicitly specified as a model input.

Representing 6-DoF grasps with grasp orientations not necessarily aligned with an arbitrary axis (such as the camera principal ray) requires an alternative grasp specification for our model. We defined the full 6-DoF grasp representation using a 3D position encoded as the grasp image coordinates (u, v) and the distance z between the camera and the gripper TCP, as well as a 3D orientation encoded as a quaternion $\mathbf{r} \in R^4$. While we used quaternions as orientation representation (e.g. similar to VGN [16]), this could be exchanged with other representations such as Euler angles. Given that the input quaternion represents an orientation, it is constrained to be a normalised unit quaternion, with $|\mathbf{r}| = 1$.

We implicitly specified the grasp image coordinates, (u, v), in the image's centre. This was achieved by projecting the gripper TCP into the image $\mathcal{I}$, cropping the image around the grasp coordinates (u, v) and subsequently resizing it to a 32×32 pixel image. The two remaining variables of the 6-DoF grasp, the distance of the TCP to the camera, z, and the grasp quaternion, $\mathbf{r}$, were provided to VGQ-CNN as an input in the pose stream, see Figure 3.6.

We kept the general model structure of GQ-CNN, consisting of parallel image- and pose-streams with convolutional and fully connected layers. The two streams were combined into a single stream towards the head end of the model. Apart from including the quaternion $\mathbf{r}$ as an extra input, we added an additional fully connected layer with 1024 nodes before the last layer. This extra layer added representational capacity to account for the additional grasp information in the pose stream and increased model performance by 4% as shown in Section 3.5.3. VGQ-CNN has a total of 6.5 million trainable parameters, as compared to GQ-CNN which has a total of 5.4 million trainable parameters.

3.5 Experiments

We trained both VGQ-CNN and GQ-CNN for 150 million training iterations on VG-dset and DexNet2.0, respectively. One epoch of training on a DexNet2.0-sized dataset is equivalent to approximately 6 million training iterations. The training took approximately 23 hours to complete on a single NVIDIA RTX 2060 GPU. We used a stochastic gradient decent optimiser with a momentum rate of 0.9, a base learning rate of 0.001 decaying every 4 million iterations by 0.95, a $L2$-regularisation of 0.0005 and sparse categorical cross-entropy loss.

In standard fashion, the depth images and z values were normalised before being input into the model. The values of the unit quaternion $\mathbf{r}$ were not normalised again for the model input since they naturally ranged within $-1, 1$. We checked performance on the validation split every 1 million iterations. We used the balanced accuracy metric to evaluate the resulting models, which is calculated as the mean of True Positive Rate (TPR) and True Negative Rate (TNR).

$$\begin{aligned} TPR &= \frac{TP}{TP + FN} \times 100\% \\ TNR &= \frac{TN}{TN + FP} \times 100\% \\ Balanced\ Accuracy &= \frac{TPR + TNR}{2} \end{aligned}$$

with TP being True Positives, i.e., ground-truth positive grasps classified as positive, FN being False Negatives, i.e., ground-truth positive grasps classified as negative, etc.

We motivate using the balanced accuracy metric due to the high imbalance in the ground-truth data. A significant imbalance between positive and negative grasps can make judging performance based on a standard accuracy metric difficult. For example, a model classifying all grasps as negative achieves an accuracy of 90% if a dataset has a positivity rate of $pr = 0.1$ and 10% if a dataset has a positivity rate of $pr = 0.9$. Using the balanced accuracy metric instead yields a balanced accuracy of 50% with $TPR = 0\%$ and $TNR = 100\%$ in both cases. We chose the best model according to the balanced accuracy metric of the validation results for all our evaluations.

In the following subsections, we investigated the performance of VGQ-CNN on a test split of VG-dset and compared the performance between GQ-CNN and VGQ-CNN on a dedicated test set only including top-grasps and overhead camera poses similar to DexNet2.0 to provide a fair basis for comparison. Our results showed the robustness of VGQ-CNN to varying camera poses, demonstrating that VGQ-CNN does not require retraining when moving the camera within the range of the parameters specified in Table 3.1. In a set of ablation studies, we showed the effects of dataset size and the extra fully-connected layer on VGQ-CNN.

3.5.1 Overall performance

To measure the overall performance, we evaluated VGQ-CNN on the test split of VG-dset (0.7 million grasps), and GQ-CNN on the test split of DexNet2.0 (0.7 million grasps). In addition, to compare the performance of VGQ-CNN and GQ-CNN, we evaluated them on a separate, new dataset with 0.3 million grasps named Top Grasp Testset (TG-tset). The test samples in TG-tset represent the intersection of possible configurations in VGQ-CNN and GQ-CNN, enabling us to compare both algorithms for these restricted input scenarios. The objects in TG-tset have not been seen by VGQ-CNN or GQ-CNN during training since they are not included in the training or validation sets of DexNet2.0 and VG-dset. TG-tset was rendered using the grasp and camera parameters of DexNet2.0 as detailed in Table 3.1.

Due to the different input requirements in the models, images were centred and rotated before being input to GQ-CNN and only centred for use in VGQ-CNN.

Model	Evaluation dataset	TPR	TNR	Balanced accuracy
GQ-CNN	DexNet2.0	70.0%	89.1%	79.5%
VGQ-CNN	VG-dset	73.9%	90.2%	82.1%
GQ-CNN	TG-tset	63.6%	89.6%	76.7%
VGQ-CNN	TG-tset	64.6%	85.7%	75.2%

Table 3.3: Results for testing VGQ-CNN and GQ-CNN on their own as well as a common evaluation set.

Note that the collision checking for VG-dset differed slightly from the approach used for DexNet2.0. In DexNet2.0, grasps were classified as collision-free if any of four linear approaches within ±10° around ω were collision-free. In VG-dset, grasps were classified as collision-free only if a linear approach along the grasp approach axis (grasp z-axis) was collision-free. To compare GQ-CNN and VGQ-CNN, we excluded grasps from TG-tset which differ in the outcome of the collision checking by these two methods, corresponding to 0.3% of all generated grasps.

We calculated TPR, TNR and balanced accuracy for all evaluations. The results can be seen in Table 3.3. The TNR for all evaluations was higher than the TPR, which reflects the positivity ratio pr in the training datasets, i.e. more negative than positive ground-truth datapoints, and led to a more conservative grasp quality estimation. We note that the balanced accuracy result for GQ-CNN with 79.5% was lower than the accuracy of 85.7% reported in [11]. The (unbalanced) accuracy of GQ-CNN in our experiments was 85.5%, exhibiting a very similar performance to the original results and thereby validating the usage of the model trained with the reference implementation[2] as a good representation of GQ-CNN's capabilities.

VGQ-CNN exhibited robust performance on VG-dset with a balanced accuracy of 82.1%, while being able to generalise to a wide range of camera poses and 6-DoF grasp poses. On the much-restricted range of camera and grasp poses in TG-tset, VGQ-CNN achieved competitive performance with a balanced accuracy of 75.2% compared to GQ-CNN's 76.7%, showing that the adjustments in training data

[2] `https://github.com/BerkeleyAutomation/gqcnn`

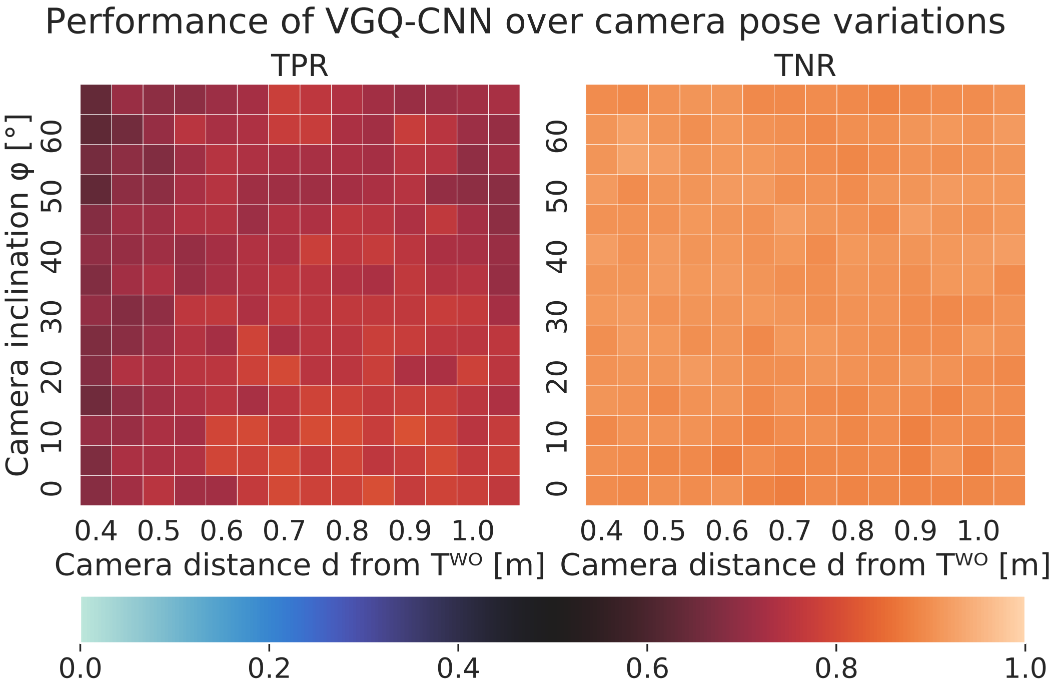


Figure 3.7: Performance of VGQ-CNN over camera pose variations with TPR (left) and TNR (right), tested on an object-wise test split of VG-dset.

and grasp representations did not result in a substantial negative impact on the performance of VGQ-CNN when tested on top-grasps with overhead cameras.

3.5.2 Performance over the camera parameter space

In addition to being comparable to GQ-CNN on the top-grasps in TG-tset, VGQ-CNN also performed well across the varied camera poses of VG-dset. We specifically examined the TPR and TNR in relation to the spherical coordinate variables of the camera, the radial distance d and the inclination angle φ and show the results in Figure 3.7.

As in the overall results, the TNR was constantly higher than the TPR, as expected due to the positivity rate pr in the training data. Furthermore, both TNR and TPR were consistent over the range of camera poses, with no visible outliers or trends. Over the full target range of camera poses with steps of $\Delta d = 0.05m$ and $\Delta\varphi = 5°$, VGQ-CNN attained a mean $TNR = 90.2\%$ with a standard deviation of

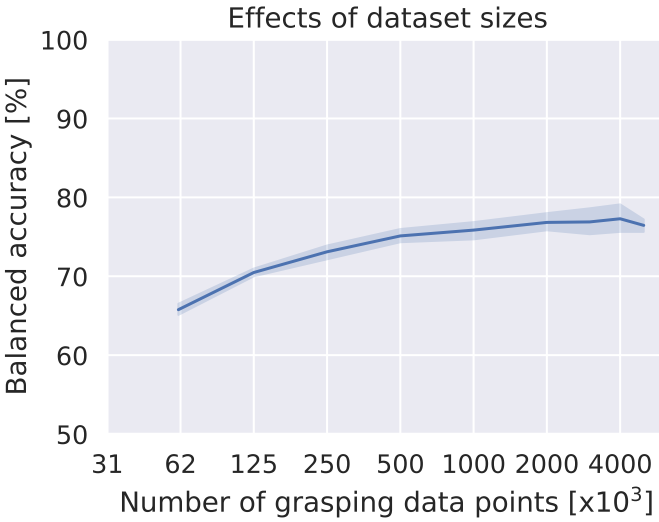


Figure 3.8: Influence of the dataset size with the number of grasping data points used for training VGQ-CNN on model performance. A low number with fewer than 1 million grasping data points exhibits signs of overfitting with a reduced balanced accuracy on the held-out test set.

0.9% and a mean $TPR = 73.9\%$ with a standard deviation of 3.5%.

3.5.3 Ablation studies

For further insight into the model performance, we conducted a set of ablation studies on the effect of the dataset size on VGQ-CNN and the extra fully-connected layer in VGQ-CNN. Note that all models were trained from scratch for 10 million training iterations.

Each model configuration was trained 8 times with the shading in Figure 3.8 corresponding to the 95% confidence interval over the results to account for the repeatability of the training procedure. The datasets for the experiments of the dataset size influence were sampled uniformly from VG-dset. While the training and validation data differed for all models in the dataset size experiments, they were tested on the same test split of VG-dset to ensure comparability.

Figure 3.8 shows how the number of training samples affected the trained model performance. The minimum number of training samples for reasonable model performance is relevant, especially when evaluating different sampling strategies in Section 3.3.2, as they can result in fewer training samples being available. When training the model with fewer than 1 million grasping data points, the model showed signs of overfitting, where the accuracy on a test set of 62 thousand grasping data points dropped to roughly 65%. Training VGQ-CNN with 1 million grasping data points or more exhibited stable results with a mean balanced accuracy of 77%. Note that a balanced accuracy of 50% can be achieved by classifying all grasps as positive or negative and hence was chosen as the minimum visible balanced accuracy in Figure 3.8 and Figure 3.10.

Adding a second fully connected layer with 1024 nodes before the last layer, as described in Section 3.4, increased balanced accuracy from a mean of 72.5% with a standard deviation of 1.8% to a mean of 76.7% with a standard deviation of 1.1%.

3.6 Fast-VGQ-CNN

While discriminative methods like VGQ-CNN have been shown to work for generating successful grasp proposals, they typically incur high latency when proposing grasps. This can be a major disadvantage for deployment on real robots, as explained in Section 3.1. We investigated an alternative model architecture for time-critical applications with the Fast Versatile Grasp Quality Convolutional Neural Network (Fast-VGQ-CNN). Our alternative model architecture can evaluate multiple grasps in a single forward pass, which we show can decrease latency when evaluating multiple grasps on a given object.

3.6.1 Model architecture

Figure 3.9 shows the differences in model architecture between VGQ-CNN and Fast-VGQ-CNN, which enabled the latter to predict multiple grasp proposals in a single forward pass.

Instead of cropping the depth images such that the TCP is centred in the image, as for VGQ-CNN, we decoupled the grasp centre from the image by turning the image coordinates of the grasp centre into explicit inputs to the model. During training, we applied random cropping [150] of the image around the grasp centre,

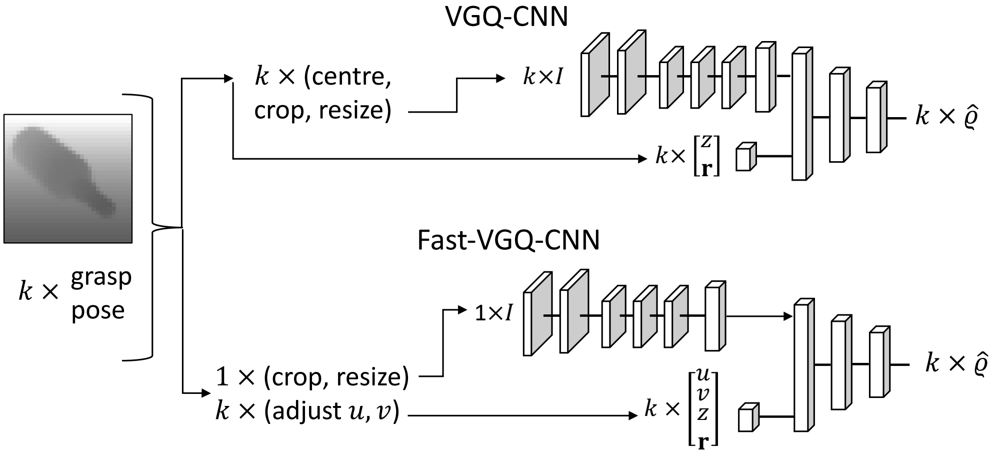

VGQ-CNN
k × (centre, crop, resize)
k×I
k×$\begin{bmatrix} z \\ \mathbf{r} \end{bmatrix}$
$k \times \hat{Q}$
k × grasp pose
Fast-VGQ-CNN
1×I
1 × (crop, resize)
k × (adjust u, v)
k × $\begin{bmatrix} u \\ v \\ z \\ \mathbf{r} \end{bmatrix}$
$k \times \hat{Q}$

Figure 3.9: Comparison of pipelines for grasp quality evaluation with VGQ-CNN and Fast-VGQ-CNN. For VGQ-CNN, k images have to be generated from a single image and input to the model with k poses. For Fast-VGQ-CNN, the image is processed once and input into the image stream of the model. The output is then input to the rest of the model together with k poses.

causing the grasp not to be centred in the image. We specified the full 6-DoF grasp pose with $g \in (u, v, z, \mathbf{r})$ as an input to the pose stream of Fast-VGQ-CNN.

Applying random cropping to the image ensured grasp- and object placement variety within the image during training and resulted in a model that can evaluate grasp proposals for varying grasp locations within the image. The coordinates of the grasp centre in the cropped and resized image (u, v) were sampled uniformly.

By introducing the new grasp representation, as well as the randomised grasp image coordinates (u, v) during training, we could split the image stream, with its high-dimensional and computationally intensive convolutional layers, from the rest of the model during inference. The image stream could be used as a shared encoder for multiple grasps visible in that image, processing the image once and using the output for a batch of k grasp poses. Utilising shared model layers to reduce run-time has been proposed and used for other model structures, e.g. for various object detection models [115].

The new architecture also changed the requirements for image pre-processing. For both VGQ-CNN and Fast-VGQ-CNN, the starting point is a single 300×300 pixel

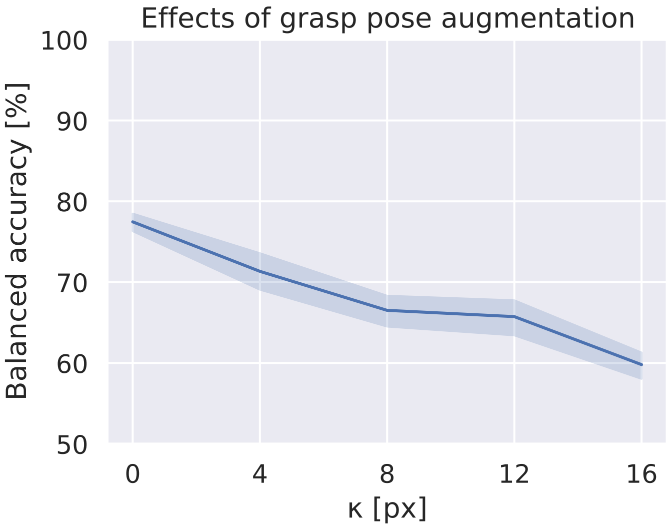


Figure 3.10: Influence of the maximum grasp distance $\kappa = max(|u|, |v|)$ to image centre on model performance.

depth image and a variable number of grasp poses k, as indicated in Figure 3.9. With VGQ-CNN, for each of the k grasps, the image had to be centred, cropped and resized, resulting in k centred 32×32 pixel depth images and k grasp poses $g \in (z, \mathbf{r})$. In contrast, for Fast-VGQ-CNN, the original image had to be cropped and resized just once, assuming all k grasp samples can be visible in a single 32×32 pixel depth image. The grasp centre coordinates (u, v) for each of the k grasp poses were determined relative to the single image. Therefore, the model input comprised a single 32×32 pixel depth image, and k grasp poses $g \in (u, v, z, \mathbf{r})$. In both pre-processing pipelines, the image and pose values were normalised, excluding the quaternion, which was normalised as a unit quaternion with $|\mathbf{r}| = 1$ already.

For prediction, the resulting images and poses were input to the model. For VGQ-CNN, the k depth images $\mathcal{I}$ and k grasp poses went into a single, multi-stream model. For Fast-VGQ-CNN, the single, depth image $\mathcal{I}$ went into the shared encoder while the resulting tensor, in combination with the k grasp poses, was input to the remaining fully connected layers of the model.

3.6.2 Performance

We hypothesised that there would exist a trade-off between the speed-up that can be achieved with Fast-VGQ-CNN and its performance compared to VGQ-CNN. This inherently relates to the spread of the distribution of grasp centre coordinates in the image. We parameterised the range over which the grasp centre coordinates (u, v) were uniformly sampled in the image by the maximum distance $\kappa = max(|u|, |v|)$ between the grasp centre and the image centre.

We varied κ for training Fast-VGQ-CNN and evaluated its performance to investigate the trade-off between the number of grasps that could be presented in an image, which is connected to the run-time of that model for grasp proposal, and its balanced accuracy. The base dataset for these experiments was VG-dset, with different values of κ applied to the same train-validation-test split. All models were trained from scratch for 10 million training iterations, in the same manner as the models from our ablation studies in Section 3.5.3.

Figure 3.10 shows a reduction in balanced accuracy as the displacement of the grasp centre in the image increases. For $\kappa = 0px$, Fast-VGQ-CNN reached a balanced accuracy of 77%, similar to VGQ-CNN's performance with the same number of training iterations shown in our ablation studies in Section 3.5.3. Fast-VGQ-CNN's performance dropped to 60% for grasps uniformly distributed over the full range defined by $\kappa = 16px$.

3.6.3 Speed

This reduction in performance with $\kappa = 16px$ should come with a speed-up based on the pipeline adaption, which substantially reduces the number of computations for a single image. To test this, we measured the inference time of VGQ-CNN and Fast-VGQ-CNN on a machine with a 2.6 GHz Intel Core i7-10750 CPU and an NVIDIA GeForce RTX 2060 GPU. Tensorflow 1.15 was used for all experiments.

We ran the tests with differing numbers of grasps, $k \in \{32, 64, 96, 128\}$, as batch sizes and measured inference as the mean over 1000 runs. All image pre-processing steps were implemented with the pillow[3] library in Python. While the inference

[3] https://pillow.readthedocs.io/

	Prediction			
k	32	64	96	128
CPU VGQ-CNN	41	72	103	138
CPU Fast-VGQ-CNN	**11**	**11**	**12**	**12**
GPU VGQ-CNN	8	12	17	21
GPU Fast-VGQ-CNN	8	**8**	**8**	**8**
	Pre-processing			
VGQ-CNN	7	14	21	28
Fast-VGQ-CNN	**0.3**	**0.5**	**0.6**	**0.8**

Table 3.4: Inference time [ms] for VGQ-CNN and Fast-VGQ-CNN.

times could be reduced further by implementing all code in a lower-level programming language, we were mainly interested in the relative speed-up of Fast-VGQ-CNN compared to VGQ-CNN. The resulting inference times for prediction and pre-processing can be seen in Table 3.4.

The speed-up of Fast-VGQ-CNN relative to VGQ-CNN increased as we raised the number of grasps predicted per batch, k. The Fast-VGQ-CNN pre-processing pipeline is up to 35 times faster than the pre-processing for VGQ-CNN, taking just $0.8ms$ to preprocess 128 grasps.

Of special interest when working with devices without a specialised GPU, Fast-VGQ-CNN could predict 128 grasps within $12ms$, while VGQ-CNN exhibited a substantial increase in runtime for the prediction of more grasps taking $138ms$ to predict 128 grasps on a CPU. When combined with an efficient grasp sampling strategy, Fast-VGQ-CNN could enable real-time grasping.

3.7 Discussion

VGQ-CNN was based on GQ-CNN with adaptations to the training data and grasp representation to extend the capabilities of the model from 4-DoF top-grasps and overhead cameras to 6-DoF grasp poses and a wide range of camera poses. VGQ-CNN exhibited a consistent performance over the space of camera pose variations when varying the radial distance of the camera d and the inclination angle φ in Figure 3.7. On top of that, when tested on top-grasp viewed from overhead cameras, the performance of VGQ-CNN was comparable to GQ-CNN, as shown in Table 3.3. This showed that the aforementioned adaptions did not negatively impact performance on the baseline.

When training VGQ-CNN, we identified sensitivities to certain dataset distributions that could prohibit the successful training of models. In particular, we found the positivity rate and the number of grasps in relation to the angle between the table normal and the grasp approach, β, to influence the model outcome substantially. As shown in Section 3.3.2 and Figure 3.5, the native, unsampled dataset featured an increase in the number of grasps and a decrease of positivity rate pr when increasing β. This led to a trained model predicting all grasps as negative due to an insufficient number of ground-truth positive samples.

By balancing both the positivity rate and the number of grasp samples over the range of β, we managed to achieve consistent performance over the whole parameter space. This showed a sensitivity of VGQ-CNN regarding dataset balance. Such a sensitivity could also be mitigated by introducing a weighted loss function to VGQ-CNN. However, due to the multi-dimensional imbalances of the positivity rate over both β and φ in the base dataset, the resulting loss function would be substantially more complicated than the current binary cross-entropy loss used for VGQ-CNN. Hence, we decided to adjust the training data for this work instead.

Overall, our results showed that by using appropriate training data, VGQ-CNN can robustly handle the different viewpoints represented in the depth images. Ultimately, this enables using VGQ-CNN from different camera viewpoints, which are flexible and not limited to being directly above the workspace. Scenarios in which this is valuable can include those using multiple cameras or having to move the camera in a scene. In particular, varying camera viewpoints are encountered by mobile manipulators where the transform between the workspace and the camera (mounted on the robot) $\mathbf{T}_c^{\mathrm{wo}}$ is not specified initially.

VGQ-CNN presents a quality prediction method for evaluating grasp candidates individually in a discriminative grasp proposal method. Such discriminative methods often have a disadvantage in run-time compared to generative methods, which directly regress to grasp proposals, since discriminative methods have to sample and evaluate each grasp candidate individually [5, 8]. In order to minimise the inference time of discriminative methods, one has to implement efficient algorithms for both sampling and evaluating candidates. We showed in Section 3.6 how a speed-up of VGQ-CNN can be achieved by using a shared image encoder and making coordinates of multiple grasp candidates an explicit variable in the model input.

Although we showed that this reduced inference times for Fast-VGQ-CNN (see Table 3.4), it also reduced the balanced accuracy of the model when increasing κ and thereby the number of grasp candidates that can be presented in a single forward pass (see Figure 3.10). There is an open question about the balance of the accuracy and the run-time of such algorithms and whether both can be improved simultaneously. In addition to achieving this balance, using VGQ-CNN or Fast-VGQ-CNN in a grasping pipeline also required an efficient sampling approach in order to generate acceptable inference times. While there are different sampling algorithms available for 6-DoF grasps [21, 24, 38, 46, 81, 151], it seems that overall generative methods are superior in terms of computational efficiency, e.g. [16, 17, 20] and research has mostly shifted towards such generative methods in recent years [5, 8].

3.8 Conclusion

In this chapter, we presented VGQ-CNN, a model for predicting the quality of 6-DoF grasps as observed from a wide range of camera poses. Removing limitations on grasp orientation and camera pose in previous methods [9, 11, 14, 23, 62, 73] and available grasp datasets [10, 11, 77] allowed for VGQ-CNN to predict the quality of 6-DoF grasps from a wide range of camera poses. The transform between the camera and the object workspace $\mathbf{T}_c^{\text{wo}}$ does not need to be defined in advance and is not limited to configurations like overhead cameras. This can be especially useful when working with mobile manipulators, which change the relationship between the camera and the object when they move.

We showed in Section 3.5.1 how VGQ-CNN exhibited consistent, high performance for varying camera viewpoints while not reducing performance for scenes with top-grasps and overhead camera poses. We take three important insights from these results: (i) A depth-image-based method using a CNN can adapt to different camera viewpoints. These different viewpoints change the appearance of the depth images significantly, as can be seen when comparing the data in VG-dset to the overhead depth images in DexNet2.0 in Figure 3.4. (ii) Encoding the 6-DoF grasp pose in the form of an implicit grasp centre in the image, a distance to the grasp and, most importantly, a quaternion $\mathbf{r} \in R^4$, where $|\mathbf{r}| = 1$ is a suitable representation when learning the quality of grasp candidates. (iii) The balance of the dataset in terms of positivity rate and relative grasp angles has to be taken into consideration in order to ensure successful training of a versatile grasp quality prediction model like VGQ-CNN. We discussed the reason for this sensitivity in Section 3.7 and implemented sampling strategies in Section 3.3.2 to enable successful training of VGQ-CNN.

Ultimately, due to the run-time disadvantages of discriminative approaches, we did not pursue these approaches further. Instead, we used the findings in this chapter to construct a generative grasp proposal method, as described in the following chapter.

CHAPTER 4

GP-net: Flexible Viewpoint Grasp Proposal

Moving from a discriminative grasping approach, like the VGQ-CNN presented in Chapter 3, to a generative approach can have major advantages in run-time [5, 7] because each grasp does not have to be sampled and evaluated individually. Sufficiently fast run-time can enable closed-loop control [12]. While this advantage already exists in 4-DoF tabletop scenarios, it becomes substantially more pronounced in the context of 6-DoF grasp proposals from unknown camera viewpoints due to the increased search space.

Such unknown camera viewpoints in relation to the workspace are encountered in domestic environments, as described in Section 1.2. Applying a 6-DoF grasp proposal algorithm then often requires an estimation of the camera-to-workspace transform $\mathbf{T}_c^{\text{wo}}$ [16, 17] or object segmentation to guide the grasp proposals [20, 24]. The pre-processing steps required to enable these solutions in setups with unknown camera viewpoints and unknown objects increase the runtime of algorithms and add a reliance on the accuracy and reliability of such pre-processing steps.

In this chapter, we present a generative approach to 6-DoF grasp proposal from flexible viewpoints with a FCNN model called the GP-net. It can propose 6-DoF grasps for single objects in the entire scene representation without needing an additional workspace definition. In other words, the model learns to differentiate between graspable and non-graspable surfaces and does not require the definition of a grasping workspace or object segmentation to guide grasp proposals to the object. For example, GP-net can be used to allow mobile manipulators to grasp unknown objects from planar surfaces like tables or the floor.

Similar to some depth-image-based generative 4-DoF approaches [12, 23], GP-net encodes grasp proposals in the form of a dense output tensor which specifies a prediction of grasp success $\hat{\varrho}$, orientation $\hat{\mathbf{r}}$ and width $\hat{\mathrm{w}}$ for each pixel in the input image. It thereby presents a solution for end-to-end learning for generating grasp proposals solely based on the depth image of a camera. We utilise a contact-based grasp anchor for the grasp representation as suggested by Sundermeyer et al. [20], which we found to increase the grasp performance substantially in our ablation studies (see Section 4.5.3).

In a real-world experiment using the 49 objects from the EGAD evaluation set [74] with a PAL TIAGo manipulator, we compared GP-net against VGN and GPD. We chose those models based on their usage as benchmark approaches across a wide number of papers in the field [17, 21, 38, 46]. GP-net achieved a grasp success of 54.4%, performing better than the VGN with 51.6% and better than the GPD with 44.2%. Importantly, GP-net achieved those results without requiring any of the pre-processing steps required for VGN and GPD. This enables GP-net to be used in scenarios with unknown camera-workspace transforms and without relying on correct results of object segmentation techniques. Additionally, GP-net was competitive in terms of runtime, especially when considering the pre-processing steps required for proposing grasps with VGN and GPD in our experiment.

The contributions presented in this chapter can be summarised as follows:

- GP-net: A model to propose 6-DoF grasps for single objects with a parallel-jaw gripper from unknown camera viewpoints without specifying the workspace or running table segmentation to filter grasp poses.
- A ROS package to run GP-net and generate grasp proposals from a depth camera.

- A procedure for generating a dataset to train GP-net or alternative model architectures featuring ground-truth grasps observed from unknown and flexible camera viewpoints. We make our code and dataset publicly available[4] so it can be used to train GP-net to adapt to different robots or grippers.

4.1 Related work

GP-net is a model to generate 6-DoF grasp proposals from flexible viewpoints and can be used for single objects on planar surfaces. Multiple potential solutions exist to such a problem, but these can only be rated against each other by considering their advantages, disadvantages and performance on a common benchmark in simulated or real-world experiments. As described in Section 2.2.2, comparing grasping algorithms to each other is particularly difficult if they have not been evaluated using the same objects and hardware in a similar setting. To assess the performance of our algorithm, we therefore conduct our real-world experiment with GP-net and two established grasp proposal methods: GPD [24] and VGN [16].

The GPD package [24] was one of the first data-driven methods to be directly applied to propose 6-DoF grasp poses based on point clouds. It identifies a region of interest in the point cloud, samples grasp proposals from points within the region of interest, encodes them in a multi-channel image, and finally uses a CNN to predict the grasp quality. It is widely used to compare against other algorithms [16, 17, 21, 38, 46], possibly due in part to the availability of a reference implementation in ROS since 2017.

When sampling the grasp proposals, one could use all points in the point cloud or only indexed points based on a region of interest, e.g. the points forming the surface of an object. When applying GPD to real-world scenarios, identifying such a region of interest in the point cloud ensures the grasp proposals fall on the objects, not background and support surfaces like the table. When using GPD in our experiment, we indexed the object points based on their position in relation to the segmented table plane. The discriminative sampling nature of the algorithm

[4] `https://aucoroboticsmu.github.io/GP-net/`

led to high latencies in the 6-DoF search space, taking an average of $14.6s$ to propose grasps in our experiment.

The second reference model used in our experiment, VGN [16], proposes grasps for objects in clutter using a TSDF representation of the workspace. Originally, the TSDF is computed by integrating a stream of depth images acquired by a wrist-mounted depth camera, where the camera follows a pre-defined scan trajectory around a defined workspace before generating the grasp proposals. In our experiment in Section 4.4, we built the TSDF from a single viewpoint instead to accommodate our robot specifics with a head-mounted camera. Once the TSDF is built, grasp proposals can be predicted based on a voxel representation of the TSDF with a resolution of $1cm^3$. While a single forward pass on the voxel grid only took $10ms$ in the original paper [16], this did not include the initial scanning procedure or any pre-processing steps required to create the TSDF of the object workspace.

In the original paper, VGN achieved good results in a real-world experiment using household objects within a tabletop scenario, which was subject to some constraints. The workspace of the fixed manipulator had to be known, and the TSDF modelled a $30 \times 30 \times 30cm^3$ workspace. When applying VGN to scenarios with an unknown camera-workspace transform $\mathbf{T}_c^w$ such as with a mobile manipulator, the approximate position of the object would have to be estimated in advance to compute the TSDF. Furthermore, if no wrist-mounted camera is available, the TSDF would have to be built using an alternative sensor, e.g. a head-mounted camera, reducing the possible camera viewpoints and thereby potentially negatively affecting the quality of the TSDF.

An advantage of GP-net over GPD and VGN is the ability to run it without pre-processing steps to identify a region of interest or a workspace transform $\mathbf{T}_c^w$. In order to rate the performance of the three models in the same scenario, we performed the minimum necessary pre-processing steps to run GPD and VGN and compared all algorithms in the same real-world experiment.

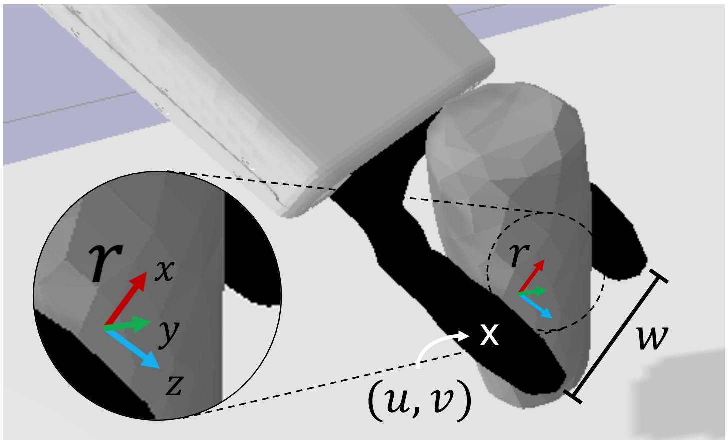


Figure 4.1: Visualisation of the grasp representation utilised in GP-net, which is anchored to the visible grasp contact with the contact coordinates in the image (u, v), grasp orientation $\mathbf{r}$ and grasp width w.

4.2 Grasp Proposal Network

We consider the problem of proposing 6-DoF grasps for a parallel-jaw gripper using a depth camera. The environment consists of a planar surface, on top of which we define a workspace coordinate frame $\mathbf{T}^{\mathrm{wo}}$ for description and camera pose sampling purposes. The pose of the depth camera frame is defined in spherical coordinates relative to the workspace $\mathbf{T}^{\mathrm{wo}}$ such that its x-axis lies parallel to the planar surface and its principal ray is pointed towards the workspace origin. The spherical coordinates are defined using the radial distance to the workspace origin d, the inclination angle φ relative to the workspace z-axis and the azimuth angle θ rotating around the workspace z-axis. The transform between the workspace and the camera frame $\mathbf{T}_c^{\mathrm{wo}}$ is unknown within the sampling bounds for the spherical coordinates as defined in Table 3.1.

A single object $o \in \mathcal{O}$ is placed in a stable resting pose $s \in SE(3)$ [149] in the workspace such that it is visible in the camera image. The goal is to propose a set of collision-free 6-DoF grasps $g \in \mathcal{G}$ to pick up the object o with a parallel-jaw gripper. The grasps are proposed based on the robot's surroundings as represented in a depth image $\mathcal{I}$.

