CONTEXT-AWARE GRASPING BY A DEXTEROUS ANTHROPOMORPHIC ROBOTIC HAND

by

HUI LI

HONGSHENG HE, COMMITTEE CHAIR
MONICA ANDERSON HERZOG
CHRIS CRAWFORD
LINA PU
MARK CHENG

A DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the Graduate School of
The University of Alabama

TUSCALOOSA, ALABAMA

2025

ABSTRACT

Dexterous manipulation is a required ability for robots to interact with the physical environment. This capability allows robots to perform a wide range of tasks, from simple actions like picking up objects to more complex operations such as assembling intricate machinery or performing delicate surgical procedures. Dexterous grasping, as the initial and pivotal step in dexterous manipulation, involves the intricate coordination of robotic hands or grippers to securely and skillfully hold objects . By mastering dexterous grasping techniques, robots can lay a solid foundation for executing complex manipulation tasks with precision and versatility.

Dexterous grasping relies on a combination of understanding object affordance and task designations. However, current research fails to tackle this challenge effectively. To solve this problem, a context-aware task-oriented dexterous grasping method is proposed. In the proposed method, multiple sensors are fused to perceive object affordances such as shape, size, position, and material in real time. A large language model is also integrated to augment the perceived information leveraging its abilities of common sense and reasoning. Task designations are learned from human experience, represented by a human knowledge dataset collected from 27.7 hours of tagged videos recorded by two housekeepers and two machinists during their regular work activities. Object affordance and task designation were then mapped to a proper grasp strategy with a multi-class multi-label deep learning network.

To deploy predicted grasp strategies efficiently and precisely while reducing task complexity, we propose the Contextual Reward Machine (CRM). The CRM decomposes the grasping task into manageable sub-tasks, each associated with stage-specific contexts, such as reward functions, action spaces, and abstracted representations, enabling efficient guidance within each stage. Transition rewards are introduced to encourage or penalize transitions between

stages. This structure enhances adaptability, reduces complexity, and improves overall performance. When integrated with Proximal Policy Optimization (PPO), the CRM significantly enhances the sample efficiency and robustness of the reinforcement learning model. A simulation environment which is identical to the real-world robotic setup is developed to train the model on grasping tasks, with the trained policy transferred to the physical robot through domain randomization.

The final experimental results on grasping tasks demonstrate that the proposed method accurately perceives object affordances in real-time. Additionally, both object affordances and task designations are mapped to the appropriate grasp topology with high precision. The real robot's performance in executing the grasping tasks highlights the efficiency and effectiveness of the proposed grasp deployment model. These results underscore the method's capability to improve real-world grasping task execution, enhancing both the adaptability and reliability of robotic manipulation systems.

DEDICATION

I dedicate this thesis to my beloved parents, whose unwavering love and support have been the cornerstone of my academic journey. Your encouragement, sacrifices, and belief in my abilities have fueled my determination to pursue excellence. Thank you for inspiring me to dream big, for instilling in me the values of hard work and perseverance, and for always being my strongest supporter, cheering me on through every triumph and challenge.

LIST OF ABBREVIATIONS AND SYMBOLS

$\mathcal{A}$	Finite set of actions called action space
a	The longest orthogonal dimension
abd.	Thumb abducted
add.	Thumb adducted
b	The second longest orthogonal dimension
c	The shortest orthogonal dimension
γ	Discount factor
$\mathcal{M}$	Markov Decision Processes
p	Transition probability
$\mathcal{R}$	Reward function that provides feedback for an action
rc	Precision circular grasp
rp	Precision prismatic grasp
$\mathcal{S}$	Finite set of states that describe the environment
S_{I}	Initial state of a multi-state system
S_{T}	Termination state of a multi-state system
wc	Power circular grasp
wh	Power heavy grasp
wp	Power prismatic grasp
wt	Power thin grasp

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Hongsheng He, for his unwavering support, invaluable guidance, and profound insight throughout the development of this thesis. His expertise, encouragement, and constructive feedback have been instrumental in shaping both my research and academic growth.

I also extend my heartfelt appreciation to the members of my thesis committee: Dr. Monica Anderson Herzog, Dr. Chris Crawford, Dr. Lina Pu, and Dr. Mark Cheng. Their insightful feedback, thoughtful critique, and scholarly guidance have greatly enriched this work and contributed significantly to its refinement.

I am truly grateful for the mentorship and support of my advisor and committee members, without whom this thesis would not have been possible.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

A context-aware system is designed to collect useful information from its environment and adapt its behavior based on the perceived context [62]. This concept is widely applied in various domains, including robot cooperation [74], patient assistance [51, 38, 24], object tracking [45], swarm robotics [36], and dexterous manipulation systems [29].

Dexterous manipulation is a critical capability for intelligent robots performing complex tasks. It involves the coordinated action of multiple manipulators or fingers to grasp and manipulate objects, such as placing fingertips at optimal contact points, applying appropriate force, and maintaining grasp stability [6]. Unlike traditional grasping approaches [44], dexterous manipulation demands precise control of both force and motion across the arm and fingers to execute complex grasping tasks.

Dexterous grasping forms the foundation of dexterous manipulation. It enables a robotic hand or gripper to securely and skillfully grasp objects with precision and adaptability. This step is crucial for effective physical interaction with the environment. Successful dexterous grasping requires the robot to identify the object's position and orientation, assess its shape, size, and surface characteristics, and determine the optimal grasp configuration. This configuration may vary depending on factors such as object geometry, weight distribution, and the specific task requirements.

An appropriate grasp topology is essential for dexterous manipulation, as it minimizes unnecessary finger movements and simplifies the overall process. Although numerous grasping strategies exist for a given object, prior research has shown that human grasps tend to

cluster across a wide range of objects, leading to the development of grasp taxonomies to streamline grasp selection [14, 8]. Studies further indicate that grasp topology selection is influenced by object affordances [40, 58] and task requirements [20]. Object affordances refer to physical properties such as size, shape, and mass, while task requirements encompass action types, applied forces, and imposed constraints. Current robotic systems face difficulties in efficiently acquiring object information, understanding task specifications, and integrating both affordances and task requirements to inform grasp strategy determination. To address these challenges, a novel context-aware, task-oriented grasping approach is proposed and developed to support complex manipulation tasks, as illustrated in Fig. 1.1.

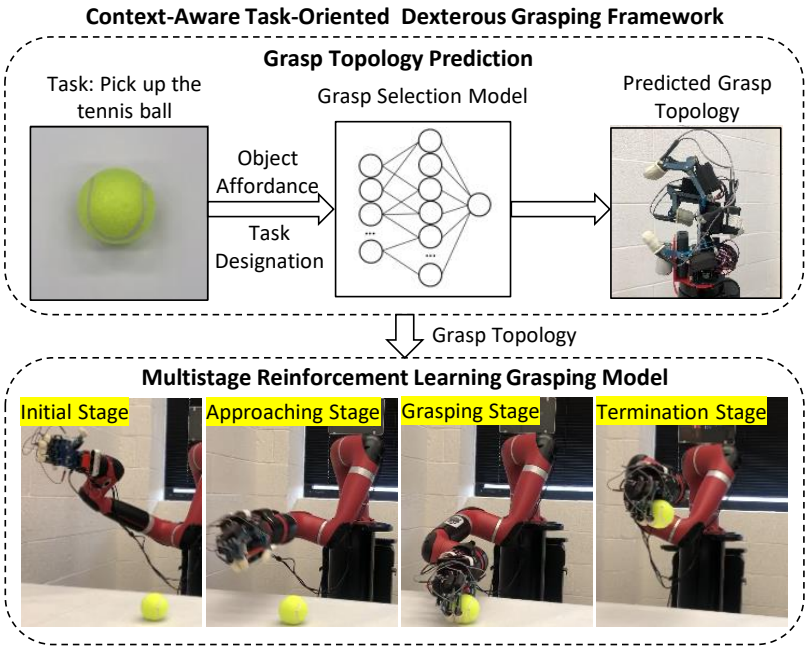

Figure 1.1: The MagicHand context-aware robotic system.

The structure of the proposed method is illustrated in Fig. 1.2. In this approach, grasping tasks are initiated through simple human instructions, which are parsed by the Controlled Robot Language (CRL) model [73] to extract the action and the object. The action is embedded using a Word2Vec network and converted into task designations. Based on the identified object, its affordances are perceived through the MagicHand platform [39]. A grasp selection network is then employed to map the extracted object affordances and task

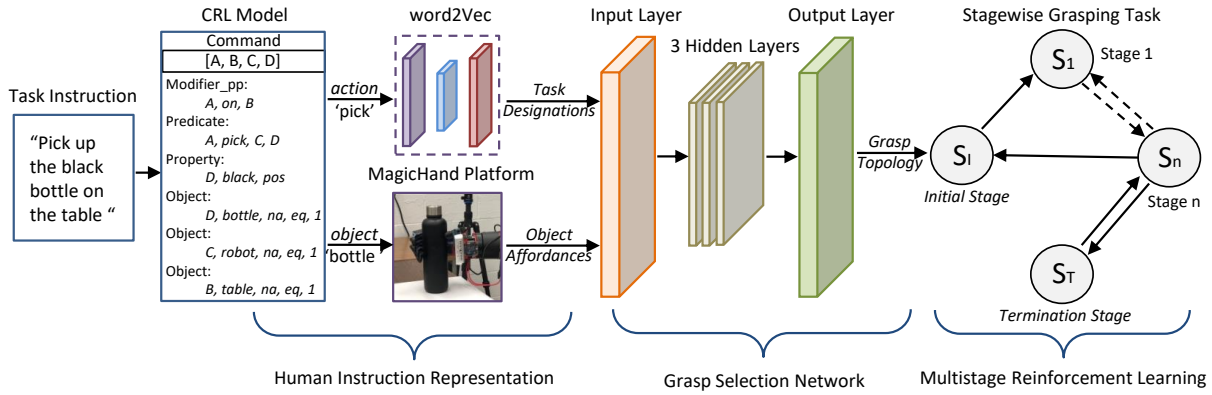

Figure 1.2: The structure of the context-aware task-oriented grasping method.

designations to appropriate grasp topologies, leveraging human experience. The predicted grasp topology is subsequently executed using a multistage reinforcement learning model, which adaptively adjusts the grasp configuration. This learning-based approach maintains a balance between grasping flexibility and dexterity by adapting to diverse grasping contexts.

