# Utilizing human-inspired dexterous picking skills for enabling context-aware robotic manipulation in multi-object scenarios

by

Anagha Dangle

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Masters of Science

in

Robotics Department

by

---

May 2024

APPROVED:

---

Dr. Berk Calli, Advisor

---

Dr. Jane Li, Faculty

---

Dr. Haichong Zhang, Faculty

**Abstract**

This research aims to emulate the dexterity and precision found in human grasping capabilities, particularly when dealing with difficult-to-pick objects and challenging manipulation scenarios. To solve these challenges, we leverage four dexterous picking skills inspired by human manipulation techniques that include sliding, pushing to a vertical surface, leveraging a horizontal surface, and flipping objects. The proposed approach extends beyond traditional methods by incorporating a decision-making process that assesses which, where, and how to apply specific manipulation skills for objects within the scene. Utilizing deep neural networks, the system identifies the most suitable manipulation skill for each object in the scene, assigns confidence scores indicating the potential success of each pick, and predicts precise skill locations. The adaptability of the proposed system is rigorously evaluated through a series of real-world experiments, encompassing scenarios involving known, unknown, and occluded objects. These experiments, comprising 45 trials with 150+ grasps, validate the system's reliability and robustness, particularly in cluttered settings. This research helps bridge the gap between human and robotic grasping, showcasing promising results in various practical scenarios.

**Acknowledgements**

I extend my profound gratitude to Prof. Berk, my thesis advisor, whose belief in my potential provided me with the opportunity to work on this project. His patience, guidance, and steadfast support were instrumental in completion of this thesis. The research I did in MER Lab represents the highlight of my master's journey and that could not have been possible without Prof. Berk.

I am also thankful to my thesis committee, Prof. Jane Li and Prof. Haichong Zhang, for their precious time and insightful feedback on my thesis. My gratitude extends to Amazon Robotics Greater Boston Tech Initiative for funding the project, and to all collaborators whose expertise and advice greatly enhanced my research experience.

I would especially like to acknowledge Denny Boby and Mihir Deshmukh, whose contributions were crucial to the completion of this thesis. Mihir Deshmukh, in particular, was of invaluable assistance for the completion of the experiments of this thesis. I'm equally thankful to my peers in the Manipulation and Environmental (MER) Lab — Galen, Sreejani, Abhinav, Albert, James, and others—for their constructive suggestions. Also, thankful to Mihir Kulkarni and Sumukh, who were always there to offer their support whenever needed.

I would like to thank my family and friends for their encouragement, support and faith at every stage in my life. Finally, I am grateful for the community at WPI—those who have cared, those who provided meals when I was busy working on research, and those from whom I have learned immensely. The experiences and knowledge acquired during my two years at WPI is something I will never forget.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Humans possess an extraordinary ability to grasp objects with remarkable precision and adaptability, which is particularly evident in how we approach each object uniquely. The intricate skill of grasping an object involves the combined effort of sensory perception, motor control, and cognitive behaviors that enable us to tailor our grasp to the specific characteristics of the target object. As a human, we employ a range of grasp strategies, including power, intermediate, and precision grasps [15], [6], each finely tuned for an object's shape, size, texture, and material properties. This adaptability allows us to interact with our environment seamlessly, manipulating everything from delicate items that require a gentle touch to robust objects needing a firm grip.

## 1.1 Motivation

Despite extensive efforts, there have been significant challenges in translating these nuances of human grasping into robotic systems. A significant issue lies in dynamically adjusting grasp strategies for a wide range of objects, especially within cluttered or unstructured environments, where the unique characteristics of each object demand a flexible and context-aware approach to grasping. These challenges underscore the necessity for robots to interpret sensory data in real time, make informed decisions that consider various manipulation possibilities, and execute these decisions with a precision akin to the human hand.

The choice of the right skill for a target object is influenced not only by its shape but also significantly by the context of its environment. This includes the relative positions of the surrounding objects and the specific configuration of the environment, such as the presence and location of table edges and vertical surfaces. For instance, consider the scenario of picking up a cylindrical object from a table. If the object is situated away from other items, a simple picking technique might suffice. However, if it is placed close to a wall, employing a more nuanced approach that leverages the wall for support could enhance the grasp's success and stability. This level of adaptability extends to scenarios involving objects with complex shapes or placement. For example, removing a flat plate from a table requires not just reaching for the plate but also considering any potential obstructions that might complicate the task, such as other objects resting on the plate or barriers between the plate and the edge of the table. We, as humans, decide how to pick any object easily considering all these factors. However, for a robot, it is challenging to navigate these obstacles,

strategically determining the best strategy and point of contact for successful manipulation. This intricate process of adapting manipulation strategies to the context of the environment and the specific characteristics of objects underscores the essence of my thesis work, which aims to bridge the gap between human dexterity and robotic capability in grasping and object manipulation.

## 1.2 Proposed framework

Motivated by the dexterity of humans, this thesis proposes a manipulation pipeline that utilizes four dexterous manipulation skills inspired by frequently employed human picking skills. These skills include sliding objects to the edge, pushing them to a vertical surface, leveraging a horizontal surface to facilitate picking and flipping them. The methodology of this system extends traditional approaches by combining the decision-making process on when and how to apply specific manipulation skills for a particular cluttered scene. This research outlines a framework for robotic manipulation that intelligently prioritizes tasks and manipulates objects based on an analysis of the environment and the specific needs of the task, particularly in cluttered scenes. The system efficiently organizes workspaces by identifying and employing the optimal skill to clear cluttered scenes through an adaptive approach.

The pipeline begins by capturing an RGB-D image of a cluttered multi-object scene. Following this, instance segmentation is carried out to determine the edges of the object in the scene and identify the most suitable skill for each object along with confidence scores for each skill. After identifying the appropriate skills, the

system predicts accurate grasp locations on the objects using heatmap generation, tailored to the specific skill to be applied. With the skills and grasp of locations established, the final stage involves motion planning, which executes the necessary movements to perform the skills on each object in the scene and effectively declutters the workspace.

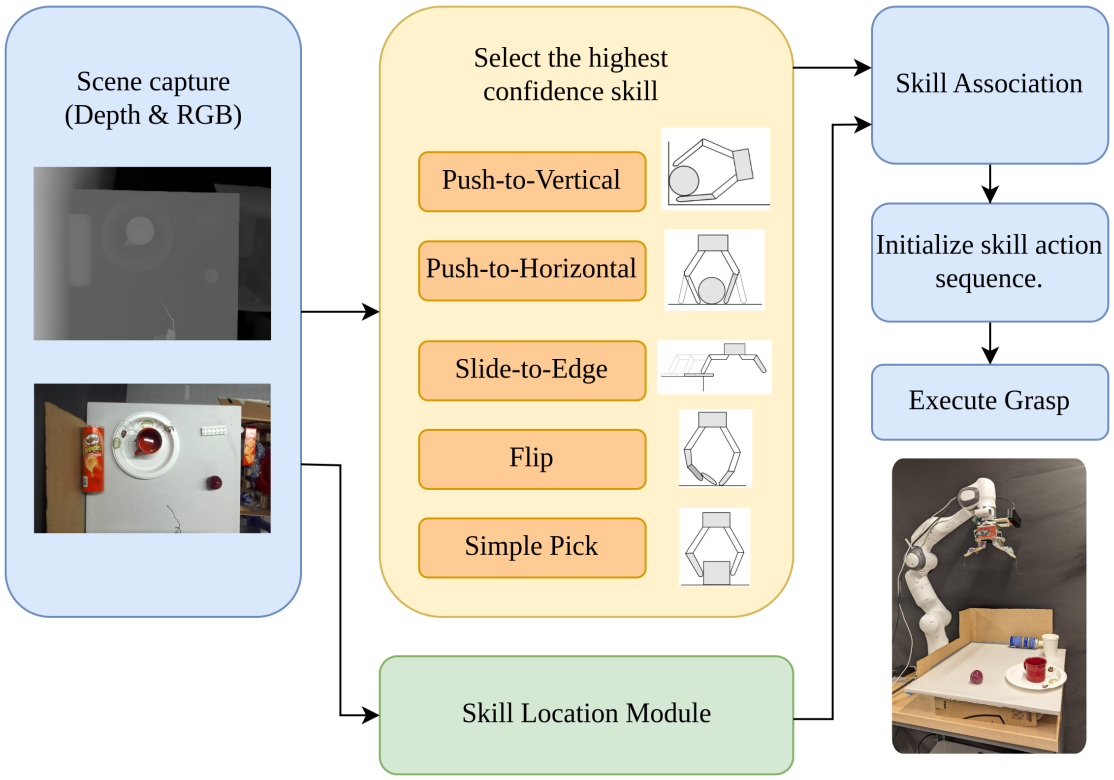



Figure 1.1: Overview of our Dexterous Picking pipeline.

## 1.3 Contributions

- Skill Detection Module: The skill detection module accurately predicts the most appropriate skills for objects within a multi-object scene. It takes an RGB-D image as the input which is captured by a camera attached to the end-effector of the manipulator. For each predicted skill, a confidence level is assigned to identify the specific skill required for handling an object. It determines the priority of objects for picking based on the confidence scores. It utilizes a context-aware approach that takes into account not only the object's shape but also the positions of nearby objects and the overall environmental configuration.

- Skill Location Module: The development of a Skill Location Module, an attention gate-based neural network allows for the precise identification of where on the object the chosen skill should be applied. The model outputs a heatmap that shows probable grasp locations on each object. The point with the highest value (between 0 to 1 output) on the heatmap is selected as the co-ordinate where the skill can be applied.

- Use of depth images: We successfully demonstrate our system's performance in real-world conditions, even with a substantial 70:30 ratio of simulation to real-world training images. This adaptability is possible due to the use of depth images, which remain consistent in simulation and when captured by real-depth cameras. Depth images, by their nature, offer a third dimension of information, which is vital for achieving a deeper spatial understanding of the

robot's environment.

- Real-World Demonstration: The culmination of this effort is a comprehensive end-to-end pipeline designed for the task of clearing objects from tabletops. This system has been put to the test in real-world conditions, undergoing 45 real-world tests that encompassed over 150 instances of object grasping. This extensive validation underscores the system's effectiveness and showcases the practical application of the proposed neural networks in robotic object manipulation.

## 1.4 Thesis structure

The remainder of this thesis has been organized into several chapters as follows:

1. **Chapter 2** reviews related work, identifying the research gap this thesis addresses.

2. **Chapter 3** discusses in detail the primitive skills necessary for robotic manipulation such as Slide-to-Edge, Push-to-Vertical, Flip, Push-to-Horizontal, and Simple-Pick.

3. **Chapter 4** details the experimental setup, including the real-world and simulation environments, and the object set used.

4. **Chapter 5** introduces the system pipeline, describing the integration of components and processes.

5. **Chapter 6** covers the development and evaluation of learning models, including dataset, skill detection, and skill location modules.

6. **Chapter 7** focuses on detailing the strategies for motion and skill planning, outlining how skills are executed and how robot motion is planned.

7. **Chapter 8** presents experiments, results, and discussions on known and unknown object manipulations.

8. **Chapter 9** concludes the thesis by summarizing the findings and reflecting on the insights gained throughout the study. It also discusses the future work, highlighting areas for further investigation.

# Chapter 2

# Related work

In this discussion, we focus on leveraging dexterous picking strategies that utilize various human-inspired hand motions. Unlike many existing works that emphasize "static" grasping strategies, we highlight the novelty of our approach in comparison to other dexterous picking methods. We aim to create a system that not only recognizes different objects in multi-object scenarios but also intelligently selects and executes the most suitable dexterous picking strategy. Our approach targets the automatic identification of skills and their application in multi-object environments, significantly advancing the adaptability and effectiveness of robotic handling in complex and varied environments.

Various studies in the literature have concentrated on motion planning and execution of dexterous manipulation skills. For instance, [4] presents methods for specific skills like slide-to-edge, push-to-vertical, and push-to-horizontal. This paper delves into the concept of exploiting environmental constraints to enhance robotic grasping

performance, drawing inspiration from human grasping strategies. Through a series of experiments involving both human and robotic grasping, the study demonstrates the effectiveness of leveraging environmental constraints for successful grasps. The introduction of mechanically compliant and highly deformable hands designed to facilitate constraint exploitation further underscores the importance of this approach in robotic manipulation tasks. By comparing different grasping strategies and emphasizing the significance of interactions between the hand, object, and environment, the research sheds light on the crucial role of constraint exploitation in developing competent robotic grasping systems. Additionally, insights from experiments on how humans adapt their grasping strategies to leverage environmental constraints, particularly support surfaces, provide valuable guidance for enhancing robotic grasping capabilities. Another study in [11] implements the flipping skill. Notably, these works focus on single-object scenarios and lack mechanisms for automatically identifying suitable skills for different objects. In contrast, the approach proposed in this thesis targets the automatic identification of skills and their application in multi-object environments. Moreover, the above-mentioned works have specific hardware constraints limiting them to performing certain skills, however, with the use of the three-fingered gripper, the proposed system becomes more generalizable. In addition, our approach also handles occlusion, for example, scenarios where objects are placed on top of each other, and the robot is unable to see all the objects at once.

As described in the paper [15], anthropomorphic soft hands provide capabilities for grasping that closely mimic human actions. This research leverages data from human grasping demonstrations and uses a deep neural network to predict grasp con-

figurations effectively. The deep neural network (DNN) architecture is fine-tuned using supervised and reinforcement learning. While these methods show promise, they encounter difficulties in control, coordination, and adaptability to different robotic hands or environments. This highlights the ongoing challenge of developing a comprehensive architecture for real-world adaptability, a challenge our work addresses by focusing on the automatic identification of skills and their application in multi-object environments.

Some existing studies, such as [2] that explores grasp strategies learned from minimal examples, face limitations. These strategies often rely heavily on RGB data, which can be ineffective in varying lighting conditions or with objects lacking color contrast. They lack depth information, as they operate solely on 2D images, hindering their feasibility in real-world applications. These strategies also don't utilize neural network models for class generation, instead relying on knowledge graphs, which have limitations in generalizing to unseen data and are less flexible when dealing with unstructured data, such as raw images and text. Moreover, their experiments are often limited to small spaces and single objects, resulting in limited system transferability. Our approach overcomes these challenges by utilizing depth data along with RGB data and DNN-based models for skill detection.

Another line of work introduces vacuum-gripper systems [17] capable of handling cluttered objects using a multi-affordance approach. The system considers multiple grasp strategies based on object geometry and context, matches real-world images with synthetic training data for robust object recognition and employs efficient mo-

tion planning algorithms. These systems predict different grasping modes, from suction to parallel-jaw grasps, without knowledge of the object's shape or properties. However, despite their adaptability to various object shapes, these systems lack the dexterity required for precise, human-like grasping and are entirely reliant on object location.

Additionally, the role of synthetic data in training deep networks for shape recognition, as proposed in the study [7], underscores the evolving landscape of grasp planning. This paper proposes a method for improving object grasping through shape recognition using synthetic data and deep neural networks. It involves generating synthetic data with diverse primitive shapes, training a deep shape recognition network, and utilizing recognized shapes for grasp planning. While these approaches improve object grasping through shape recognition, they often do not address varying grasping skills, indicating a gap our research seeks to fill by emphasizing the synergy between shape recognition and adaptive grasping skills.

The grasp generation models like GraspNet [10] and DexNet 2.0 [8] represent a significant advancement in robotic manipulation capabilities. GraspNet employs convolutional neural networks to predict grasp quality directly from RGB-D images, enabling it to generate optimal grasp autonomously poses for a wide range of objects and environments. On the other hand, Dex-Net combines geometric analysis, physics-based simulation, and deep learning to compute grasp quality metrics and generate robust grasps. These models primarily output grasps suited for non-dexterous two-fingered grippers, they lack context-specific gripping capabilities. In

contrast to these existing methods, our approach specifically predicts grasp location based on the object geometry and context for particular dexterous picking skills.

