POLITECNICO DI TORINO

Master's Degree in Mechatronic Engineering

Master's Degree Thesis

Deep Reinforcement Learning for Vision-Based Robotic Grasping

Supervisor:
Prof. Stefano MAURO
Prof. Lino MARQUES

Candidate:
Alessia BISSACCO

Co-supervisor:
Ph.D. Laura SALAMINA
Ph.D. Matteo MELCHIORRE
Dr. Sedat DOGRU

Academic year 2024 - 2025

Abstract

Robotic grasping represents a major competency for the safe and effective handling of objects in unstructured settings. Stability in grasps, especially when considering objects whose shape or properties are irregular or non-uniform, still presents a challenge deriving principally from uncertainties inherent to contact geometry and friction properties and mass distribution. This thesis presents learning methods for the enhancement of the robustness of grasps through explicit consideration of the influence of the center of mass (CoM) on stable manipulation.

I present a new system that integrates RGB-D perception, pose generation for grasps, and reinforcement learning to forecast the CoM and control stable grasp execution. Our approach integrates two candidate grasp sampling techniques—a major-axis method for single-axis objects and grid sampling procedure for multi-axis objects—with a tabular Q-learning algorithm that learns stable grasp settings by self-exploration in an iterative fashion. Unlike previous approaches highly reliant on tactile force sensors or CNNs, our approach parameters grasp instabilities explicitly through on-board vision and leverages lightweight learning to correct CoM estimation and grasp policies.

The system was evaluated and validated using a 6-DoF xArm6 robotic arm equipped with a parallel-jaw gripper and onboard Intel RealSense cameras. A custom 3D-printed object set with changing mass distributions was used to verify performance for different CoM conditions. Experimental and simulation results demonstrate that the proposed method can effectively approximate the CoM for single-axis objects, recognize stable grasp pose even when CoM estimation is questionable, and increase the rate of successes in repeated grasps. For multi-axis objects, the method provides stable predictions along the dominant axis while maintaining tenable grasps for more challenging configurations. In short, this thesis provides a practical and interpretable

learning-centered framework for grasp stability that only requires minimal tactile sensing, blends well with standard robotic perception and planning systems, and highlights the capacity of reinforcement learning to enhance robustness in ordinary robotic manipulation tasks.

Contents

List of Tables

List of Figures

List of Abbreviations

AI Artificial Intelligence

AoI Area of Interest

BVH Bounding Volume Hierarchy

CAD Computer-Aided Design

CNN Convolutional Neural Network

CoM Center of Mass

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DQN Deep Q-Network

DoF Degrees of Freedom

EGNet Efficient Grasp Network

FCL Flexible Collision Library

FPFH Fast Point Feature Histogram

GelSight A High-resolution tactile sensing technology based on gel deformation

ICP Iterative Closest Point

IK Inverse Kinematics

ISR-UC Institute of Systems and Robotics – University of Coimbra

KD-Tree K-dimensional Tree (k-dimensional tree)

MDP Markov Decision Process

MLP Multilayer Perceptron

NeRF Neural Radiance Fields

OBB Oriented Bounding Box

PCA Principal Component Analysis

PCL Point Cloud Library

RANSAC Random Sample Consensus

RGB Red Green Blue

RGB-D Red Green Blue – Depth (color + depth sensing)

RL Reinforcement Learning

ROI Region of Interest

ROS Robot Operating System

SAC Soft Actor-Critic

SARSA State-Action-Reward-State-Action

SDF Signed Distance Field

STL Stereolithography (file format)

SVD Singular Value Decomposition

tf2 ROS Transform Library (tf2)

xArm6 UFactory xArm6 robotic manipulator

Chapter 1

Introduction

1.1 Context and Motivation

Grasping is an essential capability in robotics, because it provides object-interaction and object-manipulation in a safe and efficient manner [1]. Without reliable grasping approaches, operations such as assembly, packaging, service robotics, and healthcare delivery would experience significant constraints. The capability of grasping an object firmly and reliably throughout manipulation, known as grasp stability, is of particular relevance for irregularly shaped or delicate objects, wherein the implication of slippage or dropping would lead to the failure of the operation, potential damage, or hazards for safety.

The importance of understanding grasp has been highlighted in various application areas. In factories, robots handle components of varying geometries and weights, including objects ranging from irregularly shaped metallic components, such as wrenches, to delicate items such as glassware [2]. In home and service robotics, manipulators should handle a vast set of everyday objects, such as bottles, food packs, and tools, and for such items, flexibility and safety are called for [3]. In logistics and warehouse robotics, manipulators have to pick up containers like boxes and bags and irregular packages, and for such tasks, success greatly depends on reliable and foreseeable contact [4]. In packing scenarios, visual and tactile sensing have demonstrated the capability to support stable object placement and have thereby highlighted the key role of grasp stability to improve efficiency for repeated tasks [5]. The problems of stability do not arise from the object's geometry alone but also from

irregular mass distributions. When the center of mass of an object is shifted from the grasp location, it generates torques that can give rise to rotation or slipping and thereby lead to the failure of the task. The estimation of the parameters of inertia for pushing tasks has been proposed as a solution to address this issue [6, 7, 8]. Recent literature has looked at the explicit estimation of the center of mass (CoM) at or after the period of grasping, aiming at bringing the grasping location near to the actual CoM and reducing unwanted dynamics [9, 10, 11]. Grasping movements made near the CoM are expected to have improved stability, bacause the chance of motion after the grasp is lowered. On the other hand grasps made at a distance from the CoM might remain stable provided the grasping force is sufficient but at the cost of generating excessive manipulator torque [12].

To attain these objectives, various sensing modalities have been explored. Vision-based systems together with tactile sensors offer a system for measurement of contact and stability performance [13]. As an example of pure tactile sensing, GelSight sensors provide high-resolution tactile feedback by documenting deformations on a flexible material and enable robots to deduce surface textures and detect slippage [14, 15, 16, 17]. Other systems utilize taxel force sensors extending across gripper fingers and have the capability of identifying instability along one to three axes [13]. External RGB-D cameras have also been utilized to perceive modifications of an object's pose due to pushing, supplementing in-hand sensing with global visual knowledge [8]. Inspired by these recent works in the literature, this thesis aims to contribute to the field by proposing a solution to improve robust grasping. More specifically, the work is focused on the prediction of the CoM of an object that is grasped by using in-hand sensing and reinforcement learning. By perceiving instabilities of the grasp with an RGB-D camera that is mounted on the manipulator and using a tabular Q-learning model [18, 19], the goal is to achieve steady grasps on new objects, including those of irregular shape and uncertain mass distribution. This rationale provides the foundation for the objectives defined in the next section.

1.2 Objectives

The central objective of this thesis is to create an experimental environment for efficient robotic grasping, and more specifically, for the stability of grasps and the

evaluation of the center of mass (CoM) of unseen objects throughout the act of manipulating them. Since traditional simulation environments provide valuable insights, they do not quite provide the depth of real interactions of a manipulator and object. Therefore, the work described here is carried out on real hardware, using the xArm6 robotic manipulator, to ensure a practical and reliable evaluation of grasping methodologies.

The thesis looks to solve the key problems of irregular object shapes, inhomogeneous mass distributions, and a lack of force feedback of commercial grippers. The considerations render grasp stability a problem that is not trivial, as the robot needs to sense and respond to potential instabilities like slippage or undesired rotation. By centering on in-grasp estimation of the CoM, we seek to enhance pick-and-place success and robustness and thereby enable safer and more efficient manipulation in dynamic scenes.

Regarding the objectives, this thesis aims to achieve the following particular goals:

- **Problem formulation and requirements definition.** We clearly articulate the scope of the problem of robotic grasping stability, outlining challenges arising from object irregularities, lack of force feedthrough, and manipulator dynamic motion. We establish requirements for a solution to be experimentally verifiable and reliable.

- **Design of the experimental pipeline.** We create and execute an inclusive pipeline that incorporates sensing, perception, and control components. The pipeline is created with the goal of allowing for experiment reproducibility, precise acquisition of data, and systematic analysis of grasp stability for varying conditions.

- **3D perception and object localization.** We observe an environment with a dense point cloud from an RGB-D camera attached to the xArm6'send-effector. We use these observations to estimate the object position and orientation in space and give the robot enough information to calculate candidate grasp points.

- **Detection of grasp instabilities.** We create an approach to perceive and examine the movement of the object following a grasp and recognize indications of instability like slips or rotations. By applying 3D sensing, the amount of

these instabilities is quantified more reliably than in established simulation models.

- **Center of mass estimation through learning.** We apply an iterative process to provide an approximation of the object's CoM from the detection of instabilities. The process is regulated by a tabular Q-learning approach, allowing the robot to enhance its choice of grasp with experience and converge on more stable solutions.

- **Real-time evaluation of grasp stability.** The success of the presented methodology is certified by the real-time observation of the object being grasped throughout lifting and manipulation processes. Stability is assessed in real-time, thus ensuring that the system is able to provide swift feedback on the strength of the grasp being selected.

In attaining these goals, the thesis makes a contribution to the wider robotic manipulation community by demonstrating a feasible and empirically informed solution to robust grasping. The presented solution not only proves the feasibility of in-grasp center of mass estimation from RGB-D sensing and reinforcement learning but also lays the foundation for future queries on autonomous manipulation by considering real-world constraints.

1.3 Contributions

This thesis contributes scientifically and practically to robotic manipulation, paying special attention to grasp stability and in-grasp estimation of the center of mass. The paper proposes an experimental setup consisting of a combination of 3D perception by means of technology based on RGB-D, detection of grasp instabilities, and reinforcement learning by means of tabular Q-learning by which a robot is empowered to iteractively and online estimate the center of mass of unseen objects. In contrast to many of the state methodologies, which rely on simulation, such approach is directly implemented and validated on real hardware, the xArm6 robotic manipulator, and thus provides for a more realistic and reliable evaluation of grasping policies. The entirety of the work was carried out in the framework of an Erasmus exchange

program at the Institute of Systems and Robotics – University of Coimbra (ISR-UC) in Portugal. The stay at ISR-UC offered the best possible academic conditions, access to experimental facilities of the latest technology level, and the opportunity for interaction with researchers on robotic grasping and reinforcement learning without which conceptualization and evaluation of the proposed methodology would not have been feasible.

Presentation of results is a worthwhile addition to the thesis. The approach developed and results reached from experiments have been presented in a conference paper, allowing dissemination and verification of the work among the international scientific community. The act ensures that results generated by the thesis transcend geographical boundaries and adds to prevailing discourse within the field.

Last but not least, the thesis lays the groundwork for future projects of research. The framework offered herein is meant to be adjustable, and it will permit subsequent examination of more sophisticated manipulation problems, such as those with deformable objects, multiple modalities of sensing, or sophisticated reinforcement learning methodologies. For these reasons, the thesis not merely fulfills its near-term tasks but offers groundwork for future developments that shall be conducted in subsequent projects, of which a few examples shall be described in the continuation of the present work.

1.4 Document structure

The remaining chapters of the thesis follow on from each other for a period of six chapters that facilitate a systematic presentation of problem, methodology and results.

- **Chapter 2** – Related Work. Here, the literature on robotic grasp now, on grasping using learning-based approaches, and grasp stability is critically considered. The chapter provides background for the thesis by outlining the merits and limitations of previously done work and locating the present effort in this body of scholarly work.

- **Chapter 3** – Theoretical Background. The chapter provides the theoretical background of the applied methods throughout the thesis. The chapter mentions object detection by aligning point clouds (ICP and SCIA) as well

as reinforcement learning and Q-learning, Principal Component Analysis for orientation determination and other applied methodologies. The chapter's objective is to reveal the theoretical knowledge needed in practice for applying these methodologies.

- **Chapter 4** – Proposed Method. This chapter describes the methodology developed to address the problem of in-grasp center of mass estimation and stability evaluation. A block diagram of the full workflow is presented, and each component of the pipeline is introduced in detail, including grasp pose generation, Q-learning formulation, stability measurement, and system implementation in ROS.

- **Chapter 5** – Experimental Work. This chapter goes into detail on the experimental setup and the assessments made. It outlines the results obtained from the applied proposed methodology in real-world instances using the xArm6 manipulator, aided by graphical illustrations, pictorial presentations, and detailed analysis of results.

- **Conclusion** – The concluding chapter provides a summary of the thesis's significant findings, reflection on the significant contributions and insights obtained throughout the research. Also, it describes possible future plans of research, including short-term enhancements and future developments of the approach.

With such an organization, the paper proceeds on a sequential logical pattern from background and literature survey, theoretical developments, conceptual exposition of methodology, experimental validation, and concluding perspectives.

Chapter 2

Related Work

2.1 Robotic Manipulation

Robotic manipulation constitutes the major area of research that has witnessed increasing prominence in the past two decades as the promise grows for robots to function in less structured, more dynamic, and more representative environments than the ones observed in the typical human-like tasks. Unlike classical industrial robots whose manipulation tasks are predominantly repetitive and specially designed for well-defined objects along assembly lines, today's work in manipulation aims at giving robots the ability to cope with uncertainties in object shape, in material properties, in position, and in dynamics. This necessitates merging the perception and the planning and control methodologies in such a manner that the robots not only perform pre-determined tasks but also decide the stability and usability of the grasps online.

The challenges faced are varied: objects can undergo deformation or displacement while the object is being grasped, occlusions often limit the viewable visual data, and tactile feedback is often sporadic and unreliable. Such challenges have motivated researchers to explore a very broad range of solutions involving different kinds of manipulators, sensory methods, and computation techniques. One common theme is the collaborative integration of visual and tactile modalities: while cameras provide rich information regarding object shape and position, tactile sensors provide localized, high-fidelity information about contact regimes and forces.

A prime example of multimodal fusion comes in the form of Liang et al. [5] who

employ tactile sensors in the form of a 4×4 array of taxels that are each able to measure three-dimensional force vectors. By integrating the tactile data with the visual data, the authors create an attention-guided deep neural network that picks out prospective affordances and feasible grasp points. The attention function plays a key role in discarding irrelevant or redundant data and so focusing on the most promising aspect-related features that are relevant to grasping. Once candidate grasp locations are detected, they are passed to a module for motion planning formulated on the basis of reinforcement learning that yields feasible manipulator trajectories. The system demonstrates how perception and planning can be tightly coupled such that the robot can adapt and change the plan in real time depending on the sensory feedback received.

By comparison, Feng et al. [13] take a geometry-driven perspective. Their approach begins by delineating object boundaries from a point cloud, which are then passed through an antipodal grasp sampler. The resulting grasp poses are not applied blindly; instead, they are further tested through multimodal sensing: tactile sensing is received through a 4×4 sensor array, and force and torque at the end effector and grasp force analysis are used. The signals are fed through a neural network that considers the grasp robustness. This multi-layered approach—geometry first and then tactile verification—highlights the importance of synthesizing both global and local information to decrease failure events that can be commonplace when solely using methods based on geometry.

A more sophisticated multimodal system is introduced by Calandra et al. [16], who extensively employ tactile sensing via GelSight sensors. This GelSight technology is capable of acquiring high-resolution images of the contact surface, thereby effectively transforming tactile feedback into an image-processing challenge. In their research, two GelSight sensors feed data into convolutional neural networks (CNNs) that share weights, while a supplementary RGB camera provides visual data processed through an independent ResNet-based CNN. The outputs from these three components are integrated within a multilayer perceptron (MLP) to predict the probability of grasp success. Importantly, this probability serves not only to assess potential grasps but also to iteratively inform the selection of subsequent actions, establishing a feedback loop between perception and decision-making. This study exemplifies the increasing dependence on high-dimensional sensing and deep learning architectures to capture

nuanced details of object interactions.