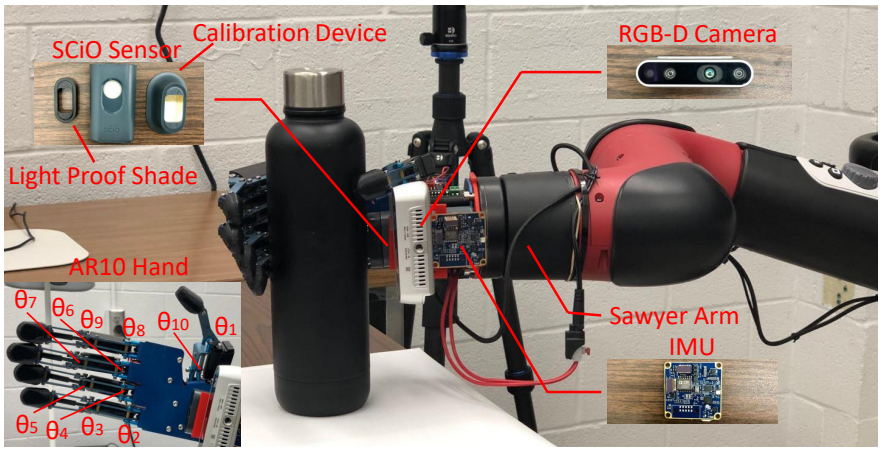

Figure 1.3: The MagicHand platform.

The MagicHand platform, shown in Fig. 1.3, is a context-aware dexterous grasping system that integrates an AR10 anthropomorphic robotic hand with a Sawyer robotic arm to perform object manipulation tasks. A SCiO sensor is utilized to identify the material composition of objects. This sensor operates in the near-infrared (NIR) spectrum, measuring absorption bands between 780 nm and 2500 nm. In this range, the molar absorption is minimal, allowing NIR light to penetrate deeply into samples. This characteristic enables

the use of near-infrared spectroscopy (NIRS) for material recognition with minimal or no sample preparation. Additionally, an RGB-D camera (Intel RealSense D435) is employed to estimate the object's dimensions, shape, and pose. Instead of using a fixed camera setup, the RGB-D sensor is mounted on the wrist of the robotic hand, allowing it to move with the hand and capture RGB-D images from multiple viewpoints of the object.

This dissertation is structured as follows: Chapter 2 presents a comprehensive literature review covering key topics such as dexterous manipulation, context-aware systems, task-oriented grasping, and optimal control. Chapter 3 elaborates on the proposed context-aware system, while Chapter 4 introduces the methodology for extracting grasp strategies from human knowledge. Chapter 5 explores a knowledge augmentation approach based on large language models. Chapter 6 discusses grasp intention interpretation in object handover. Chapter 7 introduces a multistage reinforcement learning approach that utilizes a Contextual Reward Machine to address task-oriented grasping. Finally, Chapter 8 provides concluding remarks and summarizes the contributions of this work.

CHAPTER 2

LITERATURE REVIEW

2.1 Dexterous Manipulation

Dexterous manipulation, which enables robots to interact skillfully with objects, has attracted considerable attention in robotics research. Early studies primarily focused on developing algorithms for grasping and manipulation tasks [82]. However, these approaches often encountered challenges when dealing with complex object shapes and uncertain environments [7].

Advancements in sensing and perception have enabled robots to gather rich environmental data, leading to the development of more robust manipulation techniques [4, 81]. Techniques such as tactile sensing and computer vision are pivotal in enhancing the dexterity of robotic hands [10, 79].

Furthermore, understanding object affordances and task designations has become a critical component of dexterous manipulation. Object affordances refer to the potential actions that an object enables, while task designations define the specific goals or objectives of a manipulation task [25]. Integrating these concepts into manipulation algorithms has resulted in more adaptive and versatile robotic systems [79].

However, challenges persist in achieving seamless integration across the perception, planning, and execution phases of manipulation tasks. The gap between simulated environments and real-world scenarios remains a significant obstacle to transferring learned skills from simulation to physical robots [78].

Dexterous manipulation remains a dynamic and multifaceted field of research, continuously evolving with advancements in sensing, perception, and learning. By drawing on insights from multiple disciplines, researchers aim to develop robots capable of interacting with the physical world with human-like dexterity and adaptability.

2.2 Context-Aware System

Context-aware systems adapt to dynamic conditions, providing personalized and adaptive services across various domains. Early frameworks laid the foundation for such systems but encountered scalability challenges [17]. Advances in sensing technologies have improved data acquisition, though pervasive sensing raises significant privacy concerns [55]. Reasoning techniques, such as machine learning, enable informed decision-making but often require extensive labeled data for effective operation [35].

Adaptive strategies enhance system performance, yet computational costs remain a significant hurdle [42]. These systems have found applications in diverse domains, including healthcare, where integration challenges continue to impede progress [1]. For example, context-aware healthcare systems leverage contextual information to improve patient care [75], but integrating these technologies into healthcare environments poses challenges related to data security and regulatory compliance [3].

Overall, context-aware systems have substantial potential to enhance user experiences across various domains. By leveraging advancements in sensing, reasoning, and adaptation, these systems can deliver personalized and contextually relevant services that address individual needs and preferences.

2.3 Task-Oriented Grasping

Task-oriented dexterous grasping, essential for robotic manipulation, has been a significant focus of research. Early studies concentrated on developing geometric and tactile grasping algorithms [7]. The advancement of machine learning, particularly deep learning,

has enabled the development of grasping policies learned from data [26]. Contextual understanding has become increasingly important, as it integrates object affordances and task objectives into the grasping process [43]. Hardware developments, such as multi-fingered robotic hands equipped with tactile sensing, have enhanced manipulation capabilities [12]. Despite these advances, challenges remain, particularly in achieving robustness to object variations and adaptability to dynamic environments. Future research may address these challenges through interdisciplinary approaches.

2.4 Optimal Control with Reinforcement Learning

Optimal control using reinforcement learning (RL) has emerged as a pivotal research domain, offering promising solutions to complex control problems across various fields. Over the years, significant progress has been made in developing and refining RL algorithms tailored for optimal control tasks.

In the early stages, RL algorithms such as Q-learning and policy gradient methods laid the groundwork for addressing optimal control challenges [69]. While these approaches achieved initial successes, they faced limitations when applied to complex, high-dimensional control tasks, often struggling with scalability and sample efficiency.

The advent of deep reinforcement learning (DRL) represented a major breakthrough in the field of optimal control. DRL algorithms, including Deep Q Networks (DQN) and Deep Deterministic Policy Gradient (DDPG), have demonstrated remarkable capabilities in learning optimal control policies from high-dimensional state spaces [50]. These algorithms have been successfully applied across various domains, including robotics, autonomous vehicles, and game playing, showcasing their versatility and effectiveness.

Moreover, the exploration of model-based RL methods has introduced new opportunities for solving optimal control problems. By leveraging learned dynamic models of the environment, model-based RL algorithms can plan and execute optimal control actions more efficiently than their model-free counterparts [16]. This approach offers potential for sample-

efficient learning and improved stability, making it particularly appealing for real-world applications.

Despite significant advancements, challenges remain in the field of optimal control with RL. Issues such as sample efficiency, generalization to unseen environments, and robustness to model uncertainty continue to pose substantial hurdles. Addressing these challenges remains a central focus of ongoing research.

In recent years, researchers have increasingly focused on overcoming the practical challenges of applying RL to optimal control tasks. Techniques for managing exploration-exploitation trade-offs, reward shaping, and ensuring safe exploration in real-world systems have been actively explored [63]. These efforts aim to bridge the gap between theoretical progress and practical implementation, paving the way for the widespread adoption of RL-based control strategies across diverse domains.

2.5 Summary

Numerous efforts have been dedicated to advancing dexterous manipulation capabilities. However, existing methods often struggle with the real-time extraction of critical object features such as material, texture, and rigidity. While reinforcement learning holds promise for deploying manipulation tasks, data efficiency remains a significant challenge. This work addresses these limitations by proposing a method that enables real-time feature extraction from objects and improves data efficiency through a multistage reinforcement learning approach.

CHAPTER 3

CONTEXT-AWARE GRASPING BY MULTIPLE SENSORS FUSION

3.1 Introduction

Understanding of characteristics of objects such as fragility, rigidity, texture and dimensions facilitates and innovates robotic grasping. In this chapter, a context aware grasping system named MagicHand is presented. This platform is designed to achieve dexterous grasping on target objects based on the information collected by the sensors in the platform, the performance of this system was demonstrated through multiple experiments.

3.2 Material Perception

Two deep neural networks were designed and implemented for the material recognition process: a denoising auto-encoder for nosie cancelling and a MLP neural network for material classification. We established two datasets to train the algorithm. The first dataset collected in various practical environments was called MLP dataset which contains 15936 spectra of six materials from 540 samples. The other dataset was called AE dataset (Fig. 3.1), it

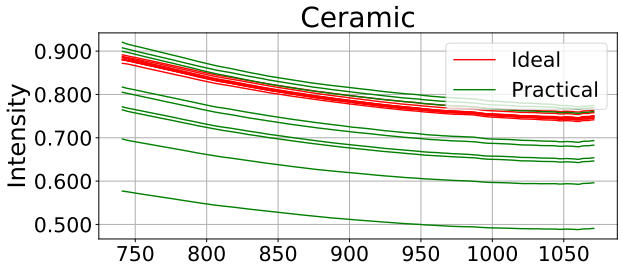

Figure 3.1: Example of ideal spectra (red) and practical spectra (green).

contains 1080 pairs of NIR spectra $(SI, SP)_k$ where SI is an idea spectrum collected from a sample in lab environment and SP is collected from the same location of the sample by the MagicHand platform. Before feeding the spectra into the network, the raw data was pre-processed with standard normal variate (SNV) to improve the performance of the whole system.

3.2.1 NIR Spectrum Denoising and Reconstruction

NIR spectra can be classified perfectly in an ideal environment where noises and errors can be neglected. However, in practical, there are many ways that able to bring noises or errors to the data such as light density, temperature or even a false operation on the sensors. These noises are not avoidable and may lead to a low classification accuracy. In this work, we train a complex structure of auto encoders (Fig. 3.2) with the AE dateset to reduce noises and to improve performance of our algorithm.

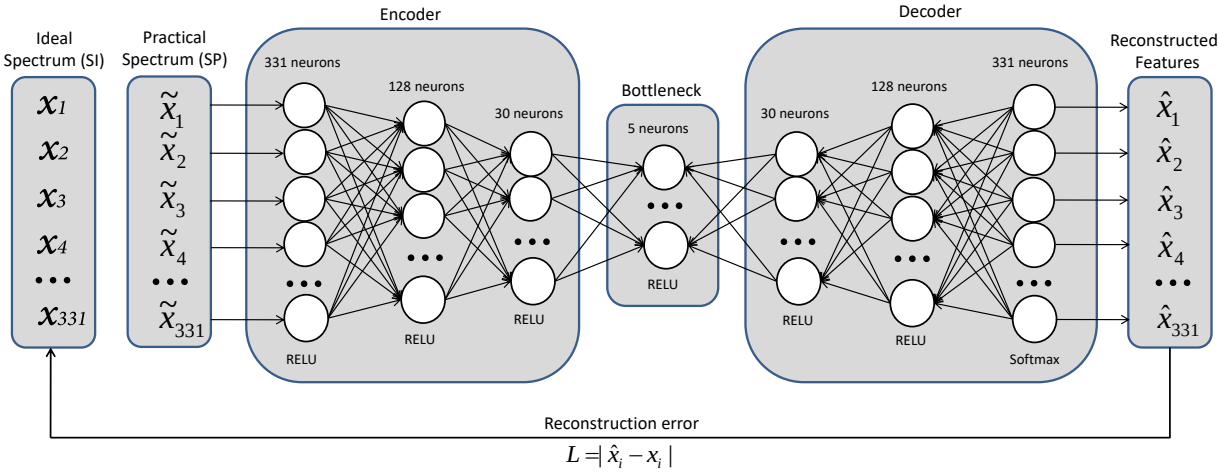

Figure 3.2: Structure of autoencoder for noise reduction

In the algorithm, x_i is the i^{th} feature in a pre-processed ideal spectrum where $i = 331$ since there are 331 features in an NIR spectrum. $\tilde{x_i}$ is the correspond practical feature of x_i in AE database. $\tilde{x_i}$was then encoded by algorithm and 5 most important features were extracted. The algorithm would then try to decode the extracted features and generate reconstructed spectrum $\hat{x_i}$.