Similar to Sundermeyer et al. [20], we anchored the grasps for the grasp representa-

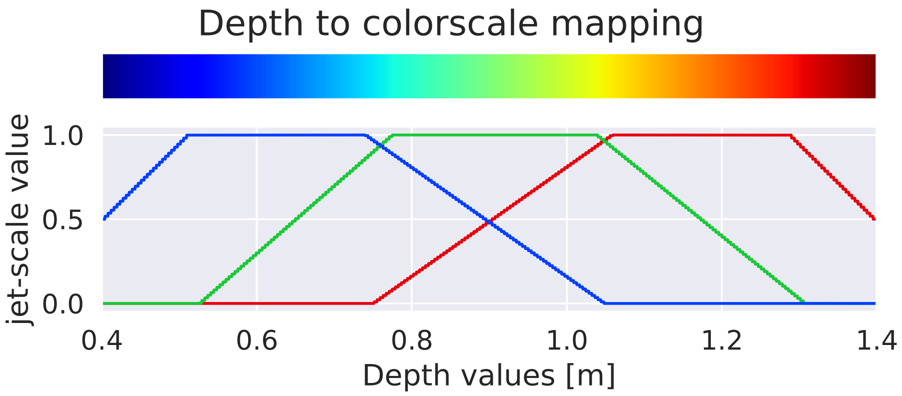


Figure 4.2: Mapping depth values to a jet colourmap with individual RGB values in the resulting normed jet-scale for GP-net.

tion to the contact point between the gripper and the object. Each pixel (u, v) in a depth image $\mathcal{I}$ described a potential grasp contact, i.e. the contact point between a gripper plate and the object during grasp execution. The full ground-truth grasp is defined as $g \in (u, v, \mathbf{r}, \mathrm{w}, \varrho)$, with the contact coordinates (u, v), the orientation of the grasp in camera coordinates $\mathbf{r}$, the width of a grasp w and the grasp success ϱ. The orientation of the grasp is given in the form of a unit quaternion $\mathbf{r}$, where $\mathbf{r} \in R^4$, and $|\mathbf{r}| = 1$. A visualisation of the grasp representation is depicted in Figure 4.1.

We based the model architecture for GP-net on a ResNet-50 model pre-trained on ImageNet. Since the rendered depth image $\mathcal{I}$ consists of a single channel and the pre-trained ResNet-50 architecture uses three input channels, we applied a jet colourmap to the depth values and represented the resulting input as a 3-channel image to the model as described by Eiter et al. [152].

When applying the jet colourmap, we set fixed normalisation boundaries of $0.4m$ and $1.4m$. The normalisation boundaries were based on the maximum grasping range of common mobile manipulators [98, 134] and the minimal perceivable depth of the RGB-D cameras used in such mobile manipulators [98]. These boundaries define a fixed relationship between the distance and size of objects in reality and the normalised image. The choice of the normalisation boundary balances the resolution and range of depth values for the model input. While narrowing the

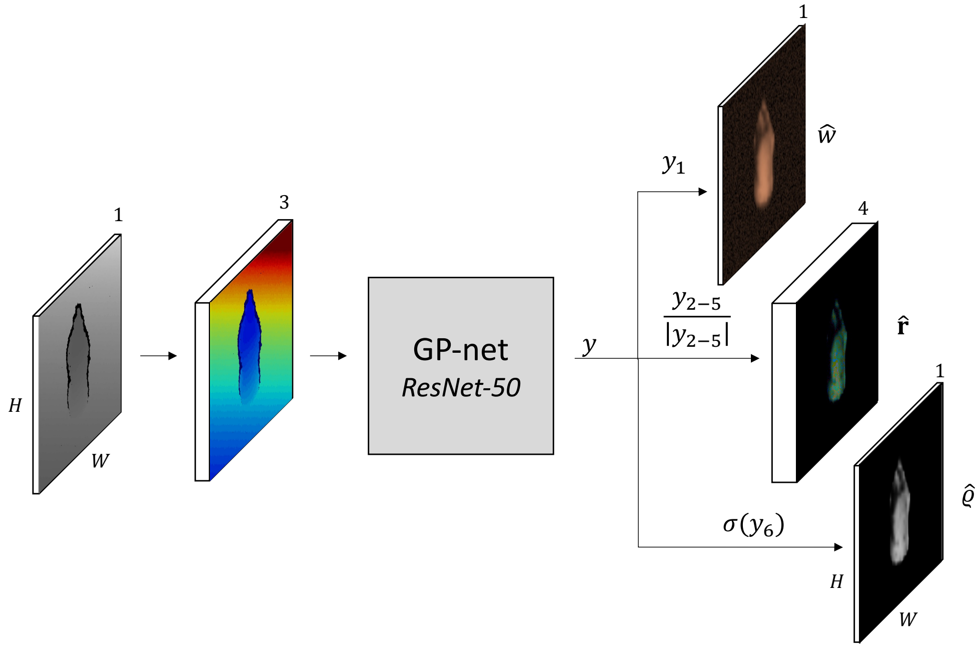


Figure 4.3: Using a depth image as input, we apply a jet colourmap, run inference on the resulting RGB image with GP-net and output a dense tensor y with 6 channels and the same spatial resolution $W \times H$ as the input image. The channels comprise a predicted grasp width $\hat{\mathrm{w}}$, a normalised unit quaternion for a predicted grasp orientation $\hat{\mathbf{r}}$ and a predicted grasp success $\hat{\varrho}$, normalised to the range $[0; 1]$ by a sigmoid function σ.

normalisation boundary increases the resolution, it also reduces the range of depth values that can be represented in the jet colourmap image.

The jet colourmap has a linear increase/decrease in individual RGB values, with a depth value corresponding to a unique composition of jet colourmap RGB values and no repetitions at the start and end of the colourmap. We visualise the mapping between the original image's depth values and the RGB values in Figure 4.2.

GP-net outputs a 6-channel tensor with the same spatial resolution as the input image, $W \times H$, see Figure 4.3. Each pixel in the output tensor represents a potential grasp proposal whose visible contact point corresponds to the 3D reprojection of that pixel in the depth image. The channels in the output tensor define the predicted width $\hat{\mathrm{w}}$ of the grasp, the predicted grasp orientation $\hat{\mathbf{r}}$ in the form of a quaternion, and the predicted grasp $\hat{\varrho}$ of that grasp. Similar to VGN [16], we normalised the quaternions to have unit norm (see Figure 4.3) and applied a sigmoid function to the predicted grasp success channel.

4.2.1 Loss function

Since creating ground-truth grasp proposals for training at each pixel of each image is computationally infeasible, we generated sparse maps containing grasp information at visible grasp contacts for up to 100 pre-sampled grasps per object. The training data generation process is described in Section 4.3. We backpropagated the loss only through the output pixels at which we have ground-truth grasp information, similar to VGN [16].

We indicate the availability of ground-truth grasp information at pixel i as $\mathbb{1}_{Grasps}(i)$ and the availability of ground-truth positive grasp information as $\mathbb{1}_{PosGrasps}(i)$, with $\mathbb{1}_{Grasps} = \mathbb{1}_{PosGrasps} \cup \mathbb{1}_{NegGrasps}$. The availability of ground-truth grasp information $\mathbb{1}_{Grasps}(i)$ includes pixels i at which the contact of one of the up to 100 pre-sampled grasps is visible and all non-object pixels, e.g., pixels showing the table. We then define our loss function as:

$$\mathcal{L} = \frac{1}{N_{Grasps}} \sum_{i}^{all\_pixel} \mathbb{1}_{Grasps}(i)\, \mathcal{L}_{\varrho,i} \quad + \\ \frac{1}{N_{PosGrasps}} \sum_{i}^{all\_pixel} \mathbb{1}_{PosGrasps}(i)(\lambda_{\mathbf{r}}\mathcal{L}_{\mathbf{r},i} + \lambda_{\mathrm{w}}\mathcal{L}_{\mathrm{w},i}) \tag{4.1}$$

with $\mathcal{L}_{\varrho}$ being the binary cross-entropy loss between the ground-truth grasp success and predicted grasp success, ϱ and $\hat{\varrho}$, $\mathcal{L}_{\mathbf{r}} = 1 - |\mathbf{r} \cdot \hat{\mathbf{r}}|$ the distance metric between the target and predicted quaternions, $\mathbf{r}$ and $\hat{\mathbf{r}}$, as defined in [153] and $\mathcal{L}_{\mathrm{w}}$ the L1 loss between the ground-truth and predicted grasp width, w and ŵ.

As ground-truth negative grasps do not have a valid configuration (grasp orientation $\mathbf{r}$ and grasp width w) for the model to learn, we only calculated the configuration part of the loss function for pixels with ground-truth positive grasps indicated as $\mathbb{1}_{PosGrasps}(i)$. We averaged all parts of the loss function by dividing the sums by the number of pixels used to calculate the loss, with the number of pixels with ground-truth grasp information N_{Grasps} and the number of pixels with ground-truth positive grasps $N_{PosGrasps}$. We set $\lambda_{\mathbf{r}}, \lambda_{\mathrm{w}} = 0.1$, which we experimentally found to give a good balance between $\mathcal{L}_{\varrho}$, $\mathcal{L}_{\mathbf{r}}$ and $\mathcal{L}_{\mathrm{w}}$.

In contrast to the loss function of VGN [16], we only allowed one valid configuration for the orientation loss $\mathcal{L}_{\mathbf{r}}$. This change was rooted in the grasp contact representation, which anchors each grasp to the visible grasp contact pixel instead of the non-visible TCP at the final grasp pose. The positive x-axis of the grasp coordinate frame points towards the second grasp contact and grasp centre (see Figure 4.1), with a mirrored version of the grasp (i.e. flipping the x-axis) resulting in a grasp in the air rather than on the object. With a TCP anchored grasp representation, as used in VGN, a mirrored version of the grasp does not change the position of the grasp centre, allowing for two valid orientations of the ground-truth grasp. The contact representation does not affect grasp variability since the parallel-jaw gripper is symmetric.

We trained GP-net for 20 epochs using an Adam optimiser with a learning rate of $3e^{-4}$ and a batch size of 32. To reduce the sim-to-real-gap for GP-net, we added simulated depth-camera noise to our depth images for training and evaluating the model in simulation. We used the noise model described in [154, 155], which simulated Gaussian-sampled random shifts in the depth based on disparity and perturbed vertices based on their normals. Overall, this resulted in more realistic depth images, with missing depth values for surfaces almost parallel to the camera principal ray, as can be seen in the input depth image in Figure 4.3. Compared to the Gaussian noise model suggested in DexNet2.0 [11], we found GP-net trained with this noise model to perform better in real-world tests.

4.2.2 Using the model output for robotic grasping

Individual grasp proposals must be extracted to use the 6-channel tensor output of GP-net for grasping objects with a robotic manipulator. As a first step, we used Non-Maximum-Suppression (NMS) to select up to $j = 10$ image coordinates (u, v) from the predicted grasp success $\hat{\varrho}$ output. To do this, we defined a minimum distance of 4 pixels between two maxima and only considered maxima in the predicted grasp success $\hat{\varrho}$ which exceeded the so-called acceptance threshold $\gamma_{\hat{\varrho}} = 0.4$. In other words, if there was a local maximum in the predicted grasp success with $\hat{\varrho} < \gamma_{\hat{\varrho}}$, NMS would not select the pixel to construct a grasp proposal. The acceptance threshold $\gamma_{\hat{\varrho}}$ was chosen based on our ablation studies in Section 4.5.3.

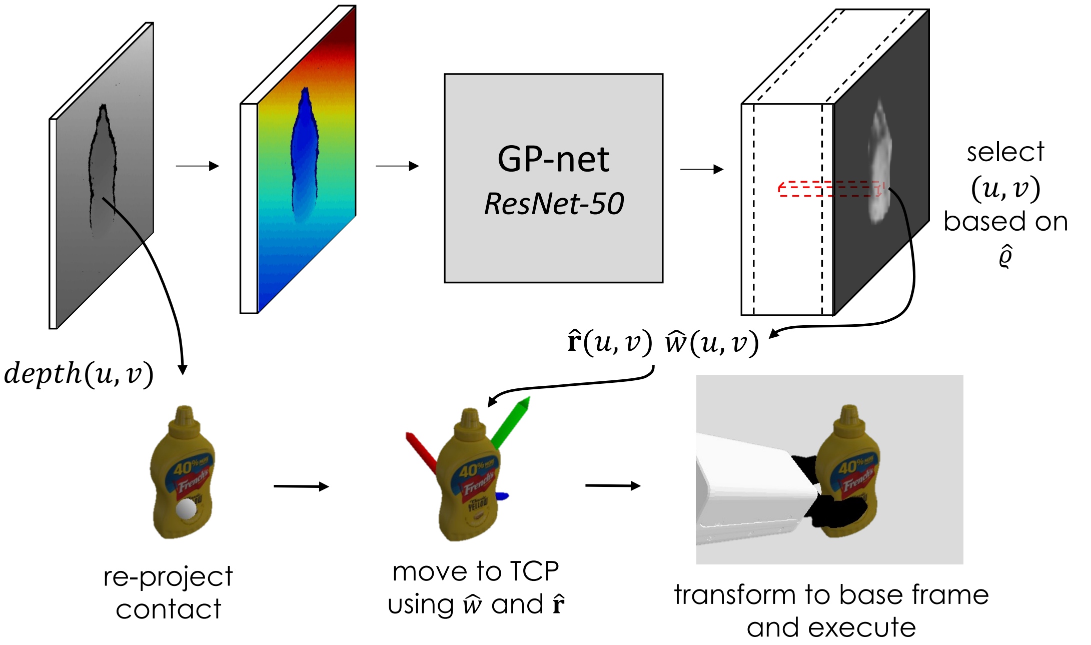


Figure 4.4: Grasp proposals for GP-net are generated by selecting pixels (u, v) with the highest predicted grasp success $\hat{\varrho}$ using NMS, re-projecting the contact point (visualised as a grey sphere) at those pixels into 3D using the depth at (u, v) from the input image and moving to the TCP (visualised as a coordinate frame) based on the predicted grasp orientation $\hat{\mathbf{r}}(u, v)$ and predicted grasp width $\hat{\mathrm{w}}(u, v)$.

Once the image coordinates (u, v) were chosen, we re-projected them into the 3D camera coordinates of the grasp contact using the camera intrinsics K and the depth value of (u, v) in the input depth image. Using the predicted quaternion $\hat{\mathbf{r}}$ and the predicted width $\hat{\mathrm{w}}$ of each pixel coordinate, we translated the grasp contact to the TCP by moving it $0.5\hat{\mathrm{w}}$ along the grasp x-axis between the gripper plates. As a final step, the grasp was transformed from the camera coordinate frame $\mathbf{T}^c$ to the robot base coordinate frame $\mathbf{T}^b$, resulting in an executable grasp pose for the robot.

Our ROS package provides this entire pipeline, as shown in Figure 4.4, by running inference with a trained model, selecting the pixel coordinates using NMS, constructing grasp poses for those coordinates from the model output and mapping them to the robot base transform. Furthermore, we provide code to use the package with a pre-trained GP-net model to plan grasps for a PAL TIAGo robot using a PAL parallel-jaw gripper. To ensure compatibility when using other gripper configurations, our code can generate an adjusted dataset, train a new model and finally use it with our ROS package to produce grasp proposals.

4.3 Training dataset

To train GP-net, we rendered a synthetic depth-image-based dataset with sparse ground-truth grasp information. Similar to Chapter 3, we used 1492 objects from the 3Dnet [144] and KIT [87] mesh datasets. We re-scaled the 3D meshes when loading them from the mesh datasets, as commonly done when generating robotic grasping datasets. In contrast to the re-scaling methods in [11, 74, 156], we re-scaled objects to randomly chosen widths, drawn uniformly from the range $6cm$ to $10cm$. Since we used a parallel-jaw gripper manufactured by PAL robotics with an opening width of $8cm$ [98], our dataset included objects that did not fit in our robotic gripper, which is a situation that will occur in real-world environments.

After re-scaling each object o, we calculated up to 25 stable resting poses [149] $\mathcal{S}(o)$ and used a modified version of the antipodal grasp sampler proposed in [11] to generate up to 100 parallel-jaw grasps $\mathcal{G}(o)$. In its original form, grasps were sampled by uniformly choosing a surface point from the mesh and then sampling the grasp x-axis (see Figure 4.1) based on the friction cone of the surface point. This procedure did not necessarily yield a good grasp for a grasp contact point. If available, our dataset should contain a good grasp of a given contact since we used it to train a model for proposing good grasps. Therefore, in our modified procedure, we generated $k = 6$ random grasps for a given contact. We calculated the robust force closure metric ϵ_{FC} [88, 108, 110] for each of the sampled grasps at one contact point and kept the grasp with the highest ϵ_{FC}. Using this method, we generated a total of $148,706$ ground-truth grasps for the 1492 objects in our dataset.

Since the robust force closure metric ϵ_{FC} of a sampled ground-truth grasp depends on the grasp contacts, it is defined by the grasp x-axis and consistent for any grasp approach axis, i.e. the direction of the grasp z-axis in Figure 4.1. However, collisions between the gripper, object and planar support surface further influence the success of a grasp. We defined the grasp success $\varrho(g)$ in our rendered dataset as

$$\varrho(g) = \begin{cases} 1 & \epsilon_{FC} \geq \delta_{FC} \text{ and } coll\_free(g) \\ 0 & otherwise \end{cases} \tag{4.2}$$

with $\delta_{FC} = 0.5$ being the robustness threshold for the robust force closure metric ϵ_{FC} and $coll\_free(g)$ indicating if the gripper is not in collision with the object at the grasp pose, similar to DexNet2.0 [11].

We aimed to choose reproducible grasp orientations for a given scene and ground-truth grasp to make the training data consistent and improve the training convergence. To achieve this, we applied the following steps for choosing the grasp approach axis orientation for a given grasp: we *hinge-rotated* the grasps in steps of $\Delta\omega = 15°$ around the contact points and thereby the grasp x-axis (see Figure 4.1) and checked for collisions between the gripper and scene at each orientation. If we only had a single, collision-free grasp orientation, we set the ground-truth grasp orientation equal to it.

If we had multiple adjacent collision-free grasp orientations, we set the ground-truth grasp orientation to be the median of these orientations to minimise the risk of collisions. There could also be multiple collision-free regions separated by grasp orientations that did result in collisions, for example, when grasping a tall and narrow object, like a bottle, at its base. Here, a top-grasp would lead to a collision of the gripper with the bottle, while horizontal grasps from both sides would be successful. In such a case, we chose the median collision-free grasp whose approach was most closely aligned with the principal ray of the camera. This resulted in grasps approaching the final grasp pose from the front of the object rather than from behind the object.

The objective for focusing on grasps more aligned with the camera principal ray is similar to the objective for excluding grasps with $\Psi > 90°$ in Section 3.3.2: maximise visibility and reduce the risk of occlusions and collisions. The approach differed from the one in Chapter 3 due to the move from a discriminative to a generative method. Since VGQ-CNN should be able to predict the quality of *any* grasp, we did not consider collisions for selecting the grasp orientation. In contrast, GP-net should propose *good* grasps if available since it is a generative method, and therefore we considered collisions when selecting the orientation of ground-truth grasps. With these steps, we fully defined the orientation of each grasp in our dataset to train GP-net in a reproducible way.

We rendered $n = 20$ images for each stable pose $s \in \mathcal{S}(o)$ of each object $o \in \mathcal{O}$. The camera poses were generated relative to the workspace origin $\mathbf{T}^{\mathrm{wo}}$ by uniformly sampling the radial distance d, the azimuth (or azimuthal) angle θ and the inclination (or polar) angle φ individually, as described in Section 3.3.1. We then projected the grasp contacts into the image plane and calculated the image coordinates of the visible contact (u, v). If grasp contacts of multiple collision-free grasps were visible at a single pixel, we chose the grasp with the highest robust force closure value ϵ_{FC}. In addition, we stored the binary segmentation mask of the object and used pixels with a visible depth point that did not show the object as ground-truth negative grasp contacts during training. The full dataset consists of $260,340$ images with an average of 88.1 grasps per image, totalling $22,944,376$ grasps.

4.4 Experiments

We evaluated the performance of GP-net with simulation and real-world experiments. While the simulation experiment enabled us to evaluate GP-net in various settings in a controlled and reproducible manner, the real-world experiment was important to evaluate the performance under real-world conditions, including sensor noise and end-effector reachability. We conducted the real-world experiment with two additional and widely used grasp proposal models, VGN [16] and GPD [24], and compared them to the performance of GP-net.

We used the 49 objects from the EGAD evaluation set [74] for both our simulation and real-world experiments to improve comparability between the experiments. The objects in EGAD can be classified by two independent metrics: “Shape complexity”, a measure of morphological complexity [89, 90], and “Grasp difficulty”, a measure of the average robustness of possible grasps on the object [91]. As proposed by the original authors of the EGAD evaluation set, we represent the results in a 7 × 7 grid, showing the performance metrics for each object according to its measure of shape complexity and grasp difficulty.

For all of our experiments, we used GSR and CR to evaluate the performance of the models overall and for each object individually. Both metrics are widely used in the community to judge the performance of grasping algorithms, as described in Section 2.2.1. We calculated GSR and CR aggregated across all evaluation scenes according to Equation 2.1 and Equation 2.2.

4.4.1 Simulation experiment

We tested the grasp success of the grasps proposed by GP-net in simulation using the PyBullet [114] physics engine. We based our simulation environment on the reference implementation provided for VGN [16], with changes to the camera viewpoints, the gripper and the tested objects to enhance comparability to our real-world experiment.

At the beginning of a grasping scene, an object was placed in a random pose within a $30 \times 30cm^2$ workspace on a planar surface. Here, the workspace was used to ensure visibility of the object from the sampled camera viewpoints, although neither having a fixed workspace nor knowing the camera-workspace transform is a requirement for using GP-net. After the object had been placed on the planar surface, a depth image was rendered from a camera pose sampled uniformly from the spherical coordinates around the centre of the workspace with the sampling bounds used for training in Section 4.3. We ran inference with GP-net on the depth image with added depth-camera noise and obtained grasp proposals from the model as described in Section 4.2.

For each scene, we tested the grasp proposal with the highest predicted grasp success $\hat{\varrho}$. A PAL parallel-jaw gripper was simulated to approach the grasp pose linearly along the grasp z-axis for $0.15m$, close the gripper at the grasp pose using a maximum force of $5N$, and lift the object vertically. The grasp was labelled successful if the gripper managed to lift the object $0.1m$ above the planar surface.

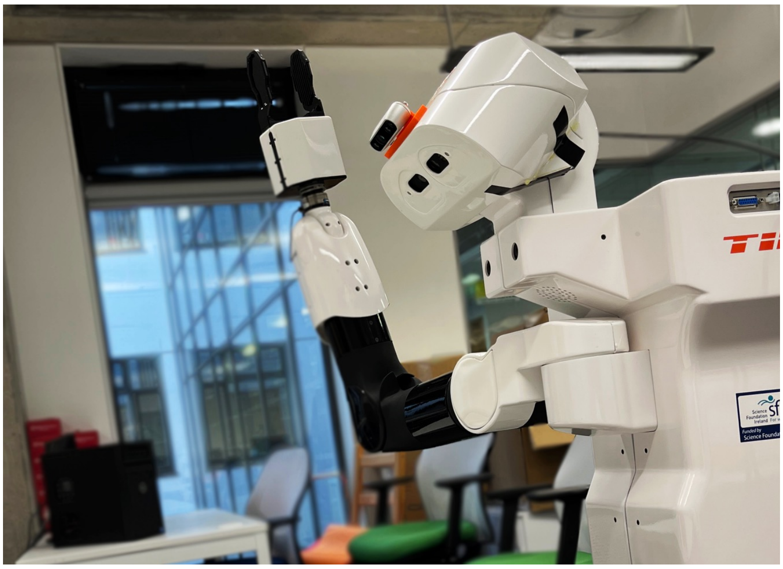


Figure 4.5: Example of a PAL TIAGo mobile manipulator with an Intel Realsense D435 depth camera mounted on its head.

Finishing a grasp attempt or failing to attempt a grasp, for example when the model has proposed no grasps, caused the scene to be abandoned. An experimental run of our simulation experiment comprised 100 grasping scenes for each of the 49 objects of the EGAD evaluation set. We decided to use the EGAD evaluation set to make our simulation experiment comparable to our real-world experiment, where we used a 3D-printed version of the same evaluation set.

4.4.2 Real-world experiment

We tested the grasping performance of GP-net, VGN and GPD in a real-world experiment using a PAL TIAGo mobile manipulator [98]. We used an Intel Realsense D435 depth camera mounted on TIAGo's head since it can perceive objects closer to the camera than the Orbbec Astra camera used in the TIAGo mobile manipulator by default. We also found the Intel RealSense camera to perform better on thin features, for example, it was able to perceive depth on thin objects such as pens at a distance $< 0.4m$. An example of the PAL TIAGo mobile manipulator with the Intel Realsense D435 camera mounted on its head is shown in Figure 4.5. For noise reduction, we averaged over 10 consecutive depth image frames to generate the model input for GP-net, VGN and GPD in each trial.

We used a 3D-printed version of the EGAD evaluation set, where we scaled each object to fit in the PAL parallel jaw gripper as instructed by the original authors [74]. For each object and each tested algorithm, a total of 10 grasp trials were executed, resulting in 490 grasp attempts per algorithm. We used two different initial robot poses for grasp planning, each having a different head tilt and torso height, resulting in two different camera viewpoints of the scene. A human operator dropped the object within a $30 \times 30cm^2$ square on the table before each grasp trial. The fixed workspace is a requirement for building the TSDF for VGN, and hence for consistency, it was used also for the evaluations with GP-net and GPD.

To apply VGN to our experiment, we manually defined the pose of the $30 \times 30 \times 30cm^3$ workspace for building the TSDF (and hence sampling grasps) using table segmentation for the height and a fixed $x - y$ position. Note that VGN was initially intended to be used with a wrist-mounted depth camera performing a grasp scan along a trajectory to build the TSDF. Since our work focused on proposing grasps from a single viewpoint, we applied VGN by building the TSDF from a single, noise-reduced depth image. While this differed from the original pipeline, there is an open question about how the number of viewpoints for building the TSDF affects the quality of VGN's grasp proposals. In later evaluations (see Section 5.5.2), we found no substantial difference between the performance of VGN when applied to a TSDF build from a single viewpoint and six different viewpoints.

VGN was originally trained and tested with a Panda gripper, which has the same maximum gripper width but a different collision shape than the PAL parallel-jaw gripper. We discuss this difference in Section 4.6, where we deem the PAL gripper to be an acceptable approximation based on a simulated comparison of both grippers.

To apply GPD to our experiment, we re-projected a point cloud from a single, noise-reduced depth image and cropped the point cloud to fixed margins to reduce the number of points and improve run-time. Furthermore, the points in the point cloud associated with objects were indexed to indicate where grasps should be sampled, applying a similar form of grasp-guiding as in our experiments in Section 5.4.2. The indexing of the point cloud was achieved by fitting a plane to the point cloud and indexing points with a distance $\geq 0.005m$ to the table plane.

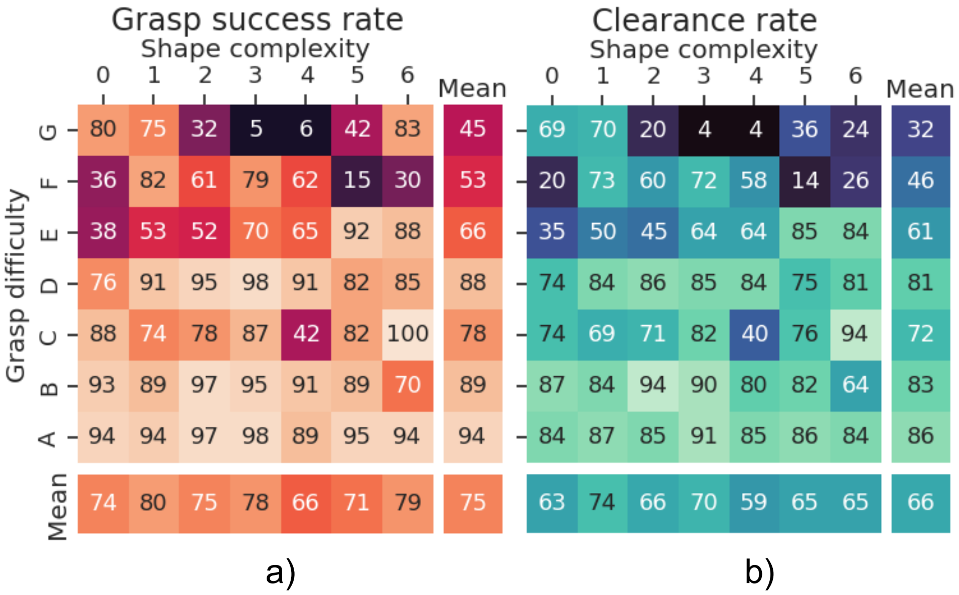
While the approaches for setting the workspace for VGN and indexing the point cloud for GPD were sufficient for our experiment, more general applications would require more sophisticated workarounds to prevent erroneous grasp proposals. For example, VGN would likely require 3D object detection to set the x-y position of the workspace if the object is placed in an unknown area on a table plane. For GPD, a robust object segmentation approach is required in order to guide the grasp samples for more complex scenarios. In contrast, GP-net does not need any workspace definition and can be directly applied to the depth image for tabletop scenarios. We executed the algorithms for all methods on an Intel i7-10750H CPU and NVIDIA RTX 2060 GPU.

4.5 Results

Overall, we found that the performance of GP-net in our experiment was superior to VGN and GPD while not having to rely on any pre-processing steps. In addition to the simulation and real-world experiments described above, we conducted two ablation studies using the simulation environment to compare a contact-anchored grasp representation to one based on the TCP and to set the acceptance threshold $\gamma_{\hat{\varrho}}$ when mapping GP-net's dense tensor output to grasp proposals.

4.5.1 Simulation experiment results

In our simulation experiment, GP-net achieved a mean GSR of 74.6% and a mean CR of 66.0% across all objects of the EGAD evaluation set. The GSR on each object is depicted in Figure 4.6 a) and the CR in Figure 4.6 b). Each object's shape complexity and grasp difficulty are defined in the EGAD evaluation set. Each object is named according to its shape complexity and grasp difficulty, e.g. object "A0" with grasp difficulty "A" and shape complexity "0". We note that both the GSR and CR decreased significantly with increasing grasp difficulty, while they were more consistent across the levels of shape complexity.



Grasp difficulty	0	1	2	3	4	5	6	Mean
G	80	75	32	5	6	42	83	45
F	36	82	61	79	62	15	30	53
E	38	53	52	70	65	92	88	66
D	76	91	95	98	91	82	85	88
C	88	74	78	87	42	82	100	78
B	93	89	97	95	91	89	70	89
A	94	94	97	98	89	95	94	94
Mean	74	80	75	78	66	71	79	75

Grasp difficulty	0	1	2	3	4	5	6	Mean
G	69	70	20	4	4	36	24	32
F	20	73	60	72	58	14	26	46
E	35	50	45	64	64	85	84	61
D	74	84	86	85	84	75	81	81
C	74	69	71	82	40	76	94	72
B	87	84	94	90	80	82	64	83
A	84	87	85	91	85	86	84	86
Mean	63	74	66	70	59	65	65	66

Figure 4.6: Results of evaluating GP-net in simulation with a) GSR [%] and b) CR [%]. 100 grasp attempts are executed with a PAL parallel-jaw gripper using the objects from the EGAD evaluation set.

This result indicated GP-net's ability to adapt to different shape complexities, while objects with a higher grasp difficulty were more challenging to grasp with the gripper. In EGAD, grasp difficulty is calculated using the average of the robust force closure ϵ_{FC} for up to 100 sampled grasps [74]. A lower average robust force closure ϵ_{FC} indicates a less robust grasp on average and, thereby, a more difficult object to grasp, resulting in a higher grasp difficulty. Therefore, we would expect a robotic grasping system to demonstrate a lower performance for shapes with a high grasp difficulty, as these objects are physically more challenging to grasp.

4.5.2 Real-world experiment results

The overall results and run-times for all three algorithms are shown in Table 4.1. GP-net performed better than VGN and GPD with a GSR of 54.4% compared to their 51.6% and 44.2%, respectively. Meanwhile, GP-net achieved a CR of 50.4%, lower than VGN's 51.2% and higher than GPD's 37.6%.

In our experiment, GP-net took 2.1s to propose grasps on a given depth image,

	GP-net	VGN [16]	GPD [19]
Grasp success [%]	**54.4**	51.6	44.2
Clearance [%]	50.4	**51.2**	37.6
Inference time [s]	2.1	**1.2**	14.7
Grasp planning time [s]	**2.7**	4.9	19.1

Table 4.1: Grasp success, inference, and grasp planning time for our real-world experiment. Inference refers to the time between model input and output, while grasp planning additionally includes pre-processing and post-processing.

while VGN and GPD required 1.2s and 14.7s, respectively. These inference times did not include the time taken for additional pre-processing steps like table segmentation and workspace definition, which were necessary for VGN and GPD. For this reason, we report the overall grasp planning time in Table 4.1, which included pre-processing, post-processing and sending data between different ROS nodes, where GP-net was faster than VGN and GPD with 2.7s compared to 4.9s and 19.1s, respectively.

Our results for GP-net with a GSR of 54.4% and CR of 50.4% were substantially lower than our results in our simulation experiment of 74.6% and 66.0%, respectively. Our simulation results were simulated using only the robotic gripper, not the full robot with the 7-DoF arm and the torso. In general, performance differences are expected when switching to a real-world environment. They are rooted in perception noise and actuation uncertainties, variations in friction, collision checking, and the performance of the path planning algorithms.

Detailed results for each algorithm with GSR and CR for each of the 49 objects in the EGAD evaluation set are shown in Figure 4.7. For all algorithms, we noticed a more prominent decline in performance for increasing grasp difficulty than for increasing shape complexity. This was consistent with the behaviour of GG-CNN [74] when tested on the EGAD evaluation set. Furthermore, we noticed some objects that were never successfully grasped by any model, e.g., objects “F0” and “G3”.

We encountered three main failure cases when testing GP-net in our real-world

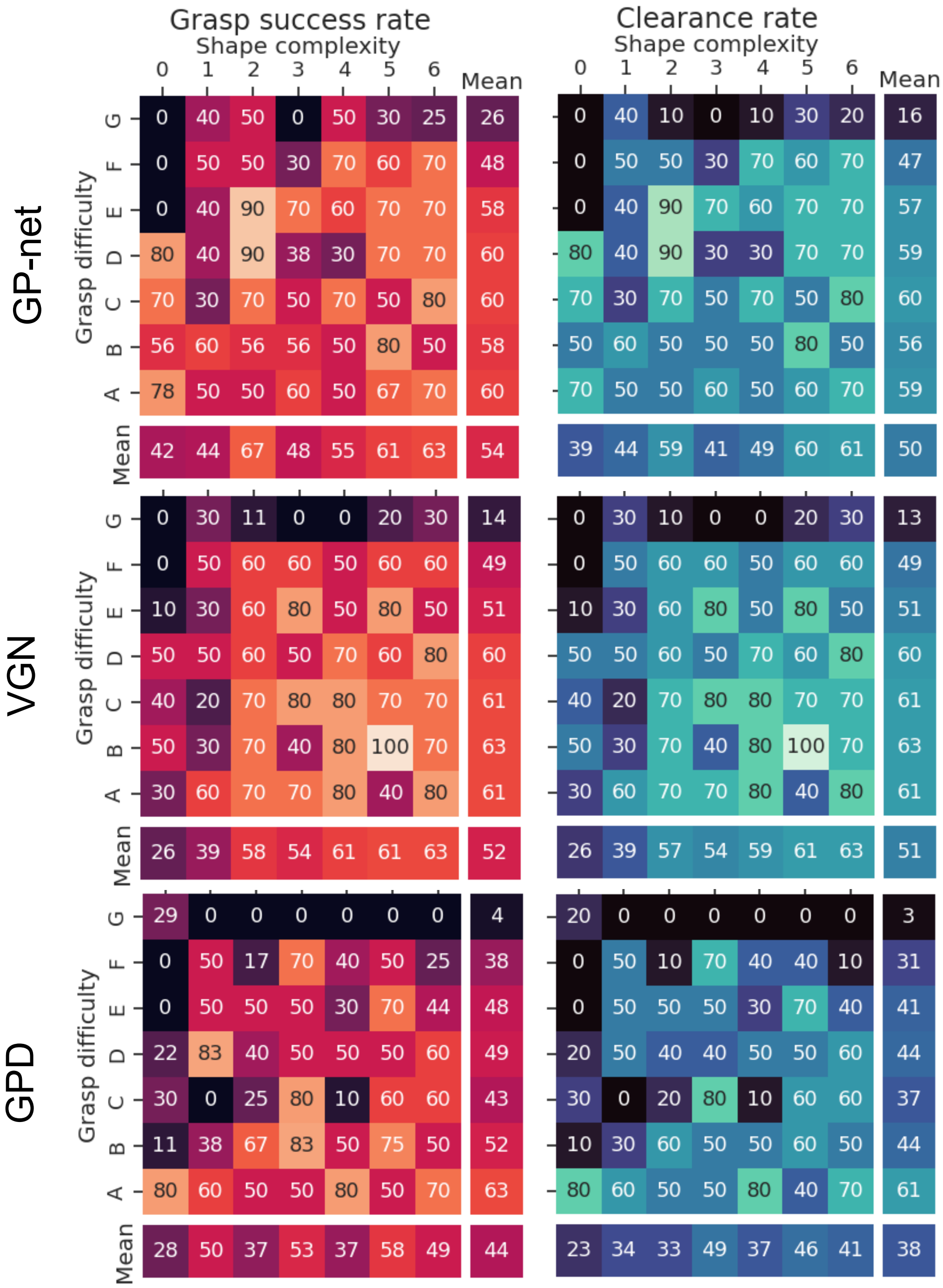


Figure 4.7: Results for evaluating grasp proposal models in a real-world experiment with a PAL TIAGo mobile manipulator. Graphs show the GSR [%] (left) and CR [%] (right) for 10 grasp attempts per object on the EGAD evaluation set using GP-net, VGN and GPD for grasp proposal.