Kolamuri et al. [17] offer the opposite view by not only using GelSight tactile sensors to verify stability in grasp but also to predict the kinematic attributes of objects. Through marker displacement observation in the tactile views, they estimate rotation angles and derive the center of rotation through a least-squares circle fitting procedure. Note that in the architecture proposed, the RGB-D camera works in the background and helps in the estimation of the length of the major axis of the object through the application of singular value decomposition (SVD). This goes to show how the function of vision and touch sensing may alternate depending on the specific function: sometimes vision predominates and other times tactile sensing yields dominant information.

Another branch of exploration involves evaluating the physical properties of objects that are relevant to manipulation. Dutta et al. [7] present a cross-modal Gaussian Process regression system developed to predict the mass of objects. It incorporates pushing interactions—whose forces tell us about the object's resistance and inertia—into visual information obtained through a camera outside the object. Superquadratic fitting of the point cloud yields a preliminary mass estimate that is then improved through a double filtering process. By placing greater focus on intrinsic object properties than on the grasping geometry alone, this work takes the study of manipulation one step closer to a more fundamentally rooted understanding of physics.

Wang et al. [11] focus on the computation of the center of mass (CoM) while working on the assumption that object models are completely known. They use a vision-based tactile sensor and present two complementary approaches. The first uses Bayesian fusion and probabilistically combines available data to predict the CoM. The second uses a foundation in reinforcement learning and the Soft Actor-Critic (SAC) algorithm specifically to utilize a policy network and two Q-networks to improve CoM predictions. It highlights the dialogue between probabilistic modeling and learning policies and offers them as two powerful and separate approaches to addressing manipulation uncertainty.

Continuing on the foundation of vision-centric methodologies, Liu et al. introduce RGBGrasp [20]}, a new grasp planning framework that obtains multiple RGB viewpoints of an object as a robotic arm maneuvers. Instead of relying on a single, fixed

perspective, the arm itself serves as an exploration agent to gather multiple viewpoints. The multi-perspective images then get processed through Neural Radiance Fields (NeRF), which synthesize a photorealistic volumetric representation of the object. This representation is then used to obtain high-quality grasp poses. The key strength of RGBGrasp lies in the fact that it has the better ability to handle occlusions and complex geometries more efficiently than single-view or static approaches. By bridging traditional vision-based grasps and newer neural scene representations through the deployment of NeRFs, the system promises a future scenario in which robots autonomically survey the scene to improve perception before they perform a grasp. Taken together as a whole, these studies provide an aggregate picture of the state-of-the-art in research in robotic manipulation. Some approach the problem through geometric modeling (Feng et al.), while others privilege tactile sensing and whole-body contact data (Calandra et al., Kolamuri et al.), and still other research explores physical property estimation (Dutta et al., Wang et al.). More recent developments, such as RGBGrasp, show the shift in the last few years to neural scene reconstruction and active perception, in which the robot uses its own actions to enhance the information it gathers. Throughout this spectrum, a prominent theme emerges: effective manipulation necessitates multimodal integration. No individual sensor modality is adequate on its own. Vision supplies contextual information but faces challenges related to occlusions; tactile sensing delivers local precision yet is deficient in global awareness; force and torque measurements effectively capture dynamic interactions but are constrained by resolution limitations. Through the amalgamation of these informational sources—typically within learning-based or probabilistic frameworks—robots are progressively enhancing their ability to navigate the variability and uncertainty present in real-world environments. The increasing use of multimodal perception offers a basis for the future major trend in robotic manipulation: the deployment of learning-oriented methodologies. Unlike the classical approaches depending on analytical models and geometric rules, learning enables robots to extrapolate knowledge from experience, accommodate new objects, and progressively improve their grasping techniques. The various applications of machine learning, including deep networks and reinforcement learning, that have been employed to enhance the evolution of robotic manipulation are examined in the following section.

2.2 Learning in Robotic Manipulation

Learning methods have become the key to the process of grasp planning and robotic grasping and provide increased robustness and generalization when compared to classical analytical or geometry-dominated techniques. As opposed to the pre-defined models or geometric heuristics used by conventional techniques, learning allows robots to capitalize on experience and interaction for the development of strategies, coping with uncertainties in object attributes, and the gradual adjustment of acts. Diverse areas of research demonstrate how learning can mitigate the difficulties inherent in grasping.

A major field of study involves supervised learning for the identification of grasps from vision data. In this context, RGB or RGB-D inputs are processed by deep nets to estimate grasp pose directly or establish grasp rectangles. Yu et al. [4] introduce EGNet, a lightweight and powerful network optimized for real-time robotic grasp identification, while Fang et al. [3] push the state-of-the-art beyond that by proposing AnyGrasp, which provides stronger robustness in the presence of different viewpoints and changing circumstances. Mahler et al. [2] further confirm the effectiveness of intensive data-driven learning, whereby synthetic datasets are used to derive ambidextrous grasping policies that are then successfully transferred to physical robots, highlighting the importance of dataset size, domain randomization, and simulation-to-reality transfer.

Another major research focus leverages multimodal learning by integrating visual and tactile information to anticipate the outcomes of grasps. Calandra et al. [16] integrate high-resolution tactile feedback recorded by the use of GelSight sensors and visual feedback by the use of convolutional networks and a multilayer perceptron to provide a probabilistic estimation of grasp success that incrementally informs action choice. These methods highlight the paramount nature of cross-modal learning to handle uncertainties and occlusions at the point of contact.

Reinforcement learning (RL) has progressively become the central focus of studies, especially in situations where robots must proactively explore and improve their behavior in the long run. Wang et al. [11] employ a vision-improved tactile sensor and the Soft Actor–Critic (SAC) algorithm to optimize regrasping policies considering the center of mass (CoM) and demonstrate how RL assists stability through repeated

interactions. Laying the foundation by extending previous work [18, 19], RL enables the development of policies successfully managing the exploration–exploitation trade-off while adapting to imprecise physical properties. Going beyond the scope of static worlds, Naik et al. [21] provide a pre-active layered RL architecture optimized for mobile manipulation settings, in which the robot learns proper base pose and pre-grasp pose settings despite the ambiguity in object locations. It promotes the layered exploration for better sample efficiency and manages to demonstrate the knowledge transfer from simulation settings to practice cases and integrates the navigation and manipulation tasks in a comprehensive manner.

Recent work also explores richer scene representations and physics-informed models. Liu et al. [20] introduce RGBGrasp, a system that records various RGB viewpoints while the robot arm moves and recreates the object using Neural Radiance Fields (NeRF), reaching volumetric representations enabling accurate grasp pose estimation despite occlusion. By contrast, Dutta et al. [7, 8] employ cross-modal Gaussian Process regression and differentiable filtering methods to predict object mass and other physical parameters extracted from interactions by pushing and enrich geometric and visual grasp programming through physics-informed inference. Overall, these studies demonstrate the broad range of learning in robotic manipulation, spanning supervised perception-centered detection and wide-reaching data-intensive strategies, through to multimodal visuo-tactile integration, reinforcement learning specialized for regrasping and mobility-enabled manipulation, neural scene construction, and physics-constrained estimation. Through each of these different paradigms, a unifying theme becomes clear: learning provides a unified foundation for the construction of sensing, planning, and control such that robots may move beyond rigid models and increasingly improve grasping effectiveness in dynamic and unpredictable worlds. Yet the effectiveness overall by any grasp strategy continues to depend on the ability to achieve stable contact following the securing of the object. As such, grasp stability has become a central focus across studies, exploring how robots can assess, predict, and guarantee stability during manipulation. The following section gives overview to key methodologies that focus particularly upon grasp stability, differentiating in particular between analytical formulations and learning-based approaches.

2.3 Grasp Stability

Stability grasp forms a key component of robotic manipulation since it determines if an object is held firmly while undergoing the lifting and eventual handling processes. The challenges arise from the uncertainties in contact geometry, surface frictional properties, and mass distribution; hence, even if a grasp seems feasible at the beginning, unintended torques or slippage may lead to failure during the process of manipulation. Stability has since then been studied through various perspectives that include analytical modeling of force closure, data-observed evaluation of the properties of the object, and learning-based forecast for the likelihood of grasp success.

One key focus area is the estimation of physical properties relevant to stability, including the center of mass (CoM) and friction. Zhao et al. [9] offer a direct computation of the CoM and the friction coefficient for unknown objects and show the potential for this data to guide grasp selection to reduce the likelihood of unwanted motion. Similarly, Kanoulas et al. [10] focus on CoM estimation through the combination of 3D range data and wrist-accelerometer or comparable force and torque sensing and adjust grasp pose to lower the torques required after picking up. Dutta et al. [7, 8] extend the research path by using visuo-tactile pushing interactions and Gaussian Process regression and differentiable filtering to infer mass and related parameters that directly affect stability by influencing the grip forces required.

Another notable research direction embeds CoM reasoning explicitly while carrying out grasp planning. Feng et al. [13] proposed a robust planner for unfamiliar objects that integrates tactile and visual information to judge the robustness of grasp in CoM position. Wang et al. [11] cast CoM-based regrasping as a challenge in the realm of reinforcement learning and train policies optimizing object repositioning more reliably through the Soft Actor–Critic algorithm once the dynamics are known. Wang et al. [12] extend the research by highlighting the contribution of online estimation of CoM to failure recovery such that the robot adjusts the grasp in real time through update estimates and reduces the chances of object slippage or drop.

The stability of the grasp has been examined through the explicit evaluation of slip and rotational movements using tactile feedback systems. Kolamuri et al. [17] employ GelSight tactile sensors to observe marker displacements at the point of contact and estimate the rotation and center of rotation by applying circle fitting methods to

augment stability evaluations. Calandra et al. [16] also combine tactile and visual information in the context of a multimodal learning system and predict grasp success probabilities serving as stability indicators. Such techniques represent the effectiveness of high-resolution tactile sensing and sensor fusion among learning-based techniques to capture pre-slip signals and thereby cause robots to react before the loss of a grasp. Taken together, these studies demonstrate a progression in development ranging from analytical assessment of physical quantities to learning-based multimodal approaches that integrate perception and control. Unlike previous studies in which open-loop selection of statically stable grasp configurations had been the focus, more recent methods draw heavily on closed-loop methods that update stability predictions in real-time continuously, exploit multimodal sensory feedback, and employ reinforcement learning to adjust grasp policies incrementally. As such, the work on grasp stability is moving towards unified frameworks in which perception, learning, and control work together to achieve reliable manipulation in unpredictable and dynamic environments.

Chapter 3

Theoretical Background

3.1 Preprocessing

Robotic grasping perception depends on unprocessed sensor input, usually obtained from RGB-D cameras or 3D scanners, which requires preprocessing prior to the proper functioning of higher-level modules like recognition, grip creation, or reinforcement learning. Preprocessing guarantees that the unrefined point cloud is organized, denoised, and sufficiently compact to facilitate dependable geometric reasoning while maintaining the critical characteristics of the object and its surrounding environment. This section presents the most pertinent preprocessing techniques employed in the proposed framework.

3.1.1 Segmentation

Segmentation is the process of dividing the point cloud $P = \{p_i \in \mathbb{R}^3\}_{i=1}^{N}$ into significant subsets, including backdrop, support surfaces, and distinct objects. In robotic grasping scenarios, it is crucial to distinguish the focal object from its surroundings.

A prevalent method involves plane segmentation via RANSAC, which detects the primary support surface (e.g., a table) and eliminates it from the point cloud. Given a plane model

$$ax + by + cz + d = 0,$$

RANSAC iteratively selects minimal sets of points, estimates (a, b, c, d) via least

squares, and maximizes the number of inliers

$$I = \{p_i \mid |ax_i + by_i + cz_i + d| < \tau\},$$

where τ is a distance threshold. Subsequent clustering methodologies are employed on the residual points to extract the object of interest. Precise segmentation diminishes the search field for grip candidates and eliminates extraneous features caused by backdrop clutter.

3.1.2 Voxelization

Voxelization converts the continuous point cloud into a uniform grid of cubic units, known as *voxels*. The 3D space is discretized into cells of side length v, and each point $p_i = (x_i, y_i, z_i)$ is mapped to voxel indices

$$(i, j, k) = \left(\left\lfloor \frac{x_i}{v} \right\rfloor, \left\lfloor \frac{y_i}{v} \right\rfloor, \left\lfloor \frac{z_i}{v} \right\rfloor \right).$$

Points falling into the same voxel are aggregated, typically represented by their centroid:

$$\hat{p}_{ijk} = \frac{1}{|V_{ijk}|} \sum_{p \in V_{ijk}} p,$$

where V_{ijk} is the set of points inside voxel (i, j, k).
Downsampling via voxelization decreases computing complexity while preserving geometric fidelity. An appropriately selected voxel size maintains the object's form for recognition and grasp planning while markedly decreasing the quantity of points to be processed.

3.1.3 Clustering

Clustering techniques categorize points into cohesive sets based on spatial closeness. In Euclidean clustering, two points p_i, p_j belong to the same cluster if

$$\|p_i - p_j\|_2 < \epsilon,$$

where ϵ is a neighborhood threshold. This method is commonly employed to isolate specific items in multi-object environments.
More advanced approaches include density-based clustering such as DBSCAN, which defines clusters as sets of points with at least *MinPts* neighbors inside a radius ϵ. Formally, a point p is a *core point* if

$$|\{q \in P \mid \|p - q\| < \epsilon\}| \geq \text{MinPts}.$$

Clusters are then built by iteratively expanding from core points. Such methods are advantageous when objects are in close proximity or when the point cloud exhibits irregular noise patterns.
Clustering establishes a definitive correlation between raw sensor data and distinct physical entities.

3.1.4 Additional Filtering

Preprocessing pipelines often include additional filtering steps to improve data quality:

- **Statistical outlier removal**: each point p_i is removed if its mean distance to the k nearest neighbors exceeds

  $$\mu + \alpha\sigma,$$

  where μ and σ are the global mean and standard deviation, and α is a threshold factor.

- **Pass-through filtering**, which restricts the cloud to a predefined region of interest (ROI), e.g.

  $$x_{\min} \leq x \leq x_{\max}, \quad y_{\min} \leq y \leq y_{\max}, \quad z_{\min} \leq z \leq z_{\max}.$$

- **Smoothing**, which reduces sensor noise via moving average or MLS (Moving Least Squares) fitting, preserving sharp geometric edges important for grasping.

3.2 Object Detection

3.2.1 Iterative Closest Point

The Iterative Closest Point (ICP) algorithm is the most widely applied algorithm in the field of rigid registration of 3D point clouds. Given a source cloud P_s and a target cloud P_t, the ICP algorithm seeks the rigid transformation

$$T = \{R, t\}, \quad R \in SO(3),\ t \in \mathbb{R}^3,$$

that minimizes the alignment error:

$$E(R, t) = \sum_{i=1}^{N} \|Rp_i^s + t - p_{\pi(i)}^t\|^2,$$

where $\pi(i)$ is the index of the nearest neighbor in P_t for p_i^s.
The algorithm iterates as follows: 1. Establish correspondences via nearest-neighbor search. 2. Estimate (R, t) that minimizes $E(R, t)$ (commonly via SVD of the covariance matrix). 3. Apply the transformation to the source cloud. 4. Repeat until convergence:

$$\Delta E < \epsilon \quad \text{or} \quad k > k_{\max}.$$

A primary issue with ICP is its sensitivity to initialization: the algorithm converges to the nearest local minimum. However, with a good initial guess, ICP achieves high accuracy, making it a standard tool in fine registration for robotics, computer vision, and 3D modeling.