3.2.2 Characteristics Classification

With NIR spectrum, material of an object can be recognized and then characteristics such as fragility, rigidity, and texture are mapped to the perceived material.

Material Classification

A MLP deep neural network was developed specifically for our MLP dataset. Rectified Linear Units (ReLU) were used as activation functions in the first five layers. In the first layer, 331 neurons were used to match the number of features in dataset. In the next four layers, 2^{10}, 2^9, 2^8 and 2^5 were employed to further calibrate the model. The output layer contains six neurons and Softmax functions were used to classify the data. The network was trained with 70% of data and achieved an overall 99.96% accuracy.

Characteristics Mapping

Table 3.1: Material to Characteristics Mapping List

Materials	Fragility	Rigidity	Texture
Wood	sturdy	rigid	rough
Glass	medium	rigid	smooth
Plastic	medium	soft	smooth
Ceramic	sturdy	rigid	slippery
Cardboard	medium	rigid	rough
Stainless Steel	sturdy	rigid	grippy

Once the material of an object was recognized, characteristics were mapped to the perceived material through a material-to-characteristics table (Table 3.1). This table was established based on an existing dataset [57]. Three characteristics including fragility, rigidity and texture were extracted and assigned to each corresponding material. After the material of an object was perceived, three characteristics (fragility, rigidity and texture) values were mapped to that material. This information would then be used to learn grasping strategies.

3.3 Object Dimension Measurement

There were two challenges in dimension estimation process. The first one was to separate the object from its background and the other one was to find orthogonal directions of the object to measure the dimensions of that object correctly.

3.3.1 Object detection in point cloud

An RGB-D point cloud/image might contain multiple objects, separating the target object from the others was necessary for dimension estimation. By utilizing an existing 2D detector named Yolo3 [59], we successfully detected the target object in an RGB image. In the next step, we reconstructed the target in 3D using Open3D [84, 11, 53] with a sequence of RGB-D images of that target object. Given depth information D and intrinsic matrix I which described the relationship between a stream's 2D and 3D coordinate systems, a pixel (px, py) in an RGB image could be deprojected to a point (x, y, z) in 3D point cloud with

$$\begin{aligned} x &= (px - ppy)D/fy \\ y &= (py - ppx)D/fx \\ z &= D \\ I &= \begin{bmatrix} fx & 0 & ppx \\ 0 & fy & ppy \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{3.1}$$

where fx and fy were focal length of the image, ppx and ppy were coordinates of center of projection, these information could be collected directly from the RGB-D sensor. To extract the target object in 3D point could, all pixels that belong to the target object detected by the 2D detector were deprojected to points in 3D space and any point within this region were cropped from the point cloud to establish a 3D model of the target object.

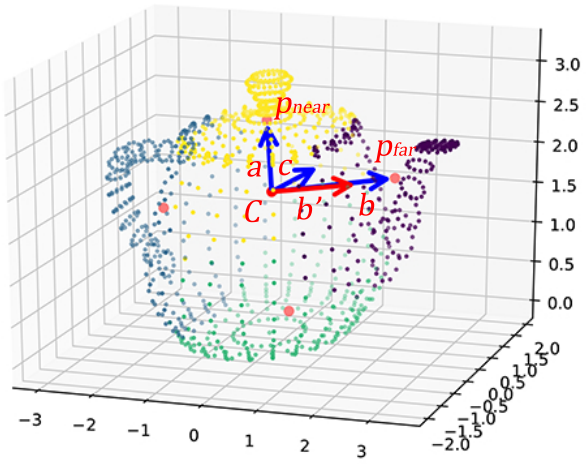

Figure 3.3: Process of orthogonal direction determination and dimension estimation

3.3.2 Axis Dimension for Grasping

While an object is placed in front of a camera, the object could be placed in different positions (distance), angles or even on a slop. All these factors may bring difficulties in estimating dimensions of the object. To address this issue, we propose a method for determining the dimensions of an object in orthogonal directions, which remains unaffected by the perspective from which the object is observed, as shown in Fig. 3.3. We first found the centroid point $\mathrm{C} = \left(\overline{\mathrm{X}}, \overline{\mathrm{Y}}, \overline{\mathrm{Z}}\right)$ where $\overline{\mathrm{X}}$,$\overline{\mathrm{Y}}$ and $\overline{\mathrm{Z}}$ were the average value of Cartesian coordinates of all points in a point cloud. Then we clustered the point cloud into m groups and found centroid points c_j of those groups as shown in 3.2 where x_i was the i^{th} point in points cloud and g_i represented the group that contained x_i.

$$
\begin{aligned}
g_i &= \arg\min_j ||x_i - c_j||^2 \\
c_j &= \frac{\sum_{i=1}^{n} 1\{g_i = j\} x_i}{\sum_{i=1}^{n} 1\{g_i = j\}} j \in [0, 1, 2 \cdots m]
\end{aligned}
\tag{3.2}
$$

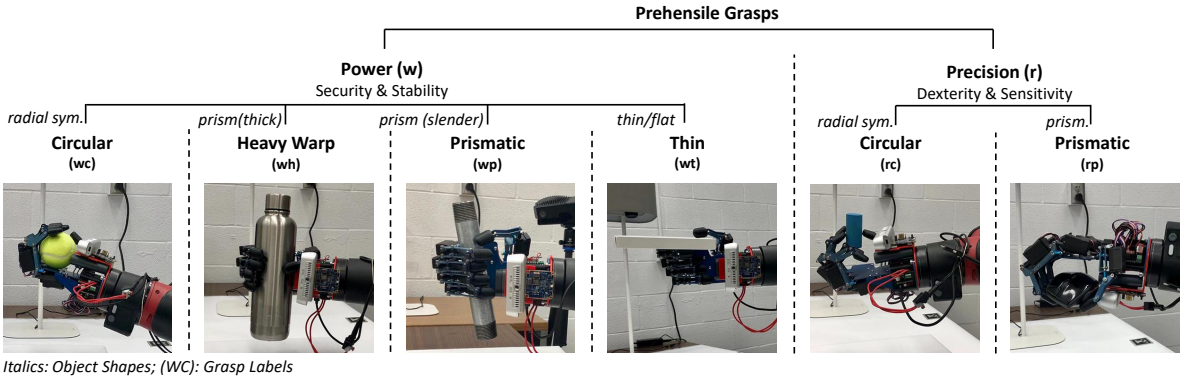

Figure 3.4: Pre-defined grasping descriptions

For the third step, euclidean distance were calculated between C and c_i and find p_{near} and p_{far} by

$$\begin{aligned} p_{near} &= \arg\min ||C - c_j|| \\ p_{far} &= \arg\max ||C - c_j|| \end{aligned} \tag{3.3}$$

Now we could determine a plane h with C, p_{near} and p_{far} and thus two vectors in h: $a = p_{near} - C$ and $b = p_{far} - C$. And we found a vector c which was orthogonal to plane h through centroid point C with equation

$$c = ||a||||b|| \sin(\theta)\lambda \tag{3.4}$$

where $||a||$ and $||b||$ were magnitude of vector a and b, θ was the angle between a and b and λ was a unit vector perpendicular to the plane h. Since the longest distance from centroid points to a point in points was usually the diagonal, the angle between a and b might not be a right angle. In this case, we needed to find a new vector b' which was perpendicular to both vector a and c through centroid point C. The lengths of vector a, b' and c could be determined by existing realsense API.

3.4 Grasping Strategies Generation

Human grasping choices are based on task and properties of target objects [34] and they trend to cluster over a large set of objects [23, 61]. Due to this fact and to simplify the problem, we defined six grasping descriptions d_i (Fig. 3.4) from [14] where d_i is a ten dimensional vector of joint angles of AR10 hand θ_i. These grasping descriptions are suitable for a wide range of objects. For example, Power Circular Grasp, Power Heavy Wrap Grasp, Power Prismatic Grasp and Power Thin Grasp are able to provide secure and stable grasping for sturdy, rigid or rough objects while Precision Circular Grasp and Precision Prismatic Grasp are more suitable for objects require dexterous or sensitive manipulations. With these pre-define grasp choices, we can simplified the problem as

$$G_i = \beta d_i$$

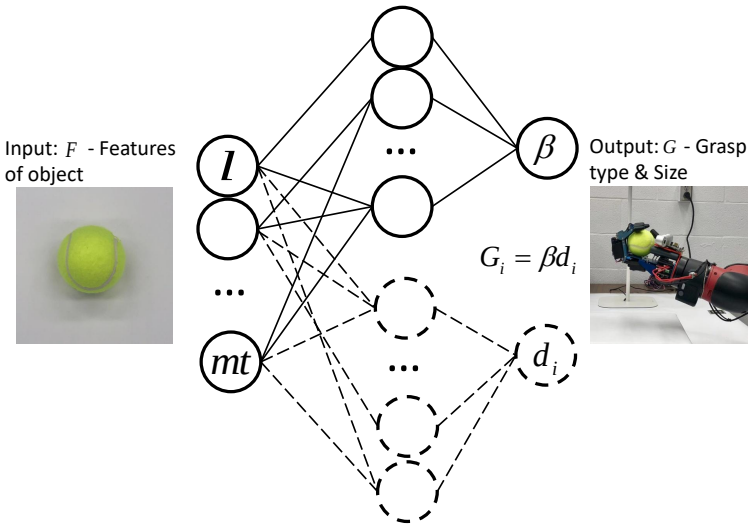

Figure 3.5: Neural Network Classification Model for grasp pose selection and grasp size determination. Two neural networks were employed, one was used for grasp type d_i selection and the other was able to decide grasp size β. Input for both networks are perceived features of target object F.

where G_i is learned grasping postures, β is a scalar that related to the size of an object and d_i is the pre-defined grasping choices. With the problem simplified, we now need to generate proper grasping strategies based on the information gathered by the MagicHand system, that is, map features of objects F to grasp G_i,

$$F = [l,\, h,\, w,\, m,\, s,\, r,\, mt] \rightarrow G_i$$

features l, h and w are dimensions along orthogonal directions of target objects and $l \geq h \geq w$. Features m, s, r and mt are stand for mass (grams), shape, rigidity and material type of the target object. We test the performance of the context-aware grasping system using grasp selection neural network model in [57, 54]. The model considers learning grasp type and grasp dimension are independent from each other and includes two separate neural networks (Fig. 3.5). The network contains a hidden layer with 75 neurons which employ ReLU as activation functions and an output layer with Softmax as its activation function. Also grid search is used to tune L_2 and α is set to 3.6. The cross-validation score for both grasp types and grasp dimensions are around 80%.