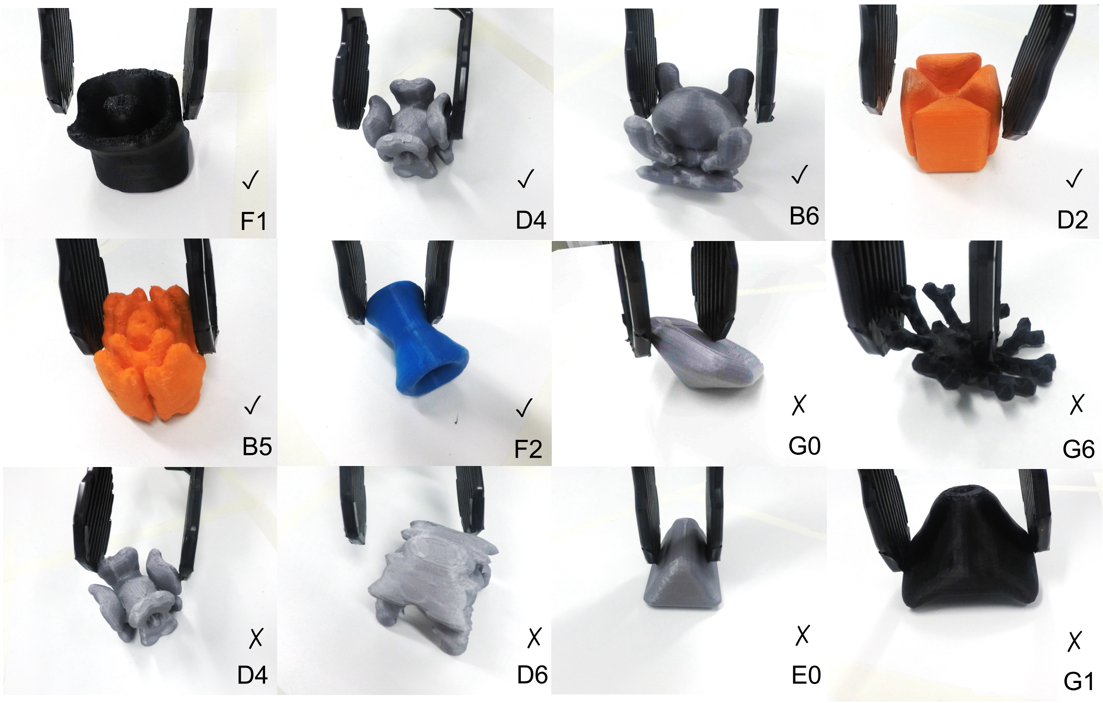


Figure 4.8: Success and failure cases when using GP-net to propose grasps on objects of the EGAD evaluation set with a PAL TIAGo mobile manipulator. We label each figure according to the corresponding object in the dataset and indicate success with ✓and failures with “x”.

experiment. An overview of different successful grasp attempts and failures is shown in Figure 4.8. The most common failure was caused by the gripper pushing the object away when approaching, as seen in the example of objects “D4” and “D6”. This failure can either be caused by badly planned grasps, e.g., a prediction of the width $\hat{w}$ that is too small or too wide, or it can be caused by perception or actuation noise. In most applications, modifying the grasping pipeline could prevent or reduce these failure types. For example, one could use closed-loop control using an RGB camera mounted on the gripper and correct the absolute gripper position during the final grasp approach.

The second most common type of failure occurred when the gripper could not close fully due to obstructions caused by the table or the object, e.g., in the examples of objects “G0” and “G6”. Once lifted, the object lost contact with the gripper, which resulted in a failed grasp attempt.

The third and last main failure type occurred when the friction between the object and the gripper was too low to compensate for the slanted surfaces being grasped, e.g. with objects “E0” and “G1” in Figure 4.8. This could be due to badly planned grasps or different friction values being expected by the model and resulted in the objects slipping out of the gripper when trying to lift the object. Note that grasping “E0” along its longitudinal axis was not an option as this side of the object is slightly wider than the gripper width, causing “E0” to be rarely grasped by our gripper.

4.5.3 Ablation studies

We conducted a set of ablation studies to investigate the performance of GP-net further. We investigated how an alternative grasp representation, anchoring the grasps to the TCP, performed compared to the contact-based grasp representation (see Figure 4.1) used for GP-net. Furthermore, we ran a simulation experiment to investigate how grasp success and the final number of grasp proposals of a given model were influenced by the acceptance threshold $\gamma_{\hat{Q}}$ used when applying NMS to select grasp proposals.

Grasp representation: The grasp representation used in GP-net is anchored to the visible grasp contacts, with each pixel in the depth image representing a visible contact of a potential grasp, see Section 4.2. This grasp representation was first proposed in Contact-Graspnet [20], where the authors suggested it facilitates learning by reducing the problem’s dimensionality. However, the grasp representation in [20] was not compared to conventional grasp representations used in data-driven grasp proposal approaches, which anchor grasps to the TCP [9, 11, 12, 14, 16, 18, 23, 24, 37, 63].

While the contact-based grasp representation can potentially increase the model performance, it can lead to problems where no grasp contact is visible. An example of such a situation is if a robot is supposed to grasp a book but is facing the spine of the book head-on, for example, as visualised in Figure 4.9. In this situation, the areas for potential grasp contacts on the left and right sides are invisible, and a contact-based method cannot produce valid grasp configurations. In these situations, repositioning the camera is necessary to produce grasp proposals.

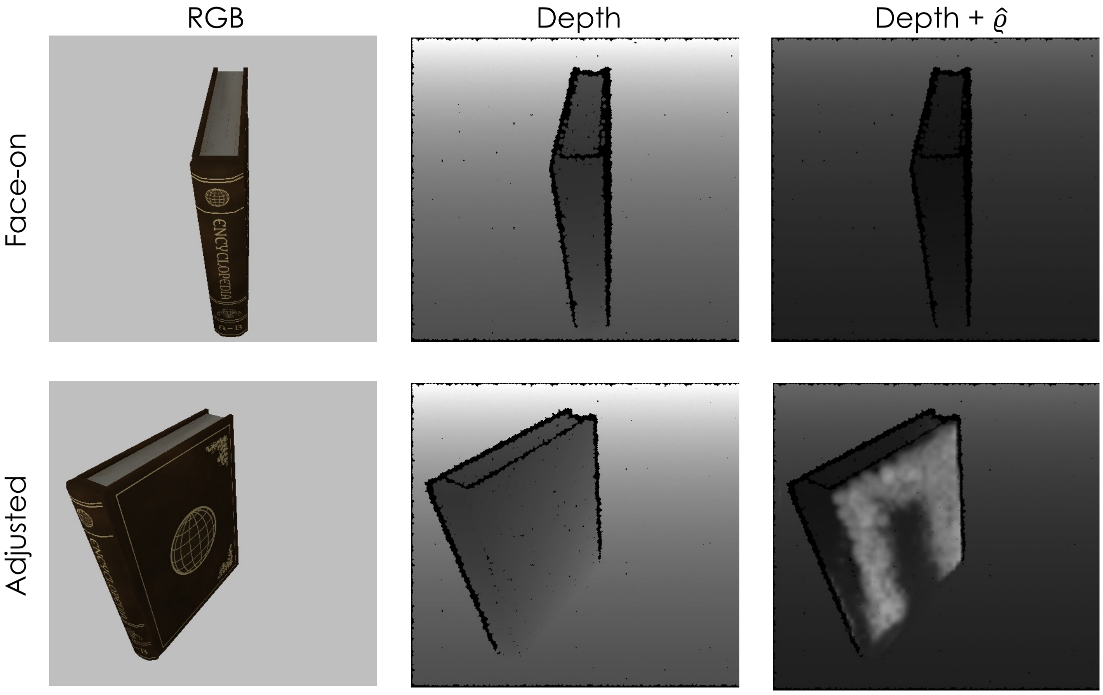


Figure 4.9: Example of a situation where a contact-based grasp representation can not represent any grasp due to visibility issues (upper row), with the RGB (left), depth (centre) and an overlayed image of the depth and predicted grasp success $\hat{\varrho}$ (right). When repositioning the camera (lower row), grasp contacts become visible, and grasps can be proposed.

To assess this limitation against potential performance gains, we compared the performance of a contact-based grasp representation to a TCP-based grasp representation. We modified GP-net to use a TCP-based grasp representation as $g \in (u, v, \Delta z, \mathbf{r}, \mathrm{w}, \varrho)$ with the image coordinates (u, v) defining the 2D position of the grasp centre, i.e. the TCP, in the image. Δz thereby describes the distance between the TCP and the visible surface depth at (u, v), since the intersection between the principal ray at (u, v) and the object surface is usually closer to the camera than the TCP. In other words, the TCP is not visible, and Δz describes how much deeper than the visible depth at (u, v) the TCP lies.

We trained the model on our dataset as described in Section 4.2 and compared performance to GP-net using the simulation analysis. Due to the extra variable Δz in the grasp representation, the model has seven output channels, and we define the adapted loss function as:

$$\begin{aligned}\mathcal{L} = \frac{1}{N_{Grasps}} \sum_{i}^{all\_pixel} \mathbb{1}_{Grasps}(i)\, \mathcal{L}_{\varrho,i} \qquad + \\ \frac{1}{N_{PosGrasps}} \sum_{i}^{all\_pixel} \mathbb{1}_{PosGrasps}(i)(\lambda_{\mathbf{r}}\mathcal{L}_{\mathbf{r},i} + \lambda_{\Delta z}\mathcal{L}_{\mathbf{\Delta z},i} + \lambda_{\mathrm{w}}\mathcal{L}_{\mathrm{w},i})\end{aligned} \tag{4.3}$$

with $\mathcal{L}_{\Delta z}$ being the L1 loss between ground-truth Δz and predicted grasp distance $\widehat{\Delta z}$. We set $\lambda_{\mathbf{r}}, \lambda_{\Delta z} = 0.1$ and $\lambda_{\mathrm{w}} = 0.01$. We reduced λ_{w} compared to the contact-based loss function since the width is not used to predict the grasp centre for the TCP-based grasp representation. As the width might still be used for grasp execution in certain applications, such as sorting grasps or setting an initial gripper width, we kept the width in the model output. After training the model for 20 epochs, we achieved a mean GSR of 27.2% and CR of 20.4% in simulation, substantially lower than GP-net's 74.6% and 66.0% with the contact-based grasp representation. We found that the rotation loss $\mathcal{L}_{\mathbf{r}}$ plateaued at almost double that of GP-net's loss with 0.24 for the TCP-based grasp representation and 0.13 with the contact-based grasp representation. The remaining parts of the loss converged similarly between both grasp representations.

Based on these results, we hypothesised that the performance difference was rooted in the reduction of orientation ambiguities when grasps are anchored to their grasp contact points. Ground-truth positive grasps would have similar orientations for neighbouring grasp contacts on an object. In contrast, TCP-based grasps could approach the same TCP position from a variety of angles.

This relationship is especially apparent when looking at cylindrical objects, where a TCP-based ground truth positive grasp could rotate the grasp axis to any orientation around the centre of the cylinder. In contrast, a contact-based ground truth positive grasp only permits one orientation such that the x-axis of the grasp passes through the centre of the cylinder. An example of this difference is shown in Figure 4.10. The three different grasp orientations result in three different grasp contacts with a contact-based grasp anchor but result in the same grasp centre and thereby TCP position in the image.

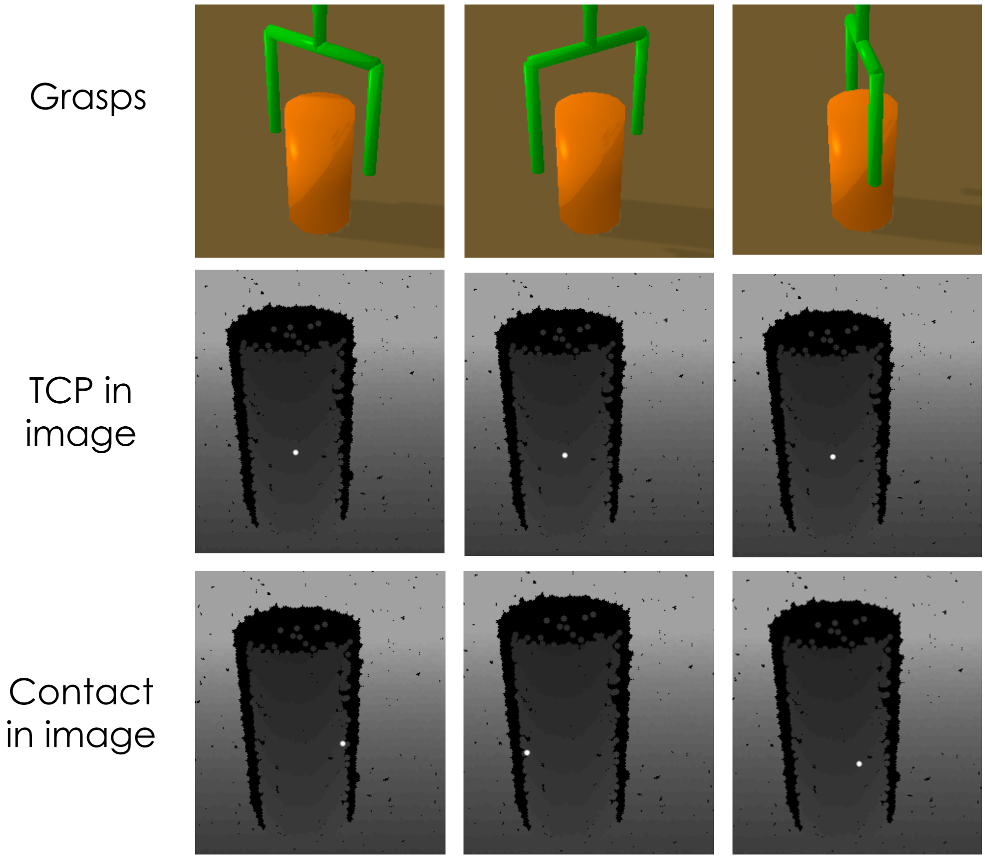


Figure 4.10: Example of the difference in TCP and contact-based anchor points for cylindrical objects. The upper row shows an RGB image of the scene with a symbolic parallel-jaw gripper in three different orientations. The corresponding grasp anchors in the image are shown as white dots in the depth images. The middle row thereby shows the TCP based grasp anchors, and the lower row shows the contact-based grasp anchors.

Acceptance threshold $\gamma_{\hat{\varrho}}$: The acceptance threshold $\gamma_{\hat{\varrho}}$ defines the minimum predicted grasp success $\hat{\varrho}$ that can result in output grasp proposals for GP-net. The coordinates of the output grasp proposals are obtained by applying NMS with $\gamma_{\hat{\varrho}}$ as a cut-off threshold to the predicted grasp success $\hat{\varrho}$ output of the model, as explained in Section 4.2. As such, $\gamma_{\hat{\varrho}}$ balances the number of proposed grasps and the confidence the model has in those grasps. Ideally, $\gamma_{\hat{\varrho}}$ should be set to propose as many grasps as possible while retaining good confidence in the quality of those grasps. To investigate how this trade-off affects GP-net, we applied a range of acceptance thresholds $\gamma_{\hat{\varrho}} = [0.1, 0.2, \ldots, 0.9]$ and investigated the influence on

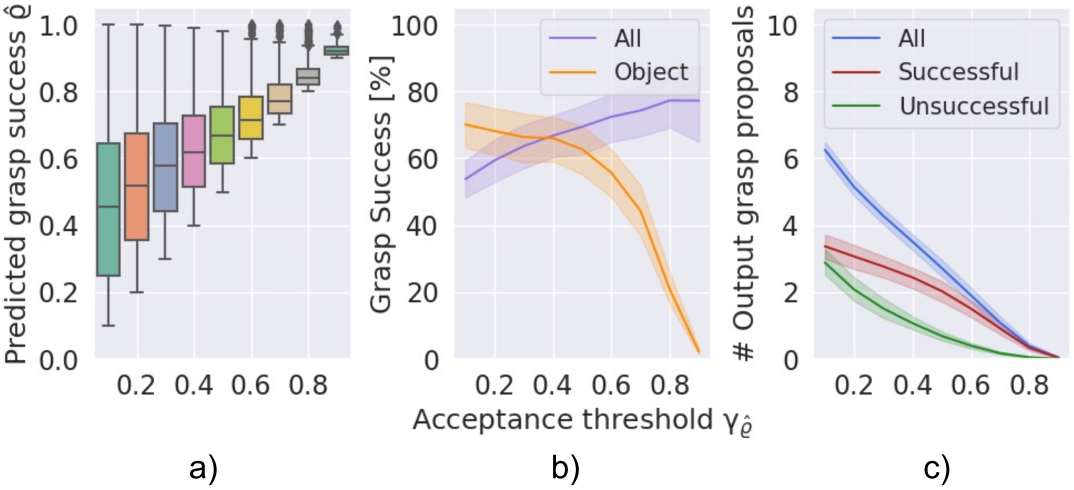


Figure 4.11: Simulation results showing the influence of varying the acceptance threshold $\gamma_{\hat{\varrho}}$ on a) the predicted grasp success $\hat{\varrho}$ of grasp proposals, b) the grasp success [%] and c) the number of output grasp proposals per object. The boxplot shows the 25^{th} to 75^{th} percentile, while the lineplots include a 95% confidence interval.

grasp performance in simulation.

The results can be seen in Figure 4.11. Increasing the acceptance threshold $\gamma_{\hat{\varrho}}$ increased the predicted grasp success $\hat{\varrho}$ in Figure 4.11 a) and decreased the number of output grasp proposals in Figure 4.11 c). The number of successful and unsuccessful grasp proposals are reduced non-linearly when increasing $\gamma_{\hat{\varrho}}$, where the number of unsuccessful grasps decreased more rapidly than the number of successful grasps for a small $\gamma_{\hat{\varrho}}$.

We show both the grasp success of all proposed grasps and the grasp success for each object in Figure 4.11 b). Note that the grasp success for each object (shown in orange) used only the output grasp proposal with the highest predicted quality $\hat{\varrho}$ and reported an unsuccessful grasp if there was no grasp proposal available, indicating the ability to pick up the object. The grasp success "All" (shown in purple) shows the ratio of grasp success to grasp attempts when simulating every proposed grasp by the model. The two curves show the trade-off between not predicting any grasp and failing to pick up the object, i.e., a low grasp success for the object, and the success of all grasp proposals, i.e., a lower grasp success on all grasp proposals if many unsuccessful grasps are proposed.

We calculated the performance over this trade-off to decide on an acceptance threshold for the remainder of our experiments. We used the average between the grasp success of all proposed grasps and the grasp success for each object. This approach balanced the ability to pick up as many objects as possible (for all scenes) while retaining a high grasp success for all grasp attempts. The maximum of the resulting curve lay at $\gamma_{\hat{Q}} = 0.4$ with 66.4%. Based on these results, we set $\gamma_{\hat{Q}} = 0.4$ for all of our other experiments using GP-net.

4.6 Discussion

The contact-based grasp representation utilised in GP-net anchors grasp proposals to visible grasp contacts. If no grasp contact on a specific object is visible, no grasp can be proposed for that object. This limitation does not occur in TCP-based grasp representations, where the grasps are anchored to the TCP of the gripper. The TCP of the gripper is then not directly anchored to any visible points in the sensor readings, as it is usually within the volume of the object. To evaluate the benefits of a contact-based grasp representation against these limitations, we compared it to a TCP-based grasp representation in our ablation studies in Section 4.5.3. We found a substantial performance advantage for a model trained with the contact-based representation, which we hypothesise is connected to a reduction in orientation ambiguities when anchoring the grasp proposals to visible grasp contacts.

In this work, we focused on applying 6-DoF grasp proposal algorithms to flexible viewpoints, for example, for use with mobile manipulators in domestic environments. The robot can move the camera and view objects from different perspectives when using such a mobile manipulator in real-world scenarios. In this way, previously invisible grasp contacts can be made visible to enable the proposal of new grasps for an object. While we evaluated GP-net on a stationary robot which can reposition itself and the camera, such a repositioning could also be achieved while approaching the object and simultaneously planning paths to enable a “smooth” movement of the robot given appropriate runtimes of the algorithms. Under these circumstances, we believe the benefits of a contact-based representation outweigh its limitations.

Another limitation, and similar to most of the commonly used methods [11, 16, 17, 18, 20, 36, 37, 38, 63, 73], is that GP-net is trained with grasp success based on experience with a specific gripper, in this case a PAL robotics parallel-jaw gripper. While using simulated grasp experience for training is a common approach in recent years [11, 16, 17, 18, 20, 38, 63], it establishes a gripper-dependence in the training data and hence the resulting model. Since the grasp success depends on the grasp being collision-free, the design of the gripper is used implicitly during dataset generation when checking for collisions, see Eq. 4.2. The resulting model learns the implicit gripper design, so the performance with alternative gripper configurations cannot be guaranteed.

Gripper designs for parallel-jaw grippers mainly differ in their maximum gripper width, collision shape and the size of their gripper plates, as shown in Figure 2.8. When using pre-trained models for different gripper designs than intended, one must take care to ensure these differences do not become an issue. In our real-world experiment, the pre-trained version of VGN is trained for a Panda gripper, which has a different design to the PAL parallel-jaw gripper used in our experiments, see Figure 2.8. While the maximum gripper width is $8cm$ for both grippers [98, 99], it cannot be guaranteed that the performance of VGN is the same when tested with a PAL parallel-jaw gripper due to differences in the collision volume and gripper plate areas.

To investigate if the PAL parallel-jaw gripper can be used instead of the Panda gripper when tested with the EGAD object set that was used in our real-world experiment, we conducted a simulation experiment as described in Section 4.4.1. Grasps proposed by VGN were evaluated with both a Panda and a PAL parallel-jaw gripper, where we got a GSR of 61.9% and 57.4%, respectively. When testing the grasps proposed by GP-net with both a Panda and a PAL parallel-jaw gripper, we achieved a GSR of 77.7% and 74.2%, respectively. This could indicate that the Panda gripper might have a general design advantage over the PAL parallel-jaw gripper.

Considering this, we decided to use the PAL parallel-jaw gripper instead of the Panda gripper for our experiment since it is available on our PAL TIAGo mobile manipulator. Furthermore, though the PAL parallel-jaw gripper potentially has a drop of up to 5% in GSR compared to the Panda gripper, it should affect all algorithms equally. Therefore, we deemed the PAL parallel-jaw gripper an acceptable approximation in our real-world experiment.

The results reported here may differ for other gripper designs or object sets, making it important for researchers to make reference implementations available and enabling others to adapt the code and model to required gripper designs. Alternatively, there has been recent progress [111, 157] in making grasp proposal models independent of the gripper design. However, this currently seems limited to discriminative approaches with 4-DoF [157] or does not consider the collision shapes of potential grippers [111].

4.7 Conclusion

In this chapter, we presented GP-net, a model to propose 6-DoF grasps based on depth images from flexible and unknown viewpoints. In contrast to widely used algorithms for robotic grasping like GPD [24] and VGN [16], the viewpoint flexibility enables GP-net to be used without the need to pre-define a workspace or filter grasps to produce good grasp proposals. Hence, GP-net can be used on mobile robots without additional pre-processing steps like plane-fitting, table segmentation or 3D object detection. To train GP-net, we created a synthetic dataset based on more than 1400 object meshes with ground-truth grasp information.

We evaluated GP-net on a PAL TIAGo mobile manipulator in a real-world experiment and compared its performance to GPD and VGN on the EGAD evaluation set [74]. GP-net achieved a grasp success of 54.4%, better than VGN's 51.6% and substantially higher than GPD's 44.2% in our setup. Importantly, GP-net achieved those superior results without requiring any of the pre-processing steps required for VGN and GPD. This enables GP-net to be used without relying on correct results of object segmentation techniques and in scenarios with unknown camera-workspace transforms. The advantage of not requiring pre-processing steps also affected the inference time in our experiment, where GP-net had a total grasp planning time that was 0.55 of VGN and 0.14 of GPD.

Both VGN and GPD work in cluttered environments, while GP-net was trained for single objects on planar surfaces. Moving from single objects on planar surfaces to cluttered objects in domestic environments is the focus of the next chapter in this thesis. Domestic environments feature cluttered objects on different furniture units like tables, desks, or shelves, all with unknown camera-workspace transforms. Therefore, a key challenge to be addressed in the next chapter is the provision of sufficiently representative training data to train a data-driven approach for grasp proposal in domestic environments.

CHAPTER 5

Grasping Objects in Domestic Environments

In Chapter 4, we proposed the GP-net, a depth-image-based 6-DoF grasp proposal method for flexible camera viewpoints. GP-net is trained to propose grasps for single objects on planar surfaces without the need for any pre-processing steps like workspace definition. When moving to grasping in domestic environments, the scenes become more complex and variable, making it important to use grasping algorithms which are robust to these conditions.

The complexity of domestic environments in the grasping context stems from a large number of unknown variables - the object type, object shape, scene surroundings and the pose of the object in relation to the robot are usually unknown. This is especially apparent when looking at mobile manipulators, which can change their own pose and that of their camera in relation to the workspace. Due to this capability, mobile manipulators are able to transport objects within the environment.

Previous robotic grasping research typically simplified the problem and search space to handle the complexity of domestic environments. While focusing on variable object types and shapes, they often placed them in tabletop scenarios with fixed cameras and 4-DoF grasp poses, e.g. [9, 10, 11, 23, 113]. While our

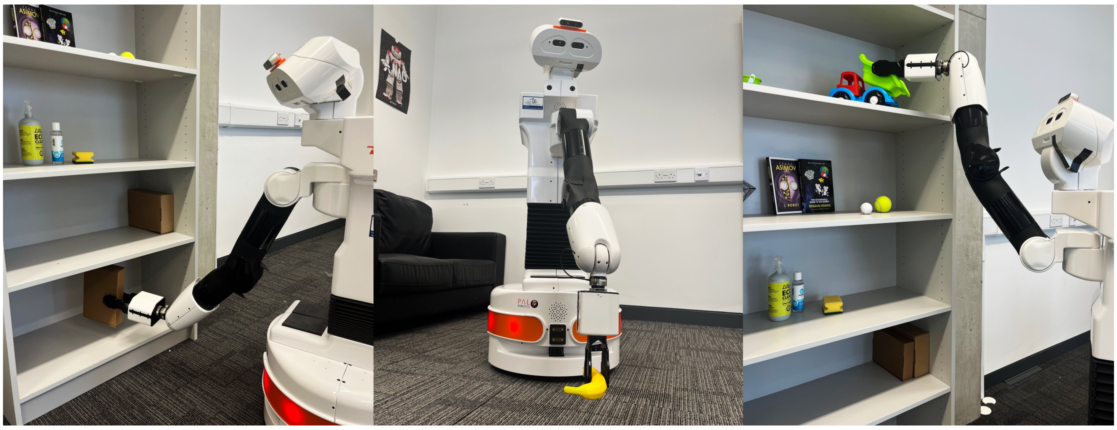


Figure 5.1: Examples of a PAL TIAGo mobile manipulator grasping objects in domestic environments.

own work, GP-net, allowed for flexible camera viewpoints and 6-DoF grasps, it was trained for single objects on planar surfaces, not allowing for more complex setups with multiple objects in clutter.

In this chapter, we focus on two substantial problems introduced when moving to grasping in domestic environments: more complex scenes (e.g. cluttered objects placed on shelves or tables) and an increased number of potential camera viewpoints (e.g. looking up to an object on a shelf or down to an object on the floor). Complex scenes make it difficult to recognise graspable surfaces and increase the risk of collisions, while a high number of potential camera viewpoints increases the complexity of the scene representation. An example of a mobile manipulator grasping objects in such surroundings is shown in Figure 5.1.

We aim to solve grasping in domestic environments with a data-driven method similar to GP-net and many grasping algorithms proposed in recent years [11, 12, 13, 14, 16, 21, 37, 38, 63, 64]. When using data-driven methods, simulated or real-world grasp experience is used to train machine learning models. The possible application scenarios for such algorithms are defined (and constrained) by the grasp experience covered in the training data. For example, if a dataset only covers tabletop grasping scenarios with a fixed overhead camera, it likely will not be suitable for grasping on shelves from variable viewpoints. In order to develop a model for domestic environments with cluttered objects on different furniture units, we had to generate training data from representative domestic scenes.

We took particular care in setting the furniture placement, object placement and camera viewpoints in a manner that would likely be encountered in real-world applications. For this, we developed reproducible setups for selecting furniture units, selecting object meshes, placing them on the furniture units and rendering camera viewpoints from realistic camera poses. We trained and evaluated an updated version of GP-net called GP-net+ on our generated data. GP-net+, which incorporates improvements to the tensor output and loss function, can be used to propose 6-DoF grasps in cluttered environments on different furniture units.

Many different architectures and scene representations for grasp proposal algorithms have been explored in recent years [5, 8], with no clear advantage of one method over the others to date. A major issue in benchmarking grasp proposal algorithms is rooted in differences in test setups, evaluation sets and robotic hardware, as we discussed in Section 2.2.2. We compared GP-net+ to widely used alternative grasp proposal solutions in a set of simulated experiments to provide quantitative evaluations and investigate its strengths and weaknesses. In those experiments, GP-net+ outperformed all models in most of the experiments while being robust to more complex, domestic scenes.

We made our code for generating the domestic training scenes and evaluating the models available on GitHub [5] so it can be used by other researchers. Our codebase can be used to generate training data for alternative scene representations, as well as train and evaluate alternative model structures. Furthermore, we made a ROS package available to run inference with GP-net+ and generate grasp proposals, which we tested in a real-world experiment with a PAL TIAGo mobile manipulator. The ROS package also includes an example of how GP-net+ can be combined with a pre-trained object detection algorithm to enable target-driven object grasping in clutter.

[5] `https://github.com/AuCoRoboticsMU/GP-netplus`

In summary, the main contributions of this chapter are:

- A benchmark dataset and a simulation environment. The dataset comprises depth images of domestic scenes with ground-truth grasp information. The simulation environment can be used for training and evaluating grasping algorithms in domestic environments. It can be used with different scene representations and parallel-jaw grippers to enhance comparability.
- GP-net+: A model to predict 6-DoF grasps on cluttered objects with a parallel-jaw gripper. The objects can be positioned on tables or shelves with an unknown camera-workspace transform.
- A ROS package for deploying GP-net+ on a real robot, for example, using a PAL TIAGo mobile manipulator. When deploying the ROS package, GP-net+ can also be used for target-driven object grasping using a pre-trained object detection algorithm.

5.1 Related Work

Domestic environments in the real world are characterised by unstructured, complex and dynamic scenes. For example, a living room typically includes several sitting arrangements, shelves or sideboards, and unknown objects. The pose of the furniture units and, especially, the objects are unknown and can change between different points in time and different rooms or households.

For a mobile manipulator to accomplish assistive tasks in these domestic environments, it should be able to grasp objects regardless of their pose or which supporting surface they stand on, as long as they are physically reachable. For example, this includes objects positioned on shelf compartments, drawers, and tables. Using a mobile manipulator, compared to a fixed robot, in these domestic environments enables accessing different furniture units and moving objects between them in the scene [3].

Research for grasping objects with mobile manipulators today is often based on hand-crafted approaches with limited flexibility and robustness under the influence

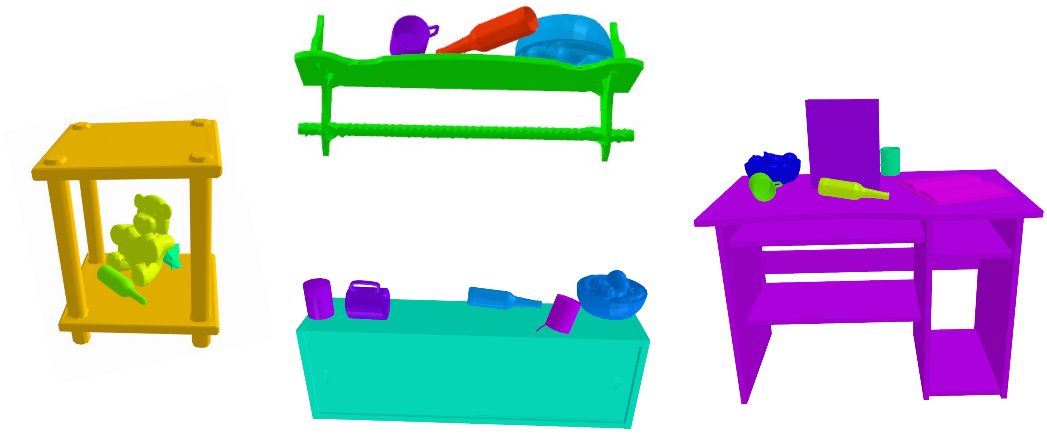


Figure 5.2: Example scenes from the ACRONYM dataset with objects placed on a table, two shelves and a desk. Note that the colour of the objects and furniture units is chosen randomly for visualisation purposes.

of uncertainty, e.g. [132, 133, 135, 136]. Example approaches include segmenting the objects in point clouds based on their relation to table surfaces [132] or applying grasp poses for known CAD models [136, 139]. As such, these approaches can likely not be applied to the domestic environments described here due to limited flexibility.

As mentioned in the introduction of this chapter, recent robotic grasping research has mostly focused on data-driven grasping algorithms, where grasping datasets are used to train machine learning models for grasp proposal. Most of these grasping datasets provide scenes with a single object [11, 76, 77, 158] or multiple cluttered objects [22, 39, 73, 159] on planar surfaces like tables. This setup is most appropriate when applying the resulting algorithms to similar scenarios where the composition of training data matches the application, for example, with fixed manipulators mounted on a table.

When moving to application in domestic scenarios, corresponding scenes must be represented in the training data. If grasping algorithms have been trained for domestic scenarios, they should be able to propose appropriate grasp proposals in such scenes, i.e. grasp proposals on the objects rather than the furniture units, and propose grasp poses that minimise the risk of collisions with the surroundings.

One training dataset that features objects placed on different furniture is the ACRONYM dataset [156], with examples shown in Figure 5.2. It is a synthetic dataset generated with a physics simulator containing a large number of ground-truth grasps for cluttered household objects. The household objects are placed on single support surfaces of furniture units from the ShapeNet [160] mesh database. While we think the ACRONYM dataset is a step in the right direction towards training domestic grasping algorithms, we identify several shortcomings for our use case specifically.

First, objects in ACRONYM are placed on a single support surface of the furniture unit rather than being distributed over multiple support surfaces. We think this could lead to algorithms being unable to propose grasps for different objects on multiple support surfaces.

Secondly, the ground-truth grasp poses for the objects in ACRONYM are only tested at the actual grasp pose; there is no collision-checking during the grasp approach or retreat. This can lead to unreachable grasp poses being proposed, especially in confined spaces like shelf compartments, where, for example, top-grasps should not be proposed as they can collide with the shelf during the grasping movements.

Lastly, scene representations are generated from camera viewpoints sampled in spherical coordinates around the object centre, similar to how camera poses are sampled in tabletop scenarios, e.g. [11]. While camera views sampled from spherical coordinates are sensible when working with tabletop scenarios, they can lead to unrealistic behaviour in more complex surroundings like shelves (e.g. occluded camera views with the camera positioned inside a shelf compartment).

When grasping objects with a mobile manipulator in such domestic environments, one would likely use a camera mounted on the robot to view the scene. The advantages of using a robot-based camera viewpoint lie in their ability to reposition with the robot and thereby access different furniture units in the entire environment. We think including such robot-based viewpoints in the training data is necessary to enable the usage of mobile manipulators for grasping objects in domestic environments. Due to these shortcomings in the ACRONYM dataset for our purposes,

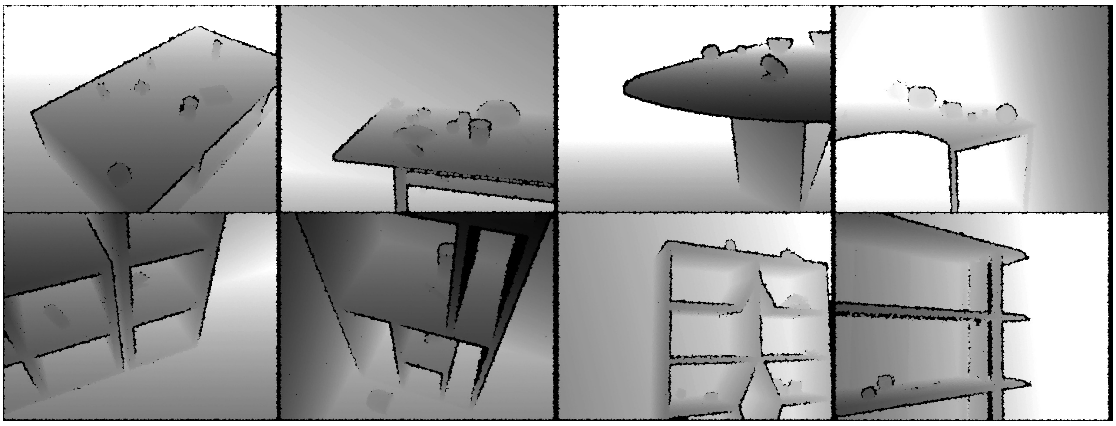


Figure 5.3: Example depth images of domestic grasping scenes.

we needed to create our own dataset, as described in the following section.

5.2 Generating domestic grasping scenes

We aimed to generate domestic grasping scenes to train and evaluate a grasp proposal algorithm for usage in real-world domestic scenarios. In contrast to our previous work in Chapter 4, this included cluttered objects on different furniture units viewed from robot-based camera poses in contrast to single objects on a table plane. We used the resulting training dataset to train GP-net+ (see Section 5.3), an extension of our previous model GP-net.