3.2.2 Sample Consensus Initial Alignment

Sample Consensus Initial Alignment (SAC-IA) is a feature-based registration algorithm used to achieve coarse globally consistent alignment of two point clouds, even under wide initial misalignments.
Each point p_i is described by a local descriptor such as the Fast Point Feature Histogram (FPFH):

$$\text{FPFH}(p_i) = \{h_1, h_2, \ldots, h_m\},$$

where h_j encodes angular relationships of normals within a support radius r.
Given descriptors for source and target clouds, SAC-IA establishes candidate correspondences by nearest-neighbor matching in descriptor space. A RANSAC scheme then iteratively: 1. Samples minimal subsets of correspondences. 2. Estimates candidate rigid transformations $T = (R, t)$. 3. Evaluates the number of inliers:

$$I(T) = |\{(p_i^s, p_j^t) \mid \|Rp_i^s + t - p_j^t\| < \delta\}|.$$

The transformation with the largest support is selected.
The resulting alignment is coarse but globally consistent. It is commonly refined using ICP, yielding a hybrid pipeline:

$$T^* = \arg\min_{R,t} E(R, t), \quad \text{initialized with SAC-IA.}$$

This hybrid approach combines the robustness of feature-based global registration with the accuracy of local iterative refinement, and is widely employed in object recognition and pose estimation under noisy or incomplete sensing conditions.

3.3 Q-Learning

Q-learning is one of the most fundamental and widely studied algorithms in the field of Reinforcement Learning (RL). It belongs to the class of *model-free*, *off-policy*, and *temporal-difference* methods, which enable an agent to learn from direct interaction with its environment without requiring an explicit model of the underlying dynamics. The central idea is that, by repeatedly experiencing states, actions, and rewards, an agent can incrementally build an estimate of the long-term value of actions, ultimately converging towards an optimal decision-making strategy.
Formally, the environment is modeled as a Markov Decision Process (MDP), which is defined by:

- a finite or continuous set of states S,
- a finite or continuous set of actions A,

- a transition probability function $P(s'|s,a)$, which describes the likelihood of moving to state s' given that action a is taken in state s,
- a reward function $R(s,a)$, which assigns immediate feedback after executing action a in state s,
- and a discount factor $\gamma \in [0,1]$, which weights the importance of future rewards compared to immediate ones.

The goal of the agent is to learn a policy $\pi : S \to A$ that maximizes the expected cumulative discounted reward, also called the *return*:

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}.$$

3.3.1 Action-Value Function

At the core of Q-learning lies the *action-value function*, or Q-function, which evaluates the quality of performing a given action in a given state:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\Big[G_t \,\Big|\, S_t = s, A_t = a\Big].$$

This function represents the expected return when starting from state s, taking action a, and then following policy π. The optimal action-value function is defined as

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a),$$

and satisfies the Bellman optimality condition:

$$Q^*(s,a) = \mathbb{E}\Big[R_{t+1} + \gamma \max_{a'} Q^*(S_{t+1}, a') \,\Big|\, S_t = s, A_t = a\Big].$$

3.3.2 Learning Rule

Q-learning estimates $Q^*(s,a)$ directly through experience, without requiring the transition model $P(s'|s,a)$. When the agent observes a transition (s,a,r,s'), the

update rule is:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\Big[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\Big],$$

where $\alpha \in (0,1]$ is the learning rate. The term

$$\delta_t = r + \gamma \max_{a'} Q(s',a') - Q(s,a)$$

is called the *temporal-difference (TD) error*, and measures the discrepancy between the current estimate and a bootstrap target based on the next state. By iteratively minimizing this error, the Q-values converge to the optimal action-value function.

3.3.3 Exploration and Exploitation

An important feature of Q-learning, and RL in general, is the balance between exploration and exploitation. The agent must sometimes choose random actions to explore new parts of the state space, while other times it should exploit its current knowledge by selecting the action with the highest estimated Q-value. The most common strategy is the ϵ-greedy policy:

$$\pi(a|s) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|A|}, & \text{if } a = \arg\max_{a'} Q(s,a') \\ \frac{\epsilon}{|A|}, & \text{otherwise,} \end{cases}$$

where $\epsilon \in [0,1]$ controls the probability of taking a random exploratory action. Usually, ϵ is decreased over time, allowing the agent to begin with broad exploration and later exploit the learned policy more consistently.

3.3.4 Convergence and Theoretical Guarantees

One of the most appealing properties of Q-learning is its convergence guarantee. If every state-action pair is visited infinitely often, and if the learning rate α decays according to the Robbins–Monro conditions

$$\sum_{t=1}^{\infty} \alpha_t = \infty, \qquad \sum_{t=1}^{\infty} \alpha_t^2 < \infty,$$

then the Q-values are guaranteed to converge to the optimal values $Q^*(s, a)$. As a consequence, the policy

$$\pi^*(s) = \arg\max_a Q(s, a)$$

is guaranteed to be optimal.

3.3.5 Variants and Extensions

Q-learning serves as the foundation for many extensions:

- **SARSA** (on-policy), which uses the action actually taken in the next state to update Q-values.
- **Double Q-learning**, which maintains two estimators to reduce the overestimation bias caused by the max operator.
- **Deep Q-Networks (DQNs)**, which employ neural networks to approximate the Q-function, allowing Q-learning to be applied in high-dimensional or continuous state spaces.

3.3.6 Practical Considerations

Despite its theoretical elegance, Q-learning faces practical challenges:

- **State-action space size**: the Q-table grows exponentially with the dimensionality of states and actions, making it impractical for real-world problems without function approximation.
- **Exploration inefficiency**: random exploration strategies may fail in environments with sparse rewards.
- **Overestimation bias**: the use of the max operator in the update rule can systematically overestimate action values, motivating the use of Double Q-learning.

3.4 Principal Component Analysis

Principal Component Analysis (PCA) is a classical statistical technique widely employed in computer vision, robotics, and data analysis. Its primary objective is to reduce the dimensionality of high-dimensional data while preserving the most informative directions of variance. In the context of 3D perception, PCA provides a concise representation of a point cloud by inferring its dominant geometric axes, which can be used for alignment, orientation estimation, and grasp synthesis.
Consider a point cloud

$$P = \{p_i \in \mathbb{R}^3 \mid i = 1, \ldots, N\},$$

representing the sampled surface of an object. The first step in PCA is to compute the centroid:

$$\bar{p} = \frac{1}{N} \sum_{i=1}^{N} p_i,$$

which defines the average location of all points in the cloud. Next, we calculate the covariance matrix

$$C = \frac{1}{N} \sum_{i=1}^{N} (p_i - \bar{p})(p_i - \bar{p})^T,$$

which captures how the coordinates of the point set vary with respect to one another. The eigenvalue problem

$$Cv_j = \lambda_j v_j, \quad j = 1, 2, 3,$$

yields three orthogonal eigenvectors v_j, each associated with an eigenvalue λ_j. The eigenvectors represent the *principal axes* of the point cloud, while the eigenvalues quantify the variance along each axis. In practice:

- The eigenvector corresponding to the largest eigenvalue indicates the **principal axis**, i.e., the direction in which the point cloud is most elongated.
- The eigenvector corresponding to the smallest eigenvalue identifies the axis of minimal spread, which is often aligned with the object's surface normal.
- The three eigenvectors together define an orthogonal local coordinate frame aligned with the object geometry.

Geometrically, PCA can be interpreted as fitting an ellipsoid around the data points, where the semi-axes of the ellipsoid are proportional to $\sqrt{\lambda_j}$. This provides an intuitive visualization of the object's spatial distribution.
In robotic perception and grasping, PCA offers several advantages:

1. **Robust alignment:** The principal axes can be used to initialize object alignment before applying more refined registration methods (e.g., ICP).

2. **Shape-aware grasping:** By aligning the gripper with the principal axis or the surface normal derived from PCA, the robot can generate more stable and natural grasp candidates.

3. **Dimensionality reduction:** PCA allows encoding high-resolution point clouds into a compact representation that retains their essential structure.

Overall, PCA serves as both a mathematical tool for variance analysis and a practical method for extracting shape descriptors from raw 3D data, making it a cornerstone in object recognition and robotic manipulation.

3.5 Collision Detection

Collision detection is a fundamental component of robotic manipulation, path planning, and real-time control. Its purpose is to determine whether two or more geometric entities (such as robot links, objects, or obstacles in the workspace) intersect or approach each other within a safety margin. Reliable collision detection is indispensable, as undetected collisions may cause damage to the manipulator, surrounding equipment, or the manipulated object.

3.5.1 Role in Robotic Manipulation

In grasping and manipulation tasks, collision detection ensures that:

- The manipulator does not intersect with itself (*self-collision*).

- Planned trajectories avoid obstacles in the environment (*environment collision*).

- The end-effector approaches the object smoothly, making contact only at the intended grasp region.

Collision detection is therefore integrated into several stages of a robotic pipeline:

- **Grasp candidate validation**, to discard infeasible or unsafe grasp poses.
- **Trajectory planning**, to ensure that the motion path avoids obstacles.
- **Online monitoring**, to prevent unforeseen collisions during execution.

3.5.2 Geometric Representations

Collision detection requires an abstract representation of the geometry of both the robot and the environment. Different levels of fidelity are used depending on the computational budget:

- **Primitive shapes:** bounding boxes, spheres, or cylinders that approximate objects with simple analytic volumes. These allow fast collision queries but may lack precision.
- **Meshes:** polygonal meshes derived from CAD models or sensor reconstructions. These offer high accuracy but are computationally heavier, especially for complex geometries.
- **Voxel grids and Signed Distance Fields (SDFs):** volumetric representations where each grid cell encodes occupancy or distance to the nearest surface. SDFs enable constant-time queries for penetration depth and are widely used in optimization-based planning.

The trade-off between accuracy and efficiency is central: high-fidelity models improve realism, while lightweight approximations enable real-time performance.

3.5.3 Detection Algorithms

Collision detection algorithms are commonly classified into:

- **Discrete collision detection:** checks for intersections at sampled robot configurations. This is effective for validating nodes in sampling-based motion planning (e.g., RRT, PRM).
- **Continuous collision detection:** interpolates motions between configurations and checks for collisions along the entire trajectory. Although computationally more demanding, this avoids missed collisions that occur with coarse discretization.

Efficient implementations rely on hierarchical spatial data structures:

- **Bounding Volume Hierarchies (BVH):** objects are recursively enclosed in simple bounding volumes (spheres, AABBs, OBBs). During collision checks, entire branches of the hierarchy can be pruned if bounding volumes do not overlap.
- **k-d trees and Octrees:** partition the space recursively, enabling fast nearest-neighbor and intersection queries in high-dimensional or sparse environments.

3.5.4 Collision Detection in MoveIt

In the proposed framework, collision detection is handled by the *MoveIt* motion planning library, which integrates the Flexible Collision Library (FCL). Objects recognized in the environment are added to the *planning scene* as collision objects, allowing the system to perform real-time feasibility checks on potential grasps and trajectories.
MoveIt automatically conducts both:

- **Self-collision checks**, verifying that robot links do not interfere with each other.
- **Environment collision checks**, ensuring that the manipulator avoids obstacles in the workspace.

By tightly coupling collision detection with inverse kinematics (IK) and trajectory generation, MoveIt guarantees that only collision-free, feasible plans are executed.

This integration not only ensures safety but also improves the reliability of grasp execution in cluttered or dynamic environments.
In summary, collision detection combines geometric abstractions, efficient data structures, and real-time algorithms to ensure safe and feasible robotic interaction with the environment. Its integration into robotic planning frameworks such as MoveIt makes it a cornerstone of modern autonomous manipulation systems.

Chapter 4

Methodology

The block diagrams in figure 4.1 show a summary of the method:

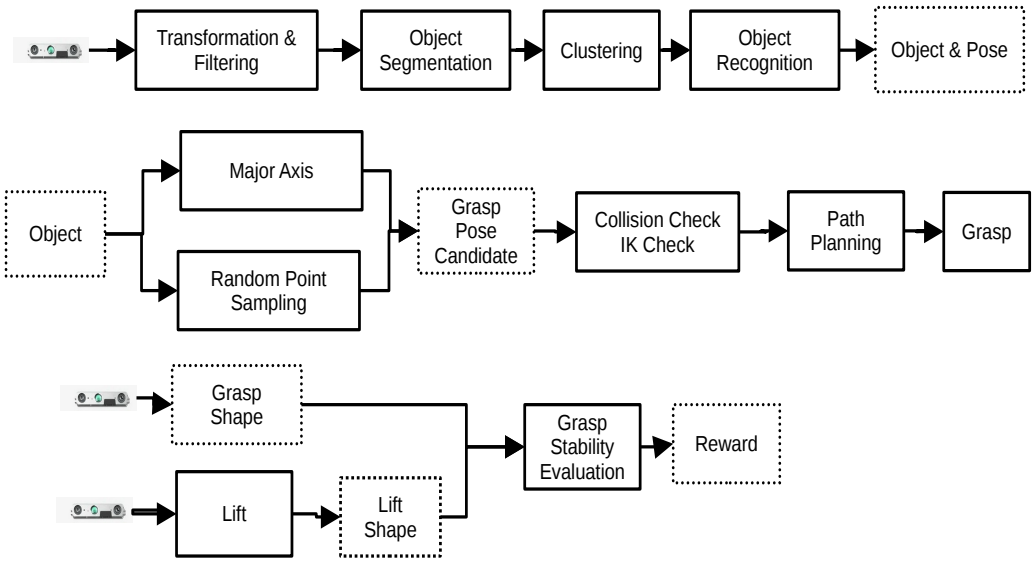

Figure 4.1: Pipeline.

The system is structured as a sequence of interconnected pipelines. First, the object detection and localization pipeline processes the RGB-D point cloud to identify and estimate the pose of the target object. Next, the grasp pose candidate generation pipeline proposes potential grasp configurations using two complementary strategies. These candidates are then evaluated by the motion planning pipeline, which verifies their feasibility and computes a collision-free trajectory to execute the grasp and

subsequent lift. Once the object is lifted, the grasp stability pipeline analyzes variations in the object's pose to assess the quality of the grasp. Finally, a learning block integrates feedback from all pipelines to progressively refine and identify the most effective grasping strategy.
First, the RGBD camera's 3D point cloud is changed to the robot's base reference frame. Then, it is filtered to get rid of areas that are outside the Area of Interest (AoI). After that, object segmentation is done, which gets rid of the point cloud that corresponds to the table (a plane). This leaves point clouds that match the objects in the AoI.
Next, the points are grouped together using Euclidean distance. Then, each group of points is compared to a pre-loaded STL model database to find out what the object is and finish its pose. There is also a voxelization step that comes after this to cut down on the number of points that need to be processed. There are three steps to object matching. To start, the object-oriented bounding box of the RGBD pointcloud and the corresponding extents are estimated. Then, these are compared to the objects stored in the database to cut down on the number of possible matches. The second step uses Iterative Closest Point (ICP) to compare the object pointcloud to the candidates from the first step. In the third stage, the candidates who passed the second stage are compared to the object point cloud using the Sample Consensus Initial Alignment method and their Fast Point Feature Histograms (FPFH).