3.5 Experiments

Ten grasping experiments are performed to test the accuracy of the system. For each trial, the system first localizes the object and the hand navigates around the object to collect RGB-D data to reconstruct the object in 3D, this process takes about 60 seconds. Then the hand put the SCiO sensor as close to the object as possible to gather NIR spectrum which is used for material detection and characteristics mapping. Based on both NIR spectrum and 3D model of the target object, a grasping strategy is generated and executed to grasp the object. This step takes about 30 seconds and the overall process takes approximately 90 seconds.

3.5.1 Materials Classification

Near-infrared spectra of six types of materials including ceramic, stainless steel, wood, cardboard, plastic and glass were collected in this work. All these materials covered a major majority of daily objects. A total of 54 different daily used objects were selected. This material/object selection pattern would expand the universality of our datasets. Two NIR datasets were established to train the material/feature detection algorithm. MLP dataset was collected to train a MLP network for material classification while AE dataset is used to train our autocoder to reduce noise in practical spectra.

Material Recognition Results

We extracted the spectra of one object from each material in the MLP dataset. This part of the data (1803 scans) was separated and used as the validation set to evaluate the final performance of the classifier. The rest of data was divided into training and testing set. 70% of the data was used to train the network and 30% of the data was used as the testing set. We tuned the hyper-parameters as follows: 64 batch size, 50 epochs and ADAM optimizer with 0.0001 learning rate. The MLP network achieved a 99.64% test accuracy and 1.95%

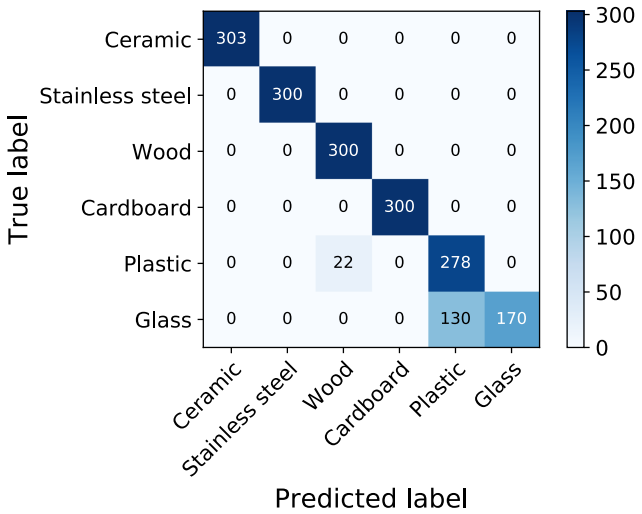

Figure 3.6: Confusion matrix for MLP network.

test loss. The pre-extracted validation data (1803 scans) which was completely new to the algorithm were tested with the algorithm. A total 1609 correct predictions were achieved by the algorithm as shown in Fig. 3.6. The results show that MLP dataset had a very efficient performance in recognizing ceramic, stainless steel, wood and card board while had a less performance on predicting plastic and glass objects due to the characteristics of those two materials. The autoencoder (Fig. 3.2) was trained with the AE dataset to reduce noises and errors.

To evaluate the performance of the whole material/features recognition system, we selected another twelve new objects which fall into the six material categories and collected spectra of those objects in a practical environment (temperature $63^{\circ}F$ and less light intensity). 60 spectra were collected (five scans per object) by the MagicHand platform and the classifying results achieved 60/60. Although the prediction result was perfect, there were two reasons that might lead to this result. Firstly, the autoencoder greatly reduced noises from practical spectrum. Secondly, there were only sixty testing samples in this experiment. As the number of testing samples increases, the accuracy is expected to decrease.

Table 3.2: Dimensions of Target Object

Objects	Measured Dimensions (mm)	Estimated Dimensions (mm)
1	$151 \times 73 \times 73$	$149.9 \times 74.0 \times 73.0$
2	$152 \times 38 \times 38$	$154.2 \times 41.4 \times 35.3$
3	$244 \times 71 \times 71$	$246.3 \times 69.6 \times 71.2$
4	$149 \times 44 \times 44$	$152.6 \times 46.4 \times 41.3$
5	$154 \times 77 \times 26$	$152.4 \times 75.4 \times 27.6$
6	$65 \times 65 \times 65$	$65.9 \times 67.6 \times 68.0$
7	$166 \times 82 \times 19$	$163.5 \times 81.2 \times 19.1$
8	$114 \times 29 \times 29$	$112.7 \times 28.5 \times 30.2$
9	$59 \times 29 \times 29$	$61.2 \times 29.7 \times 27.8$
10	$71 \times 71 \times 71$	$74.6 \times 72.9 \times 71.5$

3.5.2 Dimension Estimation

Before grasping, the robot arm took a sequence of RGB-D images around the target object to reconstruct the 3D model of the target object, dimensions were then estimated

based on the 3D model by our algorithm. The detailed dimensions are shown in table 3.2. In the table, measured dimensions indicate the real world dimensions collected with a 0-150 mm digital caliper and a steel tape, the estimated dimensions were the estimated by our algorithm and the errors were less than 4 mm.

3.5.3 Object Grasping

With the characteristics and dimensions of target objects, we tested the grasp strategy generation system on ten objects. If the robot hand could hold/lift an object for 10 seconds without dropping it, we counted the grasp as a success. The experiment results showed 8 out 10 successes on picking up the objects as shown in Fig. 3.7. There were two failures in the robot trials, the first failure was a plastic tube (object 8), the system gave a mismatch on grasp selection which led to a fail grasp. The other failure was on a wood block (object 9), the system gave a proper grasp choice, but the robotic hand slipped due to inadequate friction.

3.6 Summary

In this paper, we presented a context-aware dexterous grasping system called MagicHand. This system is able to collect NIR spectrum of object to achieve material classification with an accuracy of 99.64%. The dimensions of object is estimated based on the 3D model of target object and robot trials was performed on 10 objects with 8/10 successes. From the experiment results, we conclude that the MagicHand system is efficient in perceiving useful information about its target object and generate proper grasping strategy base on perceived information in different working environments.

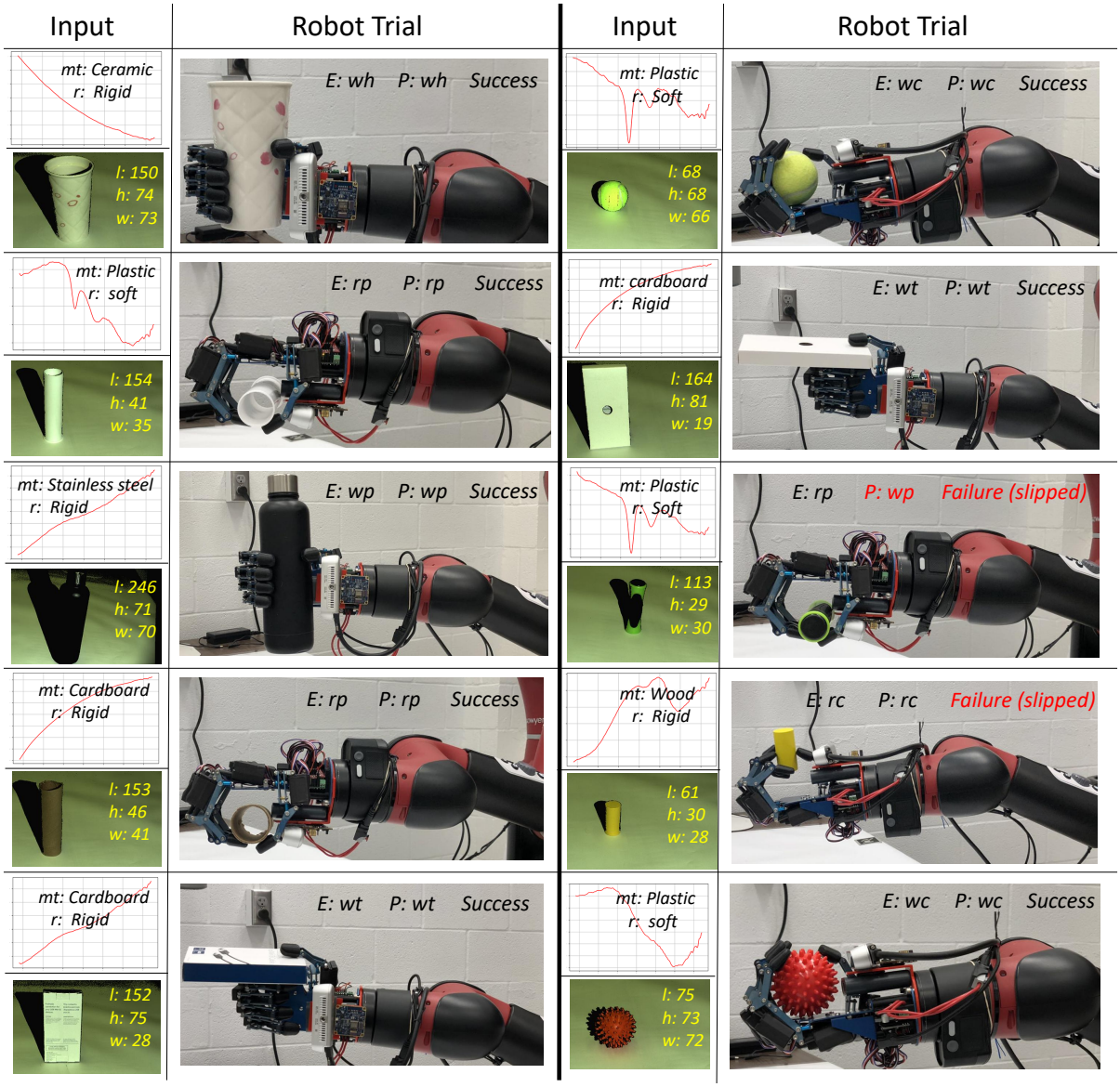

Figure 3.7: Robot grasp experiments with 10 objects. E stands for expected grasp type while P is predicted grasp type. Failure indicates the robot was not able to pick up the object and P in red means predicted grasp type does not match expected grasp type.