While works such as Pinto et al. (2015) [12] propose generating robust grasp labels through self-supervision, their approach yields rectangular grasp configurations, limiting their applicability to parallel jaw grippers. Moreover, they offer generic grasps for each object, which are independent of the grasping strategy or skill. Consequently, employing a grasp generation model of this nature is not an optimal choice for our system, which seeks to provide dexterity and skill-specific grasp locations.

To summarize, numerous existing methods primarily utilize a 2-finger gripper, which lacks dexterity, or focus on improving grasp accuracy using RGB images, or they are designed to handle a single object. Our approach overcomes these limitations by introducing a system capable of executing dexterous picking skills on objects, taking into account the surrounding factors. Our system is capable of grasping objects of various shapes and sizes due to the dexterous 3-finger gripper used. We propose a novel grasping pipeline that efficiently declutters multiple objects placed on a tabletop, regardless of their position and orientation, by automatically identifying the skills needed and the locations for grasping all objects using RGB-D images.

# Chapter 3

# Primitive skills

Our proposed approach is based on the hypothesis that commonly encountered household objects can be categorized into one or more primitive grasp skills. After a comprehensive analysis of several household object datasets [1], [3] we have identified five essential skills that underpin the manipulation of most household items. These skills are adept at utilizing the physical attributes of the environment, such as surfaces and edges, to facilitate not just the grasping of objects but also the decluttering of densely populated areas, thereby enhancing the overall efficiency of the robotic system in real-world settings. These skills form the foundation of our grasp strategy and enable our robotic system to handle a wide range of objects and scenarios with finesse.

## 3.1 Slide-to-Edge

Developed based on the ideas presented in [4], the Slide-to-Edge technique is particularly adept at handling objects with flat surfaces. It involves the robotic hand gliding along the edge of items such as plates or books to secure a firm grip ( Fig. 3.1). By utilizing a surface and an edge feature in the environment, robotic hands can effectively grasp objects like plates or books. This method leverages both the flat surface of the object and an edge, allowing the hand to snugly fit around any protruding parts of the object by sliding it to the edge of the table.

Example objects: plates, books.


Figure 3.1: Items demonstrative of the Slide-to-edge and corresponding representation diagram.

## 3.2 Push-to-Vertical

As outlined in [4], the Push-to-Vertical method is crucial for managing items located near or against vertical surfaces ( Fig. 3.2). The approach consists of gently nudging objects into corners or against walls, bins, or other vertical structures. This action

uses the vertical surface as a helpful constraint, making it easier for the robot to get a good grip on objects that are standing up or leaning against something. It's a skill that significantly aids in grasping objects in positions that might otherwise be challenging to handle.

Example objects: bottle, banana.

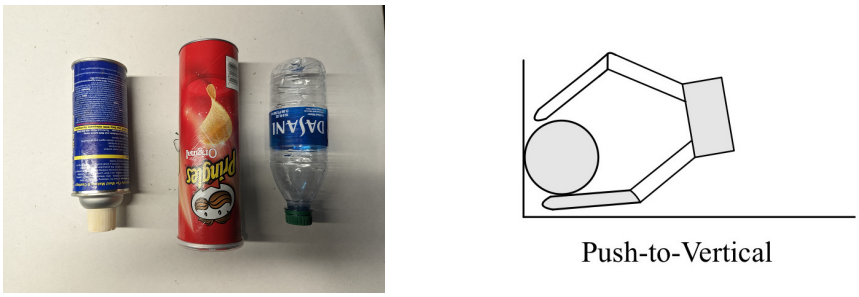
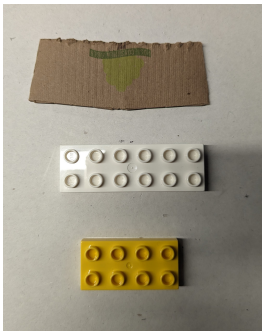



Figure 3.2: Items demonstrative of the Push-to-vertical and corresponding representation diagram.

## 3.3 Flip

This skill, inspired by insights from [11], is used for picking up small objects. It enables our system to manipulate and lift small items effectively, expanding its versatility across a range of object sizes. This skill involves a maneuver where one finger provides support on one side of the object, while another finger gently sweeps over the surface ( Fig. 3.3). This coordinated action allows the robot to make contact with the object, securely getting underneath it, and then lifting it into a graspable position.

Example: lego brick, coin.



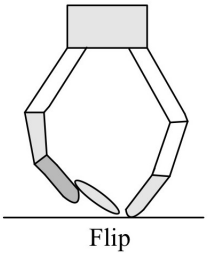



Figure 3.3: Items demonstrative of the Flip and corresponding representation diagram.

## 3.4 Push-to-Horizontal

As detailed in [4], this skill is particularly beneficial for grasping mid-sized objects. This technique likely involves manipulating objects against horizontal surfaces or constraints to optimize the grasp configuration ( Fig. 3.4). The rationale behind this approach is to mitigate the risk of misalignment; for instance, when attempting to grasp a ball, premature contact by one finger could inadvertently displace the object, leading to a failed grasp. By ensuring the fingers sweep the surface first, they collectively guide themselves around the object under a uniform motion, thus compensating for any potential discrepancies caused by measurement inaccuracies.

Example objects: ball, plum

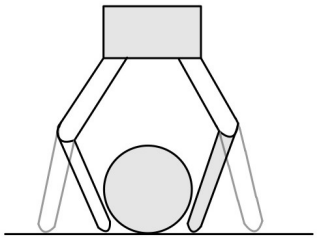



Figure 3.4: Items demonstrative of the Push-to-horizontal and corresponding representation diagram.

## 3.5 Simple-Pick

The Simple-Pick skill is characterized by its straightforward approach to grasping, wherein the robotic gripper directly approaches along the principal access and secures the object without necessitating complex adjustments or pre-manipulation strategies ( Fig. 3.5). This skill can be used to pick any object regardless of size, shape, and texture. This is used as the default skill if any of the other skills are not possible. Example objects: cup, small box.


Figure 3.5: Items demonstrative of the Simple-pick and corresponding representation diagram.

# Chapter 4

# Setup

## 4.1 Physical Environment

### 4.1.1 System

The research conducted in this thesis utilizes the Franka Emika Panda arm, a state-of-the-art 7-axis robotic manipulator known for its precision and versatility. With a payload capacity of 3 kg and an extended reach of 850 mm, the Panda arm (Fig. 4.1) is used for a wide range of applications. It boasts an impressive repeatability of 0.1 mm, ensuring highly accurate operations. For our experiments, the Panda arm used is mounted on a table of dimensions 91 x 61 cm.

Given the reach constraints of the Panda arm, our experimental setup is confined to a workspace measuring 61 x 61 cm in dimensions. This arrangement ensures that all tasks are within the robot's operational capacity. A critical component of our setup is a cardboard wall, strategically positioned on one side of the table. This

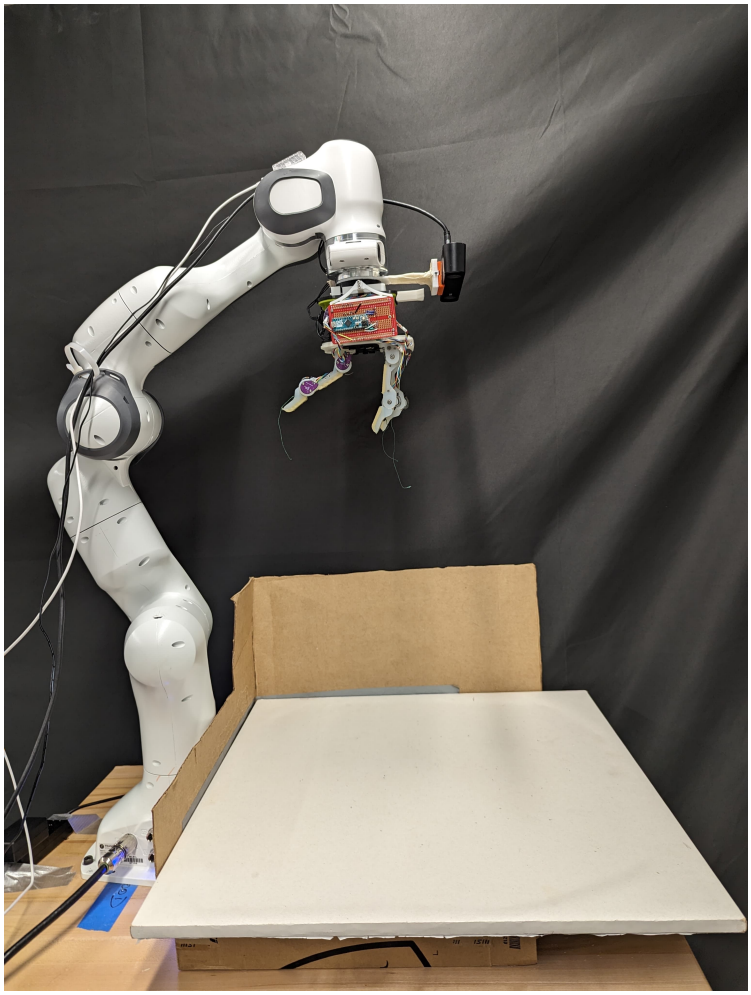


Figure 4.1: Franka Panda Emika arm with the environment setup

wall is integral to our experiments, serving as an environmental feature for testing certain manipulation skill that requires interaction with vertical surfaces. Additionally, the table's surface is covered with white card paper to maintain consistency in the visual background, aiding in the robot's perception and interaction with objects placed within its workspace. This carefully designed setup provides a controlled environment that is conducive to examining the Panda arm's capabilities in executing

the specialized skills outlined in this work.

Our setup includes a ZED camera mounted on the end-effector through an attachment mechanism (Fig. 4.2), positioned precisely 10 cm away as illustrated in the accompanying figure. The ZED 2i camera features a 120mm baseline distance providing accurate depth information at greater distances and features an operational range of 0.3m to 20m. The camera also offers an ultra-wide field of view, up to 110 degrees, enabling it to capture a broad area of the scene in front of it. Another key reason for choosing the ZED 2i camera is that the ZED SDK provides a rich set of software tools and APIs for depth sensing and easy integration with ROS.

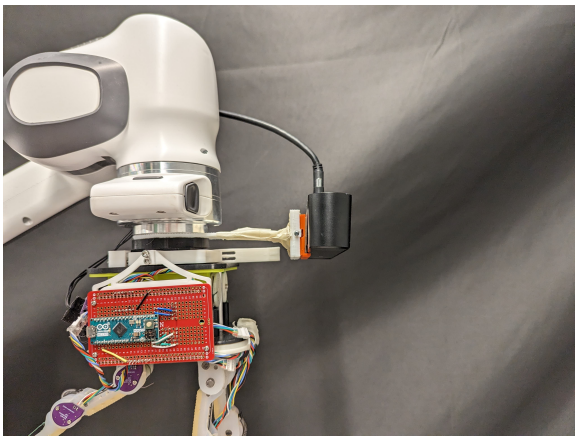


Figure 4.2: ZED2i camera attached to the end effector

The alignment of the camera's pixel frame with the robot's frame was conducted using the following parameters,

- Camera Intrinsics: Utilizing specifications provided by ZED, we used the camera's intrinsic parameters, which include focal length, optical center, and distortion coefficients.

- Camera Extrinsics: The spatial relationship between the camera and the robot's coordinate system was determined using ROS's TF2 package, establishing the camera's position and orientation relative to the robot.

We initially defined the transformation between the robot frame and the camera frame using a tf publisher, which was developed based on measured real-world dimensions. Once we obtain the rotation in the Euler angles format, we convert it from Euler to quaternion representation. This quaternion rotation along with the translation is used to transform 3D points from the camera frame to the robot frame. This process ensured the precise definition of the spatial relationship between the camera and the robotic arm. To confirm the accuracy of this transformation, we employed Rviz, a 3D visualization tool for ROS. Through Rviz, we could visually inspect the frames as seen in Fig. 4.3 and validate the rotation and translation parameters specified by the tf publisher.

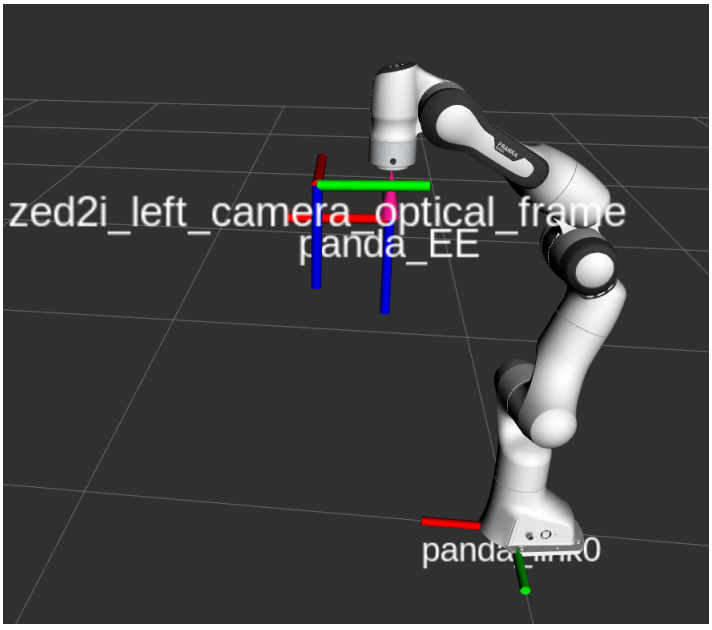



Figure 4.3: Frames used for transformation

### 4.1.2 End-effector

To achieve the desired level of compliance in our robotic grasping system, we have opted for the Yale open-hand Model-O [11] gripper design. This design employs tendon-driven underactuated fingers, allowing our system to conform to object surfaces without the need for complex sensors or feedback systems. This three-fingered gripper is powered by four Dynamixel actuators which provide the precision and control necessary for a wide range of applications.

The underactuated mechanism of the Model O gripper is central to its design, featuring a tendon-driven system that simplifies the control complexity. This system allows for passive adaptation of the fingers to the shape of the object being grasped, driven by a single actuator that controls the closing and opening of the gripper. The

tendons, made of durable material Nylon, are routed through the fingers in such a way that the applied force is evenly distributed, enabling the fingers to wrap around objects with varied geometries.

The choice of the Yale open-hand Model-O gripper, with its tendon-driven underactuated fingers, is pivotal. This design allows the gripper to conform to the surfaces of various objects passively, eliminating the need for complex sensors or control systems. The underactuated mechanism ensures that the gripper can adapt to the object's shape, facilitating a more secure grasp by evenly distributing the force applied by the tendons through the fingers. Compliance allows the system to be more adaptable to the unpredictable nature of real-world environments. It enables the system to accommodate uncertainties in object shape, size, and positioning, enhancing the robot's ability to perform tasks with a higher degree of finesse and reliability.

The Yale OpenHand Model O gripper is seamlessly integrated into robotic systems through a custom attachment mechanism ( Fig. 4.2), specifically designed and 3D printed for this purpose.