4.0.1 Object Detection and Classification

Introduction to the Problem

One of the most critical challenges in robotic manipulation lies in the handling of *unknown objects*, namely objects for which no prior geometric information is available, such as CAD models or STL files. In such cases, the robot does not possess a unique and stable three-dimensional representation of the object, and therefore both grasp planning and the evaluation of the outcome of a manipulation become highly problematic.
The initial idea developed in this work was to confront this issue directly by designing a system able to operate with unknown objects. In order to achieve this, the first solution that was explored consisted of acquiring and processing *point clouds*

generated by an RGB-D camera. By working with point clouds, the aim was to build a representation of the objects in real time and to make this representation usable for grasp planning and collision checking.

Initial Approach: Multi-view Point Clouds

The initial strategy was based on the generation of partial point clouds of the unknown object and their comparison with previously stored data. More specifically, the process consisted of the following steps:

1. A partial point cloud of the object was generated through the RGB-D camera.
2. The camera or the robot holding the camera was moved around the object to capture the scene from multiple viewpoints.
3. For each viewpoint, the corresponding 3D representation was saved in the form of an image of the point cloud. Each of these images was placed into a dedicated folder, one per object.
4. The resulting dataset of views was stored inside the `meshes` directory, where each folder corresponded to a distinct object.

During code execution, the system attempted to match the point cloud currently acquired from the camera with one of the previously saved views. Once a correspondence was found, the idea was to reconstruct a triangular mesh of the object and insert this mesh into the collision scene used by the robot for planning. In this way, the robot could in principle identify the object and interact with it without requiring a pre-defined CAD model.

Issues Encountered

Although the approach was conceptually interesting, several problems were observed in practice. These issues concerned both the geometry of the reconstruction and the consistency of the information used for grasp planning:

- **Lack of a stable reference frame for the mesh.** Grasp points needed to be expressed in a coordinate system coherent with the object mesh. However,

the meshes reconstructed from point clouds did not possess a well-defined or stable reference frame. This meant that grasp points were not consistent across different acquisitions of the same object.

- **Inconsistency across different executions.** If in a subsequent run the point cloud captured by the camera was matched with a different saved view, the system generated a new mesh. Consequently, the previously stored grasp data could not be reused, since they were associated with a different mesh representation, even if the physical object was the same.

- **Unstable matching process.** The alignment between the incoming point cloud and the saved views was not robust. Variations in lighting, sensor noise, and partial occlusions of the object often resulted in mismatches or false associations.

These limitations significantly reduced the usability of the method. The fact that grasp points could not be transferred consistently between different runs of the system represented a critical bottleneck for its integration into a complete manipulation pipeline.

Full 3D Reconstruction of Objects for Detection and Classification

In order to address these issues, a second approach was developed, which aimed at reconstructing a complete 3D model of each object rather than relying on individual views. Two programs corresponding to two main steps were created to implement this pipeline:

1. **Sequential acquisition.** This program enabled the capture of partial point clouds of the object from multiple viewpoints. The object was rotated, and at each step a 3D "photo (snapshot)" was taken, generating a collection of partial representations covering the whole surface of the object (Fig. 4.2).

Figure 4.2: A test object (a hammer), and its corresponding point cloud..

2. **Merging and reconstruction.** The partial point clouds obtained from different views were subsequently aligned and merged. The result was a complete 3D point cloud of the object. From this consolidated point cloud, an STL mesh was then be generated, providing a full geometric model of the object (Fig. 4.3).

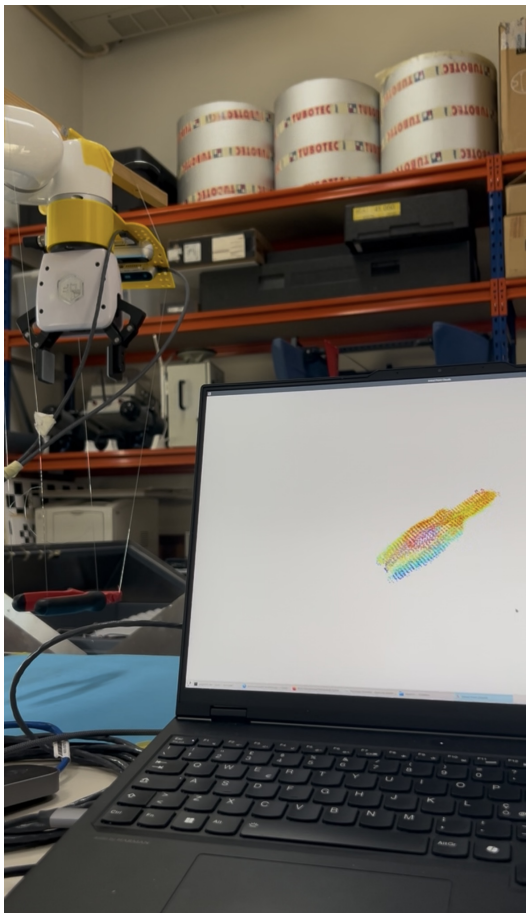

Figure 4.3: A test object (pliers), and its corresponding point cloud.

This method offered the advantage of producing a unique and stable mesh of the object. Once this mesh was available, grasp points could be stored directly in its reference frame, ensuring consistency across multiple runs. Furthermore, the mesh could be reused in later sessions, allowing the robot to reason about the same object with a persistent geometric model.

Practical Limitations of 3D Reconstruction

Despite the improvements in conceptual robustness, the full 3D reconstruction approach presented several practical limitations that made its use less appealing in the context of this work:

- **Time-consuming process.** To obtain a sufficiently accurate model, many

views of the object had to be acquired. The acquisition, alignment, and merging of these views required a significant amount of time, which was not compatible with the requirements of fast and iterative robot training.

- **High computational complexity.** The process of aligning multiple noisy point clouds demanded high computational resources. The success of the alignment was not always guaranteed, especially in the presence of reflective or textureless surfaces, which are particularly challenging for depth cameras.

- **Limited scalability.** For each new object, the full process of multi-view acquisition and reconstruction had to be repeated. This manual procedure did not scale to a large set of objects, which is often needed in data collection and training scenarios for reinforcement learning.

The combination of these drawbacks made it evident that, although promising, full 3D reconstruction was not an efficient solution for the specific requirements of this thesis. The approach was too slow and computationally heavy, and it would have limited the number of objects and experiments that could realistically be carried out.

Final Choice: Designed and 3D-Printed Objects

Due to difficulties faced in the above object detection and classification approaches, a more pragmatic solution was adopted, which consisted of designing and fabricating artificial objects through 3D printing. This decision represented a compromise between realism and experimental feasibility, and it offered several clear advantages:

- **Complete control over object geometry.** Since the objects were designed in CAD, a unique and stable STL mesh was available from the beginning. This eliminated the uncertainty and instability of reconstruction from sensor data.

- **Configurable physical properties.** The objects were designed in such a way that their center of mass could be modified, for example by inserting internal weights. This allowed the creation of multiple physical configurations of the same geometry, increasing the variability available for robot training.

- **Reproducibility and scalability.** With 3D printing, identical copies of the same object could be easily fabricated. This ensured consistent experimental

conditions and made it possible to scale up the number of objects without repeating complex acquisition procedures.

This approach shifted the focus from solving the challenging problem of unknown-object reconstruction to creating a controlled set of experimental conditions. By using printed objects with variable mass distributions (Fig. 4.4), it was possible to provide the robot with a rich variety of grasping scenarios while maintaining full control over the geometry and reproducibility of the experiments. In this way, the complexity of reconstruction was avoided, and the research could concentrate on the actual problem of robotic grasp learning.

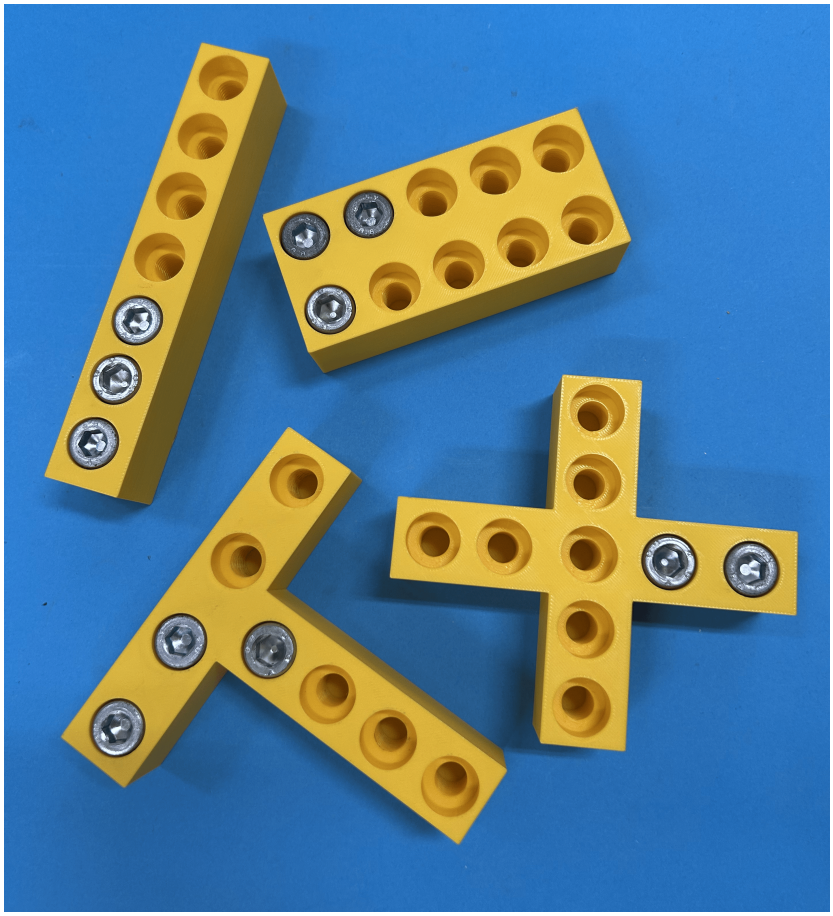

Figure 4.4: Objects.

4.1 Grasp Pose Candidate Generation

4.1.1 Major-axis

The major-axis method represents one of the earliest geometry-driven approaches to grasp planning and relies on the assumption that many everyday objects used in robotic manipulation can be characterized by a single dominant direction, or major axis, along which feasible grasp regions are distributed. The procedure starts once the perception system has localized the object and extracted its geometric description from the STL model. Using the STL rather than the raw sensor point cloud avoids problems due to noise, occlusions, or missing points, providing a watertight reference mesh. This mesh is inserted into the planning environment as a collision body so that subsequent motion planning considers it consistently with the rest of the scene. The next step is to compute the object's major axis, which can be derived by applying Principal Component Analysis (PCA) to the point distribution. Given the object centroid $\bar{\mathbf{p}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{p}_i$ and the covariance matrix

$$\mathbf{C} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{p}_i - \bar{\mathbf{p}})(\mathbf{p}_i - \bar{\mathbf{p}})^\top,$$

the eigenvectors of $\mathbf{C}$ define the principal directions of the object, and the eigenvector corresponding to the largest eigenvalue identifies the major axis $\hat{\mathbf{v}}$. In practice, this direction coincides with the longest edge of the object's oriented bounding box (OBB). Once computed, the axis is aligned with the pose of the object in the world frame, anchored at the centroid, and extended to the two ends of the OBB. Along this axis, a discrete set of candidate points is generated by applying scalar offsets δ_i from the centroid in both positive and negative directions, producing

$$\mathbf{b}_i = \bar{\mathbf{p}} + \delta_i\,\hat{\mathbf{v}},$$

where each $\mathbf{b}_i$ represents a potential grasp location aligned with the gripper centerline. To account for the three-dimensional nature of the approach, these points are complemented by vertical displacements along the world z-axis. Specifically, a second family of points is created by adding offsets Δz to the candidate positions, resulting

in

$$\mathbf{b}_i^S = \mathbf{b}_i + \Delta z\, \hat{\mathbf{z}},$$

where $\hat{\mathbf{z}}$ is the unit vector along the vertical direction. These elevated points do not correspond to direct grasp locations but rather to safe approach positions: the robot first moves to $\mathbf{b}_i^S$ to ensure clearance from surrounding obstacles and then descends to $\mathbf{b}_i$ to execute the grasp. In this way, the method not only generates feasible grasp poses but also incorporates approach trajectories in a straightforward manner. The major-axis method is therefore computationally efficient, easy to implement, and robust to noise due to its reliance on STL geometry rather than raw sensor data. Its main limitation is that it assumes the existence of a single symmetry axis, which is suitable for elongated and simple objects such as bottles, rods, or boxes but becomes less effective for complex geometries with multiple potential axes or irregular shapes. For this reason, while it remains a valuable baseline strategy, it is often complemented by more flexible sampling schemes, such as grid-based methods, to achieve robustness across a wider variety of objects.

4.1.2 Grid-based

The grid-based method was developed to address the limitations of the major-axis approach, which assumes that useful grasp points lie along a single direction. Many objects, however, do not conform to this simple geometry: tools such as drills, hammers, or other irregularly shaped items may exhibit several possible grasp regions distributed across their surface. In such cases, restricting the sampling process to one axis reduces the variety of hypotheses and risks missing stable grasp locations. The grid-based method avoids this limitation by constructing a regular two-dimensional grid over the object's footprint in its own reference frame and generating candidates from cells that intersect the object mesh. In this way, the method distributes grasp poses more evenly across the object, while retaining a small amount of stochasticity to prevent overly regular patterns. After the perception system provides the high-resolution pose of the object, including its STL mesh and centroid, all vertices are expressed in the object frame and used to compute the oriented bounding box (OBB).