CHAPTER 4

LEARNING TASK-ORIENTED DEXTEROUS GRASPING FROM HUMAN KNOWLEDGE

4.1 Introduction

We propose a task-oriented dexterous grasping approach, which can generate and deploy a proper grasping strategy based on task designation and object affordances. To represent the relation between object affordances, task designations, and human grasping strategies, a task-oriented object grasping dataset was developed, which enables learning grasping strategies of human experience. We designed a deep learning network to discover the human knowledge of grasping from the dataset. We also implemented a reinforcement learning mechanism to deploy the selected grasping strategies. The workflow of the proposed system can be found in Fig. 4.1. The learning-based grasping balances grasping flexibility and dexterity by adapting to various grasp contexts.

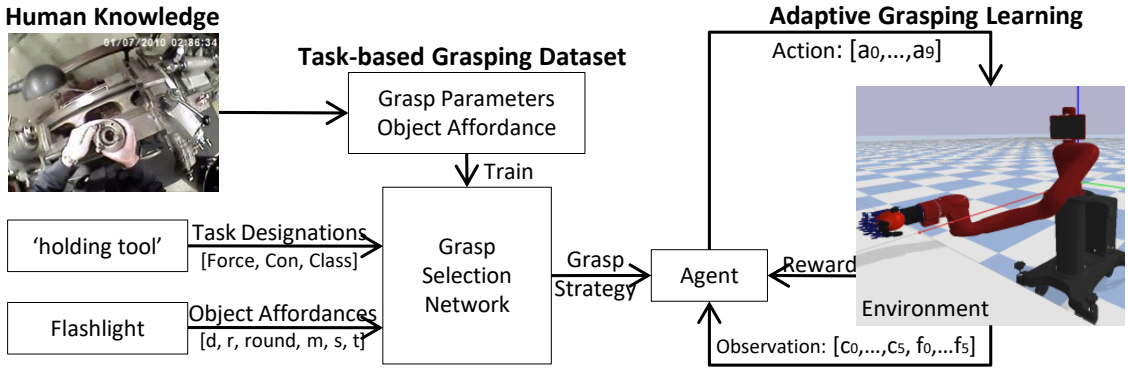

Figure 4.1: Workflow and structure of the task-oriented grasp strategy generation system. Details of object affordances and task designation can be found in Table 4.1 while action and observation are specified in Section 3.

4.2 Learning Grasp Strategies

Grasping strategies are related to object affordances and task designations [20]. The problem of grasp selection considering grasp context is a challenging problem, due to many factors such as task representation, object property measurement, and the determination of a suitable grasp strategy. To address this challenge, we implemented task-oriented dexterous grasping through learning human knowledge in grasping. We developed and trained a deep

learning neural network to predict proper grasp strategies.

4.2.1 Representing Human Grasping Knowledge

We established a dataset for task-oriented object grasping based on the Yale human grasping dataset [8]. The Yale human grasping dataset was recorded by two housekeepers and two machinists about their regular work activities including 27.7 hours of tagged video. A spreadsheet with 18210 entries was established based on the videos including task designations, object affordances, and labeled with grasp parameters. Task designations, object affordances, and grasp parameters are defined in Table 4.1.

Table 4.1: Features in the proposed task-oriented object grasping dataset

	Parameters	**Description**
Grasp	Grasp type	Grasp choices defined in [23]
	OppoType	Opposition type of a grasp: {Pad, Palm, Side}
	PIP	Power, intermediate or precision
Task	Task names	High-level task description
	Force	Type of forces required : {weight, interaction}
	Constraint	Constraints of the task
	Class	Function class of the task: {hold, feel, use}
Object	Dimensions	Dimensions of object in *cm*: {A, B, C}
	Rigidity	Rigidity of the object
	Roundness	Dimensions along which object is round
	Mass	Mass of the object in *gram*
	Shape	Basic shape class [86]
	Type	Object type, as defined by [20]
Duration	Duration	Length of the grasp instance

The grasping tasks in the Yale dataset are redundant and ambiguous in task descriptions. We, therefore, refined the task list to 74 unique tasks by combining the tasks with similar actions. The dataset includes 31 unique types of grasp topology some of which are identical or similar, and the total number of grasp types can be reduced to 15 [21]. The refined dataset includes 15 grasp topology as shown in Fig. 4.2, 255 unique objects, and 74 unique actions. Some examples of the dataset are shown in Table. 4.2. In the table, we can see that different

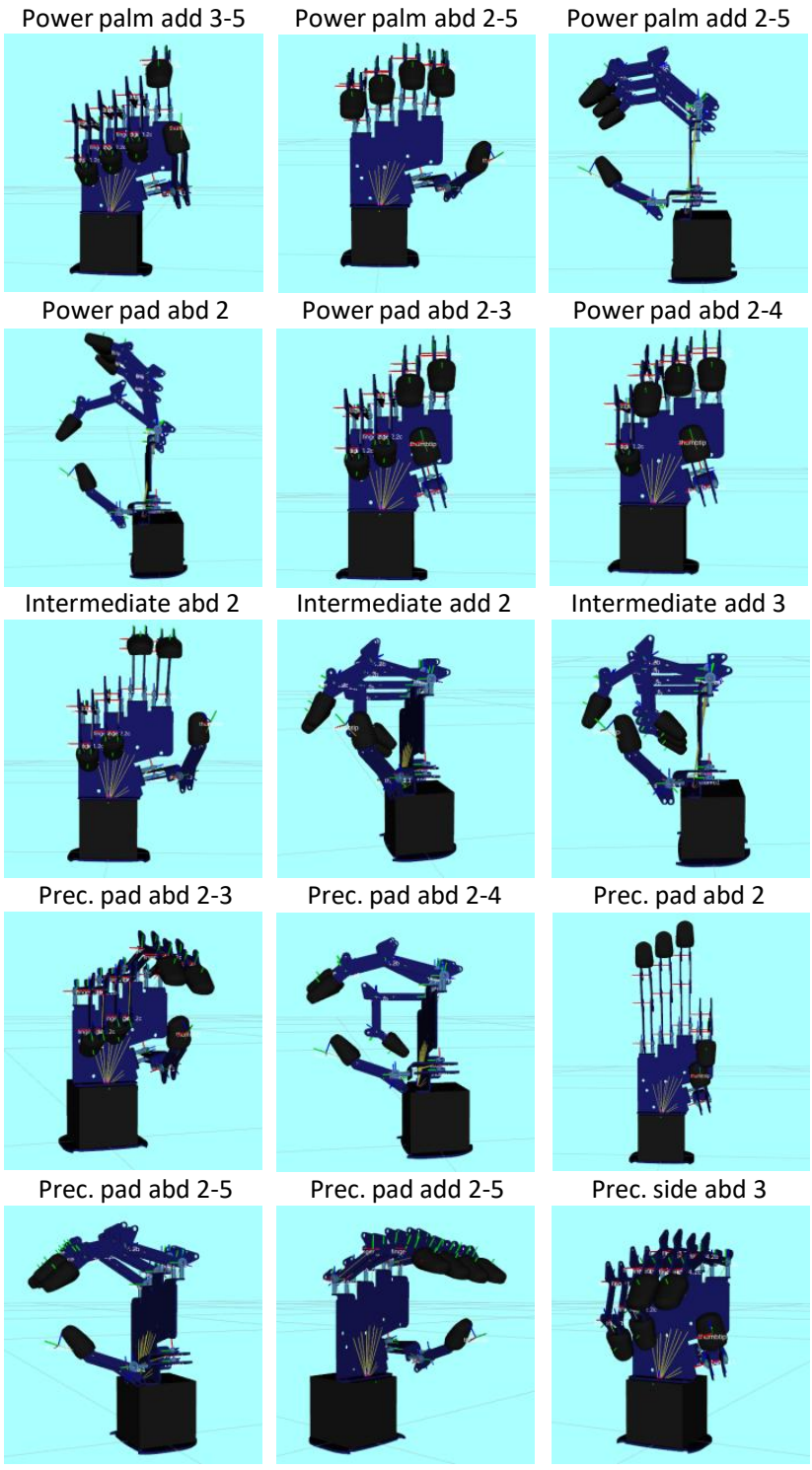

Figure 4.2: Defined grasp topology demonstrated in simulation environment.

Table 4.2: Examples of the task-object grasp dataset

Object	A	B	C	Rigidity	Roundness	Mass	Shape	Type	Duration	Task	Task_group	OppoType	PIP	Grasp_group
aerosol can cap	6	6	5	squeezable	c	16	equant	cubic cylinder	5	arranging object	arranging	Pad	Precision	Precision pad abd 2-3
aerosol can cap	6	6	5	squeezable	c	16	equant	cubic cylinder	4	securing cap	securing	Pad	Power	Power pad abd 2-3
air hose	15	3.25	1.75	rigid	a	231.5	bladed	ellipse A	11.9	blowing away chips	blowing	Palm	Power	Power palm abd 2-5
air hose	15	3.25	1.75	rigid	a	231.5	bladed	ellipse A	2.1	cleaning off part	cleaning	Palm	Power	Power palm abd 2-5
flashlight	15	2	2	rigid	a	215	prolate	cylinder	20.4	moving object	moving	Side	Intermediate	Intermediate add 3
flashlight	15	2	2	rigid	a	215	prolate	cylinder	1.64	holding tool	holding	Pad	Precision	Precision pad abd 2-3
hammer	15	2	2	rigid	a	400	prolate	cylinder	0.36	holding tool	holding	Side	Intermediate	Intermediate add 3
hammer	15	2	2	rigid	a	400	prolate	cylinder	2.04	pounding hammer	pounding	Palm	Power	Power palm add 2-5
metal Block	12.5	8.5	1	rigid	non-round	135	oblate	short prism	1.6	moving object	moving	Palm	Power	Power palm abd 2-5
paper towel roll	15	9	9	squeezable	a	200	prolate	cylinder	2	arranging object	arranging	Pad	Precision	Precision pad abd 2-5
part on machine	15	9	9	rigid	a	1000	prolate	cylinder	2	holding part	holding	Palm	Power	Power palm abd 2-5
remote	15	6	2.5	rigid	non-round	165	bladed	irregular	1.16	arranging object	arranging	Pad	Power	Power pad abd 2
remote	15	6	2.5	rigid	non-round	165	bladed	irregular	1.8	moving object	moving	Pad	Precision	Precision pad abd 2-5
rod	15	1.25	1.25	rigid	a	300	prolate	cylinder	3	adjusting rod	adjusting	Pad	Precision	Precision pad abd 2
rod	15	1.25	1.25	rigid	a	300	prolate	cylinder	3.44	holding part	holding	Palm	Power	Power palm add 2-5
ruler	15	2.25	0.25	rigid	non-round	32.5	bladed	irregular	17.04	holding tool	holding	Side	Intermediate	Intermediate add 3
ruler	15	2.25	0.25	rigid	non-round	32.5	bladed	irregular	0.4	pinching tool	pinching	Pad	Precision	Precision pad abd 2
screwdriver	15	3.5	3.5	rigid	a	100	prolate	cylinder	7.17	turning handle	turning	Pad	Precision	Precision pad abd 2-3
screwdriver	15	3.5	3.5	rigid	a	100	prolate	cylinder	1.88	holding tool	holding	Side	Intermediate	Intermediate add 3
small knob	4.5	2.5	2.5	rigid	a	150	prolate	cylinder	0.04	opening cabinet	opening	Side	Intermediate	Intermediate add 2
small knob	4.5	2.5	2.5	rigid	a	150	prolate	cylinder	10	turning knob	turning	Pad	Precision	Precision pad abd 2-3
sponge	10	6	2.5	squeezable	non-round	17.5	bladed	irregular	1	wiping counter	wiping	Pad	Precision	Precision pad abd 2
sponge	10	6	2.5	squeezable	non-round	17.5	bladed	irregular	3.9	moving object	moving	Palm	Power	Power palm add 2-5
spray bottle	15	6	3.5	squeezable	non-round	400	bladed	irregular	2	picking up	picking up	Pad	Precision	Precision pad abd 2-4
spray bottle	15	6	3.5	squeezable	non-round	400	bladed	irregular	25	holding	holding	Palm	Power	Power palm abd 2-5
thin rod	11.5	0.65	0.65	rigid	a	105	prolate	cylinder	20.36	applying gel to part	applying	Side	Intermediate	Intermediate add 3
thin rod	11.5	0.65	0.65	rigid	a	105	prolate	cylinder	8.76	turning tool	turning	Pad	Precision	Precision pad abd 2-3
wrench	15	2	0.75	rigid	non-round	250	bladed	irregular	2	holding tool	holding	Pad	Precision	Precision pad abd 2-3
wrench	15	2	0.75	rigid	non-round	250	bladed	irregular	10	cranking wrench	cranking	Palm	Power	Power palm add 2-5