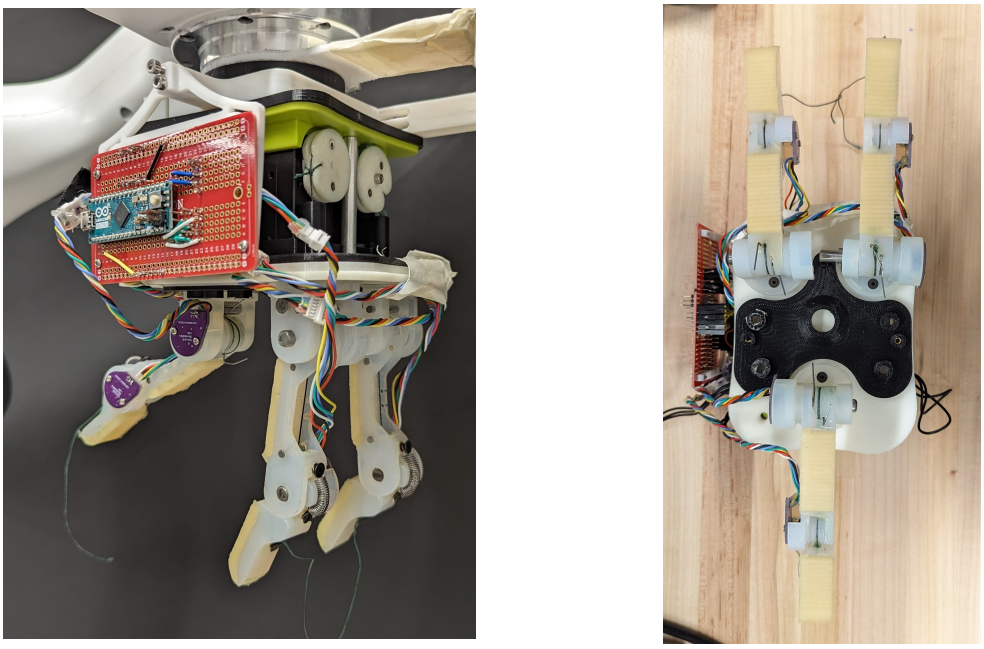


Figure 4.4: Yale OpenHand gripper in closed (left) and open (right) positions

## 4.2 Software Environment

### 4.2.1 Real-world

In our system, we employ ROS Noetic in Ubuntu 20 alongside Python 3.7. For gripper control, we integrate 'openhand' libraries, detailed later in 7. The PyZED library, a Python wrapper for the SDK provided by Stereolabs for their ZED camera, is also used for camera support. Additionally, specialized libraries for neural network models and motion planning, are used which are detailed in the subsequent sections. The version control is performed through Git, and all the data is saved on a network drive, to access later.

### 4.2.2 Simulation

In our simulation setup ( Fig. 4.5), we have replicated the real-world configuration, incorporating elements such as the camera, the robotic arm, and the table to create an environment that closely resembles our physical experimental space. Given the unavailability of direct ZED 2i camera integration within ROS, we have carefully simulated its capabilities by matching key camera parameters, including baseline distance, field of view (FOV), intrinsic parameters, and the depth range from minimum to maximum. This ensures that our virtual setup accurately reflects the visual processing capabilities present in our real-world experiments. Additionally, we have employed 3D models from the YCB dataset, which provides us with high-fidelity representations of various objects. The simulation environment, built within Gazebo, is designed to be dynamic, allowing for the specification of the number of objects per scene as well as the total number of scenes. Objects are introduced into the simulation through the use of the spawn_model_service, enabling us to systematically evaluate our system's performance across a broad spectrum of scenarios with varying complexity and object arrangements.

## 4.3 Object set

Our experimental setup encompasses a collection of 20 objects, ranging from soft and metallic to hard and transparent materials. The majority of these objects are selected from the YCB dataset [1] ( Fig. 4.6), complemented by several common household items that we found interesting in terms of shape, appearance, or texture.

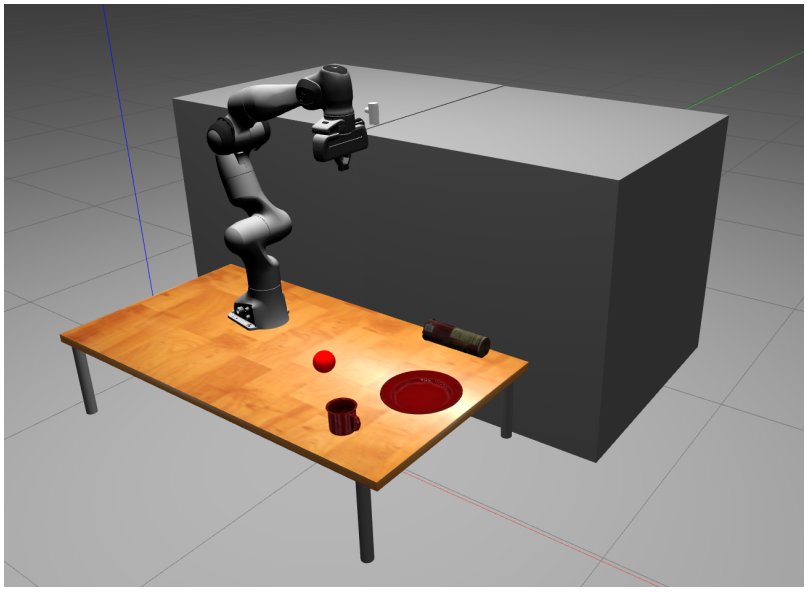


Figure 4.5: Simulation setup

This selection is intentionally varied in terms of shape and size to broaden the scope of our experiments and challenge the capabilities of our system across a wide range of manipulation scenarios ( Fig. 4.7). Table 4.1 provides the list of objects and the expected skill that should be executed for that particular object. Further information on the dataset and its utilization in model training is provided in Section 6.1.

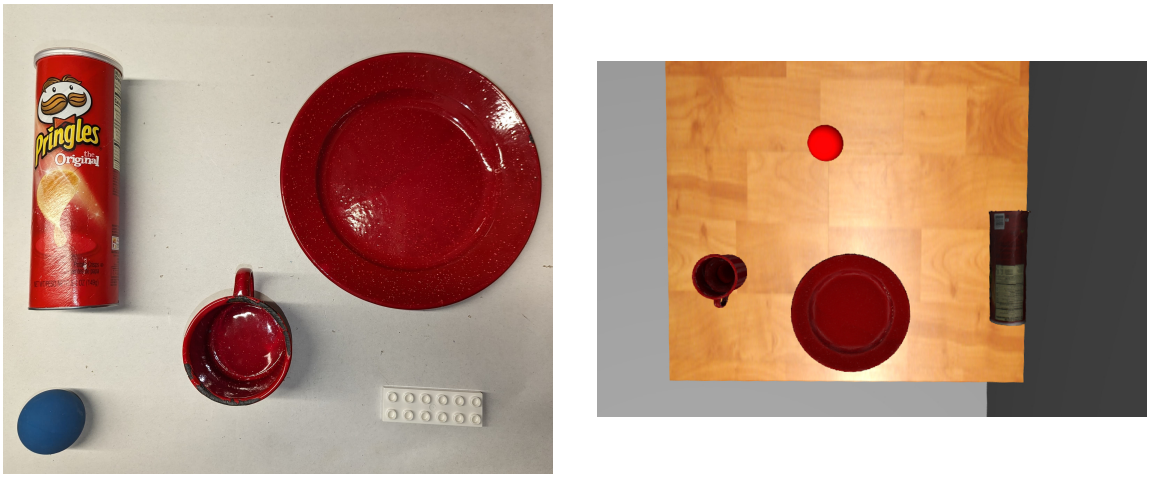



Figure 4.6: Known object set used for training models in real-world(left) and simulation(right).


Figure 4.7: The five objects on the left are "known" objects used in the model training process. The other fifteen objects on the right are "unknown" objects not included in the dataset

| **Object** | **Expected Skill to be used** | **Category** |
|---|---|---|
| Pringles can | Push-to-Vertical, Simple-pick | Known |
| Cup | Simple-pick | Known |
| Small ball | Push-to-horizontal | Known |
| Lego brick | Flip | Known |
| Plate | Slide-to-edge | Known |
| Spray can | Push-to-Vertical | Unknown |
| Dasani transparent bottle | Push-to-Vertical, Simple-pick | Unknown |
| Glove | Simple-pick | Unknown |
| Banana | Push-to-Vertical, Simple-pick | Unknown |
| Plum | Push-to-horizontal | Unknown |
| Orange | Simple-pick | Unknown |
| Paper plate | Slide-to-edge | Unknown |
| Umbrella | Push-to-Vertical, Simple-pick | Unknown |
| Tape | Push-to-horizontal | Unknown |
| White cup | Simple-pick | Unknown |
| Transparent glasses | Simple-pick | Unknown |
| Paper cup | Simple-pick | Unknown |
| Tablet bottle | Push-to-horizontal | Unknown |
| Blue box | Slide-to-edge | Unknown |
| Yellow lego brick | Flip | Unknown |

Table 4.1: Object Set

# Chapter 5

# System Pipeline

System Pipeline

1: Capture depth image using ZED camera's neural mode to reduce noise and gather scene depth information.
2: Both skill detection and grasp location models subscribe to and process the depth image.
3: Grasp location model generates a heatmap, identifying potential grasp points and selecting the point with the highest value.
4: Skill detection model segments the image into objects and associated skills, selecting the object with the highest skill probability.
5: Verify that the selected grasp point is within the object's mask and calculate the gripper's approach orientation.
6: The transformation module computes the 3D position and depth of the grasp point relative to the camera's and robot's coordinate systems.
7: Integrate this data into the motion planning pipeline to execute the selected skill action.
8: Iterate the process for all objects detected in the depth image.
9: For objects undetectable by the depth sensor (e.g., small or flat objects like coins), run RGB-based models and repeat the process.

The pipeline begins with the Panda arm at a predefined home position, from

which a depth image of the scene is captured by the ZED 2i camera attached. We use the neural depth mode of the ZED 2i, which generates processed stereo-depth images. This depth image is then published on a ROS topic and accessed by subscribing to it by both the skill detection node and the skill location node.

The subsequent step involves detecting the skills required for precise manipulation of each object present in the image. This task is performed by the skill detection node, which outputs masks for each object along with the associated skill, assigns a skill probability per object, and selects the skill with the highest probability. On the other hand, the skill location node generates a heatmap from the depth image, depicting potential grasp points on the objects. The grasp point with the highest value on the heatmap is chosen and verified to ensure it lies within the mask generated by the skill detection module. This 2D grasp point, originally in the image frame, is transformed into a 3D point in the robot frame by the grasp transform node by performing a homogeneous transformation, also considering the point's depth as given by the ZED camera. We utilize Rviz to visualize the raw depth images obtained from the camera, the segmented images with skill labels provided by the skill detection module, and the generated heatmap.

Once the 3D position and orientation of the grasp point are established, the next step is skill execution, achieved by integrating this data with the motion primitive node that generates a goal locations for the robot arm as per the detected skill. The calculated 3D location and orientation are then utilized by MoveIt as the end effector location of the robotic arm. The move group commander then executes the motion and the robotic arm successfully places the object at the designated drop location.

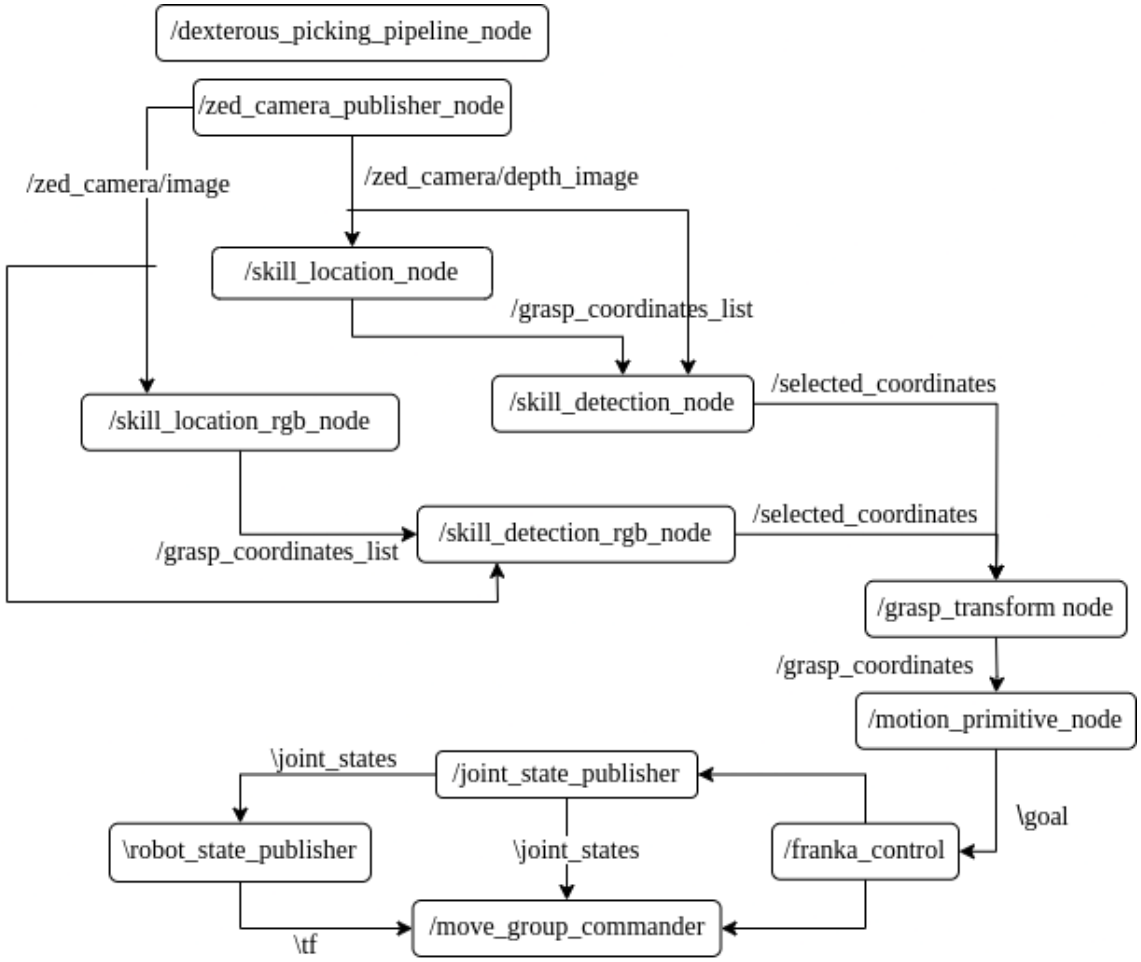



Figure 5.1: ROS pipeline

We repeat this entire process for all objects until the tabletop is cleared.

Nevertheless, there can be objects that are too small and/or too flat to be detected by the depth image (e.g. a lego brick). To address this, we utilize RGB-based models for flat objects instead of depth-based models for both the skill detection and skill location modules, continuing the process as before. Basically, once all the normal sized objects are done, we switch to RGB based detection to check if there are any

small and flat objects on the table and if there are, we just continue with the clearing of the table.

# Chapter 6

# Learning models

In this section, we detail the two deep learning models employed for skill detection and identifying the skill location. We delve into the network architecture, training process, and performance evaluation of the models in depth. Additionally, we touch upon the dataset used for training both models, including the data annotation process.

## 6.1 Dataset

To train our skill detection and skill location models, we compiled a dataset of depth images (top-down view) from 570 diverse multi-object scenes, with over 1,500 object instances. These instances are manually labeled with the appropriate skill label and skill location with all the labeling informed by the context of the scene. For example, when a cylindrical object is flushed toward a vertical wall, it is labeled as push-to-vertical, whereas, when the same object is away from the wall, it is labeled

as simple-pick. To ensure our model develops an understanding of skill applicability, we intentionally included a slightly larger number of instances depicting straightforward contexts for each skill. This approach allows the model to assign higher confidence scores to simpler scenarios during skill classification. For example, we provided slightly more examples of flat objects being at the edge of the table, than the examples when it is in the middle of clutter. This training strategy enables the model to more confidently identify the "slide-to-edge" skill when an object is near the table's edge. For every object within our dataset, we assigned labels indicating the precise location for skill application. Specifically, for the slide-to-edge skill, it is the exact point where the gripper should initiate contact to slide the object. Similarly, for other skills such as push-to-vertical, push-to-horizontal, simple-pick, and flip, it is the point where the gripper should approach to execute the skill successfully. These detailed positional labels are integral to the training of our Skill Location Model, ensuring it learns where to effectively apply each skill.