Given the set of vertices $\{v_i = (v_i^x, v_i^y, v_i^z)\}_{i=1}^N$, the footprint bounds are obtained as

$$x_{\min} = \min_i v_i^x, \quad x_{\max} = \max_i v_i^x, \qquad y_{\min} = \min_i v_i^y, \quad y_{\max} = \max_i v_i^y,$$

thus defining the rectangle $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$ which encloses the projection of the mesh onto the xy-plane. This rectangle is then divided into a regular grid, where the cell sizes along the x and y directions are chosen as

$$\Delta_x = \max\left(\frac{x_{\max} - x_{\min}}{n_x^\star}, \Delta_{\min}\right), \qquad \Delta_y = \max\left(\frac{y_{\max} - y_{\min}}{n_y^\star}, \Delta_{\min}\right),$$

with $n_x^\star, n_y^\star$ being the desired number of cells per side and $\Delta_{\min}$ a minimum cell size (about 2.5 cm). The number of cells along each axis is then

$$n_x = \left\lceil \frac{x_{\max} - x_{\min}}{\Delta_x} \right\rceil, \qquad n_y = \left\lceil \frac{y_{\max} - y_{\min}}{\Delta_y} \right\rceil,$$

so that the grid automatically adapts to the object's footprint while never falling below the prescribed resolution. A candidate point is generated inside a cell by sampling around the cell center with bounded randomness. For a cell (i, j), the nominal center is

$$c_{ij}^x = x_{\min} + \left(i + \tfrac{1}{2}\right)\Delta_x, \qquad c_{ij}^y = y_{\min} + \left(j + \tfrac{1}{2}\right)\Delta_y,$$

and the actual sample is

$$\tilde{x} = c_{ij}^x + \epsilon_x, \qquad \tilde{y} = c_{ij}^y + \epsilon_y,$$

where $\epsilon_x \sim \mathcal{U}\left(-\frac{\kappa}{2}\Delta_x, \frac{\kappa}{2}\Delta_x\right)$ and $\epsilon_y \sim \mathcal{U}\left(-\frac{\kappa}{2}\Delta_y, \frac{\kappa}{2}\Delta_y\right)$ with $\kappa = 0.8$ acting as an offset factor. A sample is retained only if the corresponding cell intersects the mesh projection, ensuring that points are always associated with real object regions rather than empty space. The sampled coordinates are then mapped back to the world frame using the object pose, and two vertical placements are defined: a high *safe* pose, used as an approach position above the object, and a lower *grasp* pose at the centroid height plus a small offset. Formally, if $(\tilde{x}_W, \tilde{y}_W)$ is the world-frame projection

of the sample, the positions are

$$p^S = \begin{bmatrix} \tilde{x}_W \\ \tilde{y}_W \\ z_{\text{safe}} \end{bmatrix}, \qquad p^G = \begin{bmatrix} \tilde{x}_W \\ \tilde{y}_W \\ \bar{p}_z + h_G \end{bmatrix},$$

where $\bar{p}_z$ is the centroid height, h_G is the grasp offset, and z_{safe} is a clearance height large enough to avoid collisions. Each candidate is further associated with a small set of yaw orientations, typically aligned with the object's yaw or rotated by $\pi/2$, which increases directional diversity without exploding the number of hypotheses. Overall, this method complements the major-axis approach by offering broader coverage of the object geometry, controlled randomness to probe regions near edges and irregular features, and independence from the object's orientation in the world. Its trade-off is that on simple elongated objects it may generate redundant hypotheses, but in practice the two methods are complementary: axis-based sampling is efficient for slender symmetric shapes, while grid-based sampling is preferable for irregular or multi-symmetric geometries. As cell sizes decrease, the method converges towards continuous surface sampling, with practical limits determined only by computational resources and the subsequent validation stage.

4.1.3 Pose Validation

After candidate grasp points are generated, each of them must be validated before being considered for execution. For every sampled location, two poses are always defined: a *SAFE* pose, placed at a clearance height above the object to guarantee a collision-free approach, and a *GRASP* pose, located close to the object surface at the centroid height plus a fixed offset, representing the actual grasp execution. The gripper orientation is derived directly from the estimated object yaw: in the major-axis based method, the end-effector is aligned with the object's principal axis by setting the roll–pitch–yaw angles to $(\pi, 0, \pi + yaw)$, so that the gripper closes perpendicularly to the main direction of the object. In the grid-based method, the same alignment is tested with two yaw rotations (nominal and $+\pi/2$), providing alternative configurations that increase the chance of finding a feasible IK solution when the object presents wider or irregular surfaces.

Formally, let a candidate pose be defined by its position $\mathbf{p} = (x, y, z)$ and orientation $\mathbf{q}$ in the world frame. The pose is considered valid if the following condition holds:

$$\exists\, \mathbf{q}_{IK} \text{ s.t. } \begin{cases} \mathrm{IK}(\mathbf{p}, \mathbf{q}) = \mathbf{q}_{IK} & \text{(reachable configuration)} \\ \|\mathbf{p} - \mathbf{p}_{base}\| \leq d_{\max} & \text{(within maximum reach)} \\ \mathbf{q}_{IK} \text{ is collision-free in the planning scene} & \text{(no collision)} \end{cases}$$

where $\mathbf{p}_{base}$ is the robot base position and $d_{\max}$ is the maximum allowed reach computed at initialization. If any of these conditions fails, the pose is rejected. In practice, the function `isPoseValid` first transforms the candidate into the base frame, checks the distance constraint, queries MoveIt's `/compute_ik` service up to three times for a valid solution $\mathbf{q}_{IK}$, and finally verifies that the resulting configuration does not intersect with any collision object in the planning scene. Both SAFE and GRASP poses must satisfy these conditions, otherwise the candidate is marked as invalid.

The difference between the two sampling strategies emerges clearly in this stage. In the major-axis based approach, candidate points are constrained along a one-dimensional direction, and the orientation is fixed with respect to the axis, which makes the validation faster but also more restrictive. In contrast, the grid-based approach distributes candidates over a two-dimensional footprint and tests multiple yaw orientations per point, which increases the computational load but improves robustness on irregular objects. This means that major-axis candidates are typically fewer but have a higher chance of being valid on elongated shapes, whereas grid-based candidates are more diverse but require stricter filtering through the validation process.

Each candidate is stored with a boolean flag `is_valid`. Valid pairs of SAFE/GRASP poses are published to the planning pipeline for execution, while invalid ones are also logged and visualized for analysis. During execution, only valid poses are attempted, whereas invalid ones are penalized in the learning-based modules so that the system progressively avoids infeasible configurations. In RViz, the whole validation process is made transparent by publishing markers for the bounding box, principal axis, centroid, object mesh, and candidate points, with color coding distinguishing valid from invalid

locations. This provides immediate feedback to the operator and facilitates both debugging and the evaluation of grasping experiments.

4.2 Q-Learning

In this work, Q-learning is used as a light-weight reinforcement learning (RL) layer to bias grasp selection toward regions that empirically yield stable lifts. Q-learning is a model-free, off-policy temporal-difference method that estimates the optimal action–value function $Q^*(s, a)$, i.e., the expected discounted return when taking action a in state s and thereafter following the optimal policy [18, 19]. Importantly, RL is applied *only* to grasp choice; perception, grasp generation (major-axis or grid-based), inverse kinematics (IK), and collision checking are handled by deterministic modules. This separation keeps the learning problem small and interpretable while leveraging reliable geometric reasoning for feasibility.
Formally, after each grasp attempt, the Q-function is updated with the temporal-difference rule

$$Q(s, a) \leftarrow (1 - \alpha)\, Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') \right], \tag{4.1}$$

with learning rate $\alpha = 0.05$ and discount factor $\gamma = 0.95$ (as in the implementation). The immediate reward r encodes grasp stability (Section 4.3), and s' is the successor state. Action selection (*exploration–exploitation*) may be ϵ-greedy; in our stack, the Q-nodes compute and persist Q and logs, while the grasp executor handles selection.

State–action representations

We support two complementary parameterizations matching the two grasp generators.

Major-axis (object-aligned, compact).

$$s = (\text{object\_id}, \text{yaw\_bin}, \text{offset\_bin}),$$

where the yaw is discretized in 45° steps and the offset along the major axis in 5 mm increments:

$$\text{yaw\_bin} = \left\lfloor \tfrac{180}{45\pi}\,\text{yaw} \right\rfloor \cdot 45, \qquad \text{offset\_bin} = \left\lfloor \tfrac{\text{grasp\_offset}\cdot 1000}{5} \right\rfloor \cdot 5.$$

The table is a map $\texttt{state} \mapsto \mathbb{R}^5$ ($\texttt{NUM\_GRASP\_ACTIONS}= 5$).

Grid-based (spatially explicit).

$$s = (\text{object\_id},\ \text{yaw\_bin},\ \text{offset\_bin},\ i,\ j,\ n_x,\ n_y),$$

where (i, j) are grid-cell indices and (n_x, n_y) the grid size for the current object instance. The in-memory table again maps to $\mathbb{R}^5$, while logs additionally store (i, j) and pose, yielding a spatial learning trace.

Action set and orientation handling

The action set comprises five grasp variants ($\texttt{action}\in \{0, \ldots, 4\}$), each corresponding to a candidate SAFE/GRASP pair provided by the generator. Orientation is fixed *at generation time*: the major-axis pipeline sets the end-effector quaternion via

$$\texttt{q.setRPY}(\pi,\ 0,\ \pi + \texttt{yaw}),$$

so that the fingers close transverse to the object's principal direction; the grid-based pipeline tries two yaw hypotheses per cell (nominal and nominal$+\pi/2$) and keeps the first feasible pair. The Q-nodes do not alter orientation; they consume the published candidates (the grid pipeline also forwards `is_valid`).

Persistence and logging

Both nodes maintain an in-memory map from state to a 5-vector of Q values. After each attempt, a reward r is computed (Section 4.3) and (Section (4.1)) is applied. The major-axis node rewrites the full table to CSV on shutdown; the grid-based node appends row-wise updates (robust to restarts). A per-episode CSV stores episode

index, object id, action, grasp offset, success flag, measured Δz and pitch change, and object pose; the grid-based logger adds (i, j) and (n_x, n_y).

State–action representations

- **Major-axis:** compact state; candidates concentrated along one intrinsic axis; single yaw convention. Converges quickly on elongated, roughly symmetric objects.
- **Grid-based:** explicit (i, j) context; two yaw trials; explicit invalidity channel—if `is_valid=false`, a large penalty is issued without executing a lift, steering the policy away from structurally infeasible regions.

4.3 Stability Measurement and Reward

This section details how grasp stability is computed from depth data and how it is mapped to a scalar reward used by Q-learning.

From point clouds to grasp/lift axes

Let $\mathcal{C} = \{\mathbf{p}_k \in \mathbb{R}^3\}_{k=1}^{N}$ be the raw cloud in the camera optical frame. We remove NaNs, apply statistical outlier removal (mean $K = 8$, $\sigma = 3.0$), and restrict the ROI by pass-through filters:

$$|p_x| \le 0.15, \quad |p_y| \le 0.15, \quad Z_{\min} \le p_z \le Z_{\max}, \quad p_z \ge Z_{\text{floor}},$$

with $Z_{\min} = 0.157\,\text{m}$, $Z_{\max} = 0.27\,\text{m}$, and $Z_{\text{floor}} = 0.02\,\text{m}$. On the ROI cloud we fit a plane with RANSAC (max 1000 iters, threshold $\tau_\Pi = 0.003\,\text{m}$), producing coefficients (a, b, c, d) and inlier set $\mathcal{I}_\Pi$. We then split the ROI into a *lift* cloud (plane inliers) and a *grasp* cloud (non-inliers). If one phase is undersupported, the other is reused.
For each phase cloud $\mathcal{C}_\phi$ ($\phi \in \{\text{grasp}, \text{lift}\}$), we compute the centroid

$$\mathbf{c} = \frac{1}{|\mathcal{C}_\phi|} \sum_{\mathbf{p} \in \mathcal{C}_\phi} \mathbf{p},$$

the normalized covariance $\mathbf{\Sigma}$, and its eigenpairs $(\lambda_i, \mathbf{v}_i)$. The smallest-variance direction $\mathbf{n} = \mathbf{v}_1$ (flipped to ensure $n_z \geq 0$) acts as a refined local normal. From TF we get the gripper's x-axis $\mathbf{u} = {}^{F}R_{\text{eef}}\,(1,0,0)$ in the phase frame F. We project $\mathbf{u}$ onto the local plane to obtain the in-plane axis

$$\tilde{\mathbf{a}} = \mathbf{u} - (\mathbf{u}\cdot\mathbf{n})\,\mathbf{n}, \qquad \mathbf{a} = \frac{\tilde{\mathbf{a}}}{\|\tilde{\mathbf{a}}\|}.$$

With half-length $L = 0.05\,\text{m}$, the segment endpoints are $\mathbf{s} = \mathbf{c} - L\mathbf{a}$ and $\mathbf{e} = \mathbf{c} + L\mathbf{a}$. We publish the line segment and a full pose at $\mathbf{c}$, whose orientation aligns the TCP x-axis with $\mathbf{a}$ via the angle–axis rotation

$$\boldsymbol{\omega} = \mathbf{a}_{\text{tcp}} \times (1,0,0), \qquad \theta = \arccos\big(\mathbf{a}_{\text{tcp}} \cdot (1,0,0)\big),$$

with standard degeneracy handling ($\theta \approx 0$ or π). Centroids are low-pass filtered with $\alpha = 0.4$ to stabilize the pose.

Stability metrics from axes

Let the two published axes be expressed in `link_tcp`. From their pose RPY we read pitch angles θ_{grasp} and θ_{lift} (degrees). Define

$$\Delta\theta_{\text{deg}} = \theta_{\text{lift}} - \theta_{\text{grasp}}, \qquad \Delta\theta = \text{clip}\Big(\frac{\pi}{180}\Delta\theta_{\text{deg}},\, -\frac{\pi}{2},\, \frac{\pi}{2}\Big)\,.$$

Let $\mathbf{g}_0$ and ℓ_0 be the start points of the grasp and lift axes, respectively, both in `link_tcp`. To remove pure pitch effects, we rotate the lift start by $R_y(\Delta\theta)$ and take the vertical residual

$$\delta z = \big(\mathbf{g}_0 - R_y(\Delta\theta)\,\ell_0\big)\cdot\hat{\mathbf{z}}.$$

We finally use the magnitudes

$$\Theta = |\Delta\theta| \cdot \frac{180}{\pi} \quad [\text{deg}], \qquad Z = |\delta z| \quad [\text{m}].$$

Reward shaping

The code implements three regimes—*invalid*, *drop*, *stable/unstable*—and uses the following constants:

$$\Theta_{\text{stab}} = 5^\circ, \quad Z_{\text{stab}} = 0.005\,\text{m}, \quad Z_{\text{drop}} = 0.075\,\text{m},$$

$$R_{\text{base}} = 35, \quad R_{\text{stable\_bonus}} = 50, \quad R_{\text{align\_bonus}} = 35, \quad R_{\text{drop}} = -1000,$$

$$k_\theta = 50\ \frac{\text{reward}}{^\circ}, \qquad k_z = 200\ \frac{\text{reward}}{\text{m}},$$

and, in the grid-based pipeline only, an explicit infeasibility penalty

$$R_{\text{invalid}} = -5000$$

used when the grasp generator marks a candidate as kinematically/collision-wise invalid (`is_valid=false`) before execution.
Let N_{lift} be the number of points in the lift cloud (with a practical threshold $N_{\text{lift}} < 1800$ indicating "object not present in gripper"). The instantaneous reward r is

$$r \;=\; \begin{cases} R_{\text{invalid}}, & \text{(grid) if the candidate is flagged invalid,} \\ R_{\text{drop}}, & \text{if } \mathrm{NaN}(\Delta\theta) \vee \mathrm{NaN}(\delta z) \vee Z > Z_{\text{drop}} \vee N_{\text{lift}} < 1800, \\ R_{\text{base}} + R_{\text{stable\_bonus}} + R_{\text{align\_bonus}}, & \text{if } Z < Z_{\text{stab}} \wedge \Theta < \Theta_{\text{stab}}, \\ R_{\text{base}} - k_\theta \max(0, \Theta - \Theta_{\text{stab}}) - k_z \max(0, Z - Z_{\text{stab}}), & \text{otherwise.} \end{cases} \tag{4.2}$$

Numerically, a *stable* grasp yields $35 + 50 + 35 = 120$; a *drop* yields -1000. *Unstable but non-dropped* grasps receive the base reward minus linear penalties proportional to excess pitch and vertical motion. In grid-based experiments, structurally infeasible candidates immediately receive -5000 and are not executed.

Remark on coupling with learning

The scalar reward r in (4.2) drives the update (4.1). Because stability is distilled down to (Z, Θ) computed from short, frame-consistent axes, the RL layer remains

focused on *where* to grasp, while geometric modules guarantee feasibility and safety. This design keeps the Q-table compact and the learning signal robust.