tasks on the same object may result in different grasp selections.

Besides, the Yale human grasping dataset does not provide 3D models of objects. We collected 157 3D object models that are similar to the objects in the dataset.

4.2.2 Learning Grasp Strategies

Task designations can be easily extracted from a task description using a natural language processing algorithm named SpaCy. A task is usually a verb while the target object is a noun. Cosine similarity S is computed between the extracted action a and each unique action A_k in the task-oriented object grasping dataset. The action A_k which maximized S will be selected as the action for this task, and other task designations of A_k would also be used for this task. Object affordances can be obtained using our previous work [57, 39].

The task-oriented object grasping dataset is further processed to train the model. The categorical features in the dataset are converted to indicator values which expand the number of features in the dataset to 127. The continuous features such as dimension and mass are normalized to improve the robustness of the network. In real-world scenarios, more than one grasp topology could be used for a certain task. We, therefore, designed a multi-class

multi-label grasp selection network to predict grasp topology. The network includes three subnetworks: the grasp topology selection network $g(f)$, the OppoType selection network $o(f)$, and the PIP selection network $p(f)$ where f is the input feature.

- The grasp type selection network $g(f)$ includes one input layer with 131 neurons, three hidden layers with 2^{11}, 2^8, and 2^6 neurons, and an output layer with 15 neurons. All units in the input layers and hidden layers are activated by ReLU while the output layer uses a sigmoid activation function. The output of $g(f)$ is a set of probability distributions over 15 grasp types.

- The OppoType selection network $o(f)$ and the PIP selection network $p(f)$ have the same structure. They share the input layer and the first two layers of hidden layers with $g(f)$ and their output layer contains three neurons with a softmax activation function.

The task-oriented object grasping dataset was divided into two parts. The training and validation part contains 90% of grasping samples, and the test data includes the rest 10% (650). The algorithm was trained with 4-fold cross-validation. The hyper-parameters include batch size 64, 5000 training epochs, and learning rate 0.001 using an Adam optimizer. We achieved training accuracy of 85% for $o(f)$ and $p(f)$, and a 100% hit rate for $g(f)$.

4.3 Adaptive Grasp Deployment

The predicted grasp strategies need to be deployed to grasp objects. Since the grasp size may vary depending on the shape, size, or even rigidity of the object, we trained a deep reinforcement learning network to deploy adaptive grasps for different objects. The traditional grasping task requires complex calculation on path and grasp points, and it is not adaptive enough for task-oriented grasping. In this paper, we achieved adaptive grasping by designing and training a reinforcement learning algorithm that allows the robot to learn to grasp by itself and respond to unforeseen environments. A Proximal Policy Optimization

(PPO) reinforcement learning algorithm [64] was implemented using actor-critic policy [37] to teach the robot to grasp and move a target object adaptively. PPO updates its policy using previous experience

$$\theta_{new} = \arg\max_{\theta} E[\min(r_t(\theta)A_t,\, clip(r_t(\theta),\, 1-\varepsilon,\, 1+\varepsilon)A_t)] \tag{4.1}$$

where θ is policy parameter, θ_{new} is updated policy parameter, E indicates empirical expectation over time steps, and r_t is the ratio between the new policy and the old policy. The term A_t is an estimator of the advantage function at time t, ε is a small number usually 0.1 or 0.2, and *clip* will clip A_t while r_t is not between $1-\varepsilon$ and $1+\varepsilon$.

Since we focus on adaptive grasp deployment, fixed grasp paths and object locations are used in this work. Each joint in the hand is controlled by a scaler $a_i \in [0, 1]$. The joint will fully extend when $a_i = 0$ and completely close while $a_i = 1$. The AR10 hand has 10 joints, and the action function is a = $[a_0,\, a_1,\, a_2,\, a_3,\, a_4,\, a_5,\, a_6,\, a_7,\, a_8,\, a_9]$. The algorithm will control the hand to grasp the object using the action function based on the configuration of the selected grasp topology. In this way, the robot is able to adjust the grasp size while keeping the configuration of the selected grasp topology.

We modeled the hand into six parts: palm, index finger, middle finger, ring finger, little finger and thumb. The observation of the algorithm is a 12 dimensions vector $\mathrm{O_b} = [c_0,\, c_1,\, c_2,\, c_3,\, c_4,\, c_5,\, f_0,\, f_1,\, f_2,\, f_3,\, f_4,\, f_5]$ where c_i is the closest distance between each part of the hand and the target object, and f_i indicates norm force provided by each part of the hand.

A steady grasp prefers more contact points, also the contact force should not be too large or it would either break the object or waste energy. To achieve optimized grasps, we define the reward function as

$$R = \begin{cases} -1000 & n = 0 \\ \beta_1 n - \beta_2 f & n > 0 \end{cases} \tag{4.2}$$

where n is the number of contact points on the target object, f is the total force, and β is the scaler to control the weight of n and f. If a grasp is a success, $R = R + 1000$. In this work, we define a grasp as a success when the hand is able to grasp the target object, move the object above a certain height, and hold it for a certain amount of time. This reward function would reward more contact points and punish large contact forces.

A total of 15 predefined grasp topology was developed to grasp and move an object in the simulation environment. A trial is terminated and reset whenever it achieves a success grasp or it runs out of time.

The training set contains 537 data samples. We also added a small noise (less than 5 mm) to the object position to improve the adaptivity of the algorithm and extended the size of the training set to 5370. The network is trained with learning rate $2.5e^{-4}$ and discount factor 0.9. The accuracy on the training objects achieved 88.2%.

4.4 Experiments

We conducted experiments to evaluate the proposed algorithm and to demonstrate its performance in a simulation environment.

4.4.1 Experiment Setup

A simulation environment as shown in Fig. 4.3 was developed to evaluate the performance of the proposed algorithm. The environment includes a robot and a workspace. The robot contains an AR10 robotic hand and a Sawyer robotic arm, which are attached together to deploy manipulation tasks. A table is placed in front of the robot as a workspace and objects are placed at a fixed location on the table. The environment is developed with Pybullet using Python 3.5 in Ubuntu 16.04.

About 10% (650) of the data from the task-oriented object grasping dataset was selected and used for testing. A random noise that follows the standard deviation was added to the continuous features of the remained data, and the size of the training set was increased to 136627.

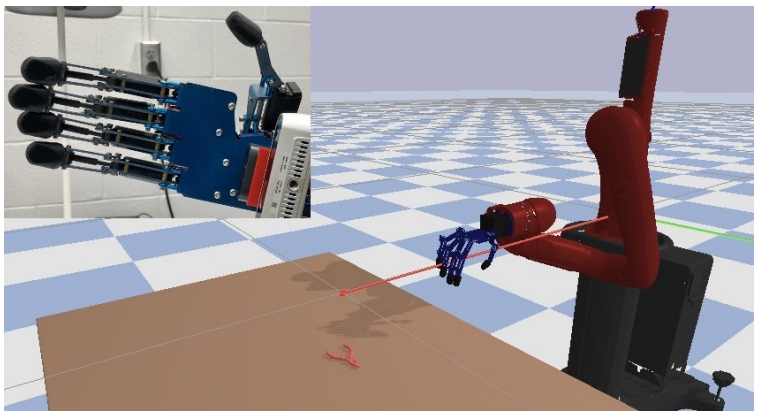

Figure 4.3: A simulation environment that allows complex grasping tasks.

Not all of the 157 3D object models were used to train/test the reinforcement learning network. Some of the 3D objects are either too large for the workspace or too small to be grasped by the robotic hand due to physical limitations of the hand. A number of 103 3D objects were used in this work, where 86 objects were used for training the adaptive grasp deployment network and 17 objects were used for testing the algorithm.

4.4.2 Grasp Strategy Learning

The proposed grasp selection network was tested with 650 data samples that are completely new to the algorithm. The result of the grasp selection network achieved a hit rate of 100% and a top-3 match rate of 98.6%. Since the grasp selection network is a multi-label model, we defined the hit rate as the probability that the top-1 predict label is in the target label set. The top-3 match rate indicates how often the top-3 predictions are in the targets. The ROC curve of each class can be found in Fig. 4.4. From the figure, we can see that the AUCs for all classes are close to 1, demonstrating the performance of the grasp selection network. The testing accuracy was a little bit higher than the training accuracy. Two reasons may lead to this result. Firstly, the training set has been augmented, so that it is large enough to cover most object-task-grasp combinations. Secondly, there were only 650 testing samples in this experiment, and the accuracy may decrease as the number of testing samples increases.

The accuracy of the OppoType selection network and the PIP selection network was

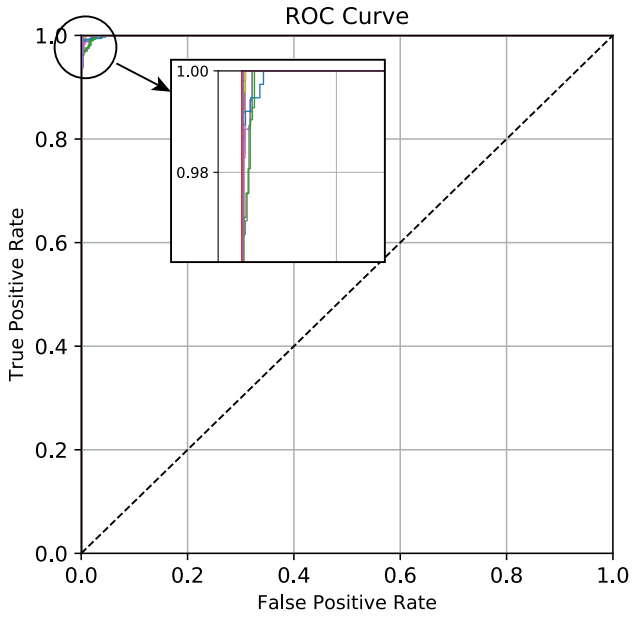

Figure 4.4: ROC curves for all 15 classes of the grasp selection network.