In our implementation, 70% of the images are from simulation. To bridge the gap between the simulated and actual depth images, we applied techniques that include functions to adjust the table area in images by isolating it through thresholding, reducing image contrast while maintaining detail, and adding artificial smudges to object edges to simulate realistic scenarios. Additionally, Gaussian blur is applied to create a smudging effect, further processing images to include realistic noise patterns. These measures aim to reduce the domain gap between simulated and real-world scenarios, improving the models' ability to adapt and generalize across different settings.

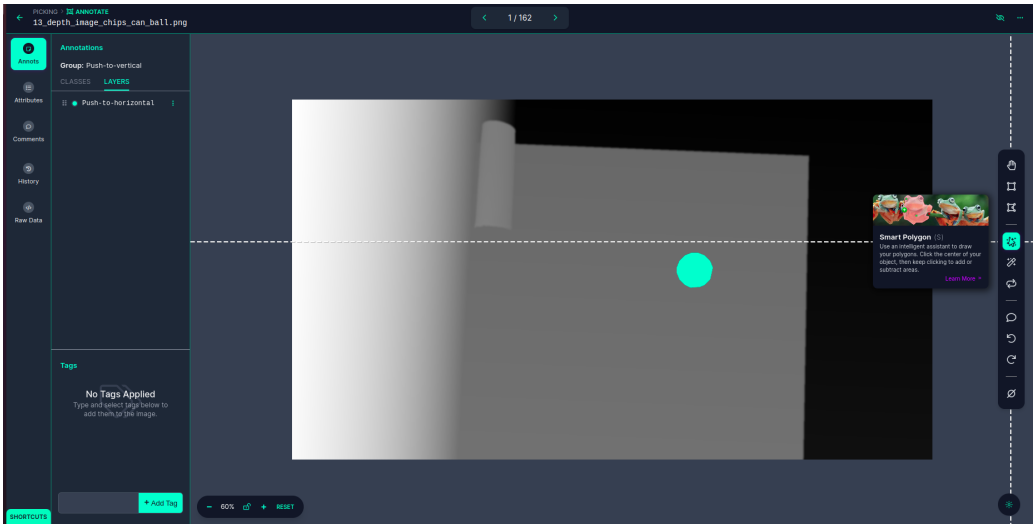



Figure 6.1: Roboflow annotation platform

However, depth data has its limitations, particularly in capturing very small or flat objects due to their minimal depth profiles. Relying solely on depth-based models would result in overlooking these objects with no significant depth signature. To address this, we also compiled a smaller RGB dataset focused exclusively on these left-out small and/or flat objects. This dataset, comprising 80 images, is specifically labeled by concentrating on the slide-to-edge and flip skills (since these are the only two skills suitable for such objects). In contrast, the depth image dataset includes other skills (except flip) and a large variety of objects.

For skill detection model annotations, we utilized the Roboflow platform Fig. 6.1. Roboflow is an end-to-end platform designed to streamline the process of preparing, labeling, and managing computer vision datasets. Roboflow has an intelligent annotation tool with which we can just click on the object in the image and it automatically creates segmentation ground truth labels for the same. The annotations

were formatted in the COCO Segmentation style and utilized JSON files compatible with Detectron2. The annotations for skill location were manually generated using a custom-built tool (Fig. 6.2). This tool allowed for the creation of circular heatmaps around selected points. The use of circular heatmaps is particularly advantageous for skill location tasks as it provides a visual representation of the area of interest with a gradient indicating the probability or confidence of the skill location.

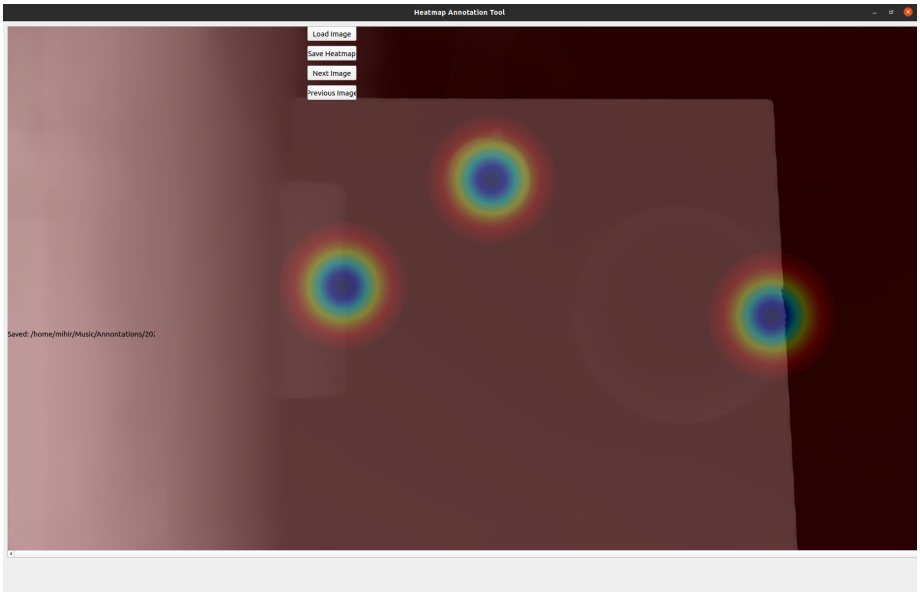



Figure 6.2: Custom annotation tool for skill location

## 6.2 Skill detection module

For the task of categorizing objects into one of five skill categories, we explored several methodologies, including Generative Adversarial Networks (GANs), Convolutional Neural Networks (CNNs), and Reinforcement Learning (RL) approaches.

GANs, while powerful for generative tasks, are not typically utilized for classification problems due to their design for generating new, synthetic data samples rather than classifying them. The goal was straightforward: to classify objects accurately within five predefined skill categories. Given this objective, we concluded that using convolutional neural network architecture was the most appropriate and effective strategy.

CNNs are renowned for their performance in image recognition tasks, offering simplicity in training and debugging, alongside robust generalization capabilities across varied datasets. Conversely, RL methods, though potentially useful in adaptive and interactive environments, tend to produce models that are highly specialized to the specific conditions they were trained for. This specialization limits their applicability and flexibility when faced with new or varied environments, making them less suited for our project's needs. After deciding on leveraging CNNs for our classification problem, the next step involved selecting a specific model that would not only achieve high accuracy but also integrate seamlessly into our workflow. We adopted Detectron2 [16], a MaskRCNN implementation [5], which has a convolutional neural network architecture designed for instance segmentation and enables the identification and delineation of objects at a pixel level within images. It provides a solid foundation with pre-trained baseline models, facilitating a quicker and more efficient training process tailored to our specific classification needs. Our implementation integrates with ROS (Robot Operating System), facilitating real-time interaction. The module subscribes to image topics from a camera, utilizing the cvBridge library to convert ROS image messages into the format suitable for processing with Detectron2.

Upon receiving an image, the system applies the trained convolutional neural network model to detect and classify objects within the scene. The model's predictions are then used to determine the most suitable dexterous manipulation skill.

### 6.2.1 Network architecture

Mask R-CNN [5] evolves from Faster R-CNN by incorporating a parallel branch for instance segmentation and employing a multi-task loss that includes losses for classification, bounding box regression, and mask prediction. The architecture ( Fig. 6.3) encompasses essential elements, including a backbone network, Region Proposal Network (RPN), ROI Alignment, multi-task learning with diverse loss functions, and the ability to adapt to custom tasks through transfer learning.

Mask R-CNN enhances object detection with instance segmentation through a two-step approach:

1. Feature Extraction and Region Proposal Mask R-CNN utilizes a backbone CNN (like ResNet) to extract a comprehensive feature map from the input image. Employs a Region Proposal Network (RPN) to identify potential object locations, generating proposals with anchor boxes of various sizes and aspect ratios. It applies RoI Align to precisely extract fixed-size feature segments from each proposal, avoiding the quantization errors of previous methods.

2. Detection and Segmentation For each proposal, it performs object classification and bounding box regression to identify object types and adjust proposal dimensions. In parallel, a segmentation branch generates pixel-level masks for each object instance by applying a Fully Convolutional Network (FCN), allowing for pre-

cise object outlines.

This architecture integrates object detection and segmentation, improving accuracy through end-to-end training with a multi-task loss that encompasses proposal classification, bounding box refinement, and mask generation.

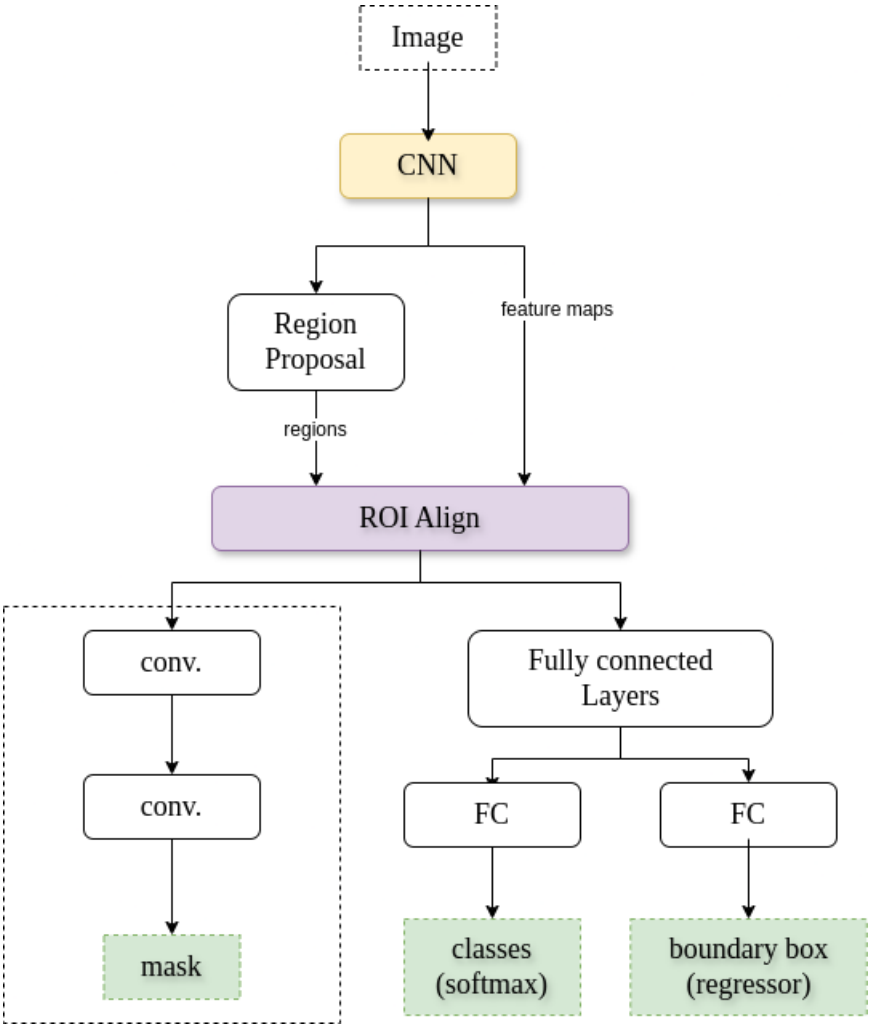



Figure 6.3: Mask R-CNN architecture diagram.

### 6.2.2 Training process

For our model's configuration, we selected the "mask_rcnn_R50_FPN_3x.yaml" from the Detectron2 model zoo, chosen for its strong baseline in instance segmentation, which we further tailored to meet our dataset's specific needs. we set the number of workers for the PyTorch dataloader to two, ensuring efficient data handling without overloading the system. The images processed in each batch are limited to 2, maintaining an optimal balance for batch processing. The model is configured to detect five unique classes (model.roi.heads.numclasses = 5), which correspond to our 4 dexterous skills namely - Push-to-horizontal, Simple-pick, Push-to-vertical, Slide-to-edge and with an additional class designated for instances requiring no skill. Additionally, we have another model trained specifically for handling RGB images, with labels as the Slide-to-edge and Flip skills.

The training process lasts for a maximum of 1000 iterations, a decision based on experimental evaluations to ensure sufficient learning while avoiding overfitting. The use of Detectron2's default trainer class simplifies the initiation and execution of training, offering built-in support for model optimization, checkpointing, and performance assessments. The model requires approximately 30-35 minutes for training on an Nvidia RTX 2080. For optimization, we employ the Stochastic Gradient Descent (SGD) method, setting the learning rate at 0.00025, momentum at 0.9, and a weight decay of 0.0001.

### 6.2.3 Performance evaluation

For assessing the accuracy of the skill detection model, our study employed a dataset comprising 570 images for training and an additional 84 images designated for testing. The evaluation metric for the skill detection model's accuracy involved calculating the Intersection over Union (IOU) with a threshold set at 0.5. Furthermore, the average IOU metric was also presented to offer a comprehensive understanding of the model's performance.

To enhance our skill detection model's evaluation, we used Average Precision (AP) scores alongside accuracy metrics. We achieved a mean Average Precision (mAP) score of 76.076 for segmentation and 68.041 for bounding box detection. AP scores, integral for understanding model precision-recall balance, reflect the model's ability to correctly identify and localize objects. This metric is especially valuable in scenarios where object detection accuracy across varying thresholds is critical.

The choice of Mask R-CNN was motivated by its superior generalization capabilities, attributed to its architecture that effectively combines object detection with pixel-wise mask prediction, ensuring detailed and accurate object segmentation crucial for our application.

Fig. 6.4 shows the input and output image, with the segmentation masks overlayed and associated confidence scores for each object.

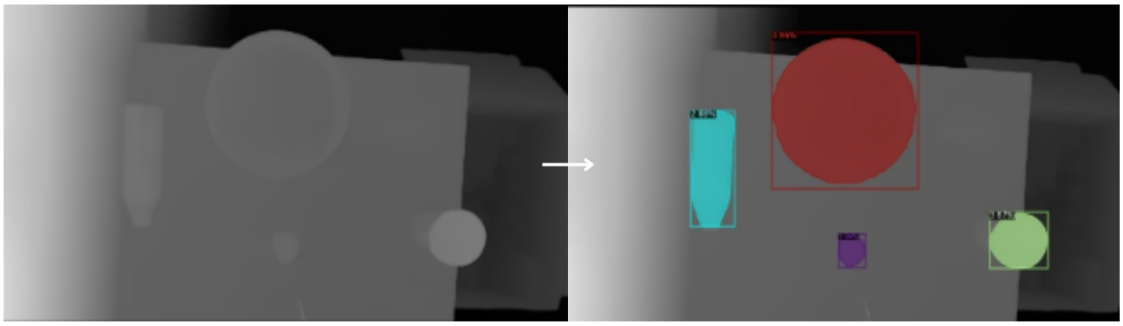


Figure 6.4: The left image is the input, and the right shows segmented objects with confidence scores.

## 6.3 Skill location module

Due to the nature of the grasping skills employed, it becomes essential to pinpoint precisely where each skill should be applied. Existing grasp generation models, such as GGCNN [9], cannot provide accurate grasping locations adapted to our specific skill set. For instance, GGCNN might suggest a center point for grasping a Pringles can, whereas our "Push-to-Vertical" skill necessitates an edge point grasp. Consequently, developing a dedicated grasp generation model that accounts for our specific skills becomes imperative.