The following describes the process for creating the domestic grasping scenes. The domestic grasping scenes consist of different kinds of graspable objects (see Section 5.2.1) arranged on a variety of furniture units (see Section 5.2.2). Example depth images of such scenes are shown in Figure 5.3. In contrast to ACRONYM, we defined camera sampling spaces and object workspaces for each furniture unit to generate the relative poses of camera viewpoints and graspable objects in a reproducible manner. We used these sampling spaces during scene generation and tested the success when grasping pre-sampled grasp poses in simulation (see Section 5.2.3). We saved the depth images and the tested grasp poses as ground-truth information for training GP-net+.

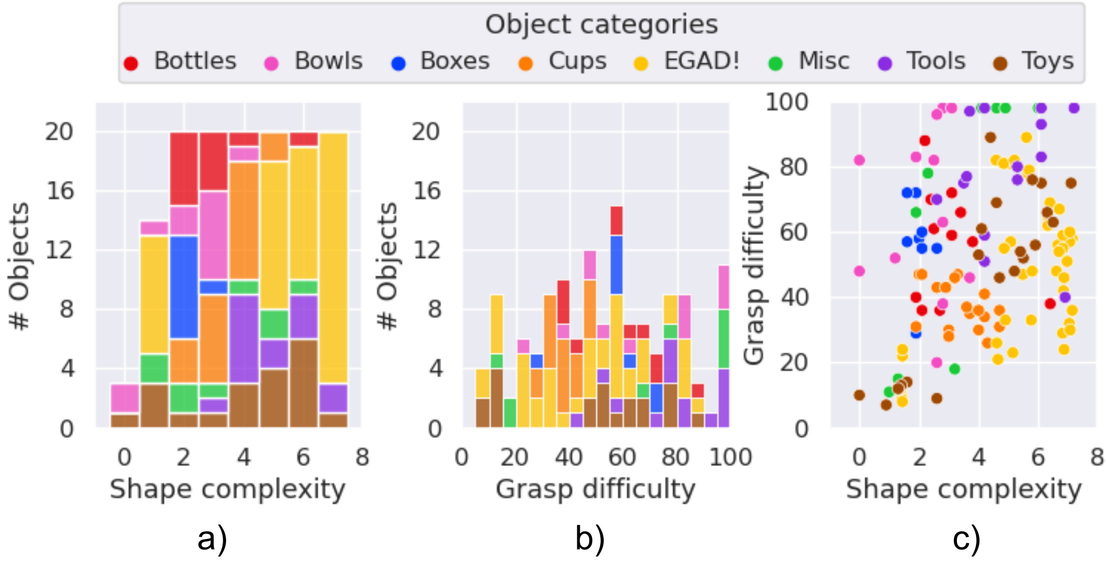


Figure 5.4: a) Shape complexity and b) grasp difficulty for graspable objects in our entire dataset with object categories. c) shows the combined scatter plot of shape complexity and grasp difficulty values for each object.

5.2.1 Graspable objects

The graspable objects should include object meshes of different complexity and variety as encountered in real domestic grasping scenarios. To achieve this, we used a selection of 137 object meshes from the YCB [85], BigBIRD [86], ShapeNet [160] and EGAD [74] datasets.

While YCB includes a wide variety of challenging objects found in household environments, BigBIRD mostly contains rectangular and cylindrical instances of various packaged household items. ShapeNet enhances the object set with various mugs and bowls, of which only very few instances are included in YCB and BigBIRD. Finally, EGAD supplies objects with highly complex and irregular shapes to prevent the object shapes from being too simple. We kept the original sizes of YCB and BigBIRD objects and reverted the normalisation of the ShapeNet objects. According to the process described in [74], we scaled the EGAD objects for our maximum gripper size of $0.08m$. Overall, the mean size of the bounding boxes of all objects in (x, y, z) was $(0.097, 0.102, 0.084)m$ with standard deviations of $(0.061, 0.062, 0.049)m$.

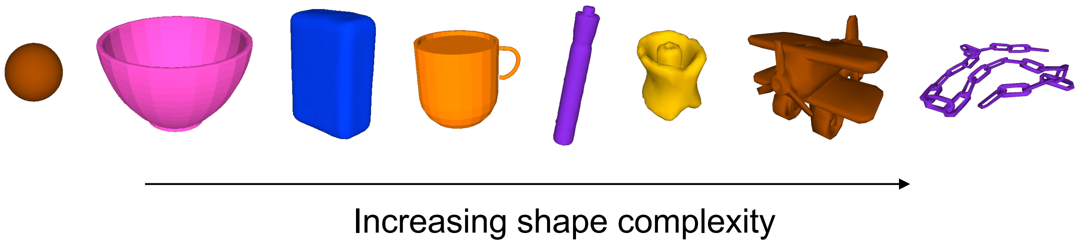


Figure 5.5: Examples of graspable objects used in the domestic grasping scenes with increasing shape complexity values as defined in EGAD [74]. The colours of the objects represent the object categories also used in Figure 5.4.

We calculated the shape complexity for each object mesh in the object sets as described in EGAD [74]. We used the shape complexity values to sample a subset of graspable objects from YCB, BigBIRD and ShapeNet. The subset contained 20 objects for every shape complexity value of $0, 1, 2 \ldots 7$, if enough examples were available in the datasets.

We added a total of 44 meshes from the EGAD training set for shape complexity intervals which did not have enough examples present in YCB, BigBIRD and ShapeNet, e.g. very complex shapes with a shape complexity of 7. Figure 5.4 a) shows the resulting distribution of shape complexity values and Figure 5.4 b) the corresponding distribution of grasp difficulty, as defined in EGAD [74][6].

The relationship between shape complexity and grasp difficulty across the dataset is visualised in Figure 5.4 c), showing the overall coverage of both dimensions. As expected and also described in EGAD [74], there were few objects with a very high grasp difficulty and very low shape complexity and vice versa. Apart from these extremes, the objects were balanced, covering a wide range of grasp difficulty and shape complexity values.

We manually assigned categories to all objects to give insight into their variety in the resulting dataset. An example of objects with increasing shape complexity is given in Figure 5.5.

[6] Note that we reverse the grasp difficulty values from the original implementation with $\textit{difficulty} = 100 - \textit{egad\_difficulty}$ to have objects that are more difficult to grasp have a greater grasp difficulty, to ensure consistency with the EGAD evaluation set and avoid confusion.

Using an antipodal grasp sampler [11] and the grasp sampling technique described in Section 4.3, we sampled an average of 377 grasps for each grasping object. To enable realistic physics simulations of concave object meshes, we leveraged a fully automatic algorithm for approximating a convex subset called "Volumetric Hierarchical Approximate Convex Decomposition" (V-HACD) [161]. V-HACD is also used when generating other robotic grasping datasets, e.g. [70, 77]. We performed V-HACD offline for each object mesh and used the result as collision mesh when generating our domestic grasping scenes, allowing for the simulation of concave object shapes.

5.2.2 Furniture units

In real-world domestic scenarios, objects are placed on various furniture units like tables, counters, sideboards, shelves or cupboards. To include a variety of furniture units in our domestic grasping scenes, we used furniture meshes from the ShapeNet [160] dataset. We uniformly sampled 50 tables and 50 shelves from ShapeNet, resulting in 100 furniture units $f \in \mathcal{F}$. The tables and shelves varied significantly in their appearance, ranging from bookshelves, cupboards and round coffee tables to sideboards, desks and dinner tables, see Figure 5.3. We subsequently use "tables"[7] for the collection of tables, sideboards, desks, and coffee tables and categorise bookshelves and cupboards as "shelves".

We added walls behind suitable furniture meshes, for example, bookshelves or desks, to make our scenes more realistic. To use the furniture units as support surfaces in our domestic grasping scenes, we defined two distinct properties: the object workspaces, i.e. where objects can be placed on the furniture, and the camera sampling spaces, i.e. how the camera can view the scene.

Object workspaces: We used blender [162] to define the object workspaces of a furniture unit. For each furniture object, we grouped adjacent faces parallel to the x-y plane, i.e. horizontal faces touching each other, into regions. We manually excluded unsuitable regions like rims and the underside of shelves since they can

[7] Note: some furniture objects classified as "tables" may have multiple support surfaces, for example, coffee tables with two levels or desks with multiple storage compartments.

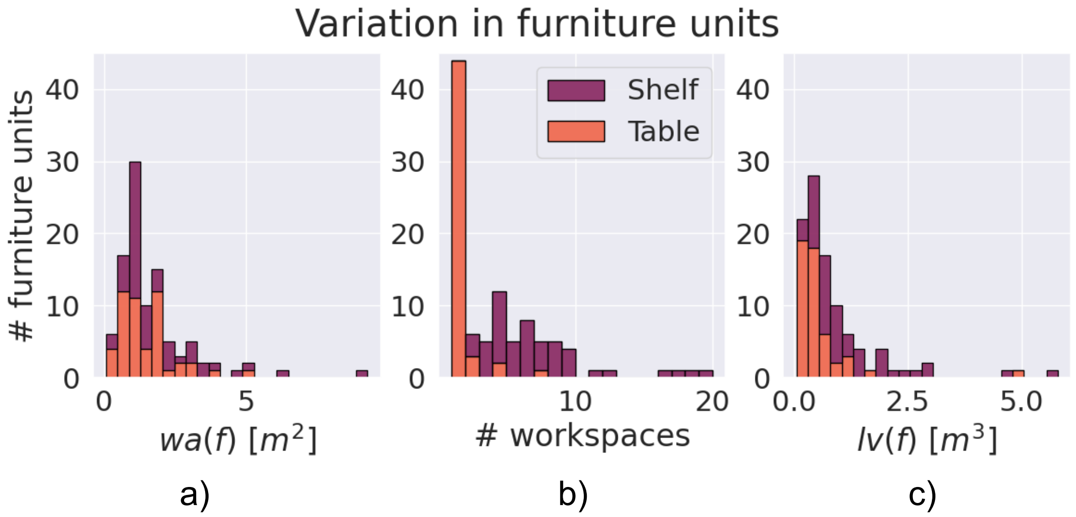


Figure 5.6: Variations in the furniture units used in our dataset, with their a) the total area of their object workspaces $wa(f)$, b) the number of object workspaces and c) the volume of their look-to-region $lv(f)$. The volume of the look-to-region results from the object workspaces and their position and is defined within the *camera sampling spaces*.

not support objects. We generated a convex hull for each region and saved them for loading and sampling during data generation.

Overall, the heights of workspaces on tables ranged from $0.061m$ to $1.07m$ (mean $= 0.63m$), and the heights of workspaces on shelves ranged from $0.01m$ to $2.18m$ (mean $= 0.84m$). The median number of workspaces was 1 (min = 1, max = 7) for tables, and 6 (min = 2, max = 20) for shelves. An overview of the variations of all furniture units in their total object workspace area $wa(f)$ and their number of object workspaces can be seen in Figure 5.6 a-b.

As the furniture units ranged from small coffee tables to large shelves, the area of the object workspaces varied significantly. To create a realistic density of objects on a given furniture unit, we defined the number of objects $n_o(f)$ based on the cumulative area of the object workspaces $wa(f)$.

Given a furniture unit f and the area of its object workspaces $wa(f)$, we calculated the expected number of objects o on the furniture unit as

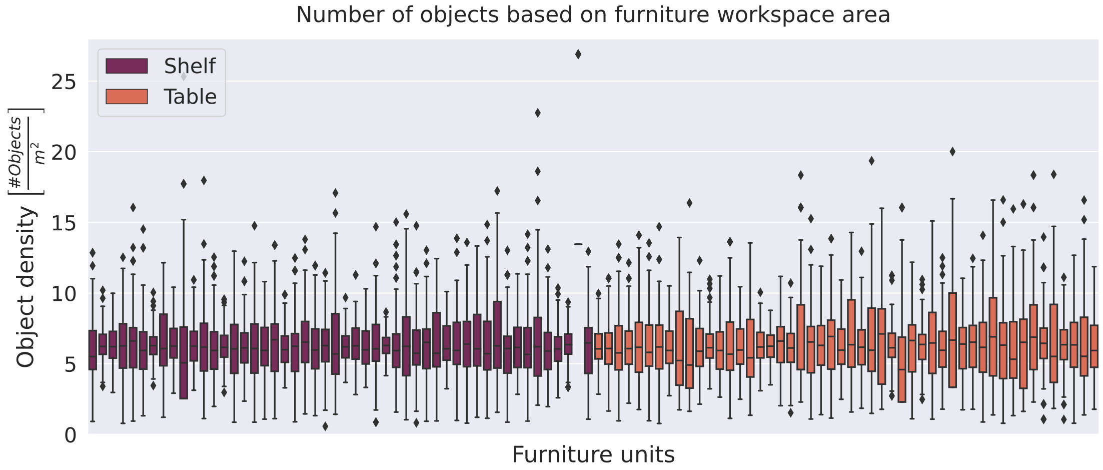


Figure 5.7: Object density for each furniture unit as observed when sampling the number of objects $n_o(f)$ for 200 times. Since the furniture units had a different total area of object workspaces (see Figure 5.6), we used the summed area of object workspaces $wa(f)$ to sample the number of objects $n_o(f)$ via a Poisson distribution. This resulted in an approximately constant *density of objects* on the object workspaces over the set of furniture units.

$$\lambda_o(f) = \frac{wa(f)}{0.16m^2} \tag{5.1}$$

resulting in a mean of one object in an area of $0.4m \times 0.4m$. We used $\lambda_o(f)$ as the expected value for a Poisson distribution and sampled from that distribution during dataset generation to obtain the number of objects $n_o(f)$ to be placed on any given furniture unit f. We further set a minimum number of objects of $\min n_o(f) = 1$ and a maximum number of objects of $\max n_o(f) = 100$ for any furniture unit. The sampling of $n_o(f)$ was actually performed during scene setup as described in Section 5.2.3.

Figure 5.7 shows how using the Poisson distribution resulted in a median of approximately 6 $Objects/m^2$ when used to sample the number of objects for a given furniture unit. (As the number of objects for each furniture unit is sampled for 200 different scenes in this figure, there is a range of object densities for each furniture unit.) The outlier shelf in the middle of the graph had a very small workspace area

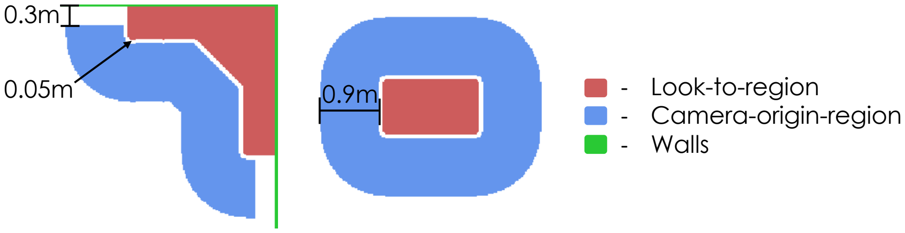


Figure 5.8: Examples of furniture units with their 2D camera sampling spaces, including the camera-origin-region, where the camera position is sampled, and the look-to-region, where the camera is pointed towards. Visualised as a top-down view of the scene.

$wa(f) = 0.07m^2$, which led to a significant jump in object density when adding a single additional object.

Camera sampling spaces: The camera sampling spaces define the range of camera poses for a given furniture mesh. Contrary to tabletop scenarios with a limited workspace where the camera pose is sampled from spherical coordinates, we aimed to sample camera poses in a robot-based manner. The resulting camera poses were more representative of those on a mobile manipulator operating in a real-world domestic scene.

We split the camera sampling spaces into a camera-origin-region and a look-to-region. From those regions, we sampled a camera-origin-point, i.e., a point in the camera-origin-region where the camera is positioned, and a look-to-point, i.e., a point in the look-to-region that the camera is directed towards. Examples of the camera-origin-regions, look-to-regions and their relation are depicted in Figure 5.8.

We defined both camera sampling spaces based on a concave hull of the footprint of a given furniture mesh. The camera-origin-region was created using disk dilation based on the concave hull of the furniture footprint, such that the robot is positioned in proximity to the furniture. The camera-origin-region had a minimum distance of $0.1m$ and a maximum distance of $1.0m$ to the mesh footprint, as well as a minimum distance of $0.3m$ to each wall. Since the camera should view the furniture unit to perceive objects placed on the furniture unit, the look-to-region was based on the concave hull of the mesh footprint dilated by $0.05m$, resulting in a minimum distance of $0.05m$ between the camera-origin-region and the look-to-region.

We pixelated the resulting 2D regions using a resolution of $0.05m$. When uniformly sampling from the regions, we augmented the resulting 2D positions with a uniformly sampled offset to compensate for the pixelation. The height of the camera-origin-point was uniformly sampled from the height range of the PAL TIAGo mobile manipulator, which can vary between $1.0m$ and $1.55m$. The look-to-point was uniformly sampled from the minimum and maximum height of the object workspaces extended by $0.2m$ above and $0.05m$ below.

Similar to the number of objects $n_o(f)$ to be placed on the furniture unit, we coupled the number of images $n_i(f)$ rendered for a scene to the scale and characteristics of the furniture unit. This was done since larger furniture units cover more space and variety than smaller ones. For example, the smallest furniture unit in our dataset had a look-to-region volume $lv(f)$ of $< 0.1m^3$ and the largest furniture unit had a look-to-region volume $lv(f)$ of $> 5.0m^3$. The increased look-to-region and, consequently, increased camera-origin-region resulted in an overall increased number of possible camera viewpoints. We used the volume of the look-to-region $lv(f)$ to calculate the expected number of rendered images of a furniture scene $n_i(f)$ as

$$\lambda_i(f) = \frac{lv(f)}{0.01m^3} \tag{5.2}$$

which resulted in 1 expected image to be rendered for a furniture volume of roughly $0.2m \times 0.2m \times 0.2m$. We used $\lambda_i(f)$ as the expected value for a Poisson distribution and sampled the number of images to render for a given furniture unit $n_i(f)$ during dataset generation. We set a minimum number of images of $\min n_i(f) = 10$ for any furniture unit.

Figure 5.9 shows how using the Poisson distribution results in a median of approximately 100 $Images/m^3$ when using it to sample the number of images $n_i(f)$ for a given furniture unit f, despite having vastly different furniture scales, as shown in Figure 5.6 c). The outlier shelf in the middle of the graph was the

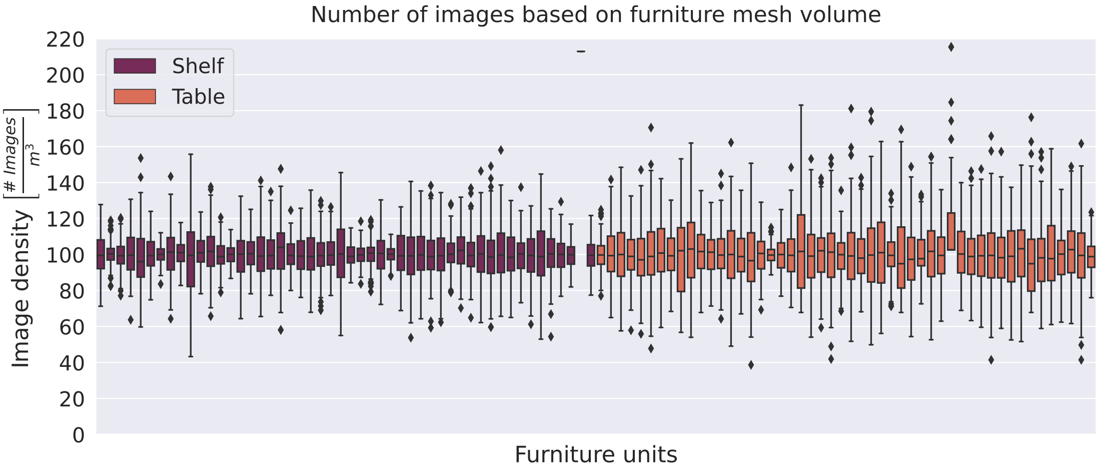


Figure 5.9: Image density for each furniture unit as observed when sampling the number of images $n_i(f)$ for 200 times. Since the furniture units had different characteristics and scales (see Figure 5.6), we used the volume of the look-to-region $lv(f)$ to sample the number of images $n_i(f)$ via a Poisson distribution. This resulted in an approximately constant *density of images* over the set of furniture units.

same as the outlier in 5.7, which was very small with a look-to-region volume of $lv(f) = 0.047m^3$.

All graspable objects and all furniture units were separated into a 70-15-15 training-validation-testing split. Moreover, this point represented the conclusion of all pre-processing steps for the furniture units, which were used as the basis for generating the domestic grasping scenes.

5.2.3 Simulating domestic grasping scenes

We used PyBullet [114] for simulating domestic grasping scenes utilising the graspable objects and furniture units described above. We describe as a domestic grasping scene a simulated environment where objects have been placed on a furniture unit. Simulating the scenes consists of three parts: scene setup, grasp testing and grasp contact projection.

Scene setup: An algorithm for the scene setup is shown in Algorithm 5.1. We started to generate the scene by uniformly choosing a furniture unit f and placing

Algorithm 5.1 Constructing a domestic grasping scene with PyBullet

```
 1: Uniformly choose one of the furniture units $f \in \mathcal{F}$ and place it at wo $= (0, 0, 0)$
 2: Sample number of objects $n_o(f)$, setting $\min n_o(f) = 1$ and $\max n_o(f) = 100$
 3: $max\_attempts = 2 \times n_o(f)$
 4: while $current\_objects < n_o(f)$ $AND$ $attempts < max\_attempts$ do
 5:     Uniformly sample one of the object meshes $o \in \mathcal{O}$
 6:     Augment object size with scaling factor $s \in [0.8; 1.2]$
 7:     Sample $x, y, z$ from the convex object workspaces
 8:     With a probability of 50%, use upright pose and randomly rotate around
        $z$, else uniformly sample random rotation matrix $\mathbf{R}$
 9:     Place object $o$ in the chosen pose such that all its parts are at least $0.01m$
        above the workspace
10:     if Object velocity $> 0.1m/s$ after $3.0s$ OR object outside of workspace
           OR object intersects with other mesh then
11:         Remove object
12:     end if
13:     $attempts += 1$
14: end while
15: Sample number of images $n_i(f)$, setting $\min n_i(f) = 10$
16: $viewpoint\_counter = 0$, $rendered\_images = 0$
17: while $rendered\_images < n_i(f)$ do
18:     Sample camera origin $\mathbf{p}_c^{\text{wo}}$ and look-to-point $\mathbf{p}_l^{\text{wo}}$ from camera regions
19:     Calculate camera extrinsic from camera origin $\mathbf{p}_c^{\text{wo}}$ and look-to-point $\mathbf{p}_l^{\text{wo}}$
20:     Render a depth image and segmentation image
21:     Add noise to the depth image
22:     $viewpoint\_counter += 1$
23:     if At least one object visible $OR$ $viewpoint\_counter > 25$ then
24:         Save noisy depth image, segmentation image, and camera pose
25:         $rendered\_images += 1$
26:     end if
27: end while
```

it in the simulation environment on the ground plane. Based on the furniture unit, we sampled the number of objects to be placed, $n_o(f)$ from a Poisson distribution with $\lambda_o(f)$ as described in Eq. 5.1, see Algorithm 5.1 line 2.

We uniformly sampled a graspable object from the object set and placed it at a uniformly sampled position within the furniture's object workspaces. The orientation of the objects is not expected to be entirely random in a domestic environment, for example, with bottles and mugs usually standing upright. To cater for these circumstances, we chose to use a hybrid of the orientation in the “piled” and “packed” procedures in VGN [16]. Approximately half the time (50% probability), objects were placed upright and randomly rotated around their z-axis (packed). The remainder of the time, objects were oriented according to a uniformly sampled random rotation matrix (piled), as described in Algorithm 5.1 line 8.

We dropped the object from $1cm$ above the workspace in the sampled pose. We ran the physics of the simulation environment to let it come to a stable resting pose by monitoring the object’s velocity. Objects were added sequentially, waiting for each object to rest before the next object was added. Furthermore, we removed objects that intersected with the furniture or other objects, fell to the ground, or were balancing over the edge of the object workspace at any point.

For a given scene, we rendered $n_i(f)$ images based on the Poisson distribution with $\lambda_i(f)$ as described in Eq. 5.2. For each rendered image, we sampled the camera’s extrinsic parameters based on the camera sampling spaces as described in Section 5.2.2, see Algorithm 5.1 lines 18-19. We rendered and saved the depth image and segmentation mask based on the sampled camera pose. We added artificial noise to the depth image as described in [154, 155]. In Chapter 4, we found adding artificial depth noise to improve model performance by reducing the sim-to-real gap.

Grasp testing: After a scene has been created with Algorithm 5.1, we proceeded to grasp testing to generate ground-truth grasp information, see Algorithm 5.2. We uniformly chose up to $n_g(o) = 300$ of the pre-sampled grasps g for each object o in the scene. We ran an initial check for each grasp to see if the gripper would be in collision when placed at the grasp pose. If the grasp pose was collision-free, grasp success was checked by approaching the grasp pose along the grasp z-axis with a PAL parallel-jaw gripper, closing the gripper with a maximum force of $5N$, moving the gripper vertically upwards and checking if it managed to lift the object from the support surface.

We repeated this process a total of 12 times for each grasp pose while rotating the gripper around ω (the grasp contacts) for $0 \ldots 360°$ in steps of $30°$, see Algo-

Algorithm 5.2 Evaluating grasp success in a given scene using a PyBullet simulation.

```
1:  for each object $o$ in scene do
2:      Load ground-truth grasps $G(o)$ and uniformly choose up to 300 of them
3:      for each grasp $g$ in chosen ground-truth grasps do
4:          for $0 \ldots 360°$ rotation of $g$ around x-axis in steps of $30°$ do
5:              if gripper does not collide AND lifting object is successful then
6:                  $\varrho(g) = 1$                  # Successful ground-truth grasp
7:              else
8:                  $\varrho(g) = 0$                # Unsuccessful ground-truth grasp
9:              end if
10:         end for
11:         Choose median grasp of neighbouring successful approach directions
12:         For multiple neighbouring successful approach direction clusters, store
            all median grasps
13:     end for
14: end for
```

rithm 5.2 line 4. This was done since the pre-sampled grasps of an object were defined by the grasp contacts and could vary in their approach direction $g.z$ (z-axis of the grasp). If any of the attempted grasps was successful, we used the median successful grasp orientation as the default orientation of the grasp, as shown in Algorithm 5.2 lines 5-11.

There could also be multiple neighbouring successful grasp approach direction clusters. For example, a bottle could be grasped from the front and rear of the object, but top-grasps could be in collision with the bottle itself. In such cases, we stored all median collision-free grasp orientations and decided the final orientation when projecting the grasp contact points into the image in Algorithm 5.3 lines 10-13.

Testing the grasps in simulation, rather than using the analytical metrics calculated during grasp sampling in Section 5.2.1, has the advantage of checking for collisions between the objects, gripper and furniture in the generated scenes. Since we worked with objects in clutter and complex furniture units, collision checking in the final scene was crucial for filtering unsuitable grasp poses.

Algorithm 5.3 Projecting grasp contacts into the image, checking their visibility and saving grasp data using a PyBullet simulation.

```
 1: for each image of the $n_i(f)$ previously rendered images do
 2:     for each stored grasp do
 3:         Calculate grasp contact points $(\mathbf{p}^c_{gcp,1}, \mathbf{p}^c_{gcp,2})$ in camera frame $c$
 4:         Choose point closer to camera $c$ with $\mathbf{p}^c_{gcp} = \min |\mathbf{p}^c_{gcp,1}|, |\mathbf{p}^c_{gcp,2}|$
 5:         Project grasp contact point $\mathbf{p}^c_{gcp}$ into image coordinates $(u, v)$
 6:         Skip grasp if coordinates $(u, v)$ outside of image boundaries
 7:         De-project image coordinates $(u, v)$ to $\mathbf{p}^c_{d\_gcp}$
 8:         Calculate distance between ground-truth and de-projected grasp con-
            tact point $dist = |\mathbf{p}^c_{gcp} - \mathbf{p}^c_{d\_gcp}|$
 9:         if $dist < 0.01m$ AND ray-test yields correct object then
                # Check if earlier grasp $g2$ is already stored at $(u, v)$
10:             if $g2$ exists at $(u, v)$ AND $\varrho(g) \leq \varrho(g2)$ then
11:                 if $\varrho(g) == 1$ AND $\Psi(g) < \Psi(g2)$ then
12:                     Overwrite $g2$ with $g$
13:                 end if
14:             else
15:                 Save grasp $g$
16:             end if
17:         end if
18:     end for
19: end for
```

Projecting grasp contacts: After all grasps in the scene have been labelled, we checked the visibility of their grasp contact points in each of the $n_i(f)$ rendered images, as described in Algorithm 5.3. Of the two grasp contact points $(\mathbf{p}^c_{gcp,1}, \mathbf{p}^c_{gcp,2})$ in the camera frame c, we selected the one closer to the camera origin, $\mathbf{p}^c_{gcp}$. This is the contact point that is visible in the camera frame unless obstructions occlude the surface point.

Since the grasp contact point is later identified by its image coordinates, we used the camera intrinsics to project the grasp contact point $\mathbf{p}^c_{gcp}$ into the 2D image, resulting in the image coordinates (u, v), see Algorithm 5.3 lines 3-6. Next, we needed to assess if the grasp contact point was visible in the image, for which we had two distinct criteria. First, we checked the Euclidean distance, *dist*, between the original grasp contact point $\mathbf{p}^c_{gcp}$ and the deprojected grasp contact point $\mathbf{p}^c_{d\_gcp}$

of the image coordinates (u, v), see Algorithm 5.3 lines 7-9. If $dist$ exceeded $0.01m$, we deemed the grasp contact point invisible due to the large aliasing error induced when projecting the point into the image. Secondly, a ray-test from the camera origin to the grasp contact point $\mathbf{p}^c_{gcp}$ had to return the object ID connected to the ground-truth grasp, as opposed to the ID of another object or the furniture unit. If both criteria were met, we deemed the grasp contact visible.

If the grasp contact was visible, we saved the image coordinates (u, v), grasp width $\mathrm{w}(g)$, orientation $\mathbf{r}(g)$ and success $\varrho(g)$ of the grasp. If more than one grasp was visible at a given set of image coordinates (u, v), we saved the successful grasp g whose approach direction was most aligned with the principal ray of the camera - resulting in a smaller Ψ - see Algorithm 5.3 lines 10-15. Using this process, we preferred grasps approaching the object from the front to those approaching the object from the rear. We repeated this process to save grasps for each depth image in a given scene. The resulting dataset with training and validation data consists of $542,055$ images with an average of 108.6 ground truth grasps visible in each scene, of which an average of 10.5 grasps are labelled as successful (i.e. ground-truth positive).

5.3 GP-net+

GP-net+ is an extension of the Grasp Proposal Network, which we introduced in Chapter 4. We made a number of changes to the loss function as well as several small changes to the model input and output to improve performance. An overview of the pipeline for GP-net+ is depicted in Figure 5.10. Similar to GP-net, GP-net+ consists of a FCNN with a ResNet-50 architecture pre-trained on ImageNetV2. We converted the input depth image into 3 channels by applying a jet colourmap using fixed normalisation boundaries of $0.2m$ and $1.5m$.

As mentioned in Section 4.2, the fixed normalisation boundaries balance the resolution and perceivable depth in the input image. Compared to GP-net, the normalisation boundaries for GP-net+ were increased by $0.3m$ to broaden the visible range. We found drops in performance when increasing the normalisation boundaries further than that, likely due to the reduced resolution in the input

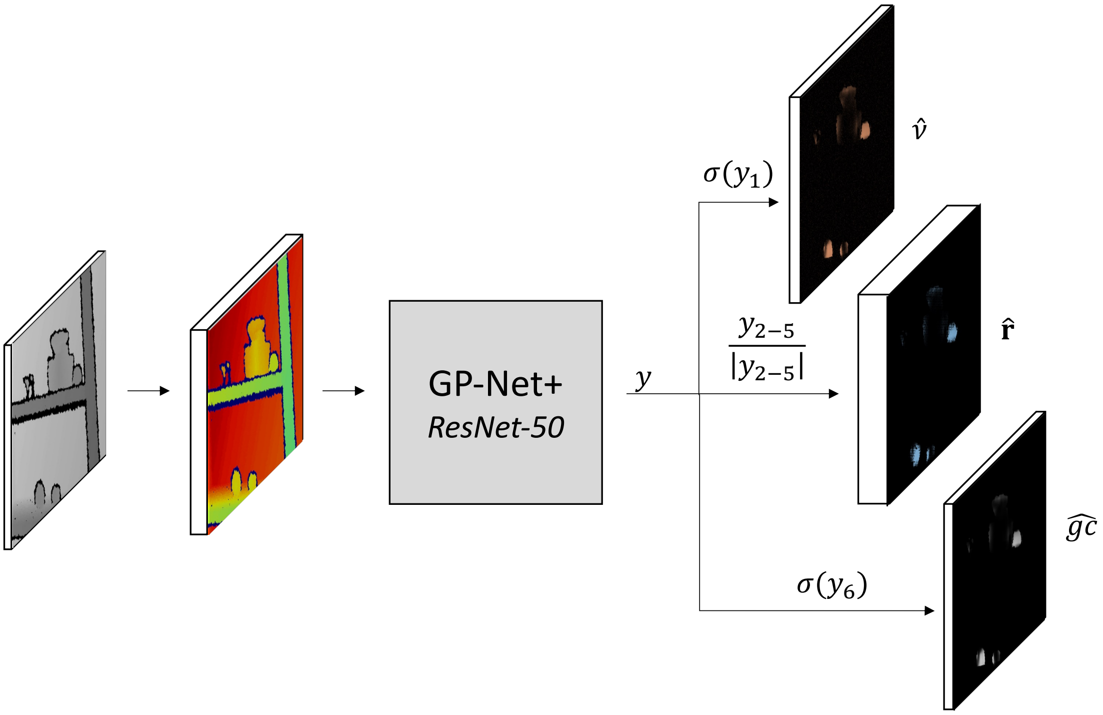


Figure 5.10: Pipeline of GP-net+ grasp proposal. Using a depth image as input, we apply a jet colourmap, run inference on the resulting RGB image with GP-net+ and output a dense tensor y with 6 channels. The channels comprise a predicted relative grasp width $\hat{\nu}$, a normalised unit quaternion for a predicted grasp orientation $\hat{\mathbf{r}}$ and a predicted grasp confidence $\hat{gc}$, normalised to the range $[0; 1]$ by a sigmoid function σ.

image.

In contrast to GP-net, we defined the grasp width in the output tensor of GP-net+ relative to the maximum gripper width and denoted it as ν. This was done since all successful grasps (as labelled in Algorithm 5.2), which are the only ones used to learn the width in the model, have a maximum width of $\max(\mathrm{w}) = 0.08m$ for a PAL parallel-jaw gripper. The relative width scales the output to the $[0; 1]$ range, thereby assimilating the scale of possible width values with the remainder of the model outputs, which we hypothesised would improve the learning process. In the model architecture, we added a sigmoid function to the width output and used the maximum grasping width as a scaling factor when mapping the model output to grasp proposals in Section 5.3.2.

In general, a pixel i in our ground-truth data could have one of three states: It could hold a successful grasp, where $\mathbb{1}_{SuccessGrasps}(i) = 1$, a negative ground-truth grasp, where $\mathbb{1}_{NegGrasps}(i) = 1$, or no ground-truth grasp information, where $\mathbb{1}_{SuccessGrasps}(i) = 0$ and $\mathbb{1}_{NegGrasps}(i) = 0$. Pixels with ground-truth negative grasps could either be based on tested, unsuccessful grasps on the objects or non-object pixels in the depth image based on the segmentation mask. Pixels with no ground-truth information were present on invisible parts of the depth image, e.g. where there was no depth information due to sensor noise, the normalisation bounds or object pixels without visible ground-truth grasp contact points.

5.3.1 Loss function

The most significant differences between GP-net and GP-net+ can be found in the loss function. In GP-net's loss function (see Equation 4.1), the loss for grasp success $\mathcal{L}_{\varrho}$ is a binary cross-entropy loss between the ground-truth and predicted grasp success. It is entirely independent of the second configuration part of the loss function and thereby does not consider the predicted grasp width $\hat{\mathrm{w}}$ or the predicted grasp orientation $\hat{\mathbf{r}}$. However, the configuration of the grasp is an important factor in the grasp prediction, where small changes in both variables can change a successful grasp proposal to an unsuccessful one.

To counter this behaviour and inspired by the box confidence in the YOLOv2 [163] loss function, we changed the predicted grasp success $\hat{\varrho}$ output from GP-net to a predicted grasp confidence $\hat{gc}$. The ground-truth grasp confidence gc was calculated as a measure of how well the predicted grasp pose with relative width $\hat{\nu}$ and orientation $\hat{\mathbf{r}}$ matched the ground-truth grasp pose. To measure how well the orientation matched the ground truth, we used the angular distance a for the predicted and ground-truth quaternions $a(\mathbf{r}, \hat{\mathbf{r}}) = \frac{2}{\pi} \arccos |\mathbf{r} \cdot \hat{\mathbf{r}}|$, normalised to 1 by dividing the term with π.

To measure how well the predicted relative grasp width $\hat{\nu}$ matched the ground truth, we calculated the absolute error between the predicted and the ground-truth value. We weighted the sum of the grasp orientation and relative grasp width error with a factor of 2 to increase the resolution of the grasp confidence gc when approaching a configuration close to the ground-truth grasp configuration.