85.9% and 86.3% (Fig. 4.5) which were lower than the accuracy of the grasp selection network. Since the OppoType and the PIP selections are not designed as multi-label models, there is only one target label for each data sample. This leads to lower accuracy on the OppoType and PIP prediction. Although each grasp type is directly related to an OppoType and a PIP, the experiment results show that the proposed network is able to predict all grasp parameters with high accuracy.

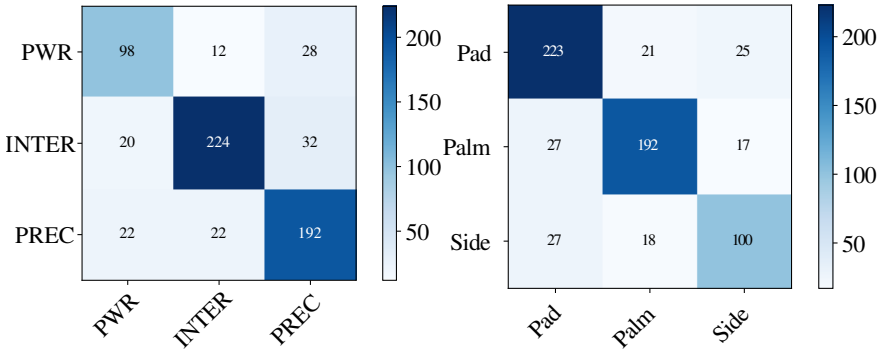

Figure 4.5: Confusion matrices of the PIP and OppoType selection network. PWR, INTER and PREC are short for Power, Intermediate and Precision.

4.4.3 Simulated Grasping of Arbitrary Objects

Since the AR10 hand only has 10 DOFs and can not perform some of the predefined grasp topology, a total of nine predefined grasp topology was employed in the simulations. Each grasp topology was tested with three new objects, and each object was grasped 100 times.

A total of 17 objects and 2700 grasping experiments were performed to evaluate the adaptive grasping learning system as shown in Fig. 4.6, and the corresponding results can be found in Table 4.3. In the experiment, if the robotic hand successfully grasped a target object, picked it up for 20 cm, and held the object for 20 time steps (0.2 seconds) without dropping it, the grasp is defined as a success. A number of 2312 successes were achieved, and the final success rate was 85.6%.

Table 4.3: Results for simulated Grasping in Fig. 4.6

Grasp Topology	**Grasping Results**		
Power pad abd 2	94	93	85
Power pad abd 2-4	91	82	75
Power palm abd 2-5	89	83	78
Power palm add 2-5	82	98	77
Prec. pad abd 2	85	84	93
Prec. pad abd 2-3	95	74	96
Prec. pad abd 2-4	91	75	78
Prec. pad abd 2-5	89	98	76
Prec. pad add 2-5	82	78	91

Three reasons may lead to these results. The first one is that the testing data was completely new to the algorithm, and some features might not be covered by the training set. Secondly, the robot hand is not dexterous enough for certain objects. For example, the success rates on the wrench were low for most of the grasp topology, since the wrench was very thin and it was hard for the robot to grasp due to physical limitations. Finally, the 3D object models are not perfect and some of the models are not accurate enough. For example, for the banana shown in Fig. 4.6, we can see that the shape collision model of the banana is not the same as its visual model.

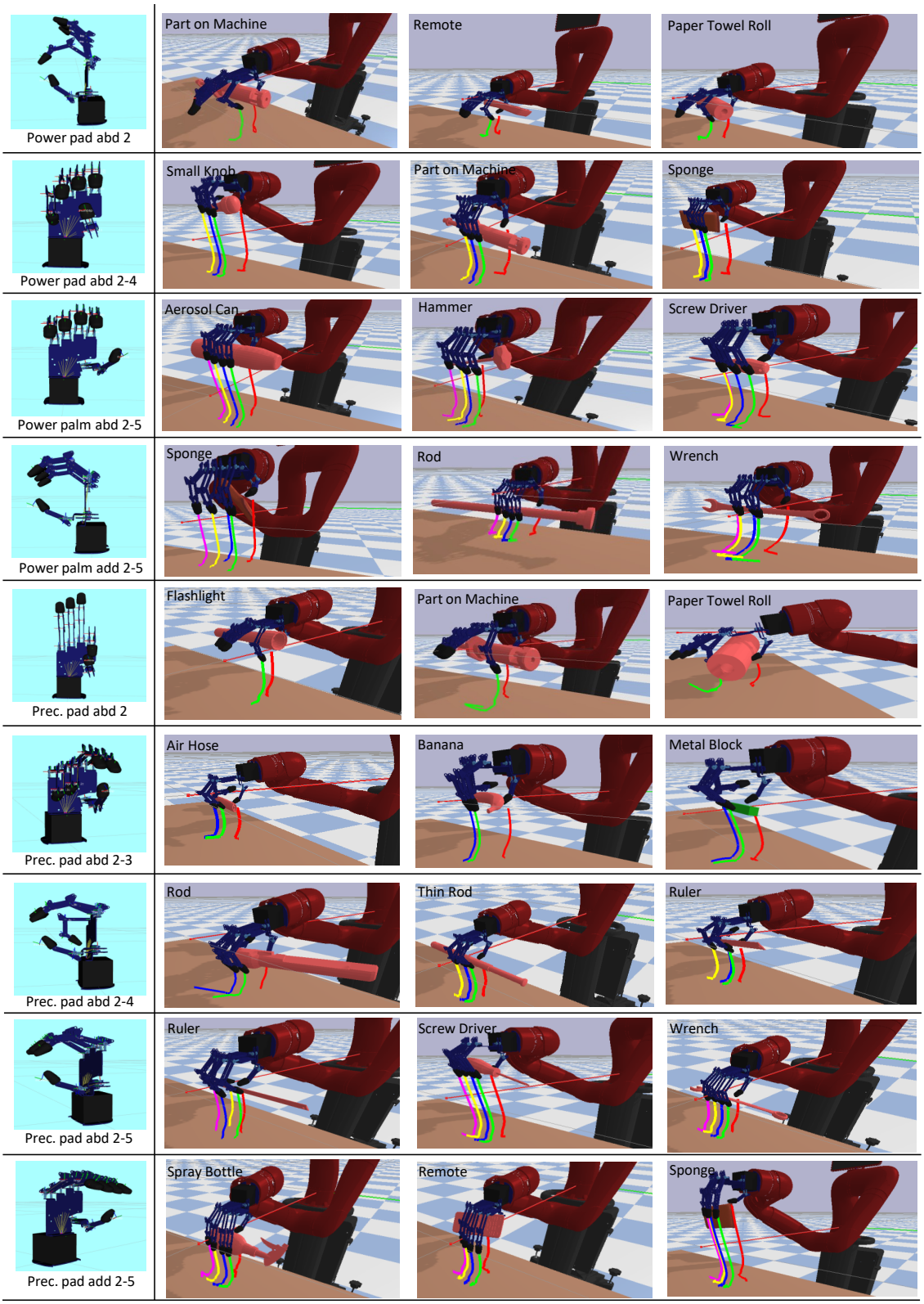

Figure 4.6: Grasping simulations on 17 arbitrary objects. Colored lines indicate trajectories of fingertips while grasping.

4.5 Summary

We presented a grasp learning algorithm that is able to learn grasp topology from human knowledge and deploy the grasp adaptively. A multi-output multi-label neural network was developed to predict grasp topology based on task designations and object affordances, and the top-1 accuracy for this network achieved 100%. A Proximal Policy Optimization reinforcement learning algorithm was implemented and trained for adaptive grasp learning. A total of 2700 grasping experiments were performed on 17 different objects and the overall accuracy achieved 85.6%. The performance of the system was demonstrated by the experiment results. Compared to deciding grasp topology for a task, choosing a proper grasp location based on a specific grasp pose, and approaching the hand to a suitable position and orientation to that location are even more challenging. Our further work will focus on those two problems, and the final goal is to develop a fully automated dexterous grasping system for both social and manufacturing environments.

CHAPTER 5

KNOWLEDGE AUGMENTATION AND TASK PLANNING USING LARGE LANGUAGE MODELS

5.1 Introduction

Previous works have successfully mapped object features to proper grasp strategies [41, 57], yet the deployment of these strategies remains a challenging task. The current research employs learning or planning approaches to address this problem, but these methods either require a large amount of data or a comprehensive understanding of the grasping environment. In this work, we further explore the intuitive reasoning abilities of the large language model to tackle this problem efficiently.

We propose a Cognition Based Grasping System, as shown in Fig. 5.1, to guide humanoid robots in accomplishing dexterous grasping of daily objects using incomplete features. The system consists of five models: Object Perception (OP), Feature Complementation (FC), Grasp Selection (GS), Task Planning (TP), and High Level Robot Control Library (HLRC). The workflow of the system is illustrated in Fig. 5.1. The Object Perception model perceives the object and gathers useful information about both the object and the environment. The Feature Complementation model estimates and complements the features collected in the Object Perception model. The Grasp Selection model maps the complemented features to appropriate grasp strategies. The Task Planning model generates functional code to guide the robot in executing the task, and the High Level Robot Control Library provides functions to adapt to the code and control the robot.

5.2 Cognition Based Grasping System

Dexterous grasping in unstructured environments could be challenging due to the lack of environmental information and the ability for intuitive reasoning and planning. We, therefore, developed a Cognition Based Grasping System to solve this problem. This system complements perceived object features, maps them to grasp strategies, and guides and performs their execution using common sense and intuitive reasoning. In this section, we provide a detailed explanation of the proposed system.