To address this gap, we have developed a grasp location model using the U2Net architecture [13], augmented with an attention-gate mechanism. This approach allows for the precise identification of optimal grasp locations by leveraging the attention gate to enhance relevant features and suppress distractions within the encoder and decoder feature maps. The synergy between the attention mechanism and the U2Net architecture ensures a more concentrated feature map, significantly improving the model's ability to generalize to unseen objects. This modified architecture is

depicted in Fig. 6.5 and detailed in following section.

### 6.3.1 Network architecture

The U2-Net architecture employs a deep nested U-structure that integrates Residual U-blocks (RSUs) to effectively capture features. The model includes convolutional layers, batch normalization, and ReLU activation functions, along with specific down-sampling and up-sampling layers to manage the flow of information across different scales. The network's depth and the specific combination of layers enable it to learn detailed features from the input images, making it robust for detecting salient objects with high precision, and generating heatmaps focusing on precise grasp locations on the object. We incorporate an attention-gate mechanism in this architecture to enhance the model's focus on relevant features by fusing encoder and decoder feature maps through the attention gate. It selectively emphasizes target regions while diminishing less relevant information. This enriched feature representation, combined with previous decoder outputs, improves the model's ability to generalize to unseen objects by providing a more detailed and focused feature map for subsequent decoding stages.

### 6.3.2 Training process

The initial step in the training process of the U2-Net model for determining skill location involves data preparation, where a manually labeled dataset is augmented through a series of transformations to standardize the input data. It includes resizing the image to a fixed size of 320 pixels along the shortest side, maintaining the aspect

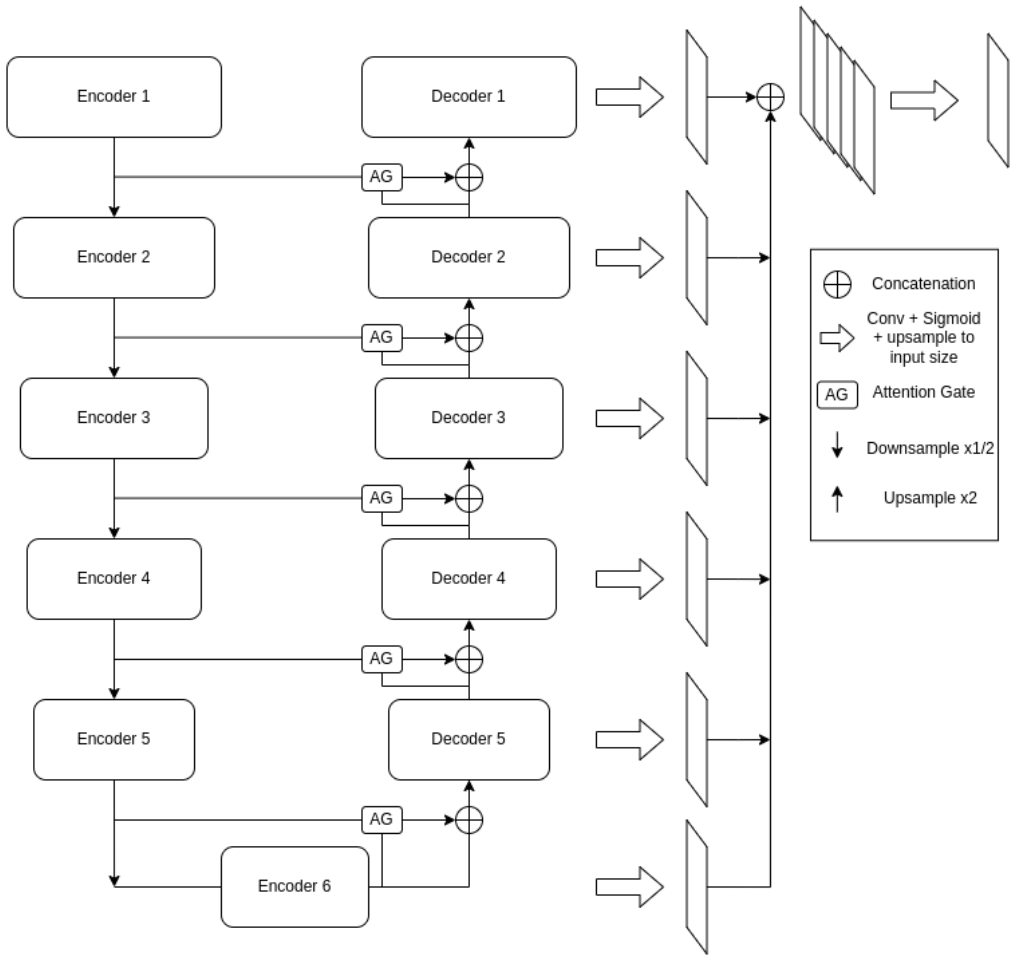



Figure 6.5: Attention gated U2Net architecture diagram

ratio and ensuring uniformity in input size for the network. We then randomly crop a region of 288x288 pixels from the rescaled image to introduce variability in the training data, helping the model to generalize better by learning from different parts of the images.

The training was structured to span over 27 epochs, with a batch size of 16 for training. The training took around 40-45 minutes to complete on the dataset de-

scribed earlier on a Nvidia RTX 2080. We use the Adam optimizer with a learning rate of 0.001, betas set to (0.9, 0.999), epsilon to 1e-08, and a weight decay of 0.0001. These parameters were instrumental in ensuring a steady and effective optimization of the model weights throughout the training process. The training process adopts a combined loss function that integrates the inverse of the Structural Similarity Index (SSIM), along with Intersection over Union (IoU) and Binary Cross Entropy (BCE). This multifaceted loss function is designed to optimize the model's performance by simultaneously considering similarity in image structure, the accuracy of object localization, and the precision of binary classification.

### 6.3.3 Performance evaluation

We employ the accuracy score metric to assess the performance of our skill detection model, while the Mean Absolute Error (MAE) and Intersection over Union (IOU) are utilized to evaluate the skill location. MAE measures the average magnitude of errors in a set of predictions, without considering direction. It's useful for understanding how close the predicted values are to the actual values, on average. IOU, on the other hand, assesses the overlap between predicted and actual object locations, providing insight into the model's localization accuracy. Specifically, the skill location model demonstrates an MAE of 0.027, an accuracy of 92.85%, and an average IOU of 0.66. The U2-Net model was chosen for its proficiency in salient object detection, closely aligning with our heatmap generation needs. Its attention mechanism effectively highlights key image areas, producing reliable heat maps for our application model.

Figure 6.6 presents the output image alongside heatmaps highlighting specific

Figure 6.6: From left to right: Input image, model output, and selected highest-probability grasp location from the heatmap.

regions of interest. From these areas, we identify and select the point with the highest probability. This point is then designated as the coordinate for the skill location. Subsequently, each identified point is correlated with the corresponding skills through the utilization of segmentation masks obtained from the skill detection model.

# Chapter 7

# Motion and skill planning

Skill execution within our system is an intertwined process that integrates programming processes with the mechanical precision of the Panda arm and the adaptive capabilities of the Yale Openhand gripper. Central to this pipeline is the integration of the OpenHand python library, which is designed to manage the intricate movements of servo motors in the gripper. We leverage the OpenHand library's functions such as open(), adduct(), moveMotor(), and power_close() to modulate the hand's posture and the force exerted by its grip, thus accommodating objects of varying shapes, sizes, and textures.

The execution sequence of a particular skill integrates an organized series of steps, starting from the robot's home position—a predefined, standard position that ensures a consistent starting point for task execution. It then involves the robot methodically progressing through a sequence of intermediate stages, each carefully calculated to optimize the robot's interaction with the object according to the specific skill being

executed.

For instance, the execution of the "push-to-vertical" skill (Fig. 7.1) for manipulating a Pringles box involves a strategic progression through four distinct stages:

1. The robot initiates from its home position, a standardized starting state that guarantees the consistency and accuracy of the images captured for skill detection and location.

2. In the second stage, the robot moves to a position near the object. It approaches with a specific orientation, aligning the gripper's two fingers parallel to the table surface and positioning them underneath the object. This careful approach is designed to minimize disruption to the object while preparing for a secure grasp.

3. The subsequent stage involves a deliberate motion where the gripper pushes toward the object. This action allows the two-finger side of the gripper to push the object vertically along the wall, while the third finger provides support from the top. This technique ensures a strong, stable grip around the object.

4. Upon securing the object, the robot then transitions to the final stage. It navigates to a position directly above the bin and releases the object.

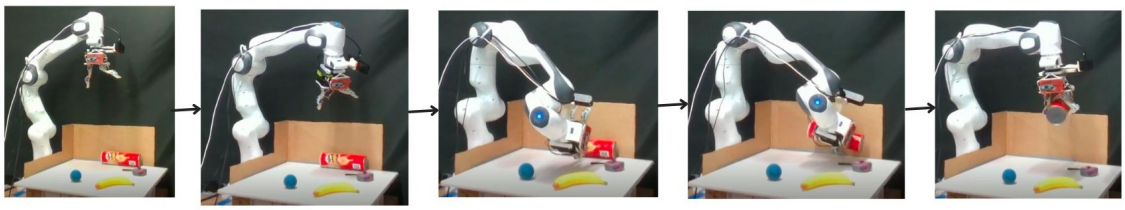


Figure 7.1: Step-wise execution of push-to-vertical skill

The execution of the "Slide-to-Edge" skill (Fig. 7.2) for manipulating a plate is executed in the following 5 steps:

1. The robot initiates from its home position, a standardized starting state that guarantees the consistency and accuracy of the images captured for skill detection and location.

2. The robot then advances to a position where the gripper's third finger makes contact with the plate's designated grasp point from above, while the other two fingers remain in an open position. This configuration is intentionally adopted to stabilize the grasp, eventually positioning the two open fingers under the plate to prepare for a secure lift.

3. With the initial grasp established, the next stage involves the robot sliding the plate across the flat surface of the tabletop. This movement continues until the plate approaches the predefined edge of the table which is a limit set based on the workspace dimensions.

4. Upon arriving at the edge of the table, the robot repositions the end effector by shifting it a small distance in the positive y direction. This adjustment places the two-finger side of the gripper directly beneath the plate, a maneuver pivotal for securing firm support from below before the gripper is engaged.

5. The final stage involves the robot lifting the plate upward along the positive z-axis before securely transporting and releasing it into the bin.

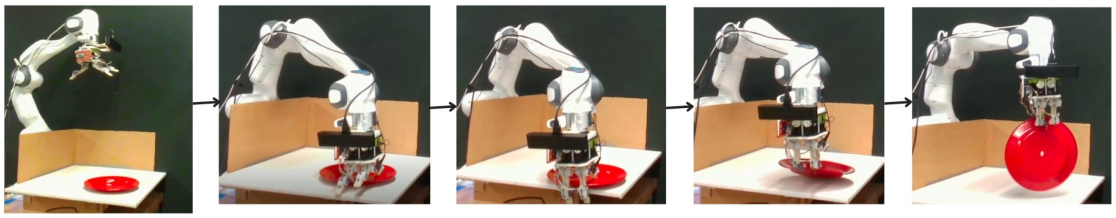


Figure 7.2: Step-wise execution of slide-to-edge skill

The execution of the "Push-to-Horizontal" skill (Fig. 7.3) for manipulating a small object (peach) is executed in the following 4 steps:

1. The robot initiates from its home position, a standardized starting state that guarantees the consistency and accuracy of the images captured for skill detection and location.

2. The robot progresses toward the designated grasp location on the object. Upon arrival, it descends along the negative z-axis by a predetermined measure, ensuring the fingers of the gripper make contact with the tabletop.

3. In the third stage, the robot gently pushes the fingers of the gripper underneath the object. This action is performed with precision to slightly elevate the object from the table surface. The lift is minimal yet significant enough to ensure that all three fingers of the gripper can support the object from below.

4. With the object securely supported by all three fingers from the bottom side, the robot then carefully lifts it off the table. Following the lift, the robot transports the object to safely release it into the bin.

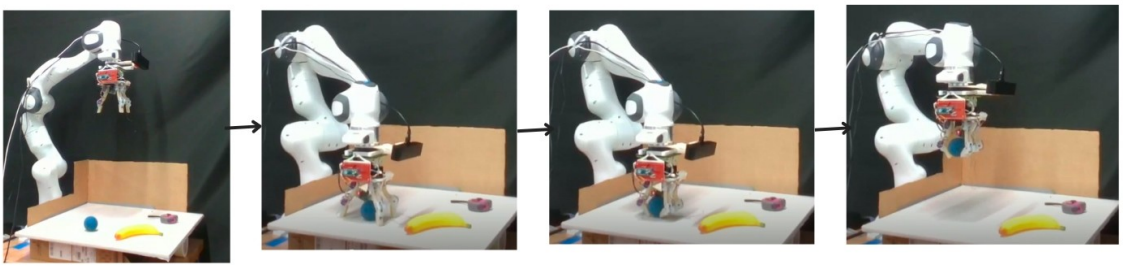


Figure 7.3: Step-wise execution of push-to-horizontal skill

The execution of the "Flip" skill (Fig. 7.4)for manipulating a lego brick is executed in the following 4 steps:

1. The robot starts from its home position and approaches the object, engaging in a preliminary and standard grasp that positions the object between the fingers of the gripper. This initial contact is on the fingertips of the gripper due to the small thickness of the object thus making it unstable.

2. Subsequently, the robot adjusts the single finger of the gripper close further which initiates the flip, carefully reorienting the object inside the gripper. This controlled flip is essential for moving the object from a less secure fingertip grasp to a more stable position within the gripper.

3. With the object securely repositioned and grasped, the robot then proceeds to lift and transport it to the designated location. The careful handling and reorientation through the flip skill mitigate the risk of dropping or improperly handling the object, particularly important for small and flat items.

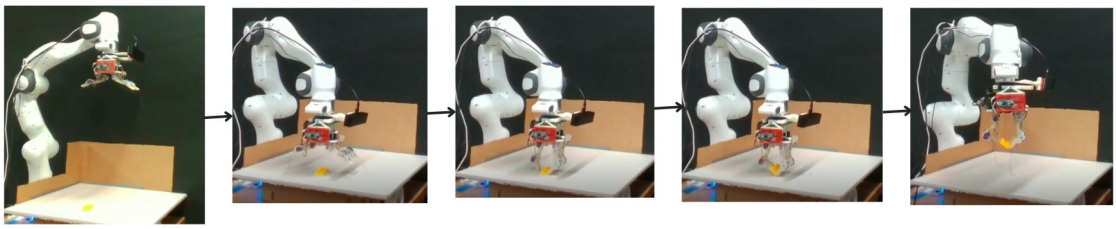


Figure 7.4: Step-wise execution of flip skill

The execution of the "Simple Pick" skill (Fig. 7.5) for manipulating an object is fairly simple and can be used for most of the objects. It is performed in the following 2 steps:

1. This skill performs the standard grasping technique where the robot starts from the home position and achieves the grasp position provided by the skill location module as well the orientation given by the skill detection module.

2. Once the desired position and orientation is achieved, the gripper simply performs the close action and grasps the object. It then moves to its final location above the bin and releases the object.