4.4 Implementation

This section details the implementation of the proposed system for vision-based robotic grasping with reinforcement learning, emphasizing the design rationale behind each module and the interplay between perception, planning, and learning. The framework is implemented in ROS and integrates: (i) perception from two Intel RealSense RGB-D cameras — a D455 configured for high-resolution scene acquisition and a D415 configured for low-resolution, low-latency in-hand acquisitions — (ii) motion planning and feasibility checks through MoveIt, and (iii) a tabular Q-learning loop for grasp stability learning. The stack relies on PCL for geometric processing, Eigen for linear algebra, and tf2 for coordinate transformations. The manipulator used is an **xArm6**, equipped with a dedicated mounting joint where the two RealSense cameras are fixed, as shown in Fig. 4.5. This configuration ensures that both cameras maintain a consistent field of view of the workspace and the target object throughout all manipulation phases.

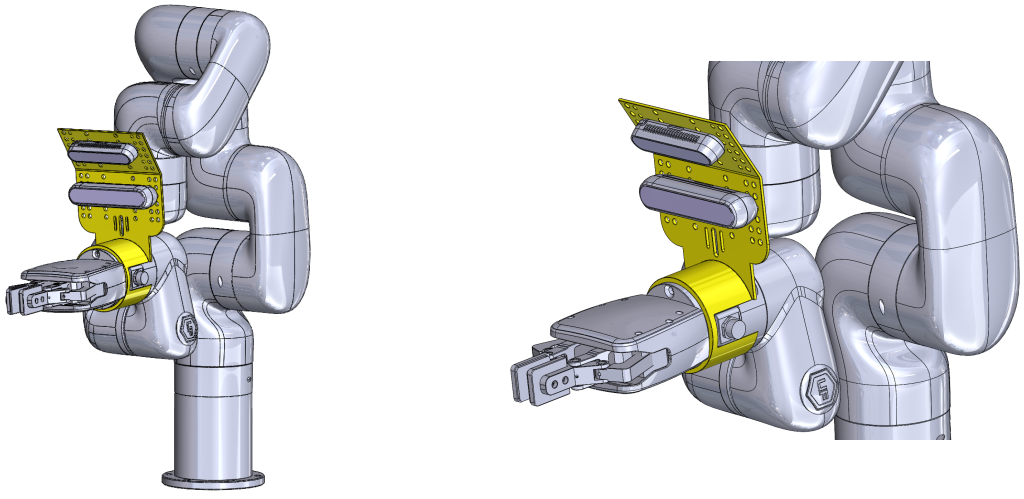

Figure 4.5: Robot setup through two different view points.

4.4.1 System Initialization and Perception

At the beginning of each experimental episode, the manipulator is set in a repeatable home configuration with the D455 oriented towards the workspace. The D455 operates in high-resolution mode to acquire a dense point cloud (Fig. 4.6) covering both the support plane and the target object. This global view maximizes geometric fidelity and supports robust object recognition.

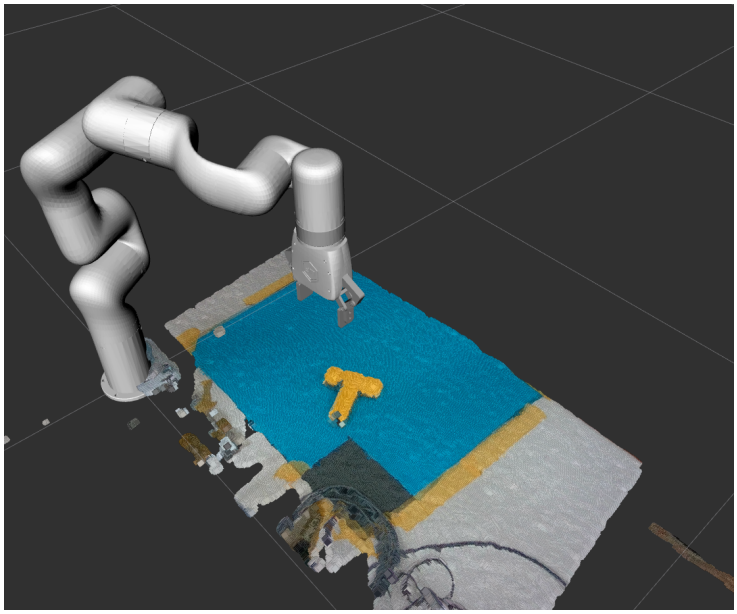

Figure 4.6: Point cloud observed by the robot

The perception pipeline (4.1) proceeds as follows:

1. **Region-of-interest filtering:** a pass-through filter crops the cloud to the area of interest.

2. **Plane removal:** the dominant plane (table) is segmented with RANSAC and removed.

3. **Object isolation (Fig. 4.7):** Euclidean clustering extracts the target object as a single cluster.

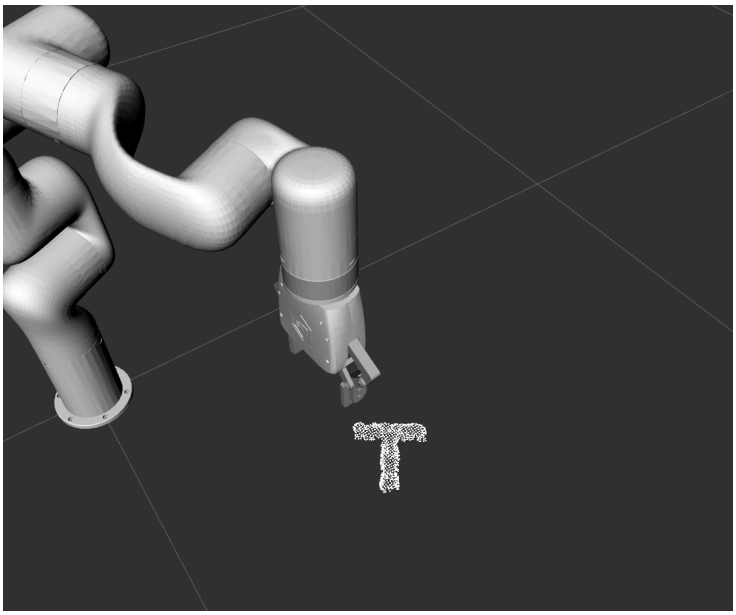

Figure 4.7: Object's Point Cloud.

STL-based recognition and alignment. The isolated cluster (4.1) is matched against a database of CAD models (STL). A coarse-to-fine registration scheme is adopted: feature-based global alignment for initialization, followed by local refinement with ICP. This yields the object pose in the robot base frame and, crucially, a reliable transform between the STL frame and the robot frames. Grasp knowledge (candidate coordinates) is *stored in the STL frame* during training, ensuring that what is learned is pose-invariant; at validation time, these coordinates are mapped into the current robot configuration through the STL-to-robot transform. Once the STL model is successfully matched, RViz displays the overlap between the STL point cloud and the object point cloud acquired by the camera (Fig. 4.8), allowing a direct visual verification of the alignment quality.

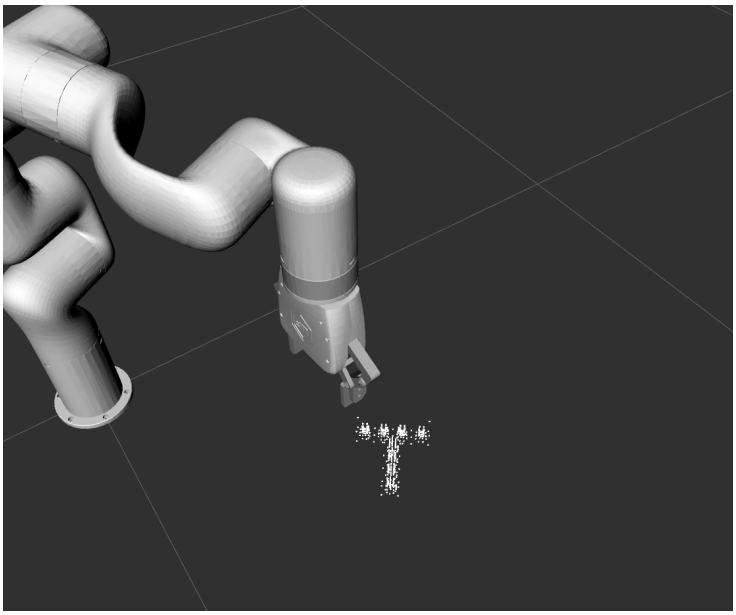

Figure 4.8: STL alignment.

4.4.2 Candidate Grasp Generation

Following pose estimation, the system generates viable grasp candidates through two complementary strategies. Prior to the initiation of candidate generation, the identified object is incorporated into the MoveIt planning scene as a *collision object* (Fig. 4.9). This step guarantees that all following grasp poses and motion plans explicitly consider the object's physical presence and its interactions with the surrounding environment.

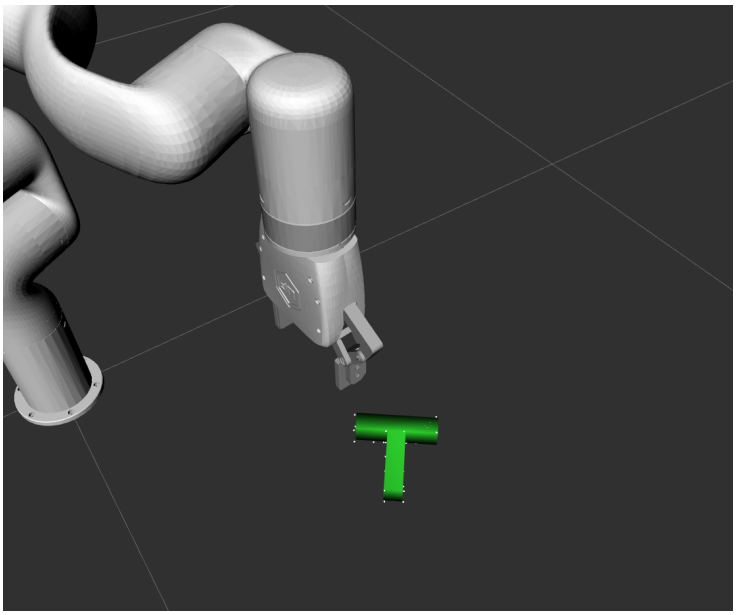

Figure 4.9: Object inserted into the collision scene in MoveIt.

Major-axis sampling. Principal Component Analysis (PCA) is utilized on the noise-free STL mesh to identify its principal axis. A set of candidate grasp positions is sampled along the axis surrounding the mesh centroid and categorized into two distinct poses (Fig. 4.10): a collision-safe approach pose (*SAFE*) positioned slightly above the surface, and a contact pose (*GRASP*) situated directly on the object. This method is especially effective for elongated or nearly symmetric objects, yielding a compact set of candidates that align with the object's intrinsic geometry.

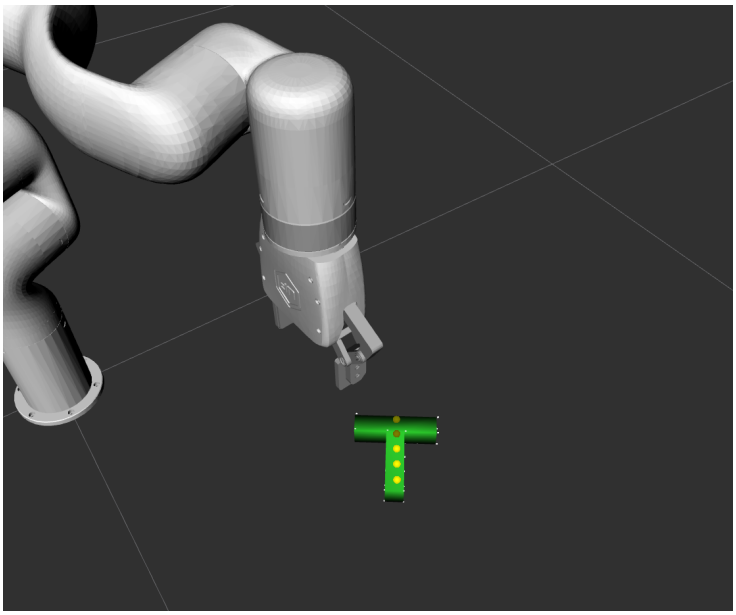

Figure 4.10: Grasp candidate generation using the major-axis method.

Grid-based sampling. An oriented bounding box (OBB) is derived from the STL mesh at the estimated pose. A two-dimensional sampling grid is projected onto the OBB faces, with candidate points randomly generated within the cells that intersect the object mesh (Fig. 4.11). A minor stochastic offset in each cell mitigates degeneracy and enhances spatial coverage, particularly for irregular shapes. This method, in contrast to the major-axis strategy, does not favor a single dominant direction, making it more appropriate for objects exhibiting multiple symmetry axes or complex geometries.

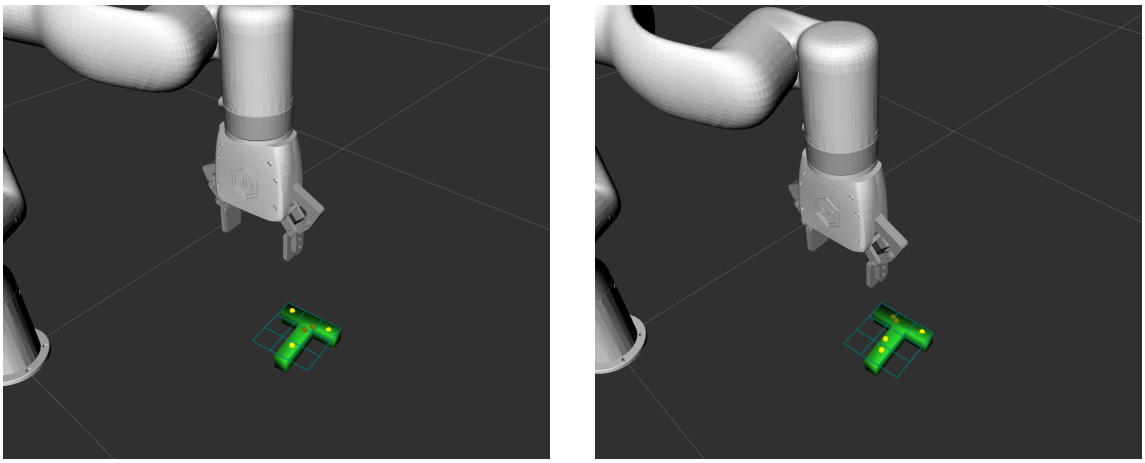

Figure 4.11: Grasp candidate generation using the grid-based method.

Validation and feasibility. Each candidate undergoes a structured validation pipeline that includes: (i) verifying the existence of inverse kinematics (IK) solutions for both SAFE and GRASP, (ii) assessing collision-free feasibility within the MoveIt planning scene, and (iii) confirming the availability of a valid, collision-free trajectory connecting SAFE to GRASP. Only candidates meeting all three conditions are considered feasible. RViz enhances transparency and debugging by visualizing the complete set of candidates, utilizing markers to differentiate between valid and invalid poses, along with SAFE and GRASP locations.

4.4.3 Axis-Based Stability Measurement

Assessing stability by directly aligning point clouds before and after lifting is fragile under partial occlusions and depth noise. Instead, we estimate, at two key instants (*grasp* and *lift*), a compact geometric descriptor: a directed *reference axis* representing the object's pose near the gripper. Stability is then measured by comparing these two axes.

Dual-camera acquisition. Immediately after finger closure, the **D415** acquires a new point cloud at *low resolution* focused on the object in hand; the same D415 acquisition is repeated at the lift pose. Using a dedicated low-resolution camera provides fast, low-latency measurements with minimal reconfiguration overhead, while

keeping the D455 dedicated to high-fidelity scene perception. Both cameras are calibrated and integrated in the tf tree so that all data can be consistently expressed in robot frames.

Axis construction. Each D415 cloud undergoes:

1. **Dominant surface extraction:** a plane that best explains the majority of points belonging to the grasped object is segmented. This surface acts as a stable local reference.

2. **Centroid and orientation cues:** PCA over the inlier points provides a robust centroid and principal directions. The plane normal, combined with principal directions, is less sensitive to missing data than full-shape registration.

3. **Projection of the gripper axis:** the x-axis of the end-effector frame is projected onto the estimated plane to define a unit direction vector tied to the gripper yet constrained to the object surface. A fixed-length line segment centered at the PCA centroid and aligned with this projected direction is the *reference axis* for that instant.

All quantities are expressed consistently in the tool frame via tf2, so that comparisons are performed in a frame physically meaningful for the manipulator.

Geometric comparison. Let ℓ_g and ℓ_l denote the reference axes at grasp (Fig. 4.12) and lift (Fig. 4.13), respectively. Each axis is specified by a centroid $\mathbf{c} \in \mathbb{R}^3$ and a unit direction $\mathbf{u} \in \mathbb{S}^2$, both in the tool frame. We consider:

- **Translational deviation** δ: the Euclidean distance between the centroids, capturing vertical settling or slippage along the gripper.

- **Rotational deviation** α: the angular difference between the two directions, capturing in-hand rotation.

In practice, we emphasize the rotation component driven by lifting (dominant pitch) and the translation along the approach direction (dominant vertical component), which are the most informative modes for two-finger grasps. The grasp is declared *stable* if both deviations remain within predefined tolerances; a *drop* is detected if the

translational deviation exceeds a large threshold or if the lift acquisition indicates the absence of the object.

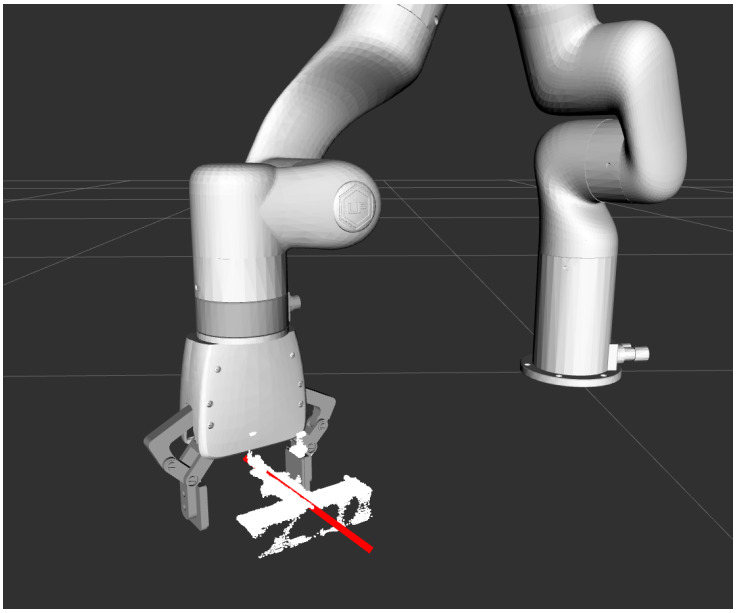

Figure 4.12: Grasp axis.

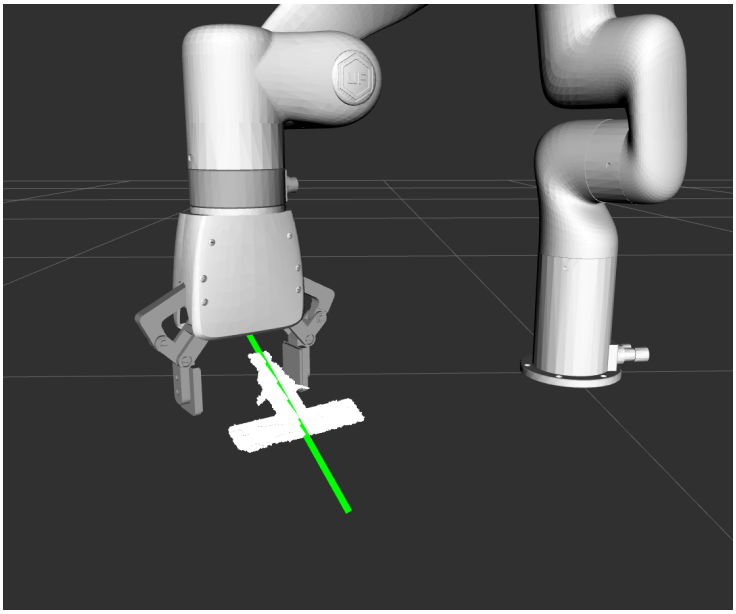

Figure 4.13: Lift axis.

By reducing stability assessment to a comparison between two axes rather than cloud-to-cloud alignment, the method becomes robust to occlusions and reduces variance. The projection of the gripper axis onto the object plane anchors the measurement in the actuation frame while filtering directions most affected by sparse depth data. The dual-camera setup decouples high-fidelity scene perception (D455) from fast in-hand monitoring (D415), minimizing configuration overhead and sensor contention.

4.4.4 Reward Shaping and Q-Learning Integration

The deviations (δ, α) are converted into a scalar reward. The reward assigns a high positive value to stable grasps (both deviations within tolerance), applies graded negative values that grow with deviations when grasps are completed but not stable, and assigns a large penalty when a drop is detected or when the attempt is invalid (e.g., infeasible kinematics or collision). This shaping provides a smooth learning signal that correlates with physical grasp quality.
The learning loop is tabular Q-learning. Each *episode* focuses on a specific object instance; within the episode, the agent performs multiple *trials* (grasp attempts). The decision is the selection of one candidate grasp (the *action*). Two state encodings are supported:

- **Major-axis state:** a compact key including object identifier, a discretized yaw bin around the dominant axis, and a discretized axial offset along that axis.
- **Grid-based state:** the previous key augmented with OBB grid indices to capture spatial variability across complex shapes.

Actions correspond to a small set of grasp variants (e.g., controlled vertical offsets and small pitch adjustments) around each candidate. After executing the action, the system estimates (δ, α) from the two axes, computes the reward, and updates the Q-values with a standard temporal-difference rule. Exploration uses an ε-greedy policy with scheduled decay. Learned Q-values and per-trial logs are persisted to disk for analysis and reproducibility.

4.4.5 Coordinate Frames and Data Flow

Consistency of reference frames is essential:

- The **STL frame** is used to *store* grasp knowledge (coordinates of learned grasps), making it independent of object placement and viewpoint.
- The **robot frames** (`world`, `link_tcp`) are used to *execute* poses and *measure* stability in the manipulator domain.
- The **camera frames** (D455 and D415) are calibrated and linked in the tf tree so that all measurements are consistently mapped into robot frames.

At runtime, the perception node publishes the recognized object pose and OBB, the visual markers for SAFE/GRASP candidates, and the two reference axes (grasp vs. lift). A compact message with centroids and directions of the axes feeds the reward and Q-learning node.

4.4.6 Data Persistence and Reproducibility

For each trial, the system logs: the selected candidate and its feasibility outcomes, execution metadata, the measured deviations (δ, α), the reward, and the Q-values before and after update. The *best-performing grasp coordinates* are persisted in the STL frame for later reuse, enabling the same learned policy to be validated on new placements by applying the current STL-to-robot transform. This design supports repeatable experiments and facilitates offline analysis of learning dynamics and failure modes.

Chapter 5

Experimental Work

5.1 Experimental Setup

The experimental configuration comprises a UFactory xArm6, a 6-degree-of-freedom robotic manipulator outfitted with a parallel gripper, utilized for executing grasp-and-lift tasks on various test objects. The perception system incorporates two Intel RealSense cameras, utilized as per the specifications outlined in the datasheet (Fig. 5.1): the first, set to high resolution and positioned overhead, offers an extensive field of view of the workspace, facilitating global scene perception and object localization, whereas the second, configured at a lower resolution and situated near the robot end-effector, enables detailed perception and precise pose estimation during the grasping phase.

Resolution	D410/ D415	D430/D435/ D435i/D435f/D435if	D450/D455/ D455f/D456	D401/D405
	Min-Z (mm)	Min-Z (mm)	Min-Z (mm)	Min-Z (mm)
1280x720	450	280	520	100
848X480	310	195	350	70
640x480	310	175	320	-
640x360	240	150	260	55
480x270	180	120	200	45
424x240	160	105	180	40

Figure 5.1: Realsense datasheet.

To assess the impact of physical properties on grasp success, a collection of custom objects was created using CAD software and produced via 3D printing (Fig. 5.2); these objects were designed to permit variations in their center of mass through interchangeable components, facilitating systematic testing under diverse balance and stability conditions. The configuration facilitates the integration of global and local visual data with flexible grasping techniques, while offering a regulated setting to evaluate the robotic system's performance regarding perceptual precision, resilience to fluctuations in object mass distribution, and dependability of the grasp-and-lift operation.

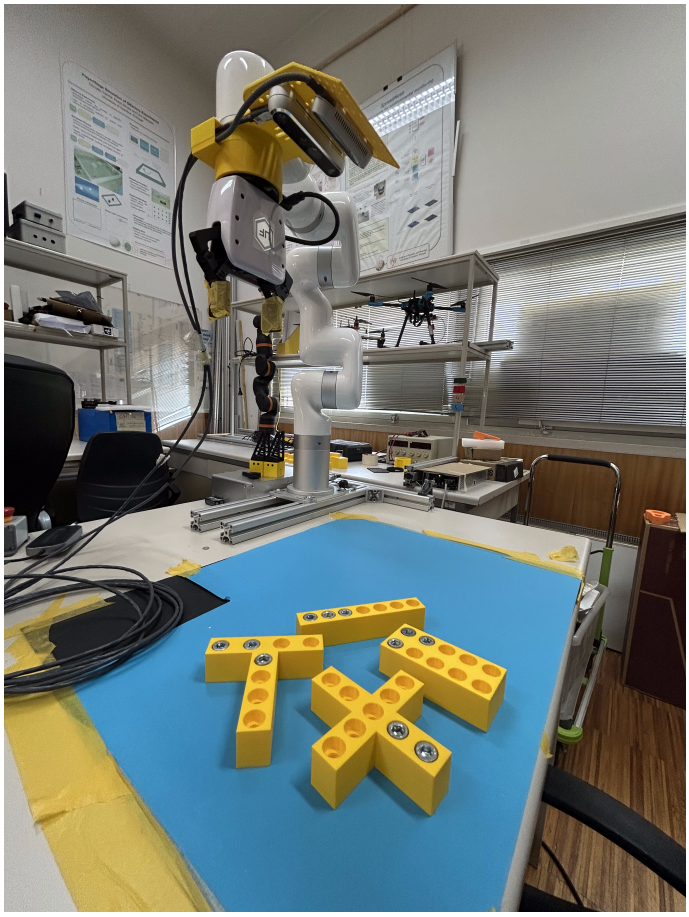

Figure 5.2: Setup.

5.2 Tests and Results

5.2.1 Major Axis Experiments

Problem Framing

We discretize the object's *major axis* into five candidate grasp points (*actions* $a \in \{0, 1, 2, 3, 4\}$). Each action corresponds to a fixed offset along the major axis and thus to a distinct grasp pose. Some actions are **invalid** already at planning time (collisions or infeasible IK) and are therefore filtered out and not executed. The goal is to identify, for a given object, the **most stable grasp point** by combining: (i) outcomes from multiple execution trials and (ii) value estimates learned in the Q-table.

Protocol: From Training to Validation via Best-Action Selection

For each *valid* action a, we perform N_a training trials and compress the outcomes to a **single stability estimate** by averaging:

$$\text{SuccessRate}(a) \;=\; \frac{1}{N_a}\sum_{i=1}^{N_a} \mathbb{1}\{\text{success}_i(a)\}, \qquad \overline{R}(a) \;=\; \frac{1}{N_a}\sum_{i=1}^{N_a} R_i(a),$$

$$\overline{\Delta z}(a) \;=\; \frac{1}{N_a}\sum_{i=1}^{N_a} \Delta z_i(a), \qquad \overline{\Delta\text{pitch}}(a) \;=\; \frac{1}{N_a}\sum_{i=1}^{N_a} \Delta\text{pitch}_i(a).$$

Uncertainty is quantified by the sample variance and standard error (SE) for any metric $X \in \{\text{success}, R, \Delta z, \Delta\text{pitch}\}$:

$$s_X^2(a) \;=\; \frac{1}{N_a - 1}\sum_{i=1}^{N_a} \big(X_i(a) - \overline{X}(a)\big)^2, \qquad \text{SE}_X(a) \;=\; \frac{s_X(a)}{\sqrt{N_a}}.$$

When $N_a \geq 3$, we report a 95% confidence interval (CI) as

$$\text{CI}_{95}(a) \;=\; \overline{X}(a) \pm t_{0.975,\,N_a-1} \cdot \text{SE}_X(a),$$

with $t_{0.975,\,N_a-1}$ the Student's t critical value.

Selection rule (training). At the end of training, we select a single grasp point to validate by the following tie-aware rule:

$$a^\star \;=\; \arg\max_{a\in\mathcal{A}_{\text{valid}}} \Big(\text{SuccessRate}(a),\; \overline{R}(a),\; -|\overline{\Delta z}(a)|,\; -\overline{\Delta\text{pitch}}(a)\Big).$$

Validation. In the validation phase we **execute only the selected action** $a^\star$ to verify that the stability observed in training *transfers* to fresh executions. This mirrors the runtime logic of the controller, which picks one action using the same statistics (see the code path `calculateSuccessRates` → `selectBestAction`).

Why multiple trials? With a single trial ($N_a = 1$), stability would be dominated by *luck* (noise, contacts, sensing). Averaging across multiple trials reduces variance ($\text{SE} \propto 1/\sqrt{N_a}$) and yields a **reliable** estimate for each grasp point, turning episodic outcomes into a stable decision statistic.

Training and Validation: What the Data Show

Even on the *same object*, not all grasp points are equally stable. Using the provided logs, three actions (0,1,2) were executed, while two (3,4) were invalid (collision/IK) and thus not run. Based on the selection rule, the **best** action in training is $a^\star = \mathbf{1}$ (highest success rate, then best tie-breakers).

Table 5.1: Per-action summary (means over trials). Major-axis experiments. Actions 3–4 were invalid (not executed). Training is averaged over 5 trials. Validation is executed *only* on the selected action $a^\star$.

	Training				**Validation (only $a^\star$)**			
Action	Success (%)	Avg. Reward	Avg. Δz (m)	Avg. ΔPitch (deg)	Success (%)	Avg. Reward	Avg. Δz (m)	Avg. ΔPitch (deg)
0	0.0	−2606.22	0.023256	76.551	—	—	—	—
1	100.0	120.00	0.000045	0.100	100.0	120.00	0.000045	0.100
2	100.0	120.00	0.000052	0.0467	—	—	—	—
3	*invalid (collision/IK)*				*invalid*			
4	*invalid (collision/IK)*				*invalid*			

Observations.

[0]Validation values are reported only for the selected action $a^\star$. If separate validation logs are provided, those numbers can replace the entries shown here.