Overall, this resulted in the ground-truth grasp confidence gc for grasp at pixel i being calculated as:

$$gc_i = \begin{cases} 1 - \min(1, 2 \cdot (a(\mathbf{r}_i, \hat{\mathbf{r}}_i) + |\nu_i - \hat{\nu}_i|)) & \text{if } \mathbb{1}_{SuccessGrasps}(i) = 1 \\ 0 & \text{if } \mathbb{1}_{SuccessGrasps}(i) = 0 \end{cases} \tag{5.3}$$

This caused the ground-truth grasp confidence gc to approach 1 for grasps with a prediction similar to the ground-truth orientation $\mathbf{r}$ and relative width ν. For bad predictions of the grasp configuration, the grasp confidence gc approached 0. If no successful grasp was available at pixel i, the ground-truth grasp confidence was $gc_i = 0$. The overall loss function consists of three parts:

- $\mathcal{L}_{SuccessGrasps}$ evaluates the predicted grasp confidence for pixels with a successful ground-truth grasp.
- $\mathcal{L}_{NegGrasps}$ evaluates the predicted grasp confidence for pixels with ground-truth negative grasp samples. Ground-truth negative grasp samples include tested, unsuccessful grasps on the objects and background pixels based on the segmentation mask of the scene.
- $\mathcal{L}_{Config}$ evaluates the predicted grasp orientation $\hat{\mathbf{r}}$ and the predicted relative grasp width $\hat{\nu}$ for pixels with a successful ground-truth grasp.

We used the distance metric for quaternions defined by Kuffner et al. [153] and used in VGN and GP-net, as well as the mean-squared error for the relative width ν in the configuration loss, resulting in:

$$\mathcal{L}_{SuccessGrasps} = \sum_{i}^{all\_pixel} \mathbb{1}_{SuccessGrasps}(i) \, (gc_i - \hat{gc}_i)^2 \tag{5.4}$$

$$\mathcal{L}_{NegGrasps} = \sum_{i}^{all\_pixel} \mathbb{1}_{NegGrasps}(i) \, \hat{gc}_i^2 \tag{5.5}$$

$$\mathcal{L}_{Config} = \sum_{i}^{all\_pixel} \mathbb{1}_{SuccessGrasps}(i)\left(1 - |\mathbf{r}_i \cdot \hat{\mathbf{r}}_i| + (\nu_i - \hat{\nu_i})^2\right) \quad (5.6)$$

$$Loss = \lambda_{SuccessGrasps}\,\mathcal{L}_{SuccessGrasps} + \lambda_{NegGrasps}\,\mathcal{L}_{NegGrasps} + \lambda_{Config}\,\mathcal{L}_{Config} \quad (5.7)$$

We set $\lambda_{NegGrasps} = 1$, $\lambda_{SuccessGrasps} = 100$ and $\lambda_{Config} = 2$, which we found experimentally to give a good balance between the individual losses. Note that we had a substantially higher number of ground-truth negative grasps than successful ground-truth grasps and summed the losses over all pixels, motivating a high $\lambda_{SuccessGrasps}$ to counter that imbalance. We trained GP-net+ for 8 epochs using an Adam Optimiser with a learning rate of $3e^{-4}$.

5.3.2 Mapping the model output to grasp proposals

Mapping the dense output tensor of GP-net+ to grasp proposals worked similarly to GP-net, see Section 4.2.2. Since GP-net+ outputs a predicted grasp confidence $\hat{gc}$ as opposed to a predicted grasp success $\hat{\varrho}$, the predicted grasp confidence was used for selecting grasp proposal pixels in the output tensor. We defined an acceptance threshold $\gamma_{\hat{gc}} = 0.29$, which we set experimentally based on the results in Section 5.5.1.

Another difference to the grasp proposal mapping for GP-net was rooted in the grasp width. In contrast to taking the absolute predicted grasp width $\hat{\mathrm{w}}$ directly to translate the grasp contact point to the TCP, we calculated it based on the predicted relative grasp width $\hat{\nu}$. This was done by multiplying the predicted relative grasp width $\hat{\nu}$ with the maximum gripper width $\max(\mathrm{w}) = 0.08m$ for a PAL parallel-jaw gripper.

5.4 Experiments

We evaluated the performance of GP-net+ in three stages using a tabletop simulation (sim-tabletop), a domestic simulation (sim-domestic) and a real-world domestic experiment (real-domestic). The combination of simulated and real-world experiments enabled us to evaluate different aspects of performance. For example, simulated experiments can be repeated in a controlled manner and be used to compare multiple models under the exact same conditions. In contrast, real-world

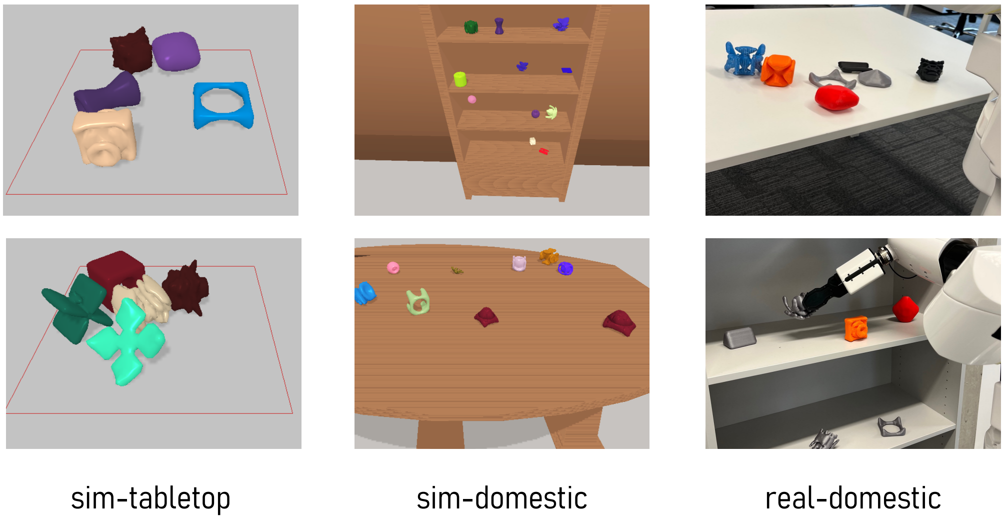


Figure 5.11: Overview of the different experiments conducted to evaluate GP-net+ and compare it against other algorithms, with tabletop simulations (sim-tabletop), domestic simulations (sim-domestic) and a real-world domestic experiment (real-domestic). The images show all experiments with the EGAD evaluation set. We tested three additional object sets in the experiments: a "simple" set (sim-tabletop), a "custom" set (sim-tabletop, sim-domestic) and a "household" set (real-domestic).

experiments evaluate the performance under real-world conditions without the assumptions and simplifications applied in a simulation environment. Figure 5.11 shows a visualisation of the different experiments.

Up to now, most research in robotic grasping is trained and evaluated using tabletop scenarios [11, 12, 13, 16, 17, 18, 19, 20, 21, 23, 25, 37, 38, 42, 62, 164]. To ensure GP-net+ is performing well in such scenarios, we compared its performance relative to VGN [16], GPD [19] and GP-net (see Chapter 4) in a tabletop simulation experiment (sim-tabletop). VGN and GPD represent two established grasp proposal methods that are frequently used as points of comparison in robotics research [16, 17, 21, 38, 46].

In a second experiment (sim-domestic), we examined how GP-net+ extends to domestic scenarios. Since domestic scenarios are more complex and variable in their object, furniture and camera placement than tabletop scenarios, we expected the performance of the models to drop relative to our first set of experiments. The simulated domestic scenarios contained objects placed in structured clutter on tables and shelves. To show the importance of using appropriate training data and models when applying grasp proposal solutions to domestic scenarios, we compared the performance of GP-net+ to the performance of GPD and GP-net on the same domestic scenarios.

Finally, we performed a real-world experiment (real-domestic) with a PAL TIAGo mobile manipulator. This real-world experiment indicated the performance of GP-net+ under real-world conditions. Such conditions include sensor noise, actuation noise, and end-effector reachability and exclude many of the assumptions and simplifications prevalent in simulated experiments.

For all experiments, we describe as a *scene* a simulated or real-world environment where objects have been placed on a furniture unit or planar surface. Multiple grasping *trials* can be executed for one scene. Each trial involves constructing a scene representation (e.g. by taking a depth image) and executing a single grasp, given that at least one grasp has been proposed by the model.

We employed four distinct evaluation sets in our experiments. Some of the evaluation sets were a constellation of virtual 3D object meshes designed for simulation purposes (virtual), and some comprised real-world objects suitable for manipulation by a physical robot (physical). The evaluation sets are: a *simple* evaluation set containing simple shapes like cuboids and cylinders (virtual, used in sim-tabletop)[8], a *custom* evaluation set based on the test split of the graspable objects described in Section 5.2.1 (virtual, used in sim-tabletop and sim-domestic), the *EGAD* [74] evaluation set (virtual and physical, used in all experiments) and a *household* evaluation set described in Section 5.4.3 (physical, used in real-domestic).

Using these varied evaluation sets allowed us to evaluate the strengths and weaknesses of the models in more detail. In the tabletop simulations (sim-tabletop) experiment, the *simple* evaluation set produced good results for all models, while the more complex *custom* evaluation set revealed distinct differences among the

[8] The simple evaluation set was first proposed in VGN [16], where it is called “blocks”

models in the same experiment. As a special case, the EGAD evaluation set was evaluated in all experiments (sim-tabletop, sim-domestic and real-domestic) and exhibited differences in performance as the complexity of the experiments increased. In the remainder of this section, we describe the setup of our experiments in detail, with the results for each experiment represented in Section 5.5.

5.4.1 Tabletop simulation experiment (sim-tabletop)

The tabletop simulation setup consisted of a $30 \times 30cm^2$ workspace positioned on top of a planar surface such that the workspace did not extend beyond the planar surface. During scene setup, n objects were dropped in a uniformly sampled pose in the workspace, corresponding to the "piled" procedure described Section 5.2.3. We evaluated two different numbers of objects with $n = 5$ and $n = 1$ objects to test cluttered scenarios and provide a basis of comparison to GP-net in Chapter 4 with single objects. We tested three different object sets in the experiment: the simple, custom and EGAD [74] evaluation set.

We generated 500 scenes for each configuration of evaluation sets and the number of objects n, totalling $500 \times 3 \times 2 = 3000$ scenes. We tested all models on those same scenes. In this experiment, we rendered images from camera poses based on spherical coordinates from the centre of the workspace, with radial distance $d = 0.6m$, inclination (or polar) angle $\varphi = 60°$ and the azimuth (or azimuthal) angle sampled uniformly from $\theta \in [0°, 360°]$ (identical to VGN [16] for comparability).

Identical to the generation of our training data in Section 5.2.3, we added noise [154, 155] to the rendered depth images. We used the noisy depth images for GP-net+ and GP-net. We also used the noisy images to generate the point cloud for GPD and the TSDF for VGN. Furthermore, we segmented the object points in the point cloud using the ground-truth segmentation mask. This segmentation was used on GPD's input point cloud to prevent grasp proposals on non-object points, e.g. the ground plane. We call this process *grasp-guiding* in our experiments. While this gives an advantage to GPD compared to the other models, it is a necessary pre-processing step for applying GPD in our experiment.

We tested all algorithms based on sensor readings from a single camera pose, not applying noise reduction techniques or stitching point clouds. An exception was VGN, where we additionally tested performance with a higher quality TSDF scan incorporating images without added noise from 6 camera viewpoints spaced at $\Delta\theta = 60°$ to reproduce the original setup for VGN [16]. We indicated this reproduced setup as "VGN*" in our results.

For each trial in each model, we executed the grasp proposal with the highest predicted grasp quality and logged the grasp as successful if the gripper was able to lift the object above the support surface. If two consecutive trials failed or, alternatively, if all objects were grasped successfully, we moved to the next scene. VGN was originally trained with a Franka Emika Panda gripper [16]. While we showed in Section 4.6 that a PAL parallel-jaw gripper can approximate a Franka Emika Panda gripper, the simulation environment enabled us to test VGN with the original gripper without requiring hardware modifications on a physical robot. Therefore, we decided to test VGN with the Franka Emika Panda gripper in our simulated tabletop experiments.

5.4.2 Domestic simulation experiment (sim-domestic)

To simulate the domestic experiments, we used the same PyBullet simulation environment that had been used to generate our training data (described in Section 5.2) but with new test scenes. Using the test split of the furniture dataset, which included a total of 15 shelves and tables, we generated 300 scenes, all with different compositions of objects, object poses and furniture units. We tested two different evaluation sets in our domestic simulation experiment: the custom and EGAD [74] evaluation set.

We rendered noisy depth images from sampled robot-based camera poses (as described in Section 5.2.2) in each scene, proposed grasps based on the image using GP-net+ and selected the grasp proposal with the highest predicted grasp quality. We executed the grasp in the simulation using a PAL parallel-jaw gripper and labelled it as successful if the gripper managed to lift the object above the support surface when it was moved vertically upwards by $0.05m$. We aborted a scene and moved to the next scene after 5 consecutive failed trials or if all objects had been grasped successfully.

We compared GP-net+ to GPD and GP-net using these domestic simulations. To apply both algorithms to such scenarios with complex furniture geometries, we used grasp-guiding to prevent them from proposing grasps on non-object points in the respective scene representation, i.e., we guided their grasp proposals exclusively to object points. For GPD, we indexed all object points in the point cloud as the region of interest for grasp sampling, as described in [24]. To do so, we utilised the ground-truth segmentation mask of the PyBullet simulation and used it when de-projecting the noisy depth image into the point cloud. This ensured grasps were proposed on the objects instead of the furniture unit. For GP-net, we filtered the output tensor of the model using the ground-truth segmentation mask. In doing so, we set the predicted grasp success $\hat{\varrho}$ of non-object pixels to zero, resulting in such pixels not being selected for grasp proposals when applying NMS (see Section 4.2).

While this gave GPD and GP-net some advantage with additional ground-truth information unavailable to GP-net+, it is a necessary and reasonable workaround when applying more "conventional" tabletop grasp proposal algorithms to domestic scenarios. For example, a mobile manipulator collecting garbage from the ground, GarbageNet [129], uses a similar approach to define grasp sampling regions with GPD. Naturally, the performance of such a workaround in the real world depends on the quality of the segmentation mask. The provision of extra information to GPD and GP-net should be considered when interpreting the results.

5.4.3 Real-world experiment (real-domestic)

We performed a real-world experiment with GP-net+ using a PAL TIAGo mobile manipulator. The real-world experiment could evaluate the performance under real conditions without many of the assumptions and simplifications used in the simulated experiments, for example, regarding object mass, inertia, friction and sensor noise. Furthermore, the real-world experiment used the full movement of the robot, in contrast to only simulating the end-effector. As a consequence, real-world limitations in terms of reachability and inaccuracies for path planning and motor control were incorporated.

The real-world experiment evaluated two separate evaluation sets: our household evaluation set and the EGAD [74] evaluation set. While the household evaluation set represented objects found in real domestic environments, the EGAD evaluation set enabled us to evaluate the performance of GP-net+ in the real world in comparison to our simulated domestic experiments (sim-domestic). We selected the 15 objects in the household evaluation set from ordinary objects available in local supermarkets and homes. While not directly obtained from any dataset, the set encompassed representative objects from all categories available in the YCB [85] object set with food, kitchenware, shape items, toys and tools. The household evaluation set can be seen in Figure 5.12. Note that for both the food can and the food box, the weight of the original contents ($> 0.5kg$), in combination with the friction between the PAL parallel-jaw gripper and object surface, caused the gripper to be unable to lift the objects. We, therefore, reduced their weight by removing the original contents and filling them with foam.

We used ROS noetic and ran GP-net+ based on depth images from an Intel RealSense D435 camera mounted on the robot's head for our real-world experiments as shown in Figure 4.5. We used the Intel RealSense D435 camera since it could perceive objects closer to the camera than the Orbbec Astra camera used in the TIAGo mobile manipulator by default. We also found the Intel RealSense camera to perform better on thin features, e.g. being able to perceive depth on thin objects like pens at a distance $< 0.4m$.

To prepare a scene for our real-world experiment, a human operator placed all objects on the furniture unit with a distance $d \leq 1.0m$ to the robot. We manually positioned the camera so that at least one of the objects was visible and started the automatic grasping trial process in the GP-net+ ROS node. The ROS node averaged over 10 consecutive depth images and input the resulting averaged depth image to GP-net+ in order to generate grasp proposals as described in Section 5.3. The resulting grasp proposals were sorted based on their predicted grasp confidence $\hat{gc}$. The automatic grasping trial process then attempted to plan a path to the pre-grasping pose of the grasp with the highest predicted grasp confidence. If no path could be planned for this grasp, the automatic grasping trial process proceeded to the next highest predicted grasp confidence until a path could be planned or no grasp proposals were left.

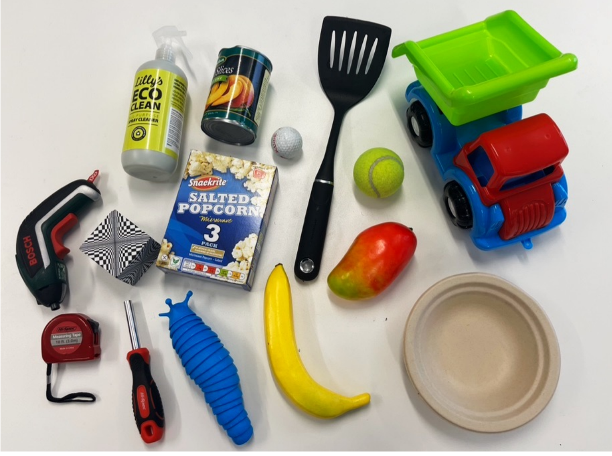


Figure 5.12: All 15 objects of the household evaluation set used for the real-world experiments, similar to objects from the YCB [85] object dataset. We include two or more objects from the categories of food items, kitchen items, shape items, toys and tools.

We planned paths using an OMPL [165] path planner in MoveIt! [72]. We further used OctoMap [71] based on the Intel RealSense point cloud with a resolution of $0.02m$ to build a planning scene for collision checking. If the pre-grasping poses could be reached, the grasp was executed with a joint movement to a pre-grasping pose $0.05m$ away from the grasping pose in negative z direction, a linear approach in positive z direction to the grasping pose and closing the gripper. Once the gripper was closed, the robot tried to lift the object above the support surface by moving the gripper vertically upwards for $0.05m$. If the gripper lifted the object above the support surface, the grasp was counted as a success.

If an object was successfully grasped, it was removed from the scene. Similar to the domestic simulation experiments, 5 consecutive failures to grasp any object caused the scene to be abandoned. When manually positioning the camera at the start of the trial, we ensured each object in a scene was visible in at least one trial. Half of the scenes feature a table as a furniture unit, and half the scenes contain a shelf. The composition of the objects for all scenes was allocated before the experiments by shuffling all available objects and selecting the objects sequentially from the shuffled list. Overall, each object appeared in 5 scenes with a table and 5 scenes with a shelf with no duplicate objects in a single scene. This resulted in 4

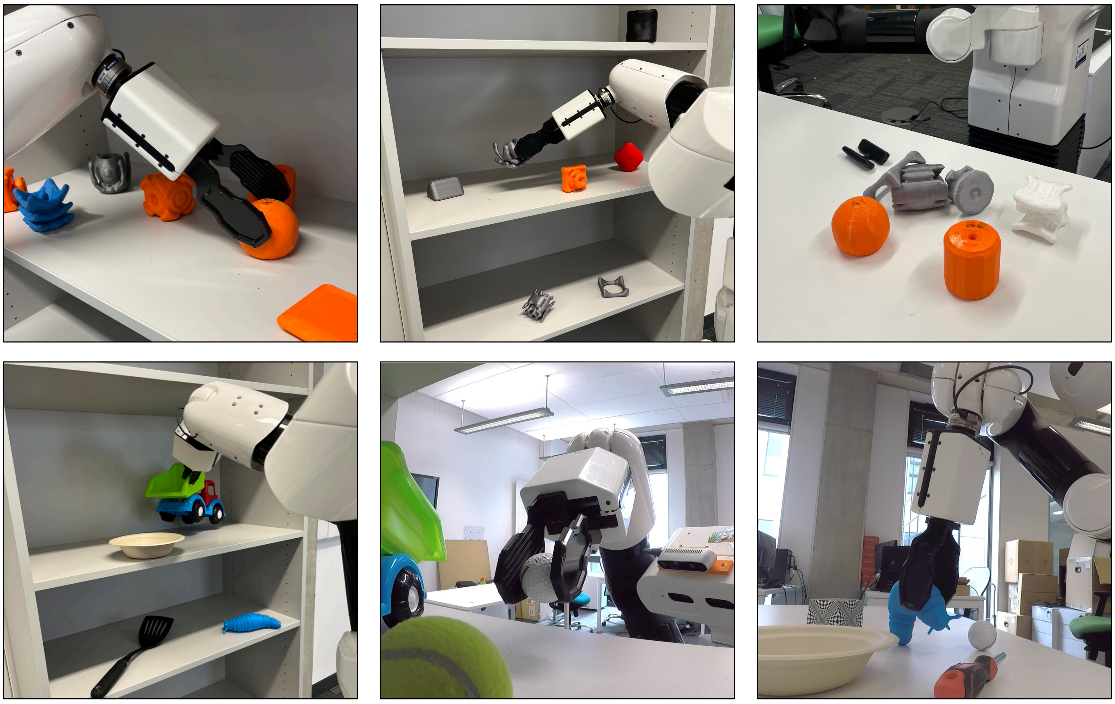


Figure 5.13: Example of the scenes tested in our real-world experiments with a PAL TIAGo mobile manipulator.

different conditions for this experiment, with 2 furniture objects and 2 evaluation sets tested. Examples of scenes tested in our real-world experiments are shown in Figure 5.13.

5.5 Results

We reported the GSR, CR and the Visible Clearance Rate (VCR), where we defined VCR specifically for our domestic simulation experiment. With the flexible setup in our domestic simulation experiment (sim-domestic), not all objects in a scene were necessarily visible in the perceptive field of the camera before the scene was abandoned. For example, if a small object was placed at the top of a shelf, it might never have been visible from a robot's perspective. We excluded such invisible objects when calculating VCR as it was impossible for any model to propose grasps on objects not included in the scene representation. We calculated GSR (Equation 2.1) and CR (Equation 2.2) and VCR aggregated across all evaluation scenes, with VCR defined as:

$$VCR = \frac{\#Objects\ cleared\ from\ scenes}{\#Visible\ objects\ in\ scenes} \tag{5.8}$$

Note that for our simulated tabletop (sim-tabletop) and real-world experiments (real-domestic), VCR = CR. While the simulated tabletop experiments naturally have all objects visible, we specifically ensured the visibility of each object in at least one trial in the real-world experiments. It is important to keep in mind that the overall performance of a model can only be judged when looking at both the grasp success with GSR and the clearance with CR (or, in our domestic simulation experiments, VCR). Ideally, a good grasp proposal pipeline should achieve high success in the grasp proposals and a high clearance of the objects in the scene.

5.5.1 Setting the acceptance threshold

When converting the dense output tensor of GP-net+ to grasp proposals, we used the so-called acceptance threshold $\gamma_{\hat{gc}}$ to define the minimum predicted grasp confidence $\hat{gc}$ that can yield a grasp proposal. As such, varying $\gamma_{\hat{gc}}$ influenced the performance of the model since both GSR and VCR vary with the predicted grasp confidence $\hat{gc}$.

To select a suitable acceptance threshold $\gamma_{\hat{gc}}$, we began by setting a low threshold of $\gamma_{\hat{gc}} = 0.01$ to generate a large set of grasp proposals for scenes using the custom evaluation set in the domestic simulation experiment (sim-domestic).

We filtered grasp proposals in steps of $\Delta\gamma_{\hat{gc}} = 0.01$ and plotted the resulting performance of GP-net+ with GSR and VCR in Figure 5.14. Both GSR and VCR are important for a good performance of a grasp proposal algorithm, as a very low performance on either would result in very few successful grasp attempts overall. The performance for the two variables showed an inverse relationship, with GSR increasing and VCR decreasing for an increasing acceptance threshold $\gamma_{\hat{gc}}$ (up until $\gamma_{\hat{gc}} > 0.8$, where fewer than 20 grasps were proposed for all objects in all scenes). Given this relationship, we found equal importance of GSR and VCR to yield the best results. Therefore, we calculated their fitness Γ as their average:

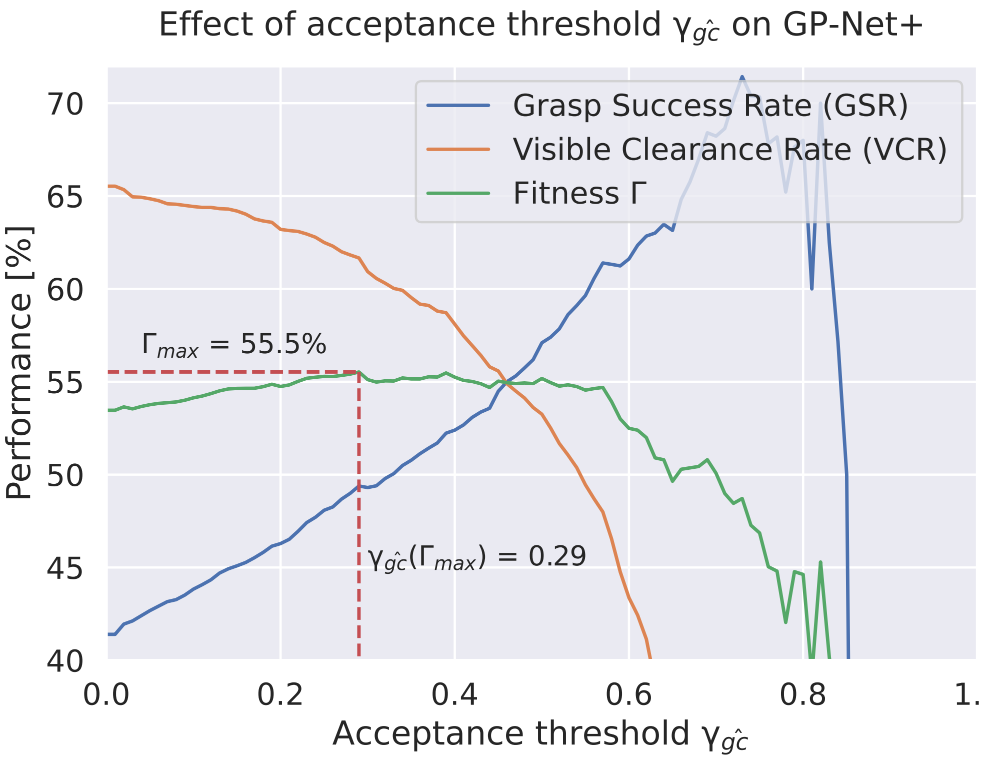


Figure 5.14: We calculate a fitness function Γ as the average of GSR and VCR to evaluate the performance of GP-net+ over variations in the acceptance threshold $\gamma_{\hat{gc}}$. Note that for $\gamma_{\hat{gc}} > 0.8$, fewer than 20 grasps were proposed for all objects in all scenes with the custom evaluation set in the domestic simulation experiment. We maximised the fitness and set the acceptance threshold $\gamma_{\hat{gc}}(\Gamma_{max}) = 0.29$ for the remainder of our experiments.

$$\Gamma = \frac{GSR + VCR}{2} \tag{5.9}$$

As shown in Figure 5.14, the fitness Γ increased initially and stayed relatively constant between $\gamma_{\hat{gc}} = 0.2$ and $\gamma_{\hat{gc}} = 0.6$, with both GSR and VCR changing distinctly in that region. This indicated the trade-off when risking more unsuccessful grasps with lower predicted grasp confidence $\hat{gc}$ and increasing the clearance rate with each additional successful grasp. In our results, the fitness Γ was maximised with $\Gamma_{max} = 55.5\%$ at $\gamma_{\hat{gc}}(\Gamma_{max}) = 0.29$, which we set as the acceptance threshold for all of our remaining experiments.

$n = 1$	Simple	EGAD	Custom
GPD	50.5 / 67.7 / 59.1	46.0 / 66.1 / 56.1	30.0 / 48.1 / 39.1
VGN	84.8 / 94.6 / 89.7	50.1 / 72.2 / 61.2	50.8 / 69.4 / 60.1
VGN*	82.5 / 90.6 / 86.6	49.4 / 66.5 / 58.0	54.0 / 70.6 / 62.3
GP-net	**88.5** / **96.8** / **92.7**	57.7 / 76.3 / 64.2	47.4 / 67.4 / 57.4
GP-net+	**90.1** / **96.2** / **93.2**	**75.6** / **89.4** / **82.5**	**64.6** / **79.2** / **71.9**
$n = 5$			
GPD	72.4 / 49.5 / 61.0	34.2 / 36.9 / 35.6	26.3 / 18.7 / 22.5
VGN	90.4 / 90.1 / 90.3	51.1 / 43.2 / 47.2	38.8 / 31.5 / 35.2
VGN*	91.8 / 87.2 / 89.5	47.5 / 31.1 / 39.3	41.3 / 29.8 / 35.6
GP-net	86.2 / 91.7 / 89.0	55.0 / 47.9 / 51.5	49.9 / 42.7 / 46.3
GP-net+	**92.3** / **96.5** / **94.4**	**63.0** / **57.5** / **60.3**	**57.8** / **51.4** / **54.6**

Table 5.1: Results of tabletop simulation experiments (sim-tabletop) with $n = 1$ and $n = 5$ objects for three different evaluation sets. Results are reported as GSR / CR / fitness with bolded items representing the best performance as measured by fitness Γ with a margin of 1%. VGN* denotes an extra run of all scenes with a full workspace scan of 6 non-noisy images, compared to a single, noisy image for all other experiments.

5.5.2 Tabletop simulation results (sim-tabletop)

Using the acceptance threshold $\gamma_{\hat{g}c} = 0.29$, GP-net+ outperformed all other models in our tested tabletop scenarios. Table 5.1 lists the full set of results for GP-net+, GPD, GP-net and VGN. GP-net+ performed substantially better than the other models, especially when using the more complex EGAD and custom evaluation sets. As expected, all models had a reduction in performance when moving from the simple evaluation set to the more complex geometries included in the EGAD evaluation set and the custom evaluation set.

The results for GP-net showed only a slight reduction in performance when moving from $n = 1$ to $n = 5$ objects. Since GP-net has never seen cluttered environments in its training data, this indicated some robustness for image-based grasp proposal methods when moving from single to cluttered objects. In general, GP-net performed well for all conditions, second in performance to GP-net+ in all but two cases.

Interestingly, VGN did not improve performance when scanning the full workspace with non-noisy images (marked as VGN* in Table 5.1). This seemed counter-intuitive since a higher-quality scan of the environment would be expected to yield a better scene representation and, hence, a higher quality of the proposed grasps.

Unfortunately, the original paper of VGN has no ablation studies regarding different numbers of scans for the TSDF integration, how it influences the grasp performance or what effect object size has on the TSDF or grasp performance. Object size might come into effect with the different evaluation sets since the maximum resolution of objects in the TSDF is proportional to the size of the voxels [166]. Regardless of the effects in play for VGN, our results for the simple evaluation set, which was used for the original simulation experiments in VGN, were similar to those of the original paper [16].

In summary, GP-net+ performed better than all tested alternatives on tabletop scenarios. This demonstrated the ability of GP-net+ to propose grasps for applications covered by tabletop methods for fixed manipulators, scenarios that are often tested in the robotic grasping literature, e.g. [10, 11, 12, 16, 20, 37, 44, 63, 73]. From these results, we moved on to more complex scenarios by evaluating GP-net+ in simulated domestic environments.

5.5.3 Domestic simulation results (sim-domestic)

In this section, we report the results of the domestic simulation experiment individually for each object set, first, the custom evaluation set and then the EGAD evaluation set [74]. Finally, we combine all domestic simulation scenes and show the performance of GP-net+ in regard to the position of the objects in the input depth image.

Custom evaluation set: An overview of all results can be seen in Table 5.2, with the individual results achieving the highest fitness Γ (as per Equation 5.9) marked in bold. Despite GPD and GP-net receiving an advantage through grasp-guiding (see Section 5.4.2) with ground-truth information, GP-net+ substantially outperformed both algorithms. GP-net was “conservative” in terms of grasp confidence,

	Tables	Shelves	All
GPD	22.3 / 26.8 / 24.5	7.3 / 8.6 / 7.9	13.6 / 16.1 / 14.8
GP-Net	70.3 / 24.6 / 47.5	49.1 / 7.0 / 28.0	63.7 / 15.4 / 39.6
GP-net+	**59.2 / 78.1 / 68.7**	**43.7 / 51.3 / 47.5**	**49.9 / 61.4 / 55.7**

Table 5.2: Performance of GP-net+ in simulations (sim-domestic) using the objects from the custom evaluation set. Results are reported as GSR / VCR / fitness with bolded items representing the best performance as measured by fitness Γ with a margin of 1%.

resulting in a high GSR, but a very low VCR due to very few grasp being tested with a total of 171 grasps compared to GP-net+'s 2009 grasps and GPD's 1674 grasps in the same scenes.

Note that the VCR values in Table 5.2 in combination with the total number of grasp attempts and the GSR can only be used to calculate the number of *visible* objects in all scenes. The number of visible objects in all scenes varied between the tested models due to the maximum number of consecutive grasp failures for a single scene and, thereby, a different number of camera viewpoints. The CR, however, could be used to calculate the total number of object appearances in the experiment. The CR was 12.0% for GPD, 5.8% for GP-net and 52.9% for GP-net+. There were a total of 1895 object appearances from the custom evaluation set in our domestic simulation experiment.

All algorithms exhibited a lower performance on shelves compared to tables. This was expected since shelves present a more challenging setting with lower visibility, more occlusions and a higher chance of collisions between the gripper and its surroundings. Examining the reasons why proposed grasps failed provides interesting insight into the performance of a grasp proposal algorithm. Figure 5.15 visualises the failure causes of GP-net+ for both furniture types. The simulation tracked five possible outcomes: "approach collides", "failed to grasp", "grasped furniture unit", "retreat collides" and "success". In our results, "failed to grasp" was the most common failure type and occurred 24.7% of the time, with a similar number of failures on tables and shelves.

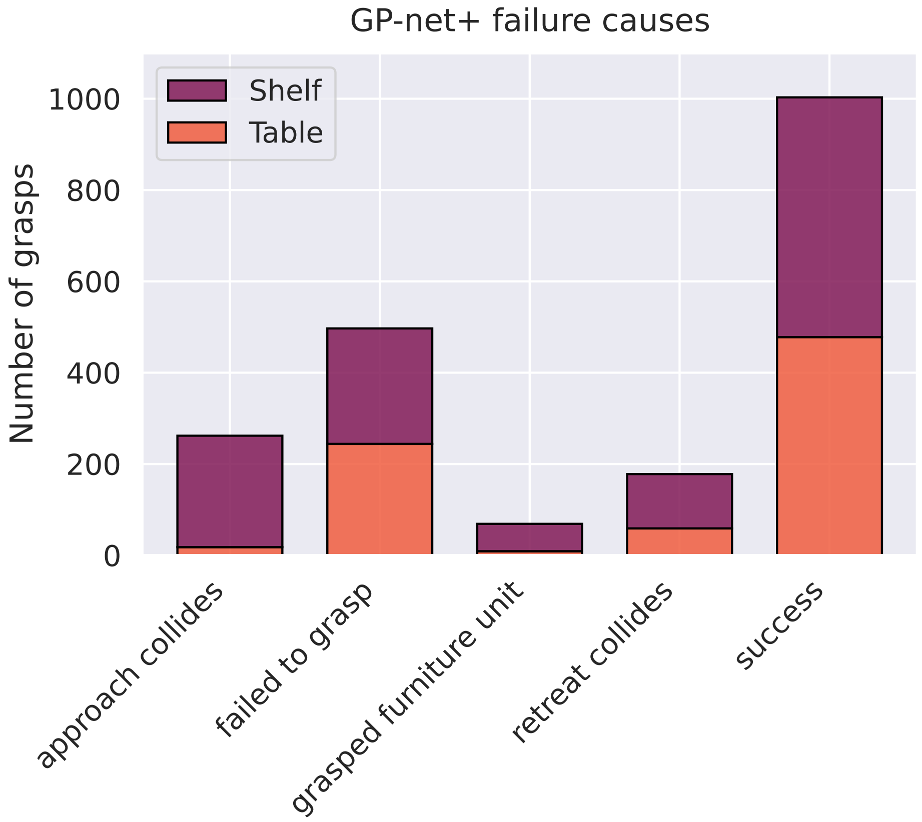


Figure 5.15: Causes of failure when testing GP-net+ grasp proposals in our domestic simulation experiment (sim-domestic) with the custom evaluation set.

Examples of grasp attempts where the gripper failed to grasp the object are shown in Figure 5.16 a). We observed two main failure situations in this category: (i) badly planned grasps, where the combination of predicted grasp orientation, width and contact position resulted in a failure, and (ii) grasps where a crucial part of the scene was occluded in the original image. An example of case (ii) is shown in Figure 5.16 a), where the gripper would have managed to grasp the mug if it had been empty. The content of the mug was not visible in the scene representation. In such situations, a measure of uncertainty might be able to differentiate the possibilities and plan sensible grasps, which we further discuss in Section 5.7.