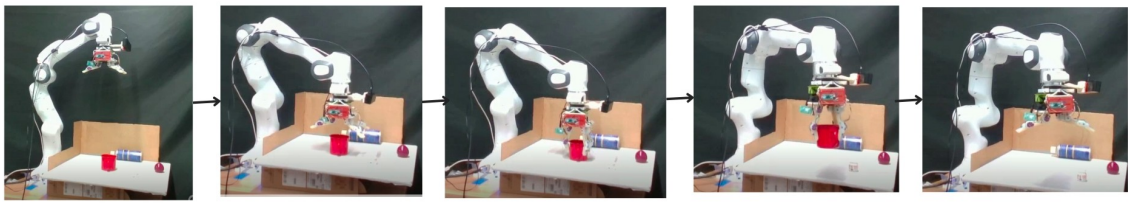


Figure 7.5: Step-wise execution of simple-pick skill

A pivotal aspect of our robotic system's functionality is the adaptive nature of the robotic hand, made possible through the power_close function of the OpenHand

library. This function is particularly significant as it allows the hand to adjust its grip dynamically, conforming to the object's shape and securing it firmly without exerting excessive force that could lead to damage. Through rigorous trial and refinement, the initial values for approaching objects and executing skills have been optimized, ensuring each motion is performed with unparalleled precision and efficiency.

For the motion planning framework of our system, we selected MoveIt for its robust support for planning operations, particularly with the Franka Panda robotic arm. The flexible plugin architecture of MoveIt enables the use of RelaxedIk [14] as the inverse kinematics solver to generate robust IK solutions, along with the Open Motion Planning Library (OMPL) as the default planner, due to its proficiency in handling complex and constrained environments. Integration with the Robot Operating System (ROS) facilitated communication between MoveIt and the robotic arm, while MoveIt's RViz interface was employed for visualizing and debugging motion plans. A significant advantage of using MoveIt is its modularity and scalability support, which simplifies the addition of more sensors or end effectors into the system, thereby accommodating the future incorporation of more dexterous manipulation skills.

To achieve the complexity required for the dexterous manipulation skills, we utilize pose goal planning as well as cartesian path planning. Pose goal planning directs the robot to move to a specified pose, ensuring the arm reaches the desired position and orientation. Cartesian path planning ensures control over the trajectory of the end-effector, which calculates and executes a trajectory through a series of waypoints, allowing for smoother and more controlled motion.

The MotionPrimitive class encapsulates the logic for executing predefined tasks such as picking, sliding, pushing, and flipping objects. This includes dynamic adjustments to the robot's grasp pose informed by real-time object coordinates from ROS topics, enhanced by a service server for flexible execution of these tasks. The system's setup involves configuring default positions and establishing ROS subscribers, publishers, and service servers for managing motion primitives based on dynamic input. Critical motion parameters for tasks like slide-to-edge and push-to-vertical are calculated, considering the direction of movement and distance needed, ensuring precise execution. The direction is set opposite to relevant normal vectors (table edge or wall), with distance $D$ determined by the formula $D = D_{max} - d_{sub}$, where $d_{sub}$ is the object's distance from the workspace limit and $D_{max}$ is the maximum distance the robot can move within the workspace, thus enabling smooth and collision-free movements.

# Chapter 8

# Experiments, results and discussion

To assess our proposed architecture, we conducted a series of table-top object grasping experiments with 25 distinct scenarios, as detailed in the earlier section. These experiments comprised approximately 100 grasping trials. Our evaluation encompassed a wide range of objects, including both known and unknown items. Our real-world experiments were designed to rigorously assess the robustness and adaptability of our algorithm in various scenarios. These experiments include:

- Known Objects Placed Randomly: In this experiment, we placed 4 known objects randomly on the table, to evaluate the algorithm's capability to handle unpredictable object placements.
- Objects with Multiple Skills: Some of the objects were configured to possess multiple graspable skills. This experiment aimed to assess the algorithm's

ability to discern and execute the appropriate skill with the highest probability for each object.

- Handling More than 4 Objects: To challenge the algorithm's capacity, we introduced more than five objects into the environment, replicating cluttered scenarios that often arise in real-world applications.

- Unknown Objects: We introduced over 15 completely unknown objects, absent from the training dataset, to gauge the algorithm's adaptability and generalization capabilities.

- Occluded Objects: Occlusion is a common occurrence in cluttered environments. We incorporated occluded objects to test the algorithm's ability to handle partially visible objects.

## 8.1 Category specific results

The table 8.1 presents the results of various experiments focused on grasping objects under different conditions. Firstly, in the "Known" category, where objects were familiar to the system, the success rate was remarkably high which is as expected, with all trials resulting in successful grasps and complete clearance of the table, indicating a robust performance in familiar environments. Moving on to the "Known + Unknown" scenario, which involved a mix of familiar and unfamiliar objects, the success rate slightly decreased to 90.83%. Despite encountering unfamiliar objects, the system still demonstrated a commendable performance, clearing the table in

the majority of attempts. Similarly, in the "Unknown" category, where all objects were unfamiliar, the success rate remained high at 93.9%. This suggests the system's adaptability and capability to handle novel objects efficiently. Additionally, in the "Occluded" experiment, where objects were partially obscured, the success rate remained high at 91.6%, indicating the system's ability to cope with challenging scenarios. For instance, when faced with scenarios like a ball placed atop a plate, our algorithm intelligently prioritized grasping the ball first. This decision-making helped prevent potential mishaps, such as dropping the ball, making the chosen sequence logical. Overall, the results demonstrate the system's robustness across various experimental conditions, showcasing its effectiveness in grasping objects reliably, even in unfamiliar or partially obscured settings.

Table 8.1: Category-Specific Grasping Results

| **Category** | **Successful Trials** | **Average % of table cleared** |
|---|---|---|
| Known | 5 / 5 | 100% |
| Known + Unknown | 16 / 20 | 90.83% |
| Unknown | 15 / 20 | 93.9% |
| Occluded | 8 / 10 | 91.6% |

## 8.2 Table clearing

Table 8.2 provides an overview of the overall success rates in clearing the table across different numbers of objects. The results demonstrate a generally high level of success across varying object counts. For trials involving two objects, the success rate stands at 80%, with an average of 1.6 objects grasped per trial. As the number of objects increases, the success rates remain consistently high, indicating the system's ability to effectively handle more complex scenarios. Specifically, for trials with three objects, the success rate is 86.67%, with an average of 2.86 objects grasped per trial. Similarly, for trials with four objects, the success rate is 76.47%, with an average of 3.76 objects grasped. Even with five objects, the system achieves a success rate of 80%, grasping an average of 4.8 objects per trial. As the complexity further increases, with six and seven objects, the system maintains a high success rate, albeit with fewer trials conducted in these categories. These results underscore the system's robust performance across a range of scenarios, showcasing its ability to efficiently clear the table even as the number of objects and the complexity of the task increase.

Table 8.2: Overall table clearing success rates

| **Number of objects** | **Successful Trials / Total # of Trials** | **Average Grasped Objects Count** |
|---|---|---|
| 2 | 4 / 5 | 1.6 |
| 3 | 13 / 15 | 2.86 |
| 4 | 13 / 17 | 3.76 |
| 5 | 4 / 5 | 4.8 |
| 6 | 1 / 2 | 5.5 |
| 7 | 1 / 1 | 7 |

## 8.3 Skill specific

The table 8.3 presents the success rates and the nature of failures associated with various grasping skills employed by the system. Note that we allowed for one retry if a skill fails for the first time and the object is still on the table. We did not do anything special for this retry mechanism; since the pipeline works cyclically, it captures a new image of the scene and runs the process again. The success rates reported in Table 8.3 also reflect those retries. Among these skills, "Push-to-Vertical" demonstrates a commendable success rate of 92%, albeit encountering occasional instances of object slippage during grasping. These occurrences of object slippage are primarily attributed to insufficient force applied during the grasping process, a challenge that could potentially be addressed in the future by implementing force-feedback mech-

anisms for the robot's fingers. In contrast, "Push-to-Horizontal" exhibits a slightly lower success rate of 85.29%, primarily due to occasional misidentification of the appropriate skill and instances of object slippage during grasping. The common reason of mis-identification seems due to it being detected as Simple-pick and can be solved by retraining the model by adding more data for appropriate skills. "Slide-to-Edge" shows a success rate of 66.66%, with failures primarily attributed to challenges in determining the optimal skill location and unachievable robot poses. We have also observed that in some cases, even though the sliding action was successful, the robot was unable to complete the pick due to the execution of the grasping motion. Specifically for slide-to-edge, 10 out of 26 picks failed in the first attempt. However, in 6 of the 10 cases, the system was successful in the second attempt. "Simple-Pick" achieves a high success rate of 92.5% but faces occasional failures caused by environmental interference. Lastly, the "Flip" skill demonstrates a success rate of 71.88%, with failures primarily stemming from difficulties in determining the optimal skill location and encountering objects that are too small for the technique to be effective. Given that objects targeted for flipping are typically small, even minor deviations in the identified skill location can lead to unsuccessful attempts, necessitating retries. Overall, these results provide valuable insights into the performance and limitations of each grasping skill, highlighting areas for improvement to enhance the system's overall efficiency and reliability.

Table 8.3: Overall Grasping Success Rates and Skill-Specific Breakdown

| **Skills** | **Success Rate** | **Algorithmic failures** | **Mechanical Failures** |
|---|---|---|---|
| Push-to-Vertical | 92% | - | Object slipped during grasping |
| Push-to-Horizontal | 85.29% | Wrong skill detected | Object slipped during grasping. |
| Slide-to-Edge | 66.66% | Optimal skill location not found. | Unachievable robot pose |
| Simple-Pick | 92.5% | - | Environmental interference. |
| Flip | 71.88% | Optimal skill location not found. | Object too small |

## 8.4 Discussing individual experiments

Experiment 41 ( Fig. 8.1) highlights our system's capability to prioritize objects and the order in which they are handled to achieve efficient decluttering. Initially, the system opts to pick up the plum using the push-to-horizontal skill, then proceeds to clear the white cup and the transparent glasses using the simple pick skill. This strategy ensures that the table surface is clear and free of obstacles when executing

the push-to-vertical maneuver for the spray can at the last. Another notable observation from this experiment is the robot's decision-making process when faced with two objects (the white cup and transparent glasses) for simple pick. It prioritizes the white cup first, simplifying the subsequent pickup of the glasses by taking into account the depth constraints of both items.

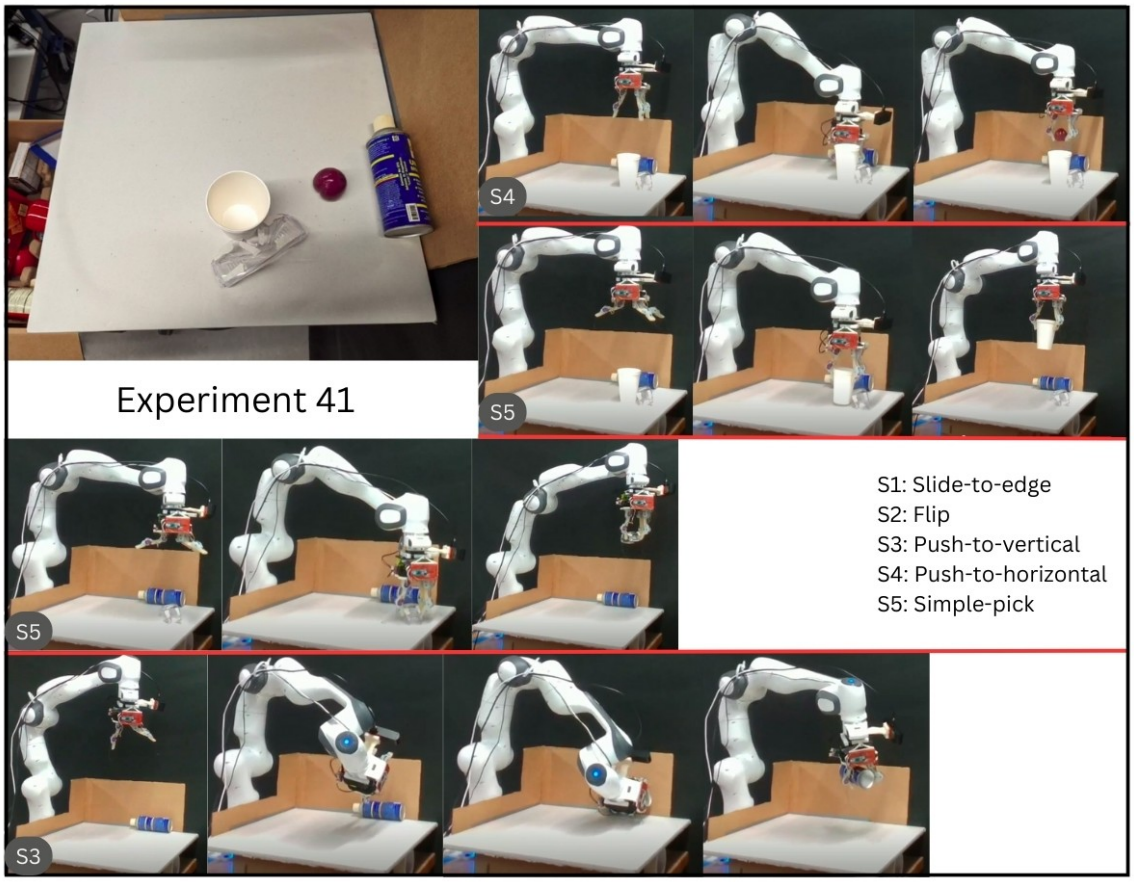



Figure 8.1: Step-wise execution of Experiment 41

In Experiment 45 ( Fig. 8.2), our system navigates a setup with a plate occluded by red and white cups. It starts by removing the plum with a push-to-horizontal skill,

essential for clearing the path for the slide-to-edge technique to retrieve the plate. Subsequently, it removes the two cups using simple pick in an optimal sequence, then applies the slide-to-edge technique for the plate. As seen, it also picks up the white cup which is a taller object, hence not obstructing the pick for cup as well. The system reserves the lego brick for last, which does not obstruct the plate, and clears it using the flip skill, utilizing the RGB model for skill detection. This demonstrates efficient clutter clearance and discerning obstacle prioritization based on their impact on task execution.

Experiment 16 ( Fig. 8.3) primarily showcases the system's ability for contextual skill selection, i.e selecting skills based on the constraints and context of the environment. Firstly, the system prioritizes unobstructed objects i.e. the white cup and blue ball through simple pick and push-to-horizontal skills respectively. The decision to use a simple pick for the pringles box, instead of push-to-vertical taking into account the the fartherness of a wall and the plate underneath. This variation in skill selection, influenced by the object's location, underscores the system's capability to adjust strategies based on context for enhanced efficiency. After clearing objects using the depth image, the system lastly employs the RGB model to identify and execute the flip skill on the lego brick, which the depth-based model could not detect due to its small size and shape. This experiment also demonstrates the integration of depth and RGB models and the ability to dynamically switch between them based on the object parameters.