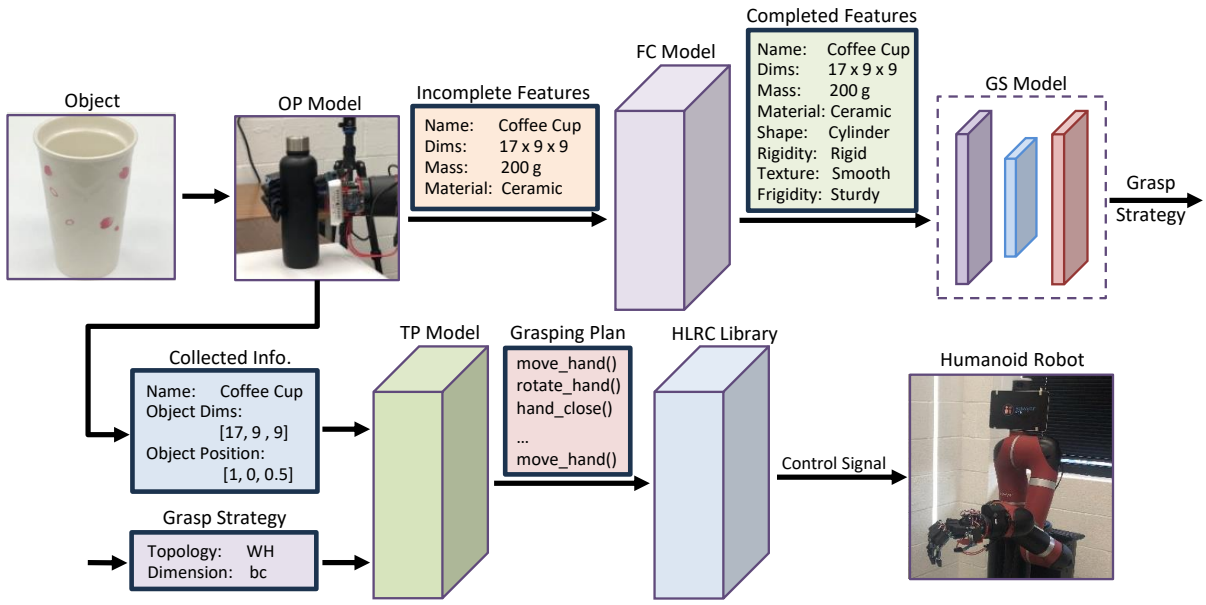

Figure 5.1: The workflow of the proposed system. In the figure, OP stands for Object Perception, FC represents Feature Complementation, GS denotes Grasp Selection, TP is the abbreviation for Task Planning, and HLRC is short for High Level Robot Control Library.

5.2.1 Object Perception

The Object Perception model utilizes the MagicHand platform to acquire object features, including its name, mass, material, and dimensions, and information about the environment such as the position and orientation of the object. The MagicHand platform integrates multiple sensors, including RGB-D cameras, Force Sensing Resistors (FSR), and a SCiO sensor, to enhance perception abilities. RGB-D cameras are used for object recognition, 3D modeling, and determination of object coordinates and orientation. FSR sensors are integrated into the tips of a five-digit robotic hand to enable the detection of contact force. The SCiO sensor collects the near-infrared spectrum of the object for material recognition [40]. An electrical scale is also employed in the model to measure the mass of the object. The dimensions of the target are derived from its 3D model. Further details of the MagicHand platform can be found in our previous work [39].

5.2.2 Feature Complementation

Although certain information about the object and the environment can be collected by the Object Perception model, perceiving features such as fragility, rigidity, shape, and textures can still be challenging. Traditional methods for acquiring these attributes, often involve costly chemistry or physical analysis and are not suitable for real-time dexterous grasping. It is straightforward to estimate these attributes based on the recognized features using common sense, similar to human reasoning. We, therefore, developed a Feature Complementation model by leveraging a large language model which has a basic understanding of common objects and common sense.

Even though Large language models possess knowledge and proficiency in common-sense and logical reasoning, they demonstrate limitations in problem-solving [5]. LLMs are sensitive to input phrasing and often output misleading and overgeneralized knowledge. To address this issue, we precisely define and categorize each feature and build prompts to prompt the LLM to generate consistent and unambiguous estimations of missing object features.

Feature Definition and Categorization

We provide clear definitions and categorizations for physical characteristics including fragility, rigidity, shape, and texture. For each feature, we explored different definitions of that feature from various dictionaries, online resources, and research papers. Then, we input each definition into the LLM and observed the output feature name. The definition that yields the most consistent and accurate results was selected as the definition for that particular feature.

While estimating the feature of an object, humans often provide a binary response, such as “smooth” or “tough” for texture and “rigid” or “soft” for rigidity. Thus, we emulate this strategy to categorize features including fragility, rigidity, and texture. However, classifying the shape of an object into binary classes is not possible. Instead, we categorize the shape

with the most basic shapes in geometry. In addition, a complex shape can be treated as a combination of these basic shapes. The definitions and categorizations for each feature are shown in Table 5.1.

Table 5.1: Feature definitions and categorizations.

Features	Definition	Categorizations
Fragility	Tendency to break, shatter, or deform when subjected to external forces or stress.	{fragile, sturdy}
Rigidity	Ability to resist deformation or bending and retain its shape and structural integrity when external forces are applied.	{soft, rigid}
Shape	The overall form and structure of an object.	{sphere, cube, cone, cylinder, cuboid, disk}
Texture	physical characteristics and qualities, such as roughness or smoothness, of the outer layer of an object	{smooth, rough}

These definitions and categorizations reduce ambiguity and enhance the LLM's comprehension of those features, thus achieving a more accurate understanding of the attributes.

Prompt Implementation

The accuracy and the consistency of LLMs highly depend on the quality of the descriptions of the problem such as phrasing, relevant details, context, and specific requirements or constraints. To have LLMs generating more close output to the desired one, we formulated various prompts to enhance problem descriptions. The prompt yields the most precise and consistent result is shown in Table 5.2. An example input "calculator 15.4 7.9 1.5 116 plastic" yields the result "{shape : cuboid, texture : smooth, rigidity : rigid, fragility : sturdy}".

Table 5.2: Prompt for Feature Complementation.

Imagine you are helping me to estimate the physical features of an object based on some known features of that object. I will give you some features of an object, and you need to complement the features with common sense. The features that need to be complemented are Fragility, Shape, Texture, and Rigidity. Shape is defined as the overall form and structure of an object and is categorized as {sphere, cube, cylinder, cone, cuboid, disk}. Texture is defined as the physical characteristics and qualities, such as roughness or smoothness, of the outer layer of an object, and categorized as {smooth, rough}. Rigidity is defined as an object's ability to resist deformation or bending and retain its shape and structural integrity when external forces are applied, and categorized as {soft, rigid}. Fragility is defined as an object's tendency to break, shatter, or deform when subjected to external forces or stress, and categorized as {fragile, sturdy}. The output should be in JSON format. Only output Fragility, Shape, Texture, and Rigidity. Do not explain your answer.

5.2.3 Grasp Selection

The grasp of an anthropomorphic robotic hand defines a set of angles of the finger joints, and the magnitude of the contact forces applied by the fingers and palm to an object at the contact points. The objective here is to emulate human grasping by mapping object features f complemented from the Feature Complementation model onto grasp prioritization H

$$f = [a, b, c, m, s, mt, r, t, fr] \rightarrow H \tag{5.1}$$

where (a, b, c) are dimensions along orthogonal directions (a, b, c) and (m, s, mt, r, t, fr) are the mass, shape, material type, rigidity, texture and fragility of the object.

Grasp Definition

The complemented object features f may be inaccurate or even erroneous due to rough measurement and estimation. To enable imprecise measure of object dimensions and position,

and improving system robustness and adaptivity, We therefore implement grasp strategies in terms of grasp topology and grasp dimension. Grasp scales are determined by the orthogonal dimensions a, b, c around which the grasp closure occurs, as illustrated in Fig. 5.2. This labeling convention is commonly adopted in the literature [23], facilitating the computation of hand closure in forward and inverse kinematics. Grasp dimension d includes all feasible dimensions that can be utilized to grasp the object and

$$d \in \{a, b, c, ab, bc, ac, abc\} \tag{5.2}$$

where ab indicates the object can be grasped either around dimension a or around dimension b.

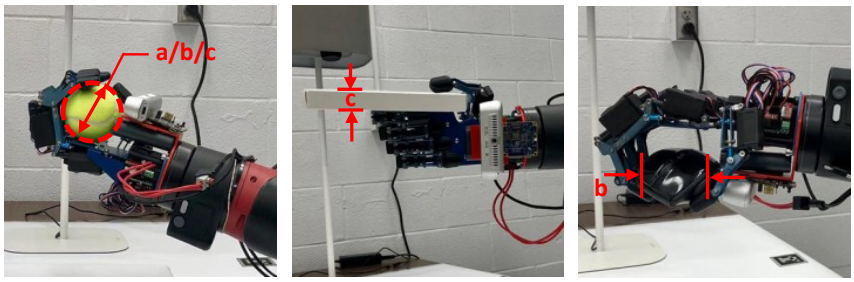

Figure 5.2: Illustrations of grasp dimension.

The grasp topology h is one of the grasp types drawn from the set of human grasp primitives

$$h \in \{wt, wp, wh, wc, rp, rc\} \tag{5.3}$$

where grasp types wt, wp, wh, wc, rp and rc, as shown in Fig. 3.4, are high level grasp topology adopted from the Grasp Taxonomy presented by Cutkosky [14]. This grasp definition can be easily applied to other type of hand-effectors by restricting the number of fingers used.

We developed a grasp classification system by combining the grasp type and dimension to overcome the inconsistencies in grasp dimensions for certain object-grasp associations.

This taxonomy ensures that each grasp topology is associated with proper dimensions and sizes. The extended grasp taxonomy is defined as

$$H \in \{rc.ab, rc.bc, rp.b, rp.c, wc.abc, wh.bc, wh.c, wp.bc, wt.c\} \tag{5.4}$$

Further details regarding this methodology can be found in our previous work [57].

Learning Grasping Strategies from Object Features

Most studies attempting to understand and codify human grasps have concluded that human grasp choice is a function of object affordances (geometry, texture etc.) and the task requirements (forces, mobility, etc.) [14, 52]. Attempts to assign one most suitable grasp for a given object-task combination have not been conclusive. The major problem is that even for the same specific object-task combination, there are multiple grasp choices possible, which offer appear to be arbitrary and not amenable for deterministic modeling. Human grasp choices nevertheless do tend to cluster when studied over a large set of objects. Both the clustering effect and the confusion between grasp types can be seen in the data presented by [23], which shows that a single object could be held in multiple different grasp types in the course of picking or handling. There is no one-to-one mapping of one object to one grasp type.

The problem of grasp selection is therefore not selecting one ideal grasp type but one of the many feasible grasp types in human grasp taxonomy for the given context. To that end, we plan to learn the mapping from features f into grasp topology distributions

$$f \rightarrow P(H|f) = [P(rc.ab|f), \cdots, P(wt.c|f)] \tag{5.5}$$

We designed a neural network to model the probability distribution over all grasp classes $\hat{P}(H|f)$, as illustrated in Fig.5.3. The network is designed with cross-entropy loss and optimized using stochastic gradient descent algorithms. The loss function is defined by cross

entropy that measures the deviation between the ground truth and predicted probability distribution

$$L(P(H|f), \hat{P}(H|f)) = \sum_i \sum_j P(H_j|f_i) \log \hat{P}(H_j|f_i) \tag{5.6}$$

where $i \in [1, N]$ with N as the number of observations and j is the index of grasp topology.