The second and third most common failures occurred when the gripper collided with the furniture unit or other objects during grasp execution. This can happen

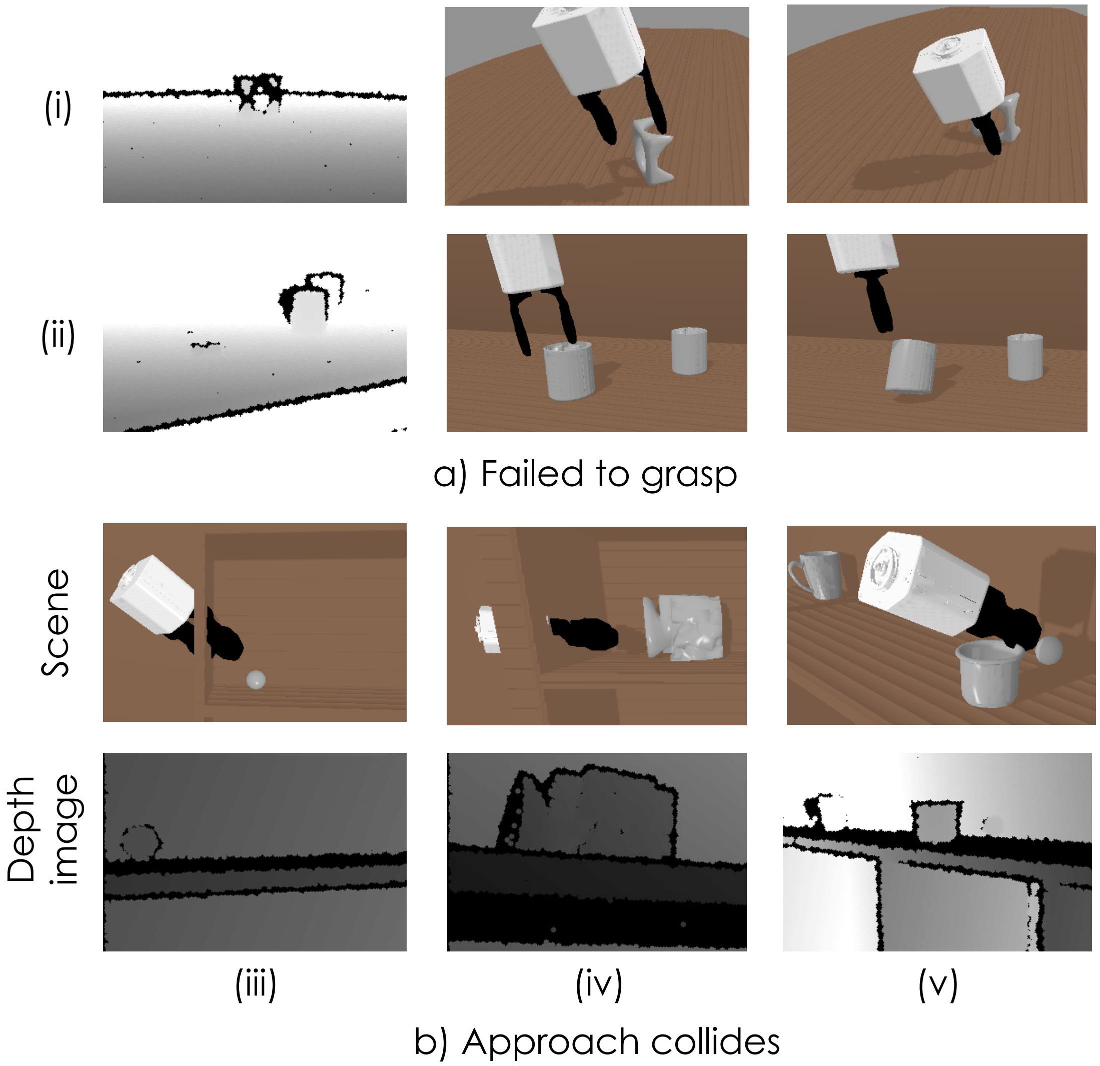


Figure 5.16: Examples of failure causes in simulation, with a) failures to grasp and b) approach collisions.

during the approach of the final grasping pose (“approach collides”, 13.0% of grasp attempts) or when lifting the gripper with the grasped object (“retreat collides”, 8.9% of grasp attempts). Most of these failures occurred when grasping objects from shelves due to the increased possibility of collision. Examples of such failures are shown in Figure 5.16 b). Similar to the “failed to grasp” category, some grasp proposals were poorly planned, e.g. (v), while others were reasonable based on the available information in the scene representation. For example, the depth images

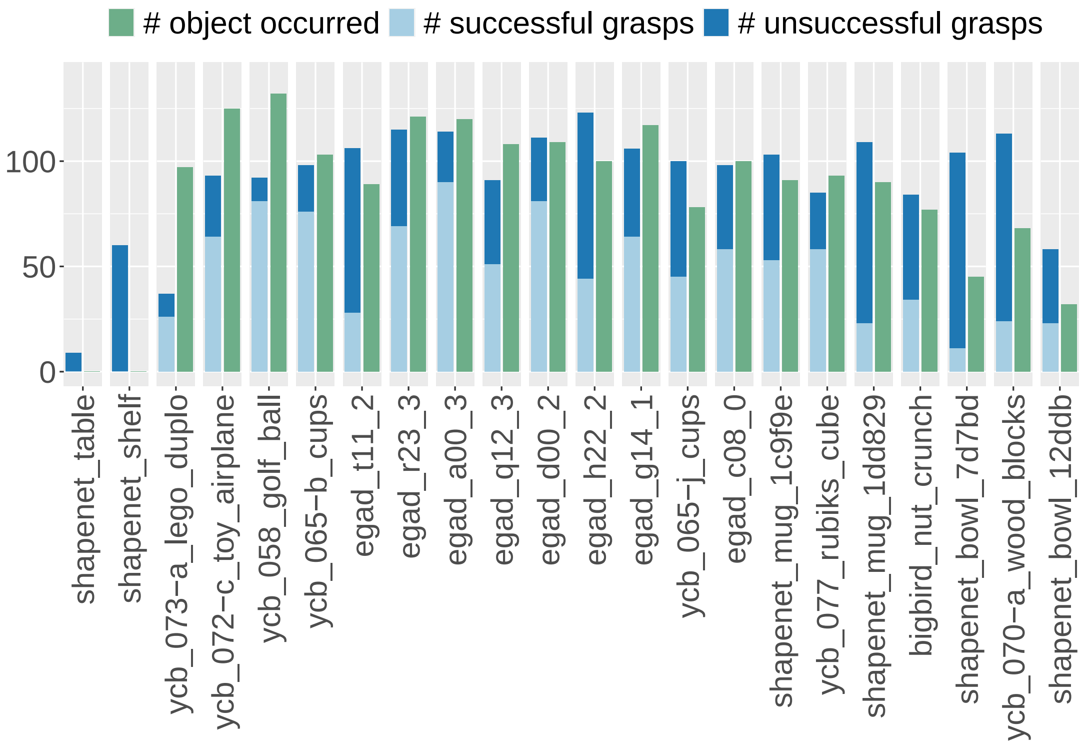


Figure 5.17: Number of times an object occurred in the simulation and number of successful/unsuccessful grasp attempts for each object in the custom evaluation set.

for (iii) and (iv) did not include the end of the outer part of the shelf, with which the gripper collided during execution. The least common failure cause of GP-net+ occurring 3.4% of the time was the "grasped furniture unit", which almost exclusively happened on the shelf and indicated a problem with distinguishing graspable object surfaces from ungraspable furniture surfaces.

To get a better insight into the performance of GP-net+ for each object, we plotted the number of successful and unsuccessful grasp attempts as well as the number of occurrences for each object in the custom evaluation set in Figure 5.17. Since tables and shelves are non-graspable objects that the robot cannot lift, they are only associated with unsuccessful grasp attempts. Furthermore, since a scene only got abandoned after 5 consecutive failed grasp attempts or when all objects have been grasped, there could be more than one attempt to grasp a single object in the

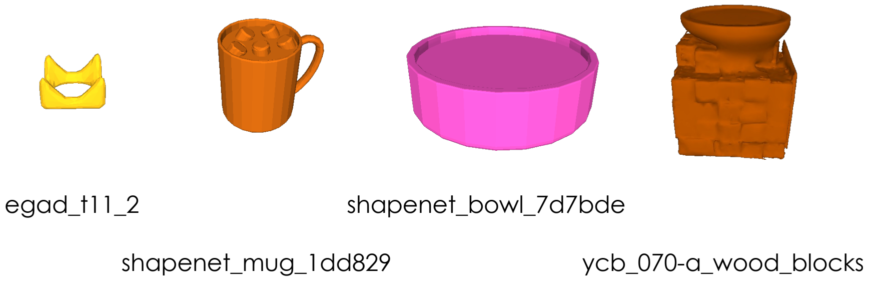


Figure 5.18: Visualisation of objects with low performance for GP-net+.

scene. This could result in the sum of successful and unsuccessful grasp attempts exceeding the number of object occurrences. For each object, the ratio between the number of successful grasps and the number of object occurrences calculates the CR (The number of unseen objects can not be referred from the graph, not allowing a calculation of VCR). Similarly, the ratio between the number of successful and unsuccessful grasp attempts results in the GSR of a given object. We sorted the graspable objects based on their mesh surface area, with the smallest object (ycb_073-a_lego_duplo) on the left and the one with the largest surface area (shapenet_bowl_12ddb) on the right.

A visualisation of the object mesh of challenging objects in the custom evaluation set is shown in Figure 5.18. One of those particularly challenging objects is “shapenet_bowl_7d7bd”, a ShapeNet bowl with a lid. The lid made it difficult for the bowl to be grasped along the rim. We noticed that most of the proposed grasps for this object were along the rim of the object, especially in cases where the camera viewed the object from below and did not perceive the lid.

Another challenging object in our custom evaluation set is a mug from ShapeNet labelled “shapenet_mug_1dd829”, which is filled with liquid and therefore could not be grasped by the rim, see Figure 5.18. Furthermore, it had a diameter of $9.9cm$, making it too wide to be grasped on the outer surface with the PAL parallel jaw gripper, which has a maximum graspable width of $8cm$. Human operators would usually grasp such an object with a form closure grasp around the handle, which is more challenging to achieve with the limited flexibility of two gripper plates with a parallel-jaw gripper.

The YCB object “ycb_070-a_wood_blocks” is a box filled with coloured wooden blocks [85] and too wide to be grasped along the bottom side of the box. Furthermore, the model exhibited traces of sensing inaccuracies in the mesh file due to the reflective surface of the object packaging (see Figure 5.18), which was a problem that can also be encountered in real-world scenarios. Despite these challenges, GP-net+ generated approximately 21% successful grasps on this object, exclusively when grasping the box lid.

The EGAD object “egad_t11_2” is another object with many failures. A failed grasp attempt on that object is also depicted in Figure 5.16 a) (i), where the gripper pushed the object away. We assumed that due to the high curvature and “spikes” on the upper part of the object as depicted in Figure 5.18, GP-net+ struggled to propose sensible grasp orientations. Furthermore, the object had a quite small height, making it difficult to grasp it when lying on the ground in a similar pose to the one shown in Figure 5.18.

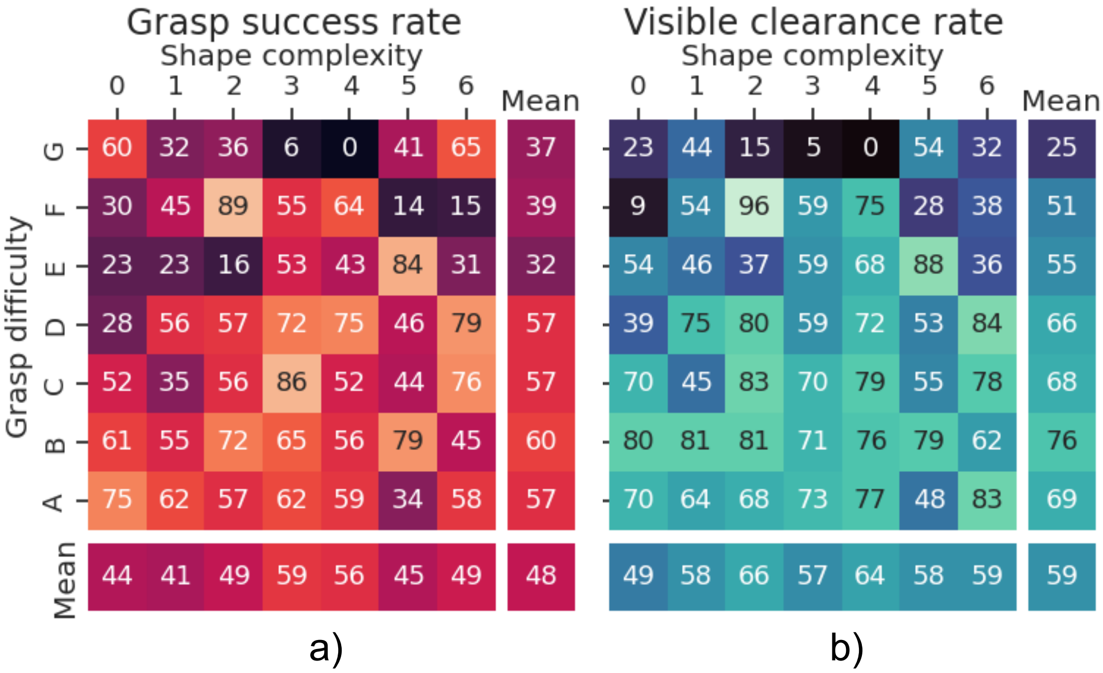
No object in Figure 5.17 had a VCR of 100%, i.e. was grasped successfully every time it appeared in a scene. This is expected in domestic scene scenarios since some objects’ poses and surroundings can make them ungraspable with a single grasping motion. For example, the “bigbird_nut_crunch” object is a rectangular food box. If the box lay face-down on a surface, it could not be grasped in a single movement with the parallel-jaw gripper as it was smaller than the height and width of the box. For grasping such objects, a robot would either require a more sophisticated gripper or combine skills like pushing and grasping objects [123, 127, 167].

EGAD evaluation set: In addition to the simulation evaluations using the custom evaluation set, we show the results for using the EGAD [74] evaluation set in Table 5.3. Similar to the results with the custom evaluation set in Table 5.2, GP-net+ outperformed both GPD and GP-net in terms of the fitness Γ for the tables and the shelves. This was despite the latter models being given an advantage through grasp-guiding with ground-truth information, as described in Section 5.4.2.

Also similar to the results in Table 5.2, GP-net was conservative in terms of grasp

EGAD	Tables	Shelves	All
GPD	21.8 / 33.9 / 27.8	6.5 / 8.3 / 7.4	14.6 / 20.6 / 17.6
GP-Net	66.4 / 22.5 / 44.4	36.6 / 4.9 / 20.8	58.4 / 14.1 / 36.3
GP-net+	**52.0 / 73.9 / 62.9**	**41.2 / 46.7 / 44.0**	**46.6 / 58.9 / 52.8**

Table 5.3: Performance of GP-net+ in our domestic simulation experiment (sim-domestic) with EGAD evaluation set. Results are reported as GSR / VCR / fitness with bolded items representing the best performance as measured by fitness Γ with a margin of 1%.



Grasp success rate	0	1	2	3	4	5	6	Mean
G	60	32	36	6	0	41	65	37
F	30	45	89	55	64	14	15	39
E	23	23	16	53	43	84	31	32
D	28	56	57	72	75	46	79	57
C	52	35	56	86	52	44	76	57
B	61	55	72	65	56	79	45	60
A	75	62	57	62	59	34	58	57
Mean	44	41	49	59	56	45	49	48

Visible clearance rate	0	1	2	3	4	5	6	Mean
G	23	44	15	5	0	54	32	25
F	9	54	96	59	75	28	38	51
E	54	46	37	59	68	88	36	55
D	39	75	80	59	72	53	84	66
C	70	45	83	70	79	55	78	68
B	80	81	81	71	76	79	62	76
A	70	64	68	73	77	48	83	69
Mean	49	58	66	57	64	58	59	59

Figure 5.19: Results of our domestic simulation experiment (sim-domestic) with a) GSR [%] and b)VCR [%] of GP-net+ on the EGAD evaluation set.

proposals, only proposing a total of 154 grasps with a high GSR of 58.4% but a low VCR with only 14.1% for all visible objects. In contrast, GPD proposed a total of 1666 grasps and GP-net+ proposed a total of 1649 grasps. Similar to the results for the custom evaluation set, the CR was 16.5% for GPD, 6.1% for GP-net and 52.3% for GP-net+.

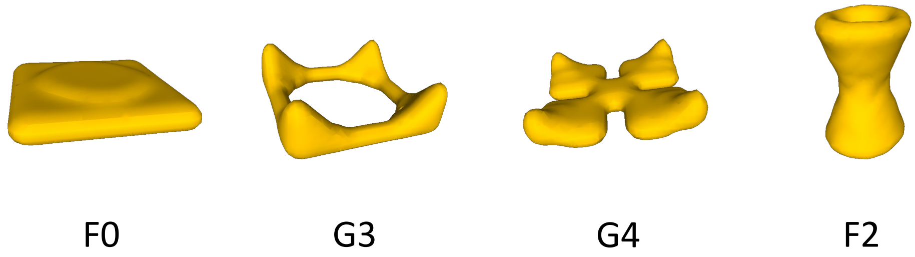


Figure 5.20: EGAD objects “F0”, “G3” and “G4”, which have a low performance, and “F2”, which has a high performance for both GP-net (see Section 4.5) and GP-net+.

We utilise the structure of the EGAD evaluation set and show the GSR and VCR for GP-net+ over the 2-dimensional space of shape complexity and grasp difficulty in Figure 5.19. There was a substantial drop in both GSR and VCR for increasing grasp difficulty, while the performance over shape complexity was relatively constant. These trends are similar to the performance evaluation of other models on the EGAD evaluation set, e.g. GG-CNN [74] or our results for GP-net in Chapter 4. There are a few objects in the EGAD evaluation set with very few successful grasp attempts overall (resulting both in a low GSR and VCR in Figure 5.19), for example, “F0”, “G3” and “G4”.

A mesh representation of these low-performance objects and a high-performing object with “F2” is visualised in Figure 5.20. We note that all low-performing objects had quite a low height, usually causing the graspable surfaces to be very close to the ground plane. Additionally, the visible surfaces of their graspable areas were small, reducing the visible surface that could be used to propose grasp contacts. This was especially apparent when comparing those low-performance objects to more successful objects like “F2”. The low-performance objects usually did not have any grasps proposed by GP-net+ in our simulated and real-world experiments.

All evaluation sets: We report the position of successfully and unsuccessfully grasped objects in image coordinates to gain more insight into where successfully grasped objects are located in the image. To do so, we saved the coordinates of

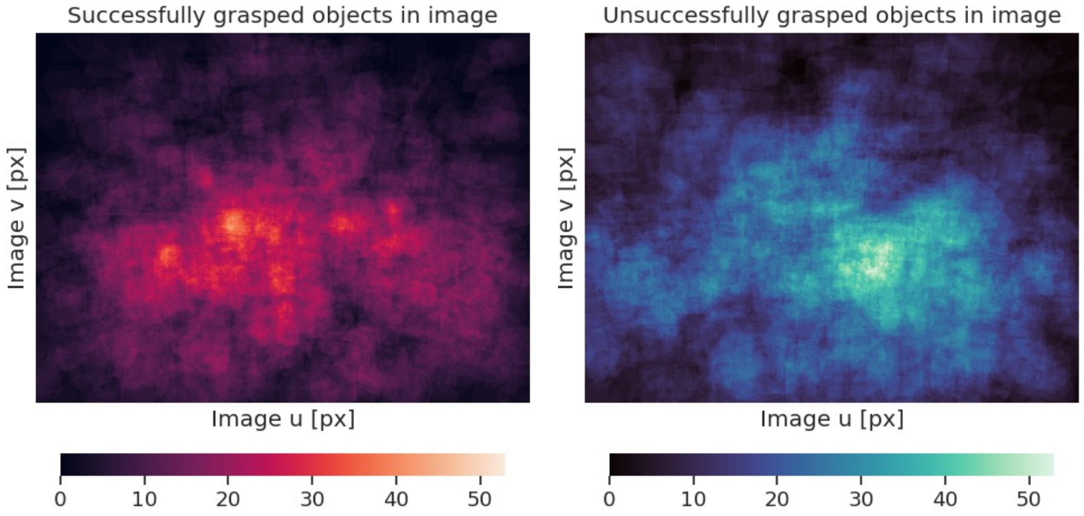


Figure 5.21: Position of successfully and unsuccessfully grasped objects in the input images. We overlay the ground-truth segmented area of the object in the image for all grasp attempts to generate these images.

all pixels of a grasped object in an image based on the ground-truth segmentation mask. After all trials had been executed, we summed the occurrences of object pixels for successfully and unsuccessfully grasped objects over all images. This resulted in the images shown in Figure 5.21.

We can see that overall, grasp proposals for both successfully and unsuccessfully grasped objects seem to be most common in the lower centre of the image. Given that the camera was placed on the robot's head and always viewed the scene upright and horizontally, this makes sense since objects towards the upper end of the image tend to be only partially visible. Additionally, there were more unsuccessfully grasped objects than successfully grasped objects towards the left and right border of the image. In these regions, the object shape and the immediate surroundings were only partly available in the scene representation.

GP-net+ could not properly judge collisions or the full grasp pose when objects were partly invisible or their immediate surroundings were out of view. We think it is important to be aware of uncertainty in such situations. While GP-net+ does not have such features at the moment, we think it would be beneficial to introduce

	Household evaluation set	EGAD evaluation set
Table	61.5 / 74.7 / 68.1	68.1 / 77.6 / 72.9
Shelf	65.8 / 33.3 / 49.6	79.3 / 37.6 / 58.5
All	63.1 / 54.7 / 58.9	71.4 / 57.6 / 64.5

Table 5.4: Results of our real-world experiment with a PAL TIAGo mobile manipulator using GP-net+ with the household and EGAD evaluation sets. Results are presented as GSR / CR / fitness Γ.

a measure of uncertainty for the scene representation or grasp proposals. We will discuss this possibility further in Section 5.7.

5.5.4 Real-world results (real-domestic)

We report both the GSR and the CR for the real-world domestic experiment using the household and EGAD evaluation set in Table 5.4. Due to the experiment setup as described in Section 5.4.3, CR was equivalent to VCR since all objects in all scenes were visible to the model at least once.

Overall, GP-net+ achieved a GSR of 63.1% and a clearance rate of 54.7% on the household evaluation set and a GSR of 71.4% and CR of 57.6% on the EGAD evaluation set. While the GSR was higher for the shelf, CR was higher for the table with both evaluation sets. This differed from the behaviour in our domestic simulation experiment (sim-domestic) in Section 5.5.3, where the GSR was lower for the shelf. This was rooted in differences in grasp execution between the simulated (sim-domestic) and real-world (real-domestic) experiments. The path planning pipeline in our real-world domestic experiment excluded grasps where the approach pose would collide with the shelf. Such grasps would count as failures in our simulated experiments (sim-domestic). As we did not attempt to execute them in real-world experiments to avoid harming the robotic arm, they could not be unsuccessful, and hence, the GSR was increased.

Household evaluation set: We plot the number of successful and unsuccessful grasp attempts for each object in Figure 5.22. While we did not have exact measures of the surface area for the real-world objects like in the simulated exper-

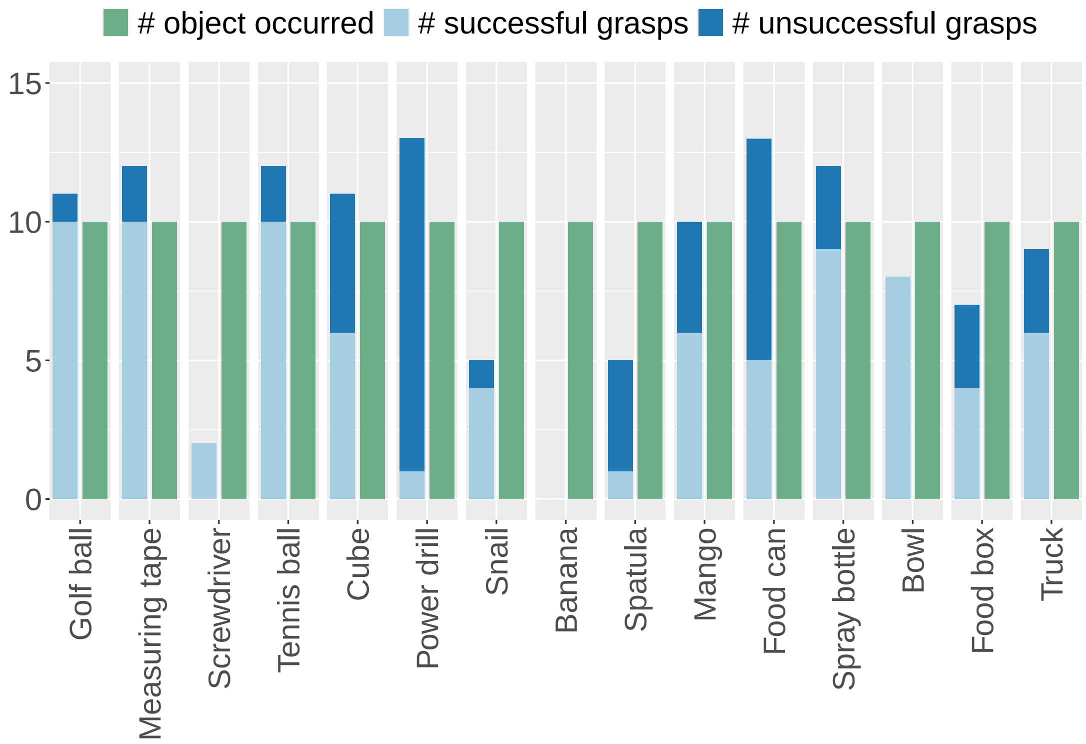


Figure 5.22: Number of successful and unsuccessful grasp attempts (stacked bars) in our real-world domestic experiment for each object in the household evaluation set.

iments, we manually sorted them according to their estimated surface area from the smallest to the largest surface area from left to right. There was a group of objects that the robot struggled to grasp. For example, objects like the banana, spatula and screwdriver had very few or no grasp attempts. GP-net+ seemed to struggle to propose grasps on objects with a small visible surface area, especially for those close to the ground plane.

The two objects with the most unsuccessful grasp attempts were the power drill and the food can. The diameter of the food can is $7.3cm$, which is close to the maximum gripper width of $8cm$. Inaccuracies in the sensor readings, gripper positioning and grasp width prediction sometimes caused the gripper to push the food can away, which we observed to be the most common error for this object. The power drill is comparatively heavy with a mass of $0.312kg$ and has a complex geometry and

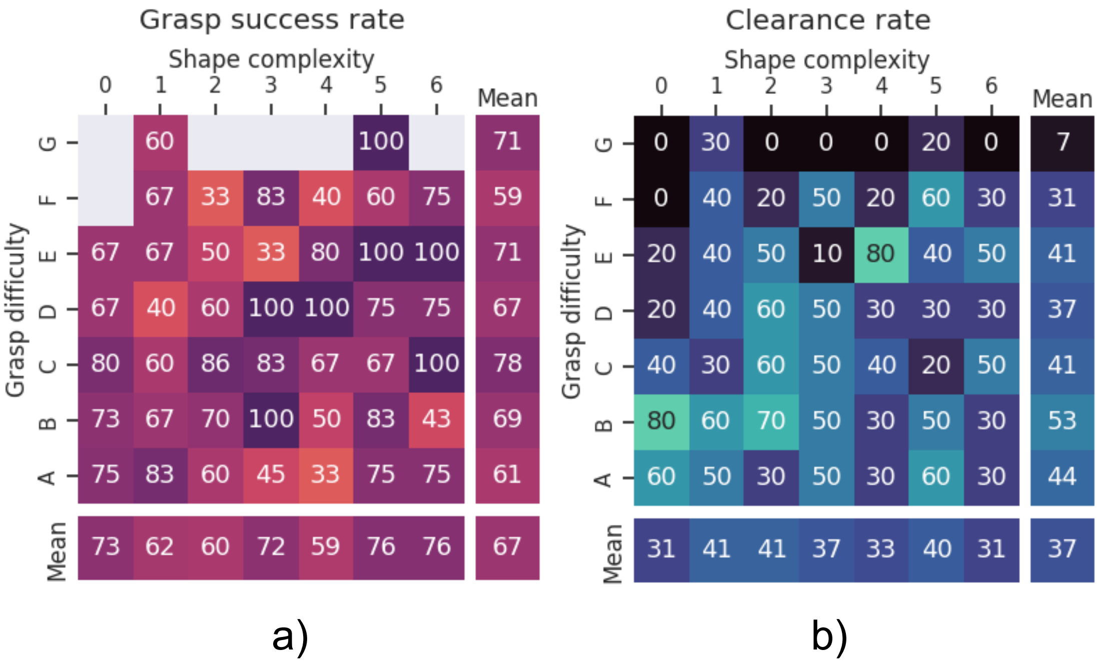


Figure 5.23: Results for our real-world domestic experiment with a) GSR [%] and b) CR [%] of GP-net+ on the EGAD evaluation set. Empty fields in a) correspond to objects where not a single grasp has been attempted.

weight distribution. It usually slipped out of the gripper when lifting the object. For both failure modes, changes to the hardware and pipeline (e.g. adding closed-loop control combined with sensor fusion) could help make the grasping process more reliable. We discuss this further in Section 5.7.

In general, we observed a high number of successfully executed grasps for objects with simpler shapes, like the golf ball, measuring tape and tennis ball, which the robot was able to pick up every time. Even objects with more complex shapes, like the truck, could be picked up more than half the time.

EGAD evaluation set: For the real-world experiments using the EGAD [74] evaluation set, GP-net+ achieved a GSR of 71.4% and CR of 57.6% across the entire dataset, see Table 5.4.

Figure 5.23 shows the GSR and CR for each object in the EGAD evaluation set. Similar to the results in our domestic simulation experiment (sim-domestic) and the results for GP-net (see Section 4.5) and for GG-CNN [74], there was a clear reduction in performance with increasing grasp difficulty. Only very few grasps were executed for objects with a high grasp difficulty of category “G”, as shown by the number of empty fields in Figure 5.23 a), where no grasps have been attempted. Overall, the low-performing objects were similar to those in the domestic simulation experiment (sim-domestic), for example, objects “F0”, “G2”, “G3” and “G4”.

In Chapter 4, Section 4.5, we observed three primary failure cases for GP-net on objects of the EGAD evaluation set. These failure cases, illustrated in Figure 4.8, involved situations where the gripper pushed objects away during the final grasp approach, was unable to fully close due to obstructions, or experienced object slippage when lifting the gripper. We found the same kind of failure cases with GP-net+, usually occurring on the same objects given their specific geometric characteristics.

In general, the performance of GP-net+ on the EGAD evaluation set in this real-world experiment was higher than that of GP-net in the real-world experiment using the same evaluation set in Chapter 4. This was despite the complexity of the setup having increased substantially for GP-net+, with multiple, cluttered objects positioned on both tables and shelves. In detail, GP-net+ achieved a GSR of 71% and CR of 58% in the experiment using the EGAD evaluation set (see Table 5.4) which was substantially higher than the results of the simpler real-world experiment in Chapter 4 for GP-net with a GSR of 54% and CR of 50%. Despite these promising results, there are several aspects which GP-net+ could be improved upon, and we discuss these in Section 5.7.

5.6 Target-driven object grasping

When using grasp proposal algorithms in real-world domestic applications, they should have the ability to be used for target-driven object grasping. For example, when using an assistive robot at home, it should be able to grasp targeted objects to transport them to a human or store them at a defined spot. To apply a grasp proposal algorithm for target-driven grasping, some form of object detection [115]

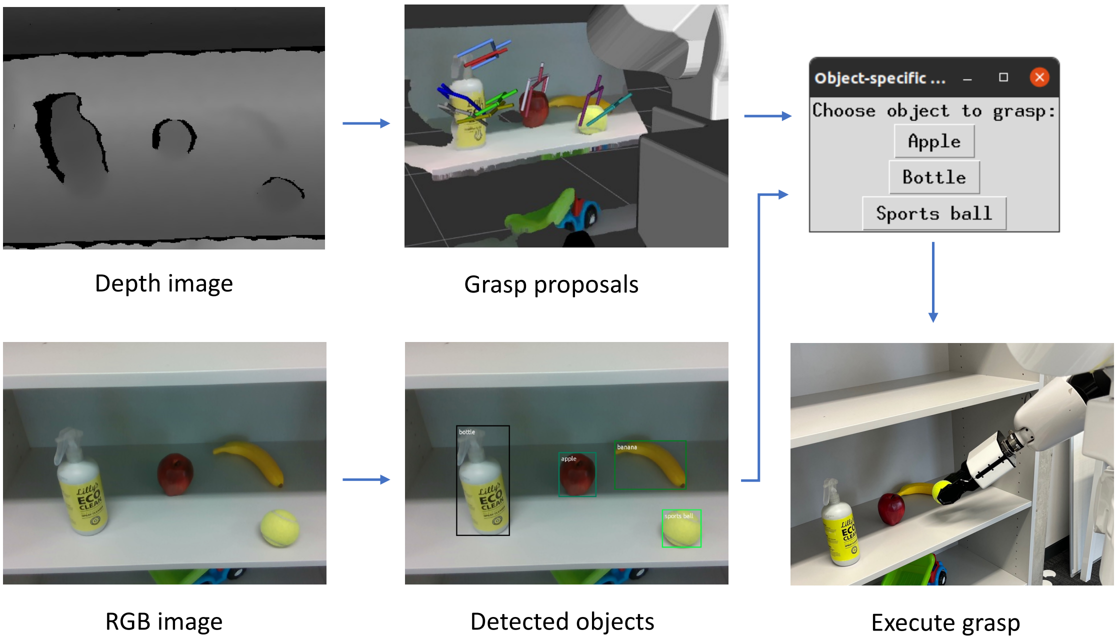


Figure 5.24: Example of target-driven object grasping with GP-net+. The user can choose an object for grasping based on the grasp proposals from GP-net+ and the object bounding boxes. The grasp proposals are filtered, paths planned, and suitable grasps without collisions are executed. Here, we show an example of grasping a sports ball successfully executed by the robot.

or instance segmentation [168] algorithm can be used to match the grasp proposals to the desired object [82, 169].

Our ROS package includes this functionality and can thereby use GP-net+ to grasp specific objects. Since the size of the dense tensor output of GP-net+ corresponds to the input image, we can easily use any image-based instance segmentation or object detection algorithm to filter grasp proposals. In our ROS package, we used a Faster R-CNN [170] object detector pre-trained on the COCO dataset [171] to filter and group GP-net+'s grasp proposals. The code can be easily amended to utilise different weights or an instance segmentation algorithm for object identification if required. In our tests, we decided to use the pre-trained Faster R-CNN object detector since we found that the available, pre-trained instance segmentation models from PyTorch performed worse, especially in domestic household environments.

In our ROS package, object labels are allocated to grasp proposals by matching the label of the smallest area-wise bounding box in which the visible grasp contact of a grasp proposal falls. Based on the labels, objects can be selected for grasping by the user through a pop-up window with choices. An example of such a process for target-driven object grasping with our ROS package and a PAL TIAGo mobile manipulator can be seen in Figure 5.24.

5.7 Discussion

When choosing a grasp proposal model to use with a robotic system, one should consider the model's relative performance and any limitations the model approach might have. In this section, we first discuss GP-net+'s performance in relation to the three other grasp proposal models we tested in our experiments in Section 5.5. Subsequently, we discuss the limitations of GP-net+ regarding depth resolution, the balance between GSR and CR, and the visibility of grasps in the scene.

In Section 5.5.2, we showed how GP-net+ outperformed all other models in all of the tested conditions in the simulated tabletop experiment. This showed GP-net+ can perform better than widely-used grasp proposal solutions in the often-tested tabletop scenarios. GP-net+ could also generalise to the different scene variations in complex, domestic environments, as shown in Section 5.5.3.

Here, GP-net+ outperformed both GPD and GP-net in a domestic simulation experiment. Despite the extra ground-truth information GPD and GP-net received for grasp guiding (see Section 5.4.2), GP-net+ achieved substantially higher performance for all tested evaluation sets. The other model tested in our simulated tabletop experiment, VGN, could not be applied directly to the domestic simulation experiment. The domestic scenes featured multiple objects in large workspace areas with an unknown pose in relation to the camera. Since VGN requires a fixed-size workspace of $30 \times 30cm^2$ with a known camera-workspace transform, such scenes cannot be solved with a single scene representation and without additional pre-processing steps to determine the pose of (multiple) workspaces in relation to the camera.

This showed the importance of using appropriate models and training data when applying grasp proposal solutions to similarly complex scenarios. Furthermore, the results for the real-world experiment in Section 5.5.4 demonstrated how GP-net+ can be used in a real-world application and obtain good results with a PAL TIAGo mobile manipulator grasping objects placed in clutter on tables and shelves.

To the best of our knowledge, there exist no other data-driven methods that can be directly applied for grasping complex objects in domestic environments without relying on workarounds like the grasp guiding required for GPD and GP-net. Some robot competitions test manipulation tasks in more controlled environments with a single shelf, for example, the Amazon Picking Challenge [172] or RoboCup@Home [132]. Complete robotic manipulation systems described for use in similar scenarios usually use hand-crafted approaches based on point clouds [132, 133, 135, 136], analytical grasp metrics like force closure using reconstructed surface patches [137, 138] or deploy offline grasp proposal based on CAD models [136, 139]. It would be beneficial to test more grasp proposal algorithms (if required with the corresponding workarounds) in complex domestic settings and evaluate their performance and applicability.