Experiment 42 ( Fig. 8.4) addresses a failure in our system where a lego brick is placed inside a plate. Initially, the robot removes the blue ball using the push-

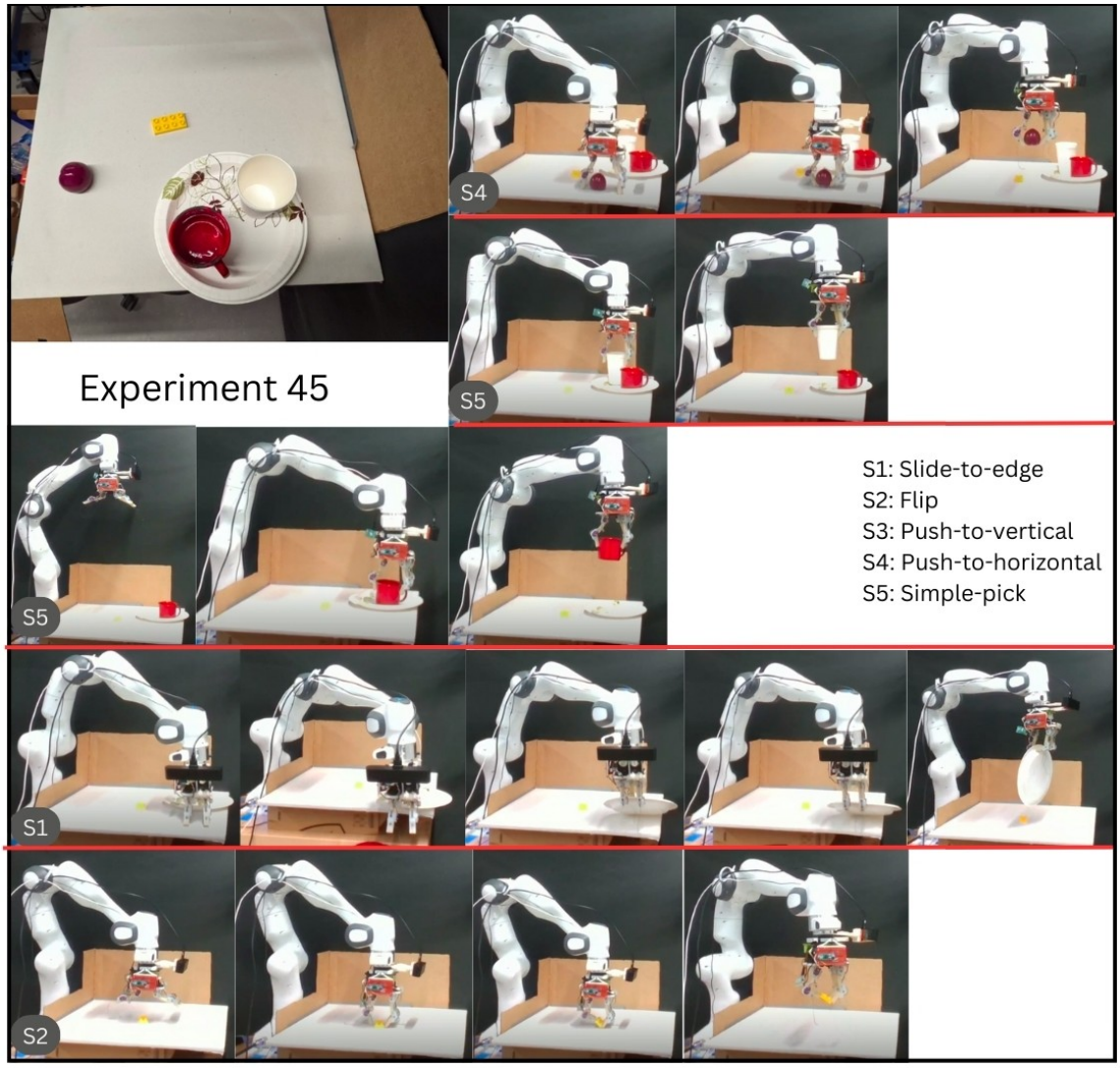



Figure 8.2: Step-wise execution of Experiment 45

to-horizontal skill but then fails to clear the lego brick before directly attempting to use the slide-to-edge skill on the plate. This oversight results in the lego brick being inadvertently dragged along and eventually falling off the plate during the slide-to-edge execution. Fortunately, in this case, the brick lands on the table surface and is

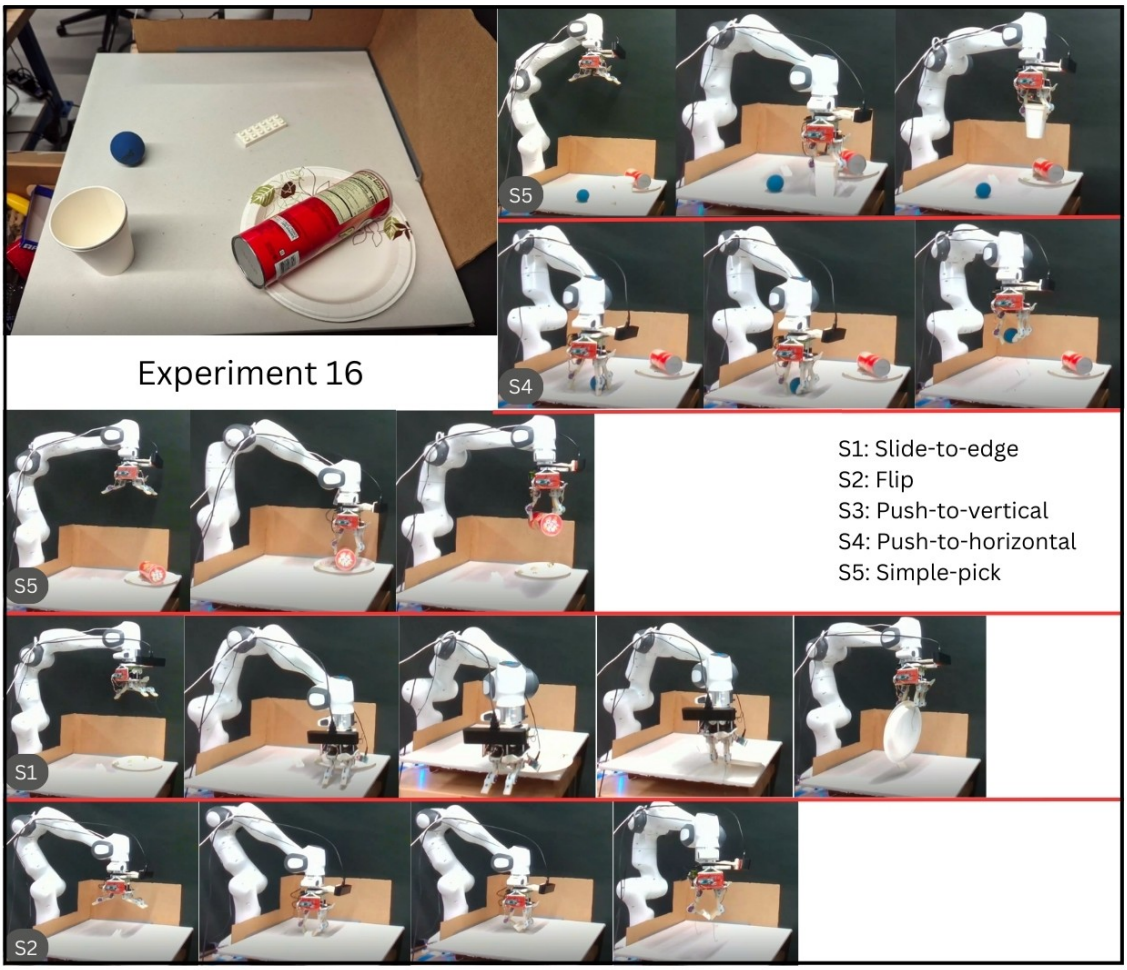



Figure 8.3: Step-wise execution of Experiment 16

subsequently picked up by the robot using the flip skill. Although this experiment highlighted a limitation, it is important for identifying specific scenarios that can lead to unexpected system behaviors and the need of a unified model.

These experiments collectively underscore our system's sophisticated manipulation capabilities, from strategizing in occluded settings to adapting skill selections based on contextual nuances. The system's intelligent decision-making, coupled with

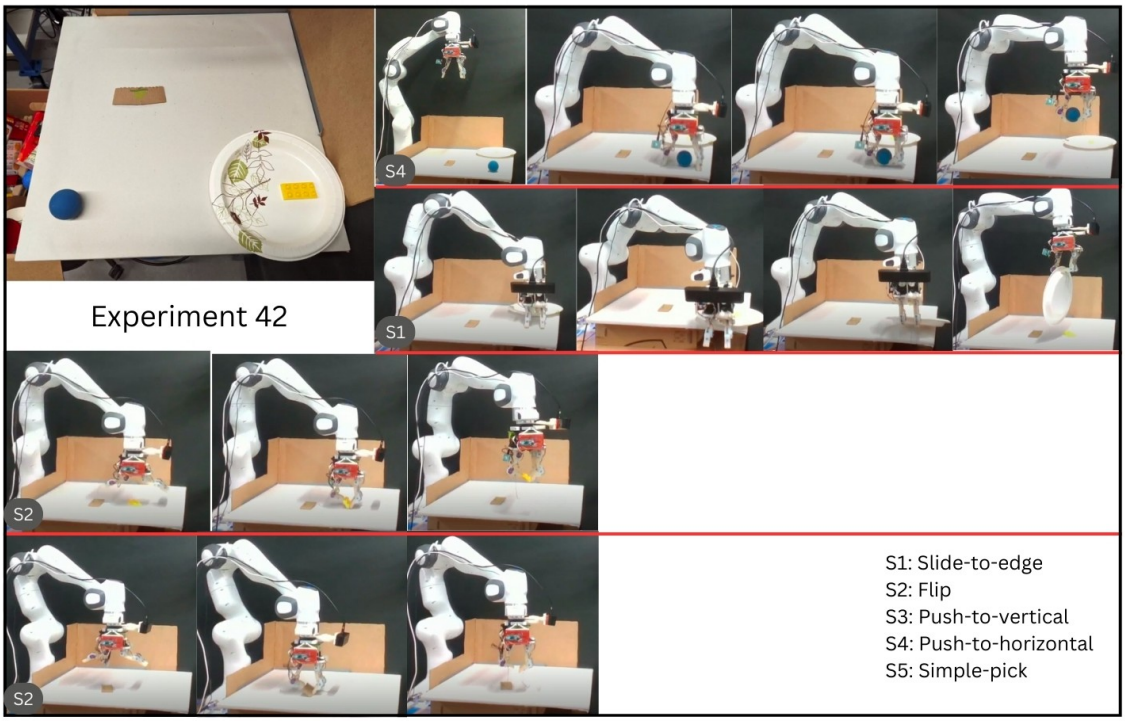



Figure 8.4: Step-wise execution of Experiment 42

its adept skill execution, promises enhanced efficiency and broader applicability in real-world scenarios.

## 8.5 Notable observations from detailed experiments

Analyzing the table from Appendix A reveals insights into the performance, adaptability, and challenges faced by the system when executing various tasks across different scenarios. Here's a summary of key observations and analysis derived from the data:

- High Success Rate in Known and Unknown Scenes: The system demonstrates

a strong ability to successfully manipulate objects in both known and unknown environments. This suggests that the robot's perception and motion planning algorithms are robust, allowing it to generalize its manipulation skills across different settings.

- Adaptation to Object Orientation Changes: We can see through the detailed experiment setup, that the gripper changes orientation according to the object. This adaptability ensures successful interaction with objects regardless of their orientation, significantly enhancing the robot's operational flexibility and effectiveness

- Handling of Complex Object Arrangements: The successful manipulation of objects in occluded scenarios, such as Experiment 12, indicates that the robot can navigate and manipulate in cluttered environments.

- Successful simulation-to-real model inference: Despite being trained on simulation data, the models demonstrate robust generalization capabilities, as evidenced by successful real-world experiments. This indicates that training on simulation data did not adversely affect the models' effectiveness when deployed in real-world scenarios.

## 8.6 Failure cases

In this section, we discuss some of the most common causes of failure in the system.

- Severe Occlusion: In scenarios where occlusion is extreme, the system might

fail to detect occluded objects. For example, a pile-up of multiple objects could lead to unseen failures in the system due to the inability to discern individual items within the clutter.

- Static Trajectory for Slide-to-Edge: Our current implementation of the slide-to-edge skill operates based on a predetermined edge of the table to which an object is slid. However, there may be situations where an object is closer to a different table edge, making it more efficient to slide the object towards that edge. Instead, our system endeavors to first clear any objects in the path towards the fixed edge before executing the slide-to-edge skill. This approach is inefficient and prone to failures in specific scenarios.

- Dynamic Execution of Skills: Consider a scenario where a push-to-vertical skill is being executed on an object, and a ball on the table rolls up near the object during execution. This unexpected obstruction can cause system failures. Integrating an online feedback mechanism during execution could allow for real-time adjustments and on-the-fly decisions.

- Separate skill detection models: Currently, our system consists of 2 separate models for skill detection: the depth model and the RGB model. Due to this, failures like Exp 42 might occur where the objects later detected by the RGB model get ignored while executing skills on the objects using the depth models. Using a combined model is a possible solution for this.

# Chapter 9

# Conclusion

In this thesis, we have designed and developed an efficient framework for the decluttering of multi-object scenes, utilizing dexterous manipulation skills that mimic human-like dexterity. We began by introducing the problem of dexterous manipulation and its underlying motivation, followed by a discussion on the challenges researchers face in imbuing robotic manipulators with human-like dexterity. A comprehensive literature review was presented, highlighting the latest advancements in dexterous manipulation and the inspiration behind our work.

We proposed a context-aware approach to tackle the problem of dexterous manipulation and detailed the deep neural network-based methods employed for detecting appropriate skills and grasp locations. Both models were assessed through practical experiments, successfully achieving skill detection and identifying skill locations for various objects. Our skill execution pipeline and the motion planning framework were elaborated upon, showcasing the methodology behind our approach. The thesis

culminated in presenting detailed results from a series of real-world experiments that demonstrate the capabilities of our system using an experimental dataset comprising a variety of everyday objects, both known and unknown. Our experimentation process, designed to be intuitive and replicable, can adapt to scenes with any number of objects. Through these experiments and their in-depth analysis, we provided insights into the generalizability of our system across different real-world conditions, underscoring the effectiveness and adaptability of our proposed framework in achieving dexterous manipulation in robotic systems.

## 9.1 Limitations and Future work

This section outlines the future work for enhancing the system's capabilities. It addresses some limitations of the current system and focuses on ways to enhance the current approach. The key areas addressed in this section include improving the deep learning models, refining skill execution with error feedback, and expanding the dataset. The aim is to make our current system better and more equipped for handling complex real-world scenarios.

### 9.1.1 Integrated Model

The current approach, which employs separate models for processing RGB and depth images, encounters challenges when small objects are in the path of other skills. Because we use RGB model at the end, it can lead to these objects being overlooked or incorrectly manipulated during specific actions, such as sliding to the edge of a

table as explained in Fig. 8.4. This issue underscores the necessity for a unified model that can provide a comprehensive solution. By combining RGB and depth data processing, the unified model aims to ensure precise detection and handling of all objects, irrespective of their size or shape, enhancing overall system performance.

### 9.1.2 Dataset Diversification

The system's training specificity to certain environmental setups limits its applicability across different real-world situations. To address this, expanding the training dataset to include a variety of table configurations, object placements, and camera perspectives is critical. Such diversification will enable the system to adapt more effectively to diverse environments, improving its versatility. This expansion should also focus on mitigating issues like object occlusion by exploring various camera angles, ensuring a thorough understanding of the scene.

### 9.1.3 Skill Refinement and Expansion

Our observations also point to the necessity of refining our implementation of certain skills. Despite selecting appropriate actions and target locations, the execution of slide-to-edge or flip sometimes falls short of expectations. Hence, more sophisticated control strategies, potentially leveraging closed-loop feedback mechanisms that draw on tactile sensing to fine-tune actions in real-time can be explored. The system currently has only five skills implemented, though in practical scenarios a broader arrays of skills are needed. The structured nature of our pipeline facilitates the integration of additional skills similar to current ones making the system more closer

to human-like behaviour.

### 9.1.4 Error recovery

At present, our system lacks a mechanism for error feedback during object grasping tasks, rendering it unable to confirm the successful execution of a grasp from start to finish. In instances where an object is dropped mid-motion, our system currently does not support error recovery processes. Integrating feedback mechanisms, such as force or tactile sensors, to verify whether an object has been securely grasped or securely released would significantly enhance the system's reliability and robustness.