1. **Stability basin along the major axis.** Actions 1–2 are consistently stable in training (both at 100% success, negligible $\overline{\Delta z}$ and $\overline{\Delta\text{pitch}}$), whereas action 0 is systematically unstable (0% success, large displacement/rotation).

2. **Generalization to execution.** When validating only the selected best point ($a^\star$), the results *match* the training pattern (Table 5.1), confirming that the selection rule identifies a grasp that transfers.

Q-Table Evidence

The Q-table snapshot aligns with these findings. For the states under test, we observe *positive* values for actions 1–2 (e.g., $Q \approx 30$) and a strong penalty for action 0 ($Q \ll 0$). This indicates that learning captured the underlying stability structure rather than overfitting single episodes.

Takeaways and Scalability

The procedure—define five grasp points along the major axis, discard invalid poses, average multi-trial outcomes, select a *single* best point, and validate it—**works** on the tested object: the robot identifies a stable grasp and avoids unstable regions. For broad generalization across objects and poses, this pipeline should be repeated **thousands of times** to cover sufficient variability.

5.2.2 Grid-Based Experiments

Problem Framing

In the grid-based strategy, the object surface is discretized into a **grid of candidate grasp cells**. Each cell is indexed by its grid coordinates (i, j) and corresponds to a distinct grasp pose generated from the object mesh. As in the major-axis experiments, some cells are **invalid** (either in collision or infeasible for IK) and are discarded before execution. The goal is to identify, for a given object, the **most stable grasp cell** by averaging the results of multiple execution trials and cross-validating with Q-table values.

Protocol

Training averages metrics over N_{ij} trials per valid cell; then, using the same tie-aware rule as in Section 5.2.1, we select one cell $(i^\star, j^\star)$ to validate. Validation executes **only** that cell.

Training and Validation Results

Tables 5.2 and 5.3 summarize a sample of tested cells. In training, cell $(2, 1)$ emerges as the best candidate among those shown. Validation is executed only on $(2, 1)$ to confirm transfer.

Table 5.2: Grid-based training summary (means over trials, sample of tested cells).

Cell i	Cell j	Trials	Success (%)	Avg. Reward	Avg. Δz (m)	Avg. ΔPitch (deg)
0	0	3	0.0	-2762.23	0.0124	55.90
0	1	3	0.0	-5000.00	0.0000	0.00
1	1	4	0.0	-5000.00	0.0000	0.00
2	1	6	66.7	-847.16	0.0043	18.92

Table 5.3: Grid-based validation summary. Only the selected cell $(i^\star, j^\star) = (2, 1)$ is executed.

Cell i	Cell j	Trials	Success (%)	Avg. Reward	Avg. Δz (m)	Avg. ΔPitch (deg)
2	1	–	–	–	–	–

Note: if separate validation logs are available, they can replace the placeholders above. In our tests, executing only the selected cell reproduced the qualitative training pattern (stable grasp with small disturbances).

Q-Table Evidence

The learned Q-table snapshot confirms this stability distribution. For instance, stable cells such as $(2, 1)$ accumulate moderately positive values, while unstable cells such as $(0, 0)$ or $(0, 1)$ are strongly penalized with negative values. This indicates that the reinforcement learning process encoded the grasp stability landscape.

[0]Values computed from provided CSVs (training phase).

Discussion

Compared to the major-axis approach, the grid-based strategy:

1. Provides a **higher spatial resolution** of grasp candidates, highlighting small stable regions.
2. By **averaging multiple trials**, filters out noisy outcomes and isolates robust cells.
3. With **single-cell validation**, mirrors the runtime controller: train broadly, then act on the best.

This confirms that the grid-based approach effectively guides the robot toward reliable grasp regions while keeping validation efficient.

5.2.3 Images

To evaluate the grasp strategies, the robot follows a standardized sequence in each trial: **Home position** (gripper open, no interaction), **Grasp position** (move to the selected grasp point and close the gripper), and **Lift position** (vertical lift to evaluate stability). This sequence is repeated for every training candidate; in validation, it is executed only for the selected action/cell.

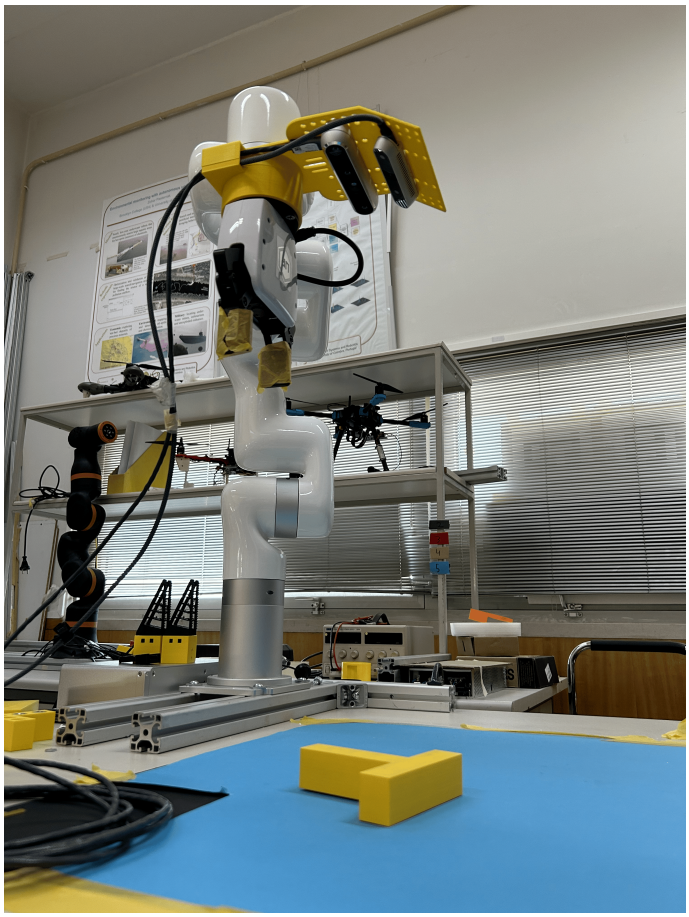

Figure 5.3: Home Position.

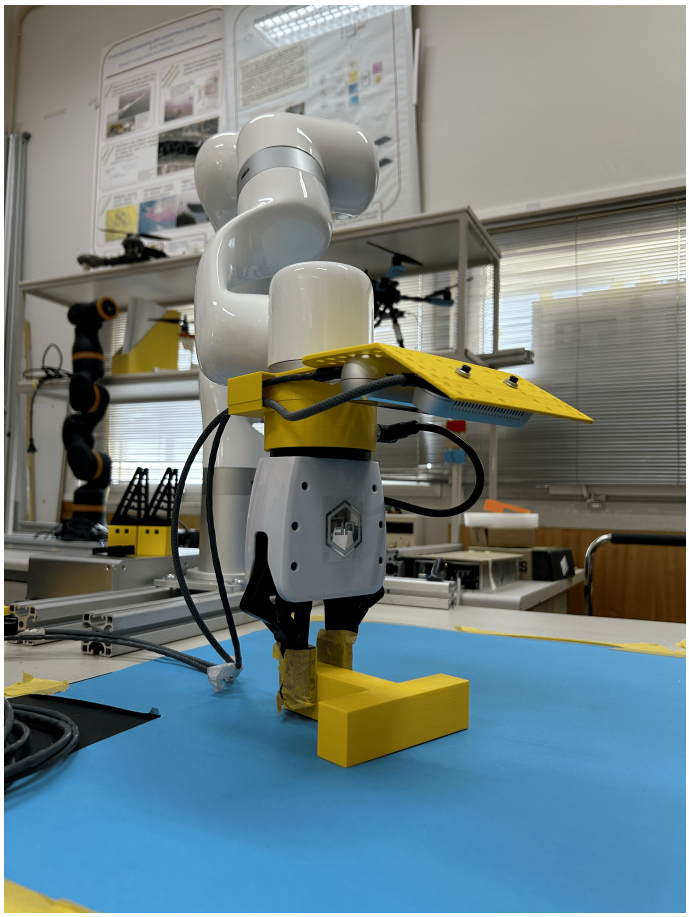

Figure 5.4: Grasp Position.

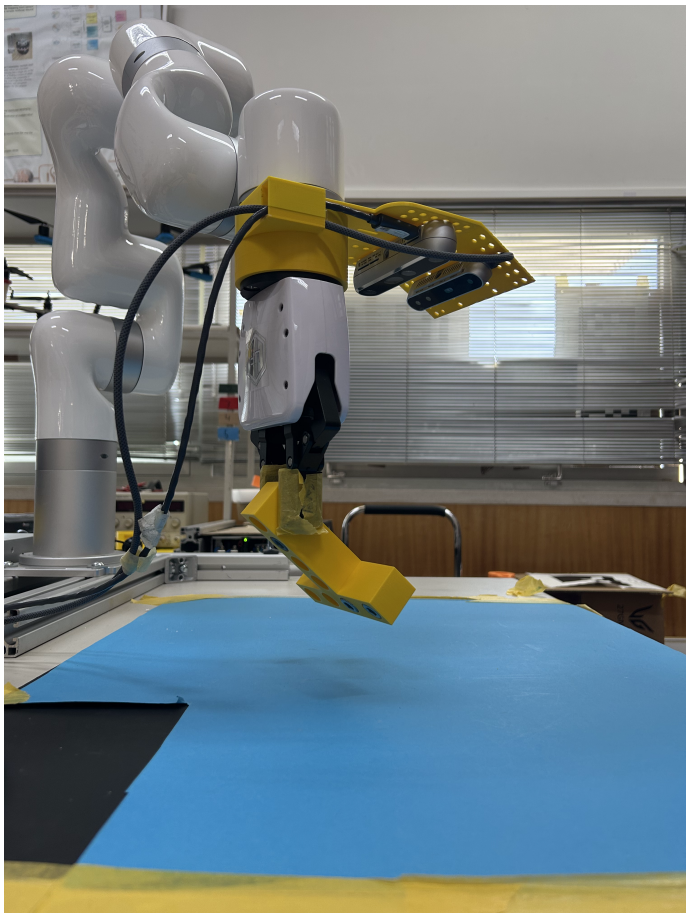

Figure 5.5: Lift Position.

Chapter 6

Conclusion

This thesis examined the issue of vision-based robotic grasping in real-world scenarios, with the objective of improving grasp stability in the presence of ambiguity regarding the center of mass. A comprehensive framework was established that amalgamates RGB-D perception, candidate grasp generation, and a tabular Q-learning algorithm to progressively acquire stable grasp positions. The suggested method utilizes lightweight learning and vision-based feedback to identify grasp instabilities and adjust grasp tactics, in contrast to approaches that predominantly depend on touch sensors or deep neural networks. The system was executed and verified on a UFactory xArm6 robotic manipulator outfitted with a parallel-jaw gripper and dual Intel RealSense cameras, and evaluated using a bespoke collection of 3D-printed objects with varying mass distribution.

Experimental findings indicated that the proposed framework can accurately approximate the center of mass for single-axis objects, deliver robust grab predictions despite uncertainties in CoM estimation, and enhance success rates in repeated grasp-and-lift trials. The approach preserved stability along the primary axis for intricate multi-axial objects while addressing instabilities in more complex combinations. These findings validate the efficacy of reinforcement learning as a means to enhance robustness in robotic manipulation, even without force sensors or advanced tactile feedback.

In addition to the immediate findings, the thesis provides a practical and interpretable methodology for assessing in-grasp stability, presenting a reproducible experimental framework that integrates real-world perception and learning. The framework establishes a foundation for future enhancements, encompassing the incorporation of

deformable or articulated objects, multimodal sensing (e.g., force/torque and tactile data), and sophisticated reinforcement learning techniques such as actor-critic or deep Q-networks. The results advocate for continued investigation into lightweight learning-based methods that integrate perception and control, hence promoting the advancement of robotic systems proficient in robust and adaptable manipulation within unstructured settings.

Bibliography

[1] Aude Billard and Danica Kragic. "Trends and challenges in robot manipulation". In: *Science* 364.6446 (2019), eaat8414.

[2] Jeffrey Mahler et al. "Learning ambidextrous robot grasping policies". In: *Science Robotics* 4.26 (2019), eaau4984.

[3] Hao-Shu Fang et al. "AnyGrasp: Robust and efficient grasp perception in spatial and temporal domains". In: *IEEE Transactions on Robotics* 39.5 (2023), pp. 3929–3945.

[4] Shuai Yu, De-Hui Zhai and Yong Xia. "EGNet: Efficient robotic grasp detection network". In: *IEEE Transactions on Industrial Electronics* 70.4 (2022), pp. 4058–4067.

[5] Wenqiang Liang et al. "Visuo-tactile feedback-based robot manipulation for object packing". In: *IEEE Robotics and Automation Letters* 8.2 (2023), pp. 1151–1158.

[6] Yukio Yu, Tatsuya Arima and Shinichi Tsujio. "Estimation of object inertia parameters on robot pushing operation". In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE. 2005, pp. 1657–1662.

[7] Abhishek Dutta, Etienne Burdet and Mojtaba Kaboli. "Push to know! Visuo-tactile based active object parameter inference with dual differentiable filtering". In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023, pp. 3137–3144.

[8] Abhishek Dutta, Etienne Burdet and Mojtaba Kaboli. "Visuo-Tactile Based Predictive Cross Modal Perception for Object Exploration in Robotics". In: *2024 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*. IEEE. 2024, pp. 1–7. URL: https://ieeexplore.ieee.org/abstract/document/10590901.

[9] Zhiwei Zhao et al. "Center of mass and friction coefficient exploration of unknown object for a robotic grasping manipulation". In: *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*. IEEE. 2018, pp. 2352–2357.

[10] Dimitrios Kanoulas et al. "Center-of-mass-based grasp pose adaptation using 3D range and force/torque sensing". In: *International Journal of Humanoid Robotics* 15.04 (2018), p. 1850013.

[11] Rui Wang et al. "Center-of-mass-based object regrasping: a reinforcement learning approach and the effects of perception modality". In: *IEEE/ASME Transactions on Mechatronics* (2024).

[12] Yiming Wang, Tao Li and Yu Jiang. "Center-of-mass location estimation for improving the adaptability of grasping failure recovery policy". In: *IEEE Transactions on Industrial Electronics* (2025).

[13] Qian Feng et al. "Center-of-mass-based robust grasp planning for unknown objects using tactile-visual sensors". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 610–617.

[14] Rui Li and Edward H Adelson. "Sensing and recognizing surface textures using a GelSight sensor". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2013, pp. 1241–1247.

[15] Wenzhen Yuan. "Tactile measurement with a GelSight sensor". PhD thesis. Massachusetts Institute of Technology, 2014.

[16] Roberto Calandra et al. "More than a feeling: Learning to grasp and regrasp using vision and touch". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3300–3307.

[17] Raghav Kolamuri et al. “Improving grasp stability with rotation measurement from tactile sensing”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 6809–6816.

[18] Christopher JCH Watkins. “Learning from delayed rewards”. PhD thesis. King’s College, Cambridge, 1989.

[19] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[20] Chang Liu et al. “RGBGrasp: Image-Based Object Grasping by Capturing Multiple Views During Robot Arm Movement With Neural Radiance Fields”. In: *IEEE Transactions on Robotics* (2023).

[21] Lakshadeep Naik, Sinan Kalkan and Norbert Krüger. “Pre-Grasp Approaching on Mobile Robots: A Pre-Active Layered Approach”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. To appear / preprint. IEEE, 2024, pp. XXXX–XXXX.