Naturally, GP-net+ comes with limitations that must be considered when choosing a grasp proposal algorithm. Since GP-net+ takes a depth image as a normalised 3-channel jet-colourscale image as input, it tries to find a balance between the maximum visibility and maximum depth resolution in the model input.

For GP-net+, we set the normalisation bounds to $[0.2m, 1.5m]$, which we found experimentally to show good convergence during training. However, this prevents the model from proposing grasps for objects further away than $1.5m$ and reduces the applicability for GP-net+ in those situations. While the graspable range of a mobile manipulator is usually smaller than $1.5m$ when stationary, an increased depth visibility could be beneficial when looking at closed-loop control with holistic grasping motions, where the robot starts to reach for the object while approaching it, e.g. [129, 130].

There is an interesting question about resolution in all scene representations. Ultimately, the representation has to balance the resolution it can achieve (within the limits of the sensor), the physical space it can represent and the memory and processing power it requires. In the case of GP-net+, the normalisation bounds limit the resolution and physical representation. In the case of VGN, these quantities are limited by the number of voxels and their size when integrating the TSDF. When using point clouds, there is a question about the number of points they contain, with possibilities to speed up the algorithms while reducing the resolution or representation by sub-sampling the point cloud.

It is also possible to combine different methods, potentially using a representation with a lower resolution and wider physical coverage for initial proposals and switching to a high resolution and low physical coverage representation in later stages of a grasp attempt to finetune grasp proposals. The hardware limits both the resolution and the physical representation, potentially influencing the best representation and resolution-representation balance for a given application.

In general, GP-net+ with the chosen acceptance threshold γ is conservative with the proposed grasps. There are objects in simulation and real-world experiments without any grasp proposals, even though they should be graspable in their respective setups. In particular, we found GP-net+ to propose few to no grasps on objects with small surface areas or thin features. When using GP-net+ in real applications, this results in objects being unable to be grasped. This deficiency in proposed grasps could potentially be improved by more targeted ground-truth grasp sampling in the training data or applying weights during training.

Finally, similar to the experiences with GP-net in Chapter 4, the contact-based grasp representation limits the ability to represent grasps for some objects. As the grasps are anchored to visible grasp contacts, no grasps can be represented if no graspable surfaces are visible in the scene representation. An example of this behaviour is represented in Chapter 4, Figure 4.9. This problem is more severe in the case of shelves, where the 6-DoF grasp poses are further limited by collisions with the furniture unit. In such cases, repositioning the robot might be necessary to grasp the objects.

5.8 Conclusion

In this chapter, we presented work for grasping objects in cluttered, domestic scenarios with a mobile manipulator. For this, we constructed domestic grasping scenes with objects in clutter positioned on varying furniture units like tables and shelves. We developed algorithms for placing objects in those scenes and sampling camera poses to generate realistic camera viewpoints. We then trained GP-net+, a 6-DoF grasp proposal model, on these scenes to be used to grasp objects in domestic environments without pre-processing.

We showed GP-net+ outperforms widely used grasp proposal algorithms on simple tabletop scenarios and in domestic scenes in two simulated experiments. In our real-world experiments, GP-net+ performed well on table scenes but, although it retained a high GSR, the VCR decreased substantially for shelves with only 35% for both evaluation sets. This corresponds to only being able to grasp a third of the available objects, which can limit applicability in practice. For GP-net+ to prove useful in real-world applications, this performance has to be improved, for example, by using more powerful model architectures and training regimes or adjusting the training data to reflect even more difficult scenarios.

The drop in performance from the simplest experiment, where GP-net+ achieved both a GSR and CR of more than 90%, to the simulated domestic experiments on shelves, where it achieved less than 50% GSR and 55% VCR, clearly shows the importance of testing models in appropriately realistic and challenging scenarios. To continue to make progress towards using assistive robotics in domestic environments, the test regimes must continue to become similarly variable and complex as their real-world counterpart.

GP-net+ concludes the work in this thesis, where we have created a viewpoint-flexible 6-DoF grasp proposal method for cluttered objects in domestic environments. We achieved this by building on a 6-DoF grasp quality prediction method in Chapter 3 and then GP-net, a 6-DoF grasp proposal method for single objects on planar surfaces in Chapter 4. Overall, GP-net+ has demonstrated the value of using a depth-imaged based FCNN for grasp proposal. Despite being a comparably simple scene representation, both GP-net and GP-net+ have performed as well as or better than widely used alternatives with more complex scene representation setups.

While we demonstrated the adaptation of GP-net+ to grasping in domestic environments in real-world experiments, there are still many challenges ahead in the path to building reliable end-to-end grasping pipelines for deployment in real-world scenarios. In the following chapter, we conclude this thesis by giving an overview of the presented research problem, highlighting some main takeaways and outlining ideas for future research.

CHAPTER 6

Concluding remarks

This thesis has presented grasp proposal algorithms for 6-DoF grasps from flexible and unknown viewpoints in complex scenes. Flexible grasp proposal algorithms are needed to allow robots to operate in dynamic real-world environments. In contrast to 4-DoF grasps, 6-DoF grasps increase the variability of grasp poses and can improve reachability when grasping objects in complex scenarios (see Section 2.5). Objects in real-world applications are also likely viewed from flexible camera viewpoints, especially when using mobile manipulators to grasp objects from different points within an environment (see Section 2.7).

Grasp quality prediction models evaluate the quality of pre-sampled grasp poses in a discriminative grasping approach, as discussed in Section 2.1. Such models are often restricted to fixed, overhead camera viewpoints and 4-DoF grasps [10, 11, 13, 14, 22], especially when their scene is represented as a depth image, see Section 2.4.1. While those models work well in practice, their limited flexibility restricts their applicability to more constrained scenarios.

To extend the flexibility of such discriminative 4-DoF models while keeping the architecture and scene representation simple, we integrated a 6-DoF grasp representation into a grasp quality prediction model and generated a balanced training dataset from versatile camera viewpoints. Training the extended architecture on this dataset resulted in a 6-DoF grasp quality prediction model for depth images observed from flexible viewpoints, called VGQ-CNN, (see Chapter 3). Experimental results showed that using appropriate training data, a CNN can evaluate the quality of 6-DoF grasps and generalise to variable camera viewpoints while retaining a similar performance as 4-DoF models for overhead cameras and top-grasps.

Discriminative approaches usually have substantial run-time disadvantages compared to generative approaches, however, since they need to sample and evaluate the quality of each grasp individually. In Chapter 4, we transitioned the positive characteristics of grasping with VGQ-CNN, including a simple scene representation for 6-DoF grasps from versatile camera viewpoints, from a discriminative to a generative approach. Instead of individually evaluating the quality of 6-DoF grasps, we constructed a generative method, called GP-net, to propose 6-DoF grasps for single objects on planar surfaces. GP-net predicts grasp poses and grasp qualities for each pixel in a depth image, which then can be used to construct grasp proposals.

A challenge in developing GP-net was rooted in the grasp representation, where we found that TCP-anchored grasp representations resulted in low convergence for the grasp orientation during training. Consequently, instead of the TCP, we based the grasp anchor for GP-net on the visible grasp contact in the scene representation, as first suggested by Contact-Graspnet [20]. Our results confirmed that this change in grasp anchor improved performance and convergence during training with our ablation studies in Section 4.5.3.

We examined the performance of GP-net not only in simulation but also conducted real-world experiments with a PAL TIAGo mobile manipulator using the EGAD evaluation set [74]. We also compared it to two widely used grasp proposal methods, VGN [16] and GPD [19], in those real-world experiments. Our results showed that GP-net performed better than GPD and VGN in these experiments with a grasp success of 54.4% compared to 44.2% and 51.6%, respectively. While performing better than alternative algorithms with this complex evaluation set, a grasp success of 54.4% should be improved before applying GP-net in commercial or real-world applications. We give ideas for such improvements in Section 6.1.

In contrast to other grasp proposal models like VGN and GPD, GP-net can propose grasps for objects in the scene representation without any additional pre-processing steps like defining a workspace in relation to the camera pose or indexing regions of interest for grasp proposals to generate suitable grasp proposals. This yields an advantage in run-time and removes the dependency on workarounds like segmentation algorithms. Furthermore, it becomes an even more important characteristic when moving from single objects on planar surfaces to cluttered objects in complex scenarios.

Domestic environments feature cluttered objects placed on workspaces within different furniture units and unknown camera-workspace configurations. We investigated training dataset design and grasp proposal approaches for such environments in Chapter 5. While the pre-processing steps for workspace or object definition required for VGN and GPD could be applied in tabletop scenarios, the wide variation in domestic environments with unknown furniture designs and an unknown camera-workspace transform makes their usage substantially more difficult in those environments. Even when using perfect ground-truth information, such workarounds can substantially reduce performance, as we show in Section 5.5.3.

The key to learning a data-driven grasp proposal technique for diverse applications lies in providing suitable data for training. Using simulated environments to generate ground-truth grasp information is a useful approach that provides better flexibility, lower cost and higher scalability than real-world experiments [74, 156, 158]. However, care must be taken with simplifications and assumptions when constructing the simulation environment to reduce the sim-to-real gap when applying the resulting models in the real world. We carefully constructed reproducible algorithms and pipelines for generating training data for 6-DoF grasp proposal on objects in clutter in domestic scenarios. We took particular care in preparing the scene, placing graspable objects on top of viable support surfaces of the furniture units and sampling camera viewpoints as they would be encountered with a camera mounted on a mobile manipulator. We used the simulation environments to generate data for training GP-net+, an extension of the grasp proposal model (GP-net) we proposed in Chapter 4.

Since real domestic environments can also feature tabletop scenarios, we showed in simulated experiments that GP-net+ performs better than VGN, GPD and GP-net in such tabletop scenarios. We further demonstrated the performance of GP-net+ in simulated domestic environments and compared it to GPD and GP-net, where we used the ground-truth segmentation mask for grasp guiding (i.e. filter grasps so they do not occur on the background or furniture unit) to be able to apply GPD and GP-net. While this was a necessary and reasonable workaround to use GPD and GP-net in domestic scenarios, utilising this ground-truth knowledge within the overall pipeline provided these algorithms with an inherent advantage. Since the pipeline was highly dependent on the quality of the segmentation mask, performance in real-world applications with estimated segmentation masks would likely be reduced.

Despite GP-net+ not being given this advantage in ground-truth information, it performed substantially better than GPD and GP-net in those simulations, regardless of the evaluation set. This demonstrated the limits of applying tabletop grasp proposal algorithms to more complex scenarios, where, despite using ground-truth information for workarounds, the increased complexity of the scene reduced the quality of the output grasp proposals substantially.

We demonstrated the applicability of GP-net+ to real scenarios with real-world experiments featuring domestic scenes with different furniture units. Finally, we showed that GP-net+ can be used for target-driven object grasping by combining the dense output tensor with an object detection or segmentation algorithm. We made our complete codebase available on GitHub[9] so it can be used to train, evaluate and apply 6-DoF grasp proposal algorithms to robots in real, domestic environments.

While GP-net+ can be applied as-is for grasping objects in domestic environments, the overall GSR of 60% to 70% (depending on the evaluation set) is too low to provide a reliable system in real domestic settings. The performance obtained in our experiments could be improved in a number of ways. For example, such options include coupling it with closed-loop control for the final grasp approach, letting the robot position itself to get a more favourable view of targeted objects,

[9] https://github.com/AuCoRoboticsMU/GP-netplus

and using a holistic motion controller for whole-body movements. We address some of our ideas for future research in the following section (for example, see Section 6.1.5 and Section 6.1.6).

6.1 Opportunities for future research

Despite the significant progress made in the robotic grasping community in recent years, much research is still needed to develop reliable grasping systems. In the following subsections, we briefly outline some opportunities for future research that can be explored to make grasping more robust and reliable.

6.1.1 Uncertainty in predictions

A robotic system in real-world environments always encounters uncertainty. This can result from sensor noise and limited visibility, e.g. with obstacles prohibiting sensor readings for parts of the scene and objects. The data of those sensor readings is used to construct the scene representation. When using this scene representation for grasp proposal, the quality and measurement uncertainty in the sensor readings will affect the quality of the grasp proposals. An example of this effect is shown in our results in Section 5.5.3, where partially occluded objects towards the edge of the scene representation experience a higher number of unsuccessful grasps than centred objects.

An approach that could limit this effect would be to estimate uncertainty in the scene representation when proposing the grasps. For example, uncertainty estimation could be used during inference to prioritise grasp proposals associated with more certain sensor measurements or directly be integrated into a grasp confidence prediction of a model. An example of estimating uncertainty in the scene representation is demonstrated by [15], where a measure of uncertainty over a shape completed voxel-grid is used for grasp selection.

6.1.2 Scene representation

A robot's (usually visual) sensor readings can be represented in various ways to propose grasps. Common representations are point clouds [20, 38, 39, 40], depth images [11, 12, 23, 41, 42, 43], RGB-D images [23, 25, 44] and voxel grids [16, 45].

As discussed in Section 5.7, a representation must balance the scene's resolution with the physical space covered and the memory and processing power it requires.

There is an open question about which scene representation is the best for robotic grasping in general and whether particular scene representations have advantages for specific applications. It would benefit the field to compare the advantages and disadvantages of different scene representations in a benchmark, particularly in relation to different object sizes, scene resolutions and computational load.

Such an evaluation may point to the potential for hybrid versions of current scene representations that could benefit the field. For example, adaptive representations could be used to balance the resolution and processing power required of the scene. There are initial explorations for using Octrees as scene representation for grasping in a Reinforcement Learning approach [173]. They can represent "unimportant" regions of the scene with a coarse resolution while representing object surfaces in more detail and could be an interesting representation to compare against more traditional methods. Furthermore, one could combine such an adaptive scene representation with an uncertainty measure (as mentioned in the previous section) to increase robustness, similar to the approach used for the UFOMap mapping framework [174].

6.1.3 Grasp representation

In tandem with the scene representation, grasps themselves must be represented in the scene to define the position and orientation of the robot's end-effector. While in this thesis, we focused on parallel-jaw grippers due to their low complexity and long lifetime (see Section 2.1), other manipulation tasks might require different gripper designs and different grasp representations.

Traditionally, grasps for parallel-jaw grippers are anchored at the TCP of the end-effector [9, 10, 11, 12, 13, 14, 15, 16, 21, 23, 37, 38, 45, 63, 64]. In Contact-GraspNet [20], a contact-based grasp representation was presented where the grasps are anchored at one of the gripper contacts. We tested this contact-based grasp representation with GP-net in Chapter 4 and compared it to a TCP-based grasp representation where we found advantages for convergence of the grasp orientation using the contact-based representation in line with the assumptions made in [20].

However, the contact-based grasps can only be anchored to visible contact points, which limits applicability in cases where a grasp is possible but the grasp contact is not visible. This can be the case when viewing objects like books from their spine and is worsened in situations with reduced visibility, for example, in shelves and cupboards. A combination of grasp representations or an entirely different approach might be needed to merge the advantages of contact-based grasp representations with a more robust availability of grasp proposals in the case of limited visibility of grasp contacts.

6.1.4 Adaptable benchmarks for robotic grasping

A major problem in robotic grasping research is comparability between different algorithms [8], as discussed in Section 2.2.2. If tested on different evaluation sets or scene setups, or when using different hardware for the tests, the results are not comparable. To properly compare algorithms against each other, they have to be tested in the same setup, either in real-world or simulated experiments.

Other disciplines in computer vision and robotics often have benchmarks adapted across a field of research and use them to compare solutions, for example, object detection [171] or SLAM [175] algorithms. In contrast, while there have been several benchmarks introduced for robotic grasping, e.g. EGAD [74], Graspnet-1-billion [39] or DexYCB [176], there exists no benchmark that has been adopted across the entire robotics grasping community so far. This is due to challenges that arise from the variability of approaches within robotics grasping research, with differences in scene representation, camera positioning, grasp flexibility, grasp representation, gripper design, object preferences, application and available hardware making it difficult to provide a single benchmark that can be used for every project.

Under these circumstances, a common framework with standardised simulation experiments could play a key role in making research more comparable. While simulation experiments always make assumptions and simplifications in contrast to real-world environments, they could provide a basis for comparison due to their high adaptability, transferability and low costs. There are several aspects that need to be considered when generating such a framework, however. These include exactly which setups to cover, how to ensure adaptability to various scene and grasp representations, and how to reduce the sim-to-real gap. If successfully implemented and used by researchers, such a simulation framework could add a common baseline for comparing the performance of grasping solutions.

6.1.5 Sensor fusion for closed-loop control

Real-world experiments with robots often demonstrate the need for closed-loop control [4], with inaccuracies in perception, actuation and dynamic environments making it crucial for a grasping system to react and reposition the gripper. One possible solution for closed-loop control in robotic grasping is fusing different sensor modalities to accumulate their advantages.

Wrist-mounted cameras can be used for closed-loop object grasping by continually re-evaluating potential grasps during the approach [12, 36, 43], for example, using a grasp proposal model during the final approach to the object to correct miss-alignments of the gripper. Furthermore, tactile sensors have been used to detect object slippage [47], reconstruct occluded regions of the object geometry [48, 53] and reduce uncertainty in object pose estimates [49].

Combining different sensor modalities could make grasping approaches more robust since the combined modalities can represent different aspects of the real world, enhancing the scene representation. When choosing to deploy such multi-modal approaches, one must ensure they are compatible with the chosen grasping pipeline. Some of the sensor fusion and control approaches described above can be combined with the grasping techniques presented in this thesis. For example, wrist-mounted cameras may be used for a closed-loop, final approach towards the object or the grasping procedure in our GP-net+ ROS package could be modified to detect object slippage.

6.1.6 Holistic grasping

Historically, when planning paths for mobile manipulators, the base movement is decoupled from the movement of the arm, and potentially the torso, of the robotic manipulator [132, 135, 140]. Hence, a robot would approach an object, stop, plan a grasping pose, plan a path for the grasping pose and finally execute the path with the arm. This approach is based on the high dimensionality and uncertainty for mobile motions, causing computationally expensive global motion planning [3].

In contrast to this decoupled approach, there has been some recent work in controlling whole-body motions for grasping with full systems like Garbage-Net [129] or holistic motion controllers [130]. Having a single, fluid motion that combines the movement of the mobile base with the movement of the robotic arm has several advantages: grasp planning and approaching the object can be executed in parallel, reducing run-time, increasing efficiency, providing more viewpoints for the grasp planning and having a “more natural” and potentially less intimidating motion of the robot. This latter point has particular relevance when operating near people in domestic environments.

It is important to ensure that grasp proposal systems fit the requirements of such methods. One has to consider their run-time for closed-loop control, their application range in terms of camera viewpoints, and the stability of their grasp proposal predictions when making small changes to the camera viewpoint and scene representation during object approach. Combining a robust grasp proposal system with a holistic motion controller could enable a safe and robust application of assistive robotic devices for grasping in complex, domestic environments.

6.1.7 General-purpose robots through Machine Learning

In the last decade, substantial developments have been made in many different areas using Machine Learning techniques. For example, CNN models have significantly improved tasks used in robots like image classification, object detection or object tracking [177]. More recently, generative architectures like transformers, adversarial networks and diffusion models have led to advancements in domains like Natural Language Processing (NLP), computer graphics and computer vision [178].

In recent years, researchers have started using multi-modal models that combine text, images, and other sensory data to increase their variety of use cases. Such models include Google Gemini [179] or GPT-4 [180]. Various efforts have been made to apply those advances to the field of robotics in order to generate foundational models that are capable of solving a diverse set of embodied tasks.

As of 2023, multi-modal language models have been successfully embodied in robot control loops to enable manipulation tasks like retrieving objects from a drawer (PaLM-E [181]) or using a sponge to clean a surface (SayCan [182]). In these examples, the large language model divides high-level tasks (such as cleaning the surface) into a series of low-level, executable instructions (such as picking up the sponge or navigating to the surface). Such an approach could be valuable in bridging the gap between the vast progress in very specific grasping and manipulation tasks and the implementation of general-purpose robotic systems.

The grasping tasks for PaLM-E and SayCan are implemented as RL algorithms [182], behavioural cloning approaches [182] or Transformer-based control models [181, 183]. So far, and to the best of our knowledge, the application of grasping tasks in the presented high-level pipelines is mostly limited to simple object shapes (for example, cylindrical, spherical or rectangular shapes [182, 183]) and has yet to be tested on grasping benchmarks containing more general object sets. However, since these models have only been developed within the last year, there is considerable opportunity for further development in more challenging scenarios. As the field advances, recent progress in the generalisation of low-level tasks, such as grasping more complex objects in domestic scenarios with grasp proposal models or RL approaches, could be incorporated with such pipelines.

There are several aspects that have to be considered when moving towards such general-purpose robotic systems. An example of a current limitation is the computational efficiency of solutions. As of today, a key to the performance of large language models is the very large number of trainable parameters they use. This, in turn, requires a large amount of suitable training data and large computational resources for training and inference. Especially when applying the solutions on mobile robotic devices, the required computing power for inference is a crucial element that might prevent the efficient usage of new concepts. While cloud-based applications might be a solution in such cases, overall efficiency should still be investigated to limit unnecessary “waste” of resources. There is no doubt that progress in Machine Learning will change the field of robotics within the next few years, and it will be interesting to observe these new developments both in terms of the problems they solve and the new opportunities they present.

6.2 Conclusions

In conclusion, the main contribution of this thesis has been to develop an approach to robotic grasping in complex, domestic environments. This was done in three steps: removing constraints on a simple grasp quality prediction model, converting it into a grasp proposal method, and finally developing simulation environments for training and evaluating a grasp proposal method in domestic scenarios. In all of the steps, we took particular care in investigating the limitations of the respective methods and, if possible, testing them in real-world experiments with a mobile manipulator.

We believe that recent developments in machine learning can accelerate the pace of developments in robotic grasping within the next decade if crucial questions like scene representations, grasp representations and reproducible benchmarking are considered in their approaches. We hope that within that time, these developments will enable safe and robust usage of mobile manipulators in domestic environments, potentially assisting with various tasks in the home such that robots can help people to live independently.

References

[1] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. "Robotics: Modelling, Planning and Control". In: 9781846286414. Springer International Publishing, 2009. Chap. Introduction, pp. 1–37. DOI: 10 . 1007/978-1-84628-642-1{_}1.

[2] Bruno Siciliano and Oussama Khatib. "1. Robotics and the handbook". In: *Springer Handbooks* (2016), pp. 1–6. ISSN: 25228706. DOI: 10.1007/978-3-319-32552-1{_}1/COVER. URL: https://link.springer.com/chapter/10.1007/978-3-319-32552-1_1.

[3] Marc Toussaint Oliver Brock Jaeheung Park. "Mobility and Manipulation". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. 2nd ed. Apr. 2008. Chap. 40, pp. 1007–1035.

[4] Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. "Data-Driven Grasp Synthesis—A Survey". In: *IEEE Transactions on Robotics* 30.2 (Apr. 2014), pp. 289–309.

[5] Rhys Newbury, Morris Gu, Lachlan Chumbley, Arsalan Mousavian, Clemens Eppner, Jurgen Leitner, Jeannette Bohg, Antonio Morales, Tamim Asfour, Danica Kragic, Dieter Fox, and Akansel Cosgun. "Deep Learning Approaches to Grasp Synthesis: A Review". In: *IEEE Transactions on Robotics* (2023). ISSN: 19410468. DOI: 10.1109/TRO.2023.3280597.

[6] Antonio Bicchi and Vijay Kumar. "Robotic grasping and contact: a review". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat.*

No.00CH37065). Vol. 1. 2000, pp. 348–353. DOI: 10.1109/ROBOT.2000.844081.

[7] Kilian Kleeberger, Richard Bormann, Werner Kraus, and Marco F Huber. “A survey on learning-based robotic grasping”. In: *Current Robotics Reports* (2020), pp. 1–11.

[8] Robert Platt. “Grasp Learning: Models, Methods, and Performance”. In: *Annual Review of Control, Robotics, and Autonomous Systems* 6 (2022).

[9] Joseph Redmon and Anelia Angelova. “Real-time grasp detection using convolutional neural networks”. In: *2015 IEEE International Conference on Robotics and Automation (ICRA).* 2015, pp. 1316–1322. DOI: 10.1109/ICRA.2015.7139361.

[10] Ian Lenz, Honglak Lee, and Ashutosh Saxena. “Deep learning for detecting robotic grasps”. In: *The International Journal of Robotics Research* 34.4-5 (2015), pp. 705–724. DOI: 10.1177/0278364914549607. URL: https://doi.org/10.1177/0278364914549607.

[11] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. “Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics”. In: *Robotics: Science and Systems (RSS).* 2017.

[12] Douglas Morrison, Peter Corke, and Jürgen Leitner. “Learning robust, real-time, reactive robotic grasping”. In: *The International Journal of Robotics Research* 39.2-3 (2019), pp. 183–201. DOI: 10.1177/0278364919859066. URL: https://doi.org/10.1177/0278364919859066.

[13] Umar Asif, Jianbin Tang, and Stefan Harrer. “EnsembleNet: Improving Grasp Detection using an Ensemble of Convolutional Neural Networks.” In: *British Machine Vision Conference (BMVC).* 2018, p. 10.

[14] Jiahao Zhang, Miao Li, Ying Feng, and Chenguang Yang. “Robotic grasp detection based on image processing and random forest”. In: *Multimedia Tools and Applications* 79 (Apr. 2020). DOI: 10.1007/s11042-019-08302-9.

[15] Jens Lundell, Francesco Verdoja, and Ville Kyrki. “Robust Grasp Planning over Uncertain Shape Completions”. In: *IEEE International Conference on Intelligent Robots and Systems* (Nov. 2019), pp. 1526–1532. ISSN: 21530866. DOI: 10.1109/IROS40897.2019.8967816.

[16] Michel Breyer, Jen Jen Chung, Lionel Ott, Siegwart Roland, and Nieto Juan. “Volumetric Grasping Network: Real-time 6 DOF Grasp Detection in Clutter”. In: *Conference on Robot Learning.* 2020.

[17] Junhao Cai, Jingcheng Su, Zida Zhou, Hui Cheng, Qifeng Chen, and Michael Yu Wang. “Volumetric-based Contact Point Detection for 7-DoF Grasping”. In: *Proceedings of The 6th Conference on Robot Learning.* PMLR, Mar. 2023, pp. 824–834. URL: https://proceedings.mlr.press/v205/cai23a.html.

[18] Jens Lundell, Francesco Verdoja, and Ville Kyrki. “Beyond Top-Grasps Through Scene Completion”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA).* 2020, pp. 545–551. DOI: 10.1109/ICRA40945.2020.9197320.

[19] Andreas Ten Pas and Robert Platt. “Using geometry to detect grasp poses in 3d point clouds”. In: *Robotics Research.* Springer, 2018, pp. 307–324.

[20] Martin Sundermeyer, Arsalan Mousavian, Rudolph Triebel, and Dieter Fox. “Contact-GraspNet: Efficient 6-DoF Grasp Generation in Cluttered Scenes”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA).* 2021, pp. 13438–13444. DOI: 10.1109/ICRA48506.2021.9561877.

[21] Zhenyu Jiang, Yifeng Zhu, Maxwell Svetlik, Kuan Fang, and Yuke Zhu. “Synergies Between Affordance and Geometry: 6-DoF Grasp Detection via Implicit Representations”. In: *Robotics: science and systems* (2021).

[22] Jeffrey Mahler and Ken Goldberg. “Learning Deep Policies for Robot Bin Picking by Simulating Robust Grasping Sequences”. In: *Proceedings of the 1st Annual Conference on Robot Learning.* Ed. by Sergey Levine, Vincent Vanhoucke, and Ken Goldberg. Vol. 78. Proceedings of Machine Learning Research. PMLR, Apr. 2017, pp. 515–524.

[23] Sulabh Kumra, Shirin Joshi, and Ferat Sahin. “Antipodal Robotic Grasping using Generative Residual Convolutional Neural Network”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 9626–9633. DOI: 10.1109/IROS45743.2020.9340777.

[24] Andreas ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. “Grasp Pose Detection in Point Clouds”. In: *The International Journal of Robotics Research* 36.13-14 (2017), pp. 1455–1473. DOI: 10.1177/0278364917735594. URL: https://doi.org/10.1177/0278364917735594.

[25] Xinke Deng, Yu Xiang, Arsalan Mousavian, Clemens Eppner, Timothy Bretl, and Dieter Fox. “Self-supervised 6D Object Pose Estimation for Robot Manipulation”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 3665–3671. DOI: 10.1109/ICRA40945.2020.9196714.

[26] Hang Yin, Anastasia Varava, and Danica Kragic. “Modeling, learning, perception, and control methods for deformable object manipulation”. In: *Science Robotics* 6.54 (May 2021), p. 8803. ISSN: 24709476. DOI: 10.1126/SCIROBOTICS.ABD8803/ASSET/0A692D57-A363-4740-82FF-766ABEF7410C/ASSETS/GRAPHIC/ABD8803-F6.JPEG. URL: https://www.science.org/doi/10.1126/scirobotics.abd8803.

[27] Jihong Zhu, Andrea Cherubini, Claire Dune, David Navarro-Alarcon, Farshid Alambeigi, Dmitry Berenson, Fanny Ficuciello, Kensuke Harada, Jens Kober, Xiang Li, Jia Pan, Wenzhen Yuan, and Michael Gienger. “Challenges and Outlook in Robotic Manipulation of Deformable Objects”. In: *IEEE Robotics and Automation Magazine* 29.3 (Sept. 2022), pp. 67–77. ISSN: 1558223X. DOI: 10.1109/MRA.2022.3147415.

[28] Domenico Prattichizzo and Jeffrey C Trinkle. “Grasping”. In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. 2nd ed. Apr. 2008. Chap. 38, pp. 955–986.

[29] Yun Jiang, Marcus Lim, Changxi Zheng, and Ashutosh Saxena. “Learning to place new objects in a scene”. In: *International Journal of Robotics Research* 31.9 (May 2012), pp. 1021–1043. ISSN: 02783649. DOI: 10.1177/

0278364912438781. URL: https://journals.sagepub.com/doi/abs/10.1177/0278364912438781.

[30] Joshua A. Haustein, Kaiyu Hang, Johannes Stork, and Danica Kragic. "Object Placement Planning and optimization for Robot Manipulators". In: *IEEE International Conference on Intelligent Robots and Systems* (Nov. 2019), pp. 7417–7424. ISSN: 21530866. DOI: 10.1109/IROS40897.2019.8967732.

[31] Ajung Moon, Minhua Zheng, Daniel M. Troniak, Benjamin A. Blumer, Brian Gleeson, Karon MacLean, Matthew K.X.J. Pan, and Elizabeth A. Croft. "Meet me where i'm gazing: How shared attention gaze affects human-robot handover timing". In: *ACM/IEEE International Conference on Human-Robot Interaction* (2014), pp. 334–341. ISSN: 21672148. DOI: 10.1145/2559636.2559656. URL: https://dl.acm.org/doi/10.1145/2559636.2559656.

[32] Matteo Melchiorre, Leonardo Sabatino Scimmi, Stefano Mauro, and Stefano Paolo Pastorelli. "Vision-based control architecture for human–robot handover applications". In: *Asian Journal of Control* 23.1 (Jan. 2021), pp. 105–117. ISSN: 1934-6093. DOI: 10.1002/ASJC.2480. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/asjc.2480%20https://onlinelibrary.wiley.com/doi/abs/10.1002/asjc.2480%20https://onlinelibrary.wiley.com/doi/10.1002/asjc.2480.

[33] Andy Zeng, Shuran Song, Kuan Ting Yu, Elliott Donlon, Francois R. Hogan, Maria Bauza, Daolin Ma, Orion Taylor, Melody Liu, Eudald Romo, Nima Fazeli, Ferran Alet, Nikhil Chavan Dafle, Rachel Holladay, Isabella Morona, Prem Qu Nair, Druck Green, Ian Taylor, Weber Liu, Thomas Funkhouser, and Alberto Rodriguez. "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching". In: *International Journal of Robotics Research* 41.7 (June 2022), pp. 690–705. ISSN: 17413176. DOI: 10.1177/0278364919868017/ASSET/IMAGES/LARGE/10.1177{_}0278364919868017-FIG8.JPEG. URL: https://journals.sagepub.com/doi/full/10.1177/0278364919868017.

[34] Paola Ardon, Eric Pairet, Ronald P.A. Petrick, Subramanian Ramamoorthy, and Katrin S. Lohan. “Learning Grasp Affordance Reasoning through Semantic Relations”. In: *IEEE Robotics and Automation Letters* 4.4 (Oct. 2019), pp. 4571–4578. ISSN: 23773766. DOI: 10.1109/LRA.2019.2933815.

[35] Anh Nguyen, Dimitrios Kanoulas, Darwin G. Caldwell, and Nikos G. Tsagarakis. “Object-based affordances detection with Convolutional Neural Networks and dense Conditional Random Fields”. In: *IEEE International Conference on Intelligent Robots and Systems* 2017-September (Dec. 2017), pp. 5908–5915. ISSN: 21530866. DOI: 10.1109/IROS.2017.8206484.

[36] Shuran Song, Andy Zeng, Johnny Lee, and Thomas Funkhouser. “Grasping in the Wild: Learning 6DoF Closed-Loop Grasping from Low-Cost Demonstrations”. In: *Robotics and Automation Letters* (2020).

[37] Lars Berscheid, Christian Friedrich, and Torsten Kröger. “Robot Learning of 6 DoF Grasping using Model-based Adaptive Primitives”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 4474–4480. DOI: 10.1109/ICRA48506.2021.9560901.

[38] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. “6-DOF GraspNet: Variational Grasp Generation for Object Manipulation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[39] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. “GraspNet-1Billion: A Large-Scale Benchmark for General Object Grasping”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[40] Hongzhuo Liang, Xiaojian Ma, Shuang Li, Michael Görner, Song Tang, Bin Fang, Fuchun Sun, and Jianwei Zhang. “PointNetGPD: Detecting Grasp Configurations from Point Sets”. In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 3629–3635. DOI: 10.1109/ICRA.2019.8794435.

[41] Anna Konrad, John McDonald, and Rudi Villing. “GP-Net: Flexible Viewpoint Grasp Proposal”. In: *2023 21st International Conference on Advanced Robotics (ICAR)* (Dec. 2023), pp. 317–324. DOI: 10.1109/ICAR58858.2023.10406781. URL: https://ieeexplore.ieee.org/document/10406781/.

[42] Philipp Schmidt, Nikolaus Vahrenkamp, Mirko Wächter, and Tamim Asfour. “Grasping of Unknown Objects Using Deep Convolutional Neural Networks Based on Depth Images”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 6831–6838. DOI: 10.1109/ICRA.2018.8463204.

[43] Ulrich Viereck, Andreas Pas, Kate Saenko, and Robert Platt. “Learning a visuomotor controller for real world robotic grasping using simulated depth images”. In: *Conference on Robot Learning*. 2017, pp. 291–300.

[44] Andy Zeng, Shuran Song, Stefan Welker, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. “Learning Synergies between Pushing and Grasping with Self-Supervised Deep Reinforcement Learning”. In: *IEEE International Conference on Intelligent Robots and Systems* (Dec. 2018), pp. 4238–4245. ISSN: 21530866. DOI: 10.1109/IROS.2018.8593986.

[45] Jacob Varley, Chad DeChant, Adam Richardson, Joaquin Ruales, and Peter Allen. “Shape completion enabled robotic grasping”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 2442–2447. DOI: 10.1109/IROS.2017.8206060.

[46] Marcus Gualtieri, Andreas ten Pas, Kate Saenko, and Robert Platt. “High precision grasp pose detection in dense clutter”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 598–605. DOI: 10.1109/IROS.2016.7759114.

[47] Wei Chen, Heba Khamis, Ingvars Birznieks, Nathan F. Lepora, and Stephen J. Redmond. “Tactile Sensors for Friction Estimation and Incipient Slip Detection—Toward Dexterous Robotic Manipulation: A Review”. In: *IEEE Sensors Journal* 18.22 (2018), pp. 9049–9064. ISSN: 1558-1748. DOI: 10.1109/JSEN.2018.2868340.

[48] Cristiana de Farias, Naresh Marturi, Rustam Stolkin, and Yasemin Bekiroglu. “Simultaneous Tactile Exploration and Grasp Refinement for Unknown Objects”. In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 3349–3356. ISSN: 2377-3766. DOI: `10.1109/LRA.2021.3063074`.

[49] Hamza Merzić, Miroslav Bogdanović, Daniel Kappler, Ludovic Righetti, and Jeannette Bohg. “Leveraging Contact Forces for Learning to Grasp”. In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 3615–3621. ISBN: 2577-087X. DOI: `10.1109/ICRA.2019.8793733`.

[50] Dimitrios Kanoulas, Jinoh Lee, Darwin G. Caldwell, and Nikos G. Tsagarakis. “Center-of-Mass-Based Grasp Pose Adaptation Using 3D Range and Force/-Torque Sensing”. In: *https://doi.org/10.1142/S0219843618500135* 15.4 (Aug. 2018). ISSN: 02198436. DOI: `10.1142/S0219843618500135`.

[51] Chao Wang, Xizhe Zang, Xuehe Zhang, Yubin Liu, and Jie Zhao. “Parameter estimation and object gripping based on fingertip force/torque sensors”. In: *Measurement* 179 (July 2021), p. 109479. ISSN: 0263-2241. DOI: `10.1016/J.MEASUREMENT.2021.109479`.

[52] Gabriel Zöller, Vincent Wall, and Oliver Brock. “Active Acoustic Contact Sensing for Soft Pneumatic Actuators”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 7966–7972. ISBN: 2577-087X. DOI: `10.1109/ICRA40945.2020.9196916`.