# Bibliography


[1] Calli, B., Walsman, A., Singh, A., Srinivasa, S., Abbeel, P., and Dollar, A. M. Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set. *IEEE Robotics  Automation Magazine 22*, 3 (2015), 36–52.

[2] Collodi, L., Bacciu, D., Bianchi, M., and Averta, G. Learning with few examples the semantic description of novel human-inspired grasp strategies from rgb data. *IEEE Robotics and Automation Letters 7*, 2 (2022), 2573–2580.

[3] Depierre, A., Dellandréa, E., and Chen, L. Jacquard: A large scale dataset for robotic grasp detection. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), 3511–3516.

[4] Eppner, C., Deimel, R., Álvarez Ruiz, J., Maertens, M., and Brock, O. Exploitation of environmental constraints in human and robotic grasping. *The International Journal of Robotics Research 34*, 7 (2015), 1021–1038.

[5] He, K., Gkioxari, G., Dollár, P., and Girshick, R. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 2980–2988.

[6] Li, Y., Wang, P., Li, R., Tao, M., Liu, Z., and Qiao, H. A survey of multifingered robotic manipulation: Biological results, structural evolvements, and learning methods. *Frontiers in Neurorobotics 16* (2022).

[7] Lin, Y., Tang, C., Chu, F.-J., and Vela, P. A. Using synthetic data and deep networks to recognize primitive shapes for object grasping. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), pp. 10494–10501.

[8] Mahler, J., Liang, J., Niyaz, S., Laskey, M., Doan, R., Liu, X., Ojea, J. A., and Goldberg, K. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics, 2017.

[9] Morrison, D., Corke, P., and Leitner, J. Closing the Loop for Robotic Grasping: A Real-time, Generative Grasp Synthesis Approach. In *Proc. of Robotics: Science and Systems (RSS)* (2018).

[10] Mousavian, A., Eppner, C., and Fox, D. 6-dof graspnet: Variational grasp generation for object manipulation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), 2901–2910.

[11] Odhner, L., Ma, R. R., and Dollar, A. M. Open-loop precision grasping with underactuated hands inspired by a human manipulation strategy. *IEEE Transactions on Automation Science and Engineering 10* (2013), 625–633.

[12] Pinto, L., and Gupta, A. K. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. *2016 IEEE International Conference on Robotics and Automation (ICRA)* (2015), 3406–3413.

[13] Qin, X., Zhang, Z., Huang, C., Dehghan, M., Zaiane, O. R., and Jagersand, M. U2-net: Going deeper with nested u-structure for salient object detection. *Pattern Recognition 106* (Oct. 2020), 107404.

[14] Rakita, D., Mutlu, B., and Gleicher, M. RelaxedIK: Real-time Synthesis of Accurate and Feasible Robot Arm Motion. In *Proceedings of Robotics: Science and Systems* (Pittsburgh, Pennsylvania, June 2018).

[15] Santina, C. D., Arapi, V., Averta, G., Damiani, F., Fiore, G., Settimi, A., Catalano, M. G., Bacciu, D., Bicchi, A., and Bianchi, M. Learning from humans how to grasp: A data-driven architecture for autonomous grasping with anthropomorphic soft hands. *IEEE Robotics and Automation Letters 4*, 2 (2019), 1533–1540.

[16] Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., and Girshick, R. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[17] Zeng, A., Song, S., Yu, K.-T., Donlon, E., Hogan, F. R., Bauza, M., Ma, D., Taylor, O., Liu, M., Romo, E., Fazeli, N., Alet, F., Dafle, N. C., Holladay, R., Morena, I., Qu Nair, P., Green, D., Taylor, I., Liu, W., Funkhouser, T., and Rodriguez, A. Robotic pick-and-place of

novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (2018), pp. 3750–3757.

# Appendix A

# Detailed experiments table

| Exp No | Success | Scene | Object | No. of tries | Observation | Skill |
|---|---|---|---|---|---|---|
| 1 | Yes | Known + Unknown 4 of 4 | Plum | 1 | | P2H |
| | | | Advil | 1 | | P2H |
| | | | Cup | 1 | | SP |
| | | | Spray metal | 1 | | P2V |
| 2 | Yes | Unknown 3 of 3 | Glove | 1 | | SP |
| | | | Glasses | 2 | Slipped on the first attempt | SP |
| | | | Umbrella | 1 | | SP |
| 3 | Yes | Known + Unknown 4 of 4 | Pringles | 1 | | P2V |
| | | | Banana | 1 | | SP |
| | | | Tape | 1 | | P2H |
| | | | Ball | 1 | | P2H |
| 4 | No | Known + Unknown 3 of 4 | Pringles | 1 | | SP |
| | | | Banana | 0/2 | Gripper not closing enough | P2V |
| | | | Plum | 1 | | P2H |
| | | | Paper plate | 1 | | S2E |

| Exp No | Success | Scene | Object | No. of tries | Observation | Skill |
|---|---|---|---|---|---|---|
| 5 | Yes | Known + Unknown<br><br>6 of 6 | Spray metal | 1 | No change in orientation observed | SP |
| | | | Tape | 1 | Initial detection was SP; however, a P2H approach is required. | SP |
| | | | Glasses | 1 | | SP |
| | | | Paper plate | 1 | | S2E |
| | | | Umbrella | 1 | | P2V |
| | | | Cup | 1 | | SP |
| 6 | Yes | Unknown<br>4 of 4 | Umbrella | 1 | | SP |
| | | | Tape | 1 | | P2H |
| | | | Dasani | 1 | | P2V |
| | | | Spray metal | 2 | Slipped on the first attempt | P2V |
| 7 | No | Known + Unknown<br><br>2 of 3 | Plum | 2 | Grasp point was off centre | P2H |
| | | | Glove | 1 | | SP |
| | | | Plate | 0/2 | Pose was not achievable by the robot | S2E |
| 8 | Yes | Unknown<br>4 of 4 | Dasani | 1 | | SP |
| | | | Glasses | 1 | | SP |
| | | | Paper plate | 2 | Slipped from the fingers while picking (after sliding) in the first attempt. Successfully grasped in the second attempt. | S2E |
| | | | Airdopes | 2 | Object slipped on the first attempt | P2H |

| Exp No | Success | Scene | Object | No. of tries | Observation | Skill |
|---|---|---|---|---|---|---|
| 9 | Yes | Unknown 3 of 3 | Dasani | 1 | | SP |
| | | | Banana | 1 | | SP |
| | | | Spray metal | 1 | | P2V |
| 10 | Yes | Known + Unknown 3 of 3 | Pringles | 1 | | SP |
| | | | Cup | 1 | | SP |
| | | | Paper Cup | 1 | | SP |
| 11 | Yes | Known + Unknown 3 of 3 | Ball | 1 | | P2H |
| | | | Dasani Bottle | 1 | | P2V |
| | | | Cup | 1 | | SP |
| 12 | Yes | Occluded 5 of 5 | Paper plate | 1 | | S2E |
| | | | Spray metal | 1 | | P2V |
| | | | Paper Cup | 1 | | SP |
| | | | Plum | 1 | | P2H |
| | | | Mug | 1 | | SP |
| 13 | No | Known + Unknown | Pringles | 0/0 | Wrong priority list - hand will collide with the plate | P2V |
| | | 0 of 2 | Paper plate | 0/0 | | S2E |
| 14 | Yes | Known + Unknown | Paper plate | 1/2 | Unachievable pose in the first attempt | S2E |
| | | 4 of 4 | Lego brick | 1 | | F |
| | | | Dasani bottle | 1 | | P2V |
| | | | Plum | 1 | | P2H |
| 15 | No | Known + Unknown 3 of 4 | Lego brick | 1 | | F |
| | | | Cardboard small | 0/2 | Kept slipping between the fingers | F |
| | | | Spray metal | 1 | | P2V |
| | | | Paper plate | 1 | | S2E |
| 16 | Yes | Occluded | Pringles | 1 | Grasp was unstable | SP |
| | | 5 of 5 | Paper plate | 1 | | S2E |
| | | | Paper Cup | 1 | | SP |
| | | | Ball | 1 | | P2H |
| | | | Lego brick | 1 | | F |

| Exp No | Success | Scene | Object | No. of tries | Observation | Skill |
|---|---|---|---|---|---|---|
| 17 | No | Unknown<br><br>4 of 5 | Lego brick | 2 | Slippage occurred on the first attempt. | F |
| | | | Paper Cup | 1 | | SP |
| | | | Dasani Bottle | 1 | | SP |
| | | | Plum | 1 | | P2H |
| | | | Cardboard tray | 0/2 | Slipped from the fingers while picking (after sliding) | S2E |
| 18 | Yes | Known<br>3 of 3 | Pringles | 1 | | P2V |
| | | | Cup | 1 | | SP |
| | | | Ball | 1 | | P2H |
| 19 | Yes | Known + Unknown<br>3 of 3 | Paper Cup | 1 | | SP |
| | | | Orange | 1 | Struck the object before grasping | P2H |
| | | | Dasani Bottle | 1 | | P2V |
| 20 | Yes | Known<br>3 of 3 | Pringles | 1 | | P2V |
| | | | Plate | 1 | Grip was unstable | S2E |
| | | | Cup | 1 | | SP |
| 21 | Yes | Known<br>3 of 3 | Pringles | 1 | | P2V |
| | | | Ball | 1 | | P2H |
| | | | Cup | 1 | | SP |
| 22 | Yes | Known + Unknown<br><br>4 of 4 | Paper plate | 2 | Gripper pushes the plate inside while trying to grasp in the first attempt | S2E |
| | | | Lego brick | 1 | Did not flip properly | F |
| | | | Lego brick | 2 | Slipped in the first attempt | F |
| | | | Paper Cup | 1 | | SP |

| Exp No | Success | Scene | Object | No. of tries | Observation | Skill |
|---|---|---|---|---|---|---|
| 23 | Yes | Known + Unknown 3 of 3 | Lego brick | 1 | | F |
| | | | Lego brick | 1 | | F |
| | | | Dasani Bottle | 1 | | SP |
| 24 | Yes | Occluded 4 of 4 | Lego - small | 1 | | F |
| | | | Tape | 1 | | P2H |
| | | | Paper Cup | 1 | | SP |
| | | | Paper plate | 1 | | S2E |
| 25 | Yes | Known + Unknown 5 of 5 | Dasani Bottle | 1 | | SP |
| | | | Paper plate | 1 | | S2E |
| | | | Cup | 1 | | SP |
| | | | Spray metal | 1 | Grasp was unstable | P2V |
| | | | Lego brick | 1 | | F |
| 26 | Yes | Occluded 5 of 5 | Lego brick | 1 | | F |
| | | | Lego brick | 1 | | F |
| | | | Paper plate | 1 | | S2E |
| | | | Pringles | 1 | | SP |
| | | | Plum | 1 | | P2H |
| 27 | No | Unknown 3 of 4 | Paper plate | 2 | Pose was not achievable on the first attempt. | S2E |
| | | | Plum | 1 | | P2H |
| | | | Paper Cup | 1 | | SP |
| | | | Cardboard | 0/2 | Slipped between the fingers | F |
| 28 | Yes | Unknown 3 of 3 | Umbrella | 1 | Slipped during grasp but didn't fall | SP |
| | | | Paper plate | 1 | | S2E |
| | | | Glasses | 2 | Gripper wasn't completely closed | SP |
| 29 | Yes | Unknown 3 of 3 | Dasani Bottle | 1 | | SP |
| | | | Tape | 1 | | P2H |
| | | | Spray metal | 1 | | P2V |

| Exp No | Success | Scene | Object | No. of tries | Observation | Skill |
|---|---|---|---|---|---|---|
| 30 | Yes | Unknown 4 of 4 | Paper Cup | 1 | | SP |
| | | | Dasani Bottle | 1 | | P2V |
| | | | Plum | 1 | | P2H |
| | | | Tape | 1 | Wrong skill was detected | SP |
| 31 | Yes | Unknown | Paper plate | 2 | Pose was not achievable on the first attempt. | S2E |
| | | 4 of 4 | Dasani Bottle | 1 | | P2V |
| | | | Plum | 1 | | P2H |
| | | | Cardboard Cup Sleeves | 1 | Got two masks around object | F |
| 32 | Yes | Occluded 2 of 2 | Umbrella | 1 | | SP |
| | | | Paper plate | 1 | | S2E |
| 33 | No | Occluded 2 of 3 | Paper Cup | 1 | | SP |
| | | | Spray metal | 1 | | SP |
| | | | Paper plate | 0/2 | Pose was not achievable by the robot | S2E |
| 34 | Yes | Unknown 7 of 7 | Paper plate | 1 | | S2E |
| | | | Paper Cup | 1 | | SP |
| | | | Banana | 1 | | SP |
| | | | Lego brick - small | 1 | | F |
| | | | Spray metal | 1 | | P2V |
| | | | Advil | 1 | | P2H |
| | | | Plum | 1 | | P2H |
| 35 | Yes | Known 4 of 4 | Cup | 1 | | SP |
| | | | Ball | 1 | | P2H |
| | | | Lego brick | 1 | | F |
| | | | Pringles | 1 | | P2V |
| 36 | Yes | Known 2 of 2 | Lego brick | 1 | | F |
| | | | Lego brick | 1 | | F |

| Exp No | Success | Scene | Object | No. of tries | Observation | Skill |
|---|---|---|---|---|---|---|
| 37 | No | Unknown 3 of 4 | Paper plate | 1 | | S2E |
| | | | Paper Cup | 1 | | SP |
| | | | Spray metal | 2 | Object slipped on the first attempt due to slant position and improper depth | SP |
| | | | Lego brick - small | 0/0 | Was pushed out of the workspace when sliding the plate | F |
| 38 | Yes | Unknown 2 of 2 | Cardboard Cup Sleeves | 0/2 | Object kept slipping persistently | F |
| | | | Gloves | 1 | | SP |
| 39 | Yes | Occluded 2 of 2 | Lego brick - small | 1 | | F |
| | | | Cardboard tray | 1 | | S2E |
| 40 | No | Unknown 5 of 6 | Tape | 1 | | P2H |
| | | | Paper Cup | 2 | Slipped due to cup being at a slant angle | SP |
| | | | Glasses | 1 | | SP |
| | | | Advil | 0/2 | Fingers didn't close enough to grasp | P2H |
| | | | Lego brick - small | 1 | | F |
| | | | 3D print - pink | 1 | Object was not flipped properly | F |
| 41 | Yes | Unknown 4 of 4 | Paper Cup | 1 | | SP |
| | | | Glasses | 1 | | SP |
| | | | Plum | 1 | | P2H |
| | | | Spray metal | 1 | | P2V |

| Exp No | Success | Scene | Object | No. of tries | Observation | Skill |
|---|---|---|---|---|---|---|
| 42 | No | Known + Unknown 3 of 4 | Cardboard Cup Sleeves | 0/2 | Kept slipping between the fingers | F |
| | | | Lego brick - small | 1 | | F |
| | | | Ball | 1 | | P2H |
| | | | Paper plate | 1 | | S2E |
| 43 | Yes | Occluded 3 of 3 | Cup | 1 | | SP |
| | | | Pringles | 1 | | SP |
| | | | Paper Plate | 1 | | S2E |
| 44 | Yes | Occluded 3 of 3 | Banana | 2 | Object slipped on the first attempt due to improper depth | SP |
| | | | Dasani Bottle | 1 | | SP |
| | | | Advil | 2 | Wrong skill detected | SP |
| 45 | Yes | Occluded 4 of 5 | Paper plate | 2 | Unachieavable pose | S2E |
| | | | Paper Cup | 1 | | SP |
| | | | Cup | 1 | | SP |
| | | | Plum | 1 | | P2H |
| | | | Lego brick- small | 1 | | F |