[53] Simon Ottenhaus, Daniel Renninghoff, Raphael Grimm, Fabio Ferreira, and Tamim Asfour. “Visuo-Haptic Grasping of Unknown Objects based on Gaussian Process Implicit Surfaces and Deep Learning”. In: *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*. 2019, pp. 402–409. ISBN: 2164-0580. DOI: `10.1109/Humanoids43949.2019.9035002`.

[54] J W Soto Martell and Giuseppina Gini. “Robotic hands: Design review and proposal of new design process”. In: *World Academy of Science, Engineering and Technology* 26.5 (2007), pp. 85–90.

[55] Claudio Melchiorri and Makoto Kaneko. “Robot Hands”. In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Cham: Springer International Publishing, 2016, pp. 463–480. ISBN: 978-3-319-32552-1. DOI: 10.1007/978-3-319-32552-1{_}19. URL: https://doi.org/10.1007/978-3-319-32552-1_19.

[56] Jun Shintake, Vito Cacucciolo, Dario Floreano, and Herbert Shea. “Soft robotic grippers”. In: *Advanced materials* 30.29 (2018), p. 1707035.

[57] Debanik Roy. “Development of novel magnetic grippers for use in unstructured robotic workspace”. In: *Robotics and Computer-Integrated Manufacturing* 35 (Oct. 2015), pp. 16–41. ISSN: 0736-5845. DOI: 10.1016/J.RCIM.2015.02.003.

[58] Adrian Peidro, Mahmoud Tavakoli, Jose Maria Marín, and Oscar Reinoso. “Design of compact switchable magnetic grippers for the HyReCRo structure-climbing robot”. In: *Mechatronics* 59 (May 2019), pp. 199–212. ISSN: 0957-4158. DOI: 10.1016/J.MECHATRONICS.2019.04.007.

[59] Jeffrey Mahler, Matthew Matl, Xinyu Liu, Albert Li, David Gealy, and Ken Goldberg. “Dex-Net 3.0: Computing Robust Robot Suction Grasp Targets in Point Clouds using a New Analytic Model and Deep Learning”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 5620–5627. DOI: 10.1109/ICRA.2018.8460887.

[60] Jeffrey Mahler, Matthew Matl, Vishal Satish, Michael Danielczuk, Bill DeRose, Stephen McKinley, and Ken Goldberg. “Learning ambidextrous robot grasping policies”. In: *Science Robotics* 4.26 (2019). DOI: 10.1126/scirobotics.aau4984. URL: https://robotics.sciencemag.org/content/4/26/eaau4984.

[61] Menglong Guo, David V Gealy, Jacky Liang, Jeffrey Mahler, Aimee Goncalves, Stephen McKinley, Juan Aparicio Ojea, and Ken Goldberg. “Design of parallel-jaw gripper tip surfaces for robust grasping”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 2831–2838. DOI: 10.1109/ICRA.2017.7989330.

[62] Yanan Song, Liang Gao, Xinyu Li, and Weiming Shen. “A novel robotic grasp detection method based on region proposal networks”. In: *Robotics and Computer-Integrated Manufacturing* 65 (2020), p. 101963. ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2020.101963. URL: https://www.sciencedirect.com/science/article/pii/S0736584519308105.

[63] Vishal Satish, Jeffrey Mahler, and Ken Goldberg. “On-Policy Dataset Synthesis for Learning Robot Grasping Policies Using Fully Convolutional Deep Networks”. In: *IEEE Robotics and Automation Letters* (2019).

[64] Xinghao Zhu, Lingfeng Sun, Yongxiang Fan, and Masayoshi Tomizuka. “6-DoF Contrastive Grasp Proposal Network”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 6371–6377. DOI: 10.1109/ICRA48506.2021.9561954.

[65] Chenxi Wang, Hao-Shu Fang, Minghao Gou, Hongjie Fang, Jin Gao, and Cewu Lu. “Graspness Discovery in Clutters for Fast and Accurate Grasp Detection”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 15964–15973.

[66] Sulabh Kumra and Christopher Kanan. “Robotic grasp detection using deep convolutional neural networks”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 769–776. DOI: 10.1109/IROS.2017.8202237.

[67] Umar Asif, Jianbin Tang, and Stefan Harrer. “Densely Supervised Grasp Detector (DSGD)”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01 (July 2019), pp. 8085–8093. ISSN: 2374-3468. DOI: 10.1609/AAAI.V33I01.33018085. URL: https://ojs.aaai.org/index.php/AAAI/article/view/4816.

[68] Xinwen Zhou, Xuguang Lan, Hanbo Zhang, Zhiqiang Tian, Yang Zhang, and Narming Zheng. “Fully Convolutional Grasp Detection Network with Oriented Anchor Box”. In: *IEEE International Conference on Intelligent Robots and Systems* (Dec. 2018), pp. 7223–7230. ISSN: 21530866. DOI: 10.1109/IROS.2018.8594116.

[69] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. "Efficient grasping from rgbd images: Learning using a new rectangle representation". In: *2011 IEEE International conference on robotics and automation.* 2011, pp. 3304–3311.

[70] Chaozheng Wu, Jian Chen, Qiaoyu Cao, Jianchi Zhang, Yunxin Tai, Lin Sun, and Kui Jia. "Grasp proposal networks: an end-to-end solution for visual learning of robotic grasps". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 13174–13184.

[71] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. "OctoMap: an efficient probabilistic 3D mapping framework based on octrees". In: *Autonomous Robots* 34.3 (2013), pp. 189–206. ISSN: 1573-7527. DOI: 10.1007/s10514-012-9321-0. URL: https://doi.org/10.1007/s10514-012-9321-0.

[72] David Coleman, Ioan A. Șucan, Sachin Chitta, and Nikolaus Correll. " Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study". In: *Journal of Software Engineering for Robotics* 5 (May 2014), pp. 3–16.

[73] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection". In: *The International Journal of Robotics Research* 37.4-5 (2018), pp. 421–436. DOI: 10.1177/0278364917710318. URL: https://doi.org/10.1177/0278364917710318.

[74] Douglas Morrison, Peter Corke, and Jürgen Leitner. "EGAD! An Evolved Grasping Analysis Dataset for Diversity and Reproducibility in Robotic Manipulation". In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4368–4375. DOI: 10.1109/LRA.2020.2992195.

[75] Robert Krug, Yasemin Bekiroglu, Danica Kragic, and Maximo A. Roa. "Evaluating the quality of non-prehensile balancing grasps". In: *Proceedings - IEEE International Conference on Robotics and Automation* (Sept. 2018), pp. 4215–4220. ISSN: 10504729. DOI: 10.1109/ICRA.2018.8461078.

[76] Ian Lenz, Honglak Lee, and Ashutosh Saxena. *Cornell Grasping Dataset.* 2013.

[77] Amaury Depierre, Emmanuel Dellandréa, and Liming Chen. “Jacquard: A Large Scale Dataset for Robotic Grasp Detection”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* 2018, pp. 3511–3516. DOI: `10.1109/IROS.2018.8593950`.

[78] Ashutosh Saxena, Justin Driemeyer, Justin Kearns, and Andrew Y Ng. “Robotic Grasping of Novel Objects”. In: *Advances in Neural Information Processing Systems* 19 (2006).

[79] Changhyun Choi, Wilko Schwarting, Joseph Delpreto, and Daniela Rus. “Learning Object Grasping for Soft Robot Hands”. In: *IEEE Robotics and Automation Letters* 3.3 (July 2018), pp. 2370–2377. ISSN: 23773766. DOI: `10.1109/LRA.2018.2810544`.

[80] Jesse Davis and Mark Goadrich. “The relationship between precision-recall and ROC curves”. In: *ACM International Conference Proceeding Series* 148 (2006), pp. 233–240. DOI: `10.1145/1143844.1143874`. URL: `https://dl.acm.org/doi/10.1145/1143844.1143874`.

[81] Jens Lundell, Francesco Verdoja, and Ville Kyrki. “DDGC: Generative Deep Dexterous Grasping in Clutter”. In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 6899–6906. DOI: `10.1109/LRA.2021.3096239`.

[82] Adithyavairavan Murali, Arsalan Mousavian, Clemens Eppner, Chris Paxton, and Dieter Fox. “6-DOF Grasping for Target-driven Object Manipulation in Clutter”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA).* 2020, pp. 6232–6238. DOI: `10.1109/ICRA40945.2020.9197318`.

[83] Dmitry Berenson. “Manipulation of deformable objects without modeling and simulating deformation”. In: *IEEE International Conference on Intelligent Robots and Systems* (2013), pp. 4525–4532. ISSN: 21530858. DOI: `10.1109/IROS.2013.6697007`.

[84] Jeffrey Ichnowski*, Yahav Avigal*, Justin Kerr, and Ken Goldberg. "Dex-NeRF: Using a Neural Radiance field to Grasp Transparent Objects". In: *Conference on Robot Learning (CoRL)*. 2020.

[85] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. "Benchmarking in Manipulation Research: Using the Yale-CMU-Berkeley Object and Model Set". In: *IEEE Robotics and Automation Magazine* 22.3 (2015), pp. 36–52. DOI: 10.1109/MRA.2015.2448951.

[86] Arjun Singh, James Sha, Karthik S Narayan, Tudor Achim, and P Abbeel. "BigBIRD: A large-scale 3D database of object instances". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)* (2014), pp. 509–516.

[87] Alexander Kasper, Zhixing Xue, and Rüdiger Dillmann. "The KIT object models database: An object model database for object recognition, localization and manipulation in service robotics". In: *The International Journal of Robotics Research* 31.8 (2012), pp. 927–934. DOI: 10.1177/0278364912445831. URL: https://doi.org/10.1177/0278364912445831.

[88] Jeffrey Mahler, Florian T Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kröger, James Kuffner, and Ken Goldberg. "Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 1957–1964.

[89] David L. Page, Andreas F. Koschan, Sreenivas R. Sukumar, B. Roui-Abidi, and Mongi A. Abidi. "Shape analysis algorithm based on information theory". In: *IEEE International Conference on Image Processing* 1 (2003), pp. 229–232. DOI: 10.1109/ICIP.2003.1246940.

[90] Joshua E. Auerbach and Josh C. Bongard. "Environmental Influence on the Evolution of Morphological Complexity in Machines". In: *PLOS Computational Biology* 10.1 (2014), e1003399. ISSN: 1553-7358. DOI: 10.1371/JOURNAL.PCBI.1003399. URL: https://journals.plos.org/ploscompbiol/

`article?id=10.1371/journal.pcbi.1003399%20http://www.darpa.mil/%20http://www.nasa.gov/`.

[91] David Wang, David Tseng, Pusong Li, Yiding Jiang, Menglong Guo, Michael Danielczuk, Jeffrey Mahler, Jeffrey Ichnowski, and Ken Goldberg. "Adversarial grasp objects". In: *IEEE International Conference on Automation Science and Engineering* 2019-August (Aug. 2019), pp. 241–248. ISSN: 21618089. DOI: `10.1109/COASE.2019.8843059`.

[92] Tao Chen, Jie Xu, and Pulkit Agrawal. "A System for General In-Hand Object Re-Orientation". In: *Proceedings of the 5th Conference on Robot Learning.* Ed. by Aleksandra Faust, David Hsu, and Gerhard Neumann. Vol. 164. Proceedings of Machine Learning Research. PMLR, Apr. 2022, pp. 297–307. URL: `https://proceedings.mlr.press/v164/chen22a.html`.

[93] Jens Lundell, Enric Corona, Tran Nguyen Le, Francesco Verdoja, Philippe Weinzaepfel, Grégory Rogez, Francesc Moreno-Noguer, and Ville Kyrki. "Multi-FinGAN: Generative Coarse-To-Fine Sampling of Multi-Finger Grasps". In: *2021 IEEE International Conference on Robotics and Automation (ICRA).* 2021, pp. 4495–4501. DOI: `10.1109/ICRA48506.2021.9561228`.

[94] Wei Wei, Daheng Li, Peng Wang, Yiming Li, Wanyi Li, Yongkang Luo, and Jun Zhong. "DVGG: Deep Variational Grasp Generation for Dextrous Manipulation". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 1659–1666. DOI: `10.1109/LRA.2022.3140424`.

[95] Tanwi Mallick, Partha Pratim Das, and Arun Kumar Majumdar. "Characterizations of noise in Kinect depth images: A review". In: *IEEE Sensors Journal* 14.6 (2014), pp. 1731–1740. ISSN: 1530437X. DOI: `10.1109/JSEN.2014.2309987`.

[96] Michael Riis Andersen, T. Jensen, P. Lisouski, A. K. Mortensen, M. K. Hansen, T. Gregersen, and P. J. A. U. Ahrendt. *Kinect depth sensor evaluation for computer vision applications.* Tech. rep. Aarhus University, 2012, pp. 1–37.

[97] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and guiseppe Oriolo. "Kinematics". In: *Advanced Textbooks in Control and Signal Processin* (2009), pp. 39–103. ISSN: 25103814. DOI: 10.1007/978-1-84628-642-1{_}2/COVER. URL: https://link.springer.com/chapter/10.1007/978-1-84628-642-1_2.

[98] PAL Robotics. *TIAGo technical specifications.* 2021. URL: https://pal-robotics.com/wp-content/uploads/2021/07/Datasheet-complete_TIAGo-2021.pdf.

[99] Franka Emika. *Franka Hand - Product Manual.* Dec. 2022. URL: https://download.franka.de/documents/220010_Product%20Manual_Franka%20Hand_1.2_EN.pdf.

[100] Cliff Fitzgerald. "Developing baxter". In: *2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA).* 2013, pp. 1–6. ISBN: 2325-0534. DOI: 10.1109/TePRA.2013.6556344.

[101] Robotiq. *Robotiq 2F-85 gripper specifications.* URL: https://assets.robotiq.com/website-assets/support_documents/document/online/2F-85_2F-140_TM_InstructionManual_HTML5_20190503.zip/2F-85_2F-140_TM_InstructionManual_HTML5/Content/6.%20Specifications.htm.

[102] Franka Emika. *Franka Emika Robot - Product Manual.* Oct. 2021. URL: https://download.franka.de/documents/100010_Product%20Manual%20Franka%20Emika%20Robot_10.21_EN.pdf.

[103] Richard M. Murray, Zexiang Li, and S. Shankar Sastry. "A mathematical introduction to robotic manipulation". In: *A Mathematical Introduction to Robotic Manipulation* (Jan. 2017), pp. 1–456. DOI: 10.1201/9781315136370/MATHEMATICAL-INTRODUCTION-ROBOTIC-MANIPULATION-RICHARD-MURRAY-ZEXIANG-LI-SHANKAR-SASTRY. URL: https://www.taylorfrancis.com/books/mono/10.1201/9781315136370/mathematical-introduction-robotic-manipulation-richard-murray-zexiang-li-shankar-sastry.

[104] Van-Duc Nguyen. "Constructing force-closure grasps". In: *The International Journal of Robotics Research* 7.3 (1988), pp. 3–16.

[105] Antonio Bicchi. "On the closure properties of robotic grasping". In: *The International Journal of Robotics Research* 14.4 (1995), pp. 319–334.

[106] Li Han, J C Trinkle, and Z X Li. "Grasp analysis as linear matrix inequality problems". In: *IEEE Transactions on Robotics and Automation* 16.6 (2000), pp. 663–674. DOI: `10.1109/70.897778`.

[107] Carlo Ferrari and John Canny. "Planning optimal grasps". In: *Proceedings 1992 IEEE International Conference on Robotics and Automation.* 1992, pp. 2290–2295. DOI: `10.1109/ROBOT.1992.219918`.

[108] Jonathan Weisz and Peter K Allen. "Pose error robust grasping from contact wrench space metrics". In: *2012 IEEE International Conference on Robotics and Automation.* 2012, pp. 557–562.

[109] Máximo A. Roa and Raúl Suárez. "Grasp quality measures: review and performance". In: *Autonomous Robots* 38.1 (Jan. 2015), p. 65. ISSN: 15737527. DOI: `10.1007/S10514-014-9402-3`. URL: `/pmc/articles/PMC4457357/%20/pmc/articles/PMC4457357/?report=abstract%20https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4457357/`.

[110] Daniel Seita, Florian T Pokorny, Jeffrey Mahler, Danica Kragic, Michael Franklin, John Canny, and Ken Goldberg. "Large-scale supervised learning of the grasp robustness of surface patch pairs". In: *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR).* 2016, pp. 216–223.

[111] Lin Shao, Fabio Ferreira, Mikael Jorda, Varun Nambiar, Jianlan Luo, Eugen Solowjow, Juan Aparicio Ojea, Oussama Khatib, and Jeannette Bohg. "UniGrasp: Learning a Unified Model to Grasp With Multifingered Robotic Hands". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2286–2293. DOI: `10.1109/LRA.2020.2969946`.

[112] Puhao Li, Tengyu Liu, Yuyang Li, Yiran Geng, Yixin Zhu, Yaodong Yang, and Siyuan Huang. "GenDexGrasp: Generalizable Dexterous Grasping". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)* (May 2023), pp. 8068–8074. DOI: `10.1109/ICRA48891.2023.10160667`. URL: `https://ieeexplore.ieee.org/document/10160667/`.

[113] Lerrel Pinto and Abhinav Gupta. "Supersizing self-supervision: Learning to grasp from 50K tries and 700 robot hours". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 3406–3413. DOI: `10.1109/ICRA.2016.7487517`.

[114] Erwin Coumans and Yunfei Bai. *Pybullet, a python module for physics simulation for games, robotics and machine learning*. 2016.

[115] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. "Object detection with deep learning: A review". In: *IEEE transactions on neural networks and learning systems* 30.11 (2019), pp. 3212–3232.

[116] Shaochen Wang, Zhangli Zhou, and Zhen Kan. "When Transformer Meets Robotic Grasping: Exploits Context for Efficient Grasp Detection". In: *IEEE Robotics and Automation Letters* 7.3 (July 2022), pp. 8170–8177. ISSN: 23773766. DOI: `10.1109/LRA.2022.3187261`.

[117] Xibai Lou, Yang Yang, and Changhyun Choi. "Learning to Generate 6-DoF Grasp Poses with Reachability Awareness". In: *Proceedings - IEEE International Conference on Robotics and Automation* (Oct. 2019), pp. 1532–1538. ISSN: 10504729. DOI: `10.1109/ICRA40945.2020.9197413`. URL: `https://arxiv.org/abs/1910.06404v3`.

[118] Yuzhe Qin, Rui Chen, Hao Zhu, Meng Song, Jing Xu, and Hao Su. "S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes". In: *Conference on Robot Learning*. 2020, pp. 53–65.

[119] Wenkai Chen, Hongzhuo Liang, Zhaopeng Chen, Fuchun Sun, and Jianwei Zhang. *TransSC: Transformer-based Shape Completion for Grasp Evaluation*. 2021.

[120] Hamidreza Kasaei and Mohammadreza Kasaei. “MVGrasp: Real-time multi-view 3D object grasping in highly cluttered environments”. In: *Robotics and Autonomous Systems* 160 (2023), p. 104313. ISSN: 0921-8890. DOI: `https://doi.org/10.1016/j.robot.2022.104313`. URL: `https://www.sciencedirect.com/science/article/pii/S0921889022002020`.

[121] Haojie Huang, Dian Wang, Xupeng Zhu, Robin Walters, and Robert Platt. “Edge Grasp Network: A Graph-Based SE(3)-invariant Approach to Grasp Detection”. In: *Proceedings - IEEE International Conference on Robotics and Automation* 2023-May (2023), pp. 3882–3888. ISSN: 10504729. DOI: `10.1109/ICRA48891.2023.10160728`.

[122] Xupeng Zhu, Dian Wang, Guanang Su, Ondrej Biza, Robin Walters, and Robert Platt. “On Robot Grasp Learning Using Equivariant Models”. In: (June 2023). URL: `https://arxiv.org/abs/2306.06489v1`.

[123] Marios Kiatos, Iason Sarantopoulos, Leonidas Koutras, Sotiris Malassiotis, and Zoe Doulgeri. “Learning Push-Grasping in Dense Clutter”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 8783–8790. DOI: `10.1109/LRA.2022.3188437`.

[124] Hai Nguyen and Hung La. “Review of Deep Reinforcement Learning for Robot Manipulation”. In: *2019 Third IEEE International Conference on Robotic Computing (IRC)*. 2019, pp. 590–595. DOI: `10.1109/IRC.2019.00120`.

[125] Marwan Qaid Mohammed, Kwek Lee Chung, and Chua Shing Chyi. “Review of deep reinforcement learning-based object grasping: Techniques, open challenges, and recommendations”. In: *IEEE Access* 8 (2020), pp. 178450–178481. ISSN: 21693536. DOI: `10.1109/ACCESS.2020.3027923`.

[126] Cheng Zhang, Liang Ma, and Alexander Schmitz. “A sample efficient model-based deep reinforcement learning algorithm with experience replay for robot manipulation”. In: *International Journal of Intelligent Robotics and Applications* 4.2 (June 2020), pp. 217–228. ISSN: 2366598X. DOI: `10.1007/S41315-020-00135-2/TABLES/3`. URL: `https://link.springer.com/article/10.1007/s41315-020-00135-2`.

[127] Kechun Xu, Hongxiang Yu, Qianen Lai, Yue Wang, and Rong Xiong. "Efficient Learning of Goal-Oriented Push-Grasping Synergy in Clutter". In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 6337–6344. DOI: 10.1109/LRA.2021.3092640.

[128] Liyiming Ke, Jingqiang Wang, Tapomayukh Bhattacharjee, Byron Boots, and Siddhartha Srinivasa. "Grasping with Chopsticks: Combating Covariate Shift in Model-free Imitation Learning for Fine Manipulation". In: *Proceedings - IEEE International Conference on Robotics and Automation* 2021-May (2021), pp. 6185–6191. ISSN: 10504729. DOI: 10.1109/ICRA48506.2021.9561662.

[129] Jingyi Liu, Pietro Balatti, Kirsty Ellis, Denis Hadjivelichkov, Danail Stoyanov, Arash Ajoudani, and Dimitrios Kanoulas. "Garbage Collection and Sorting with a Mobile Manipulator using Deep Learning and Whole-Body Control". In: *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*. 2021, pp. 408–414. DOI: 10.1109/HUMANOIDS47582.2021.9555800.

[130] Jesse Haviland, Niko Sünderhauf, and Peter Corke. "A Holistic Approach to Reactive Mobile Manipulation". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 3122–3129. DOI: 10.1109/LRA.2022.3146554.

[131] Cong Wang, Qifeng Zhang, Qiyan Tian, Shuo Li, Xiaohui Wang, David Lane, Yvan Petillot, and Sen Wang. "Learning Mobile Manipulation through Deep Reinforcement Learning". In: *Sensors* 20.3 (2020). ISSN: 1424-8220. DOI: 10.3390/s20030939. URL: https://www.mdpi.com/1424-8220/20/3/939.

[132] Jörg Stückler, Ricarda Steffens, Dirk Holz, and Sven Behnke. "Efficient 3D object perception and grasp planning for mobile manipulation in domestic environments". In: *Robotics and Autonomous Systems* 61.10 (2013), pp. 1106–1115. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2012.08.003. URL: https://www.sciencedirect.com/science/article/pii/S0921889012001297.

[133] Michalis Logothetis, George C Karras, Shahab Heshmati-Alamdari, Panagiotis Vlantis, and Kostas J Kyriakopoulos. “A Model Predictive Control Approach for Vision-Based Object Grasping via Mobile Manipulator”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* 2018, pp. 1–6. DOI: 10.1109/IROS.2018.8593759.

[134] Takashi Yamamoto, Tamaki Nishino, Hideki Kajima, Mitsunori Ohta, and Koichi Ikeda. “Human Support Robot (HSR)”. In: *ACM SIGGRAPH 2018 Emerging Technologies.* SIGGRAPH ’18. New York, NY, USA: Association for Computing Machinery, 2018. ISBN: 9781450358101. DOI: 10.1145/3214907.3233972. URL: https://doi.org/10.1145/3214907.3233972.

[135] Fei Chen, Mario Selvaggio, and Darwin G Caldwell. “Dexterous Grasping by Manipulability Selection for Mobile Manipulator With Visual Guidance”. In: *IEEE Transactions on Industrial Informatics* 15.2 (2019), pp. 1202–1210. DOI: 10.1109/TII.2018.2879426.

[136] Sachin Chitta, E Gil Jones, Matei Ciocarlie, and Kaijen Hsiao. “Perception, planning, and execution for mobile manipulation in unstructured environments”. In: *IEEE Robotics and Automation Magazine, Special Issue on Mobile Manipulation* 19.2 (2012), pp. 58–71.

[137] Radu Bogdan Rusu, Andreas Holzbach, Rosen Diankov, Gary Bradski, and Michael Beetz. “Perception for mobile manipulation and grasping using active stereo”. In: *2009 9th IEEE-RAS International Conference on Humanoid Robots.* 2009, pp. 632–638. DOI: 10.1109/ICHR.2009.5379597.

[138] Michael Hegedus, Kamal Gupta, and Mehran Mehrandezh. “Towards an Integrated Autonomous Data-Driven Grasping System with a Mobile Manipulator”. In: *2019 International Conference on Robotics and Automation (ICRA).* 2019, pp. 1601–1607. DOI: 10.1109/ICRA.2019.8793759.

[139] Ulrich Klank, Dejan Pangercic, Radu Bogdan Rusu, and Michael Beetz. “Real-time CAD model matching for mobile manipulation and grasping”. In: *2009 9th IEEE-RAS International Conference on Humanoid Robots.* 2009, pp. 290–296. DOI: 10.1109/ICHR.2009.5379561.

[140] Petr Štibinger, George Broughton, Filip Majer, Zdeněk Rozsypálek, Anthony Wang, Kshitij Jindal, Alex Zhou, Dinesh Thakur, Giuseppe Loianno, Tomáš Krajník, and Martin Saska. "Mobile Manipulator for Autonomous Localization, Grasping and Precise Placement of Construction Material in a Semi-Structured Environment". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 2595–2602. DOI: `10.1109/LRA.2021.3061377`.

[141] Tim Welschehold, Christian Dornhege, and Wolfram Burgard. "Learning mobile manipulation actions from human demonstrations". In: *2017 IEEE / RSJ International Conference on Intelligent Robots and Systems (IROS).* 2017, pp. 3196–3201. DOI: `10.1109/IROS.2017.8206152`.

[142] Zhijun Li, Ting Zhao, Fei Chen, Yingbai Hu, Chun-Yi Su, and Toshio Fukuda. "Reinforcement Learning of Manipulation and Grasping Using Dynamical Movement Primitives for a Humanoidlike Mobile Manipulator". In: *IEEE/ASME Transactions on Mechatronics* 23.1 (2018), pp. 121–131. DOI: `10.1109/TMECH.2017.2717461`.

[143] Benno Staub, Ajay Kumar Tanwani, Jeffrey Mahler, Michel Breyer, Michael Laskey, Yutaka Takaoka, Max Bajracharya, Roland Siegwart, and Ken Goldberg. "Dex-Net MM: Deep Grasping for Surface Decluttering with a Low-Precision Mobile Manipulator". In: *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE).* 2019, pp. 1373–1379. DOI: `10.1109/COASE.2019.8842901`.

[144] Walter Wohlkinger, Aitor Aldoma, Radu B Rusu, and Markus Vincze. "3DNet: Large-scale object class recognition from CAD models". In: *2012 IEEE International Conference on Robotics and Automation.* 2012, pp. 5384–5391. DOI: `10.1109/ICRA.2012.6225116`.

[145] Lucia Biagetti, Amrit Kochar, Cristina Cristalli, and Simonetta Boria. "Cognitive Grasping System: A Grasping Solution for Industrial Robotic Manipulation using Convolutional Neural Network". In: *Procedia Manufacturing* 51 (Apr. 2020), pp. 32–37. DOI: `10.1016/j.promfg.2020.10.006`.

[146] Alexandre Gariepy, Jean Christophe Ruel, Brahim Chaib-Draa, and Philippe Giguere. "GQ-STN: Optimizing One-Shot Grasp Detection based on Ro-

bustness Classifier”. In: *IEEE International Conference on Intelligent Robots and Systems* (Nov. 2019), pp. 3996–4003. ISSN: 21530866. DOI: 10.1109/IROS40897.2019.8967785.

[147] Ping Jiang, Yoshiyuki Ishihara, Nobukatsu Sugiyama, Junji Oaki, Seiji Tokura, Atsushi Sugahara, and Akihito Ogawa. “Depth Image–Based Deep Learning of Grasp Planning for Textureless Planar-Faced Objects in Vision-Guided Robotic Bin-Picking”. In: *Sensors* 20.3 (2020). ISSN: 1424-8220. DOI: 10.3390/s20030706. URL: https://www.mdpi.com/1424-8220/20/3/706.

[148] PickNik Robotics. *Moveit2 Tutorial - Deep Grasps.* 2023. URL: https://moveit.picknik.ai/humble/doc/examples/moveit_deep_grasps/moveit_deep_grasps_tutorial.html.

[149] Ken Goldberg, Brian V. Mirtich, Yan Zhuang, John Craig, Brian R. Carlisle, and John Canny. “Part pose statistics: Estimators and experiments”. In: *IEEE Transactions on Robotics and Automation* 15.5 (1999), pp. 849–857. ISSN: 1042296X. DOI: 10.1109/70.795790.

[150] Connor Shorten and Taghi M Khoshgoftaar. “A survey on image data augmentation for deep learning”. In: *Journal of Big Data* 6.1 (2019), pp. 1–48.

[151] Andreas ten Pas, Colin Keil, and Robert Platt. “Efficient and Accurate Candidate Generation for Grasp Pose Detection in SE(3)”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* IEEE, Sept. 2021. DOI: 10.1109/iros51168.2021.9636215.

[152] Andreas Eitel, Jost Tobias Springenberg, Luciano Spinello, Martin Riedmiller, and Wolfram Burgard. “Multimodal deep learning for robust RGB-D object recognition”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* 2015, pp. 681–687. DOI: 10.1109/IROS.2015.7353446.

[153] James J. Kuffner. “Effective sampling and distance metrics for 3D rigid body path planning”. In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004.* Vol. 4. 2004, pp. 3993–3998. DOI: 10.1109/ROBOT.2004.1308895.

[154] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J. Davison. "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM". In: *Proceedings - IEEE International Conference on Robotics and Automation* (Sept. 2014), pp. 1524–1531. ISSN: 10504729. DOI: `10.1109/ICRA.2014.6907054`.

[155] Jonathan T Barron and Jitendra Malik. "Intrinsic Scene Properties from a Single RGB-D Image". In: *CVPR* (2013).

[156] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. "ACRONYM: A Large-Scale Grasp Dataset Based on Simulation". In: *2021 IEEE Int. Conf. on Robotics and Automation, ICRA*. 2020.

[157] Zhenjia Xu, Beichun Qi, Shubham Agrawal, and Shuran Song. "Adagrasp: Learning an adaptive gripper-aware grasping policy". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 4620–4626.

[158] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. "A Billion Ways to Grasp: An Evaluation of Grasp Sampling Schemes on a Dense, Physics-based Grasp Data Set". In: *Proceedings of the International Symposium on Robotics Research ({ISRR})*. Vol. abs/1912.05604. Hanoi, Vietnam, 2019. URL: `http://arxiv.org/abs/1912.05604`.

[159] Abhinav Gupta, Adithyavairavan Murali, Dhiraj Prakashchand Gandhi, and Lerrel Pinto. "Robot Learning in Homes: Improving Generalization and Reducing Dataset Bias". In: *Advances in Neural Information Processing Systems*. Ed. by S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett. Vol. 31. Curran Associates, Inc., 2018. URL: `https://proceedings.neurips.cc/paper/2018/file/febefe1cc5c87748ea02036dbe9e3d67-Paper.pdf`.

[160] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. *ShapeNet: An Information-Rich 3D Model Repository*. Tech. rep. arXiv:1512.03012 [cs.GR]. Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[161] Khaled Mamou, E Lengyel, and A Peters. “Volumetric hierarchical approximate convex decomposition”. In: *Game Engine Gems 3*. AK Peters, 2016, pp. 141–158.

[162] The Blender Foundation. *Blender - a 3D modelling and rendering package*. 2023.

[163] Joseph Redmon and Ali Farhadi. “YOLO9000: Better, Faster, Stronger”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 7263–7271.

[164] Yaoxian Song, Yuejiao Fei, Chun Cheng, Xiangqing Li, and Changbin Yu. “UG-Net for Robotic Grasping using Only Depth Image”. In: *2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. 2019, pp. 913–918. DOI: 10.1109/RCAR47638.2019.9044116.

[165] Ioan A Şucan, Mark Moll, and Lydia E Kavraki. “The Open Motion Planning Library”. In: *IEEE Robotics & Automation Magazine* 19.4 (Dec. 2012), pp. 72–82. DOI: 10.1109/MRA.2012.2205651.

[166] Malte Splietker and Sven Behnke. “Directional TSDF: Modeling Surface Orientation for Coherent Meshes”. In: *IEEE International Conference on Intelligent Robots and Systems* (Nov. 2019), pp. 1727–1734. ISSN: 21530866. DOI: 10.1109/IROS40897.2019.8968264.

[167] Bingjie Tang, Matthew Corsaro, George Konidaris, Stefanos Nikolaidis, and Stefanie Tellex. “Learning Collaborative Pushing and Grasping Policies in Dense Clutter”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 6177–6184. DOI: 10.1109/ICRA48506.2021.9561828.

[168] Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. “A survey on instance segmentation: state of the art”. In: *International Journal of Multimedia Information Retrieval* 9.3 (Sept. 2020), pp. 171–189. ISSN: 2192662X. DOI: 10.1007/S13735-020-00195-X/TABLES/3. URL: https://link.springer.com/article/10.1007/s13735-020-00195-x.

[169] Xibai Lou, Yang Yang, and Changhyun Choi. “Collision-Aware Target-Driven Object Grasping in Constrained Environments”. In: *Proceedings - IEEE International Conference on Robotics and Automation* 2021-May (2021), pp. 6364–6370. ISSN: 10504729. DOI: `10.1109/ICRA48506.2021.9561473`.

[170] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *Advances in Neural Information Processing Systems* 28 (2015). URL: `https://github.com/`.

[171] Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. “Microsoft COCO: Common objects in context”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8693 LNCS.PART 5 (2014), pp. 740–755. ISSN: 16113349. DOI: `10.1007/978-3-319-10602-1{\_}48/COVER`. URL: `https://link.springer.com/chapter/10.1007/978-3-319-10602-1_48`.

[172] Nikolaus Correll, Kostas E. Bekris, Dmitry Berenson, Oliver Brock, Albert Causo, Kris Hauser, Kei Okada, Alberto Rodriguez, Joseph M. Romano, and Peter R. Wurman. “Analysis and observations from the first Amazon picking challenge”. In: *IEEE Transactions on Automation Science and Engineering* 15.1 (Jan. 2018), pp. 172–188. ISSN: 15583783. DOI: `10.1109/TASE.2016.2600527`.

[173] Andrej Orsula, Simon Bøgh, Miguel Olivares-Mendez, and Carol Martinez. “Learning to Grasp on the Moon from 3D Octree Observations with Deep Reinforcement Learning”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 4112–4119. ISBN: 2153-0866. DOI: `10.1109/IROS47612.2022.9981661`.

[174] Daniel Duberg and Patric Jensfelt. “UFOMap: An Efficient Probabilistic 3D Mapping Framework That Embraces the Unknown”. In: *IEEE Robotics and Automation Letters* 5.4 (Oct. 2020), pp. 6411–6418. ISSN: 23773766. DOI: `10.1109/LRA.2020.3013861`.

[175] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. “A benchmark for the evaluation of RGB-D SLAM systems”. In: *IEEE International Conference on Intelligent Robots and Systems* (2012), pp. 573–580. ISSN: 21530858. DOI: `10.1109/IROS.2012.6385773`.

[176] Yu-Wei Chao, Wei Yang, Yu Xiang, Pavlo Molchanov, Ankur Handa, Jonathan Tremblay, Yashraj S Narang, Karl Van Wyk, Umar Iqbal, Stan Birchfield, Jan Kautz, and Dieter Fox. “DexYCB: A Benchmark for Capturing Hand Grasping of Objects”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* June 2021, pp. 9044–9053.

[177] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. “Recent advances in convolutional neural networks”. In: *Pattern Recognition* 77 (May 2018), pp. 354–377. ISSN: 0031-3203. DOI: `10.1016/J.PATCOG.2017.10.013`.

[178] Mladan Jovanović. “Generative Artificial Intelligence: Trends and Prospects”. In: *IEEE Computer Society* (2022).

[179] Gemini Team. “Gemini: A Family of Highly Capable Multimodal Models”. 2023.

[180] OpenAI. “GPT-4 Technical Report”. In: (Mar. 2023). URL: `https://arxiv.org/abs/2303.08774v3`.

[181] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. “PaLM-E: An Embodied Multimodal Language Model”. In: (Mar. 2023). ISSN: 26403498. URL: `https://arxiv.org/abs/2303.03378v1`.

[182] Michael Ahn et al. “Do As I Can, Not As I Say: Grounding Language in Robotic Affordances”. In: *Proceedings of Machine Learning Research* 205 (Apr. 2022), pp. 287–318. ISSN: 26403498. URL: `https://arxiv.org/abs/2204.01691v2`.

[183] Anthony Brohan et al. “RT-1: Robotics Transformer for Real-World Control at Scale”. In: (Dec. 2022). DOI: `10.15607/rss.2023.xix.025`. URL: `https://arxiv.org/abs/2212.06817v2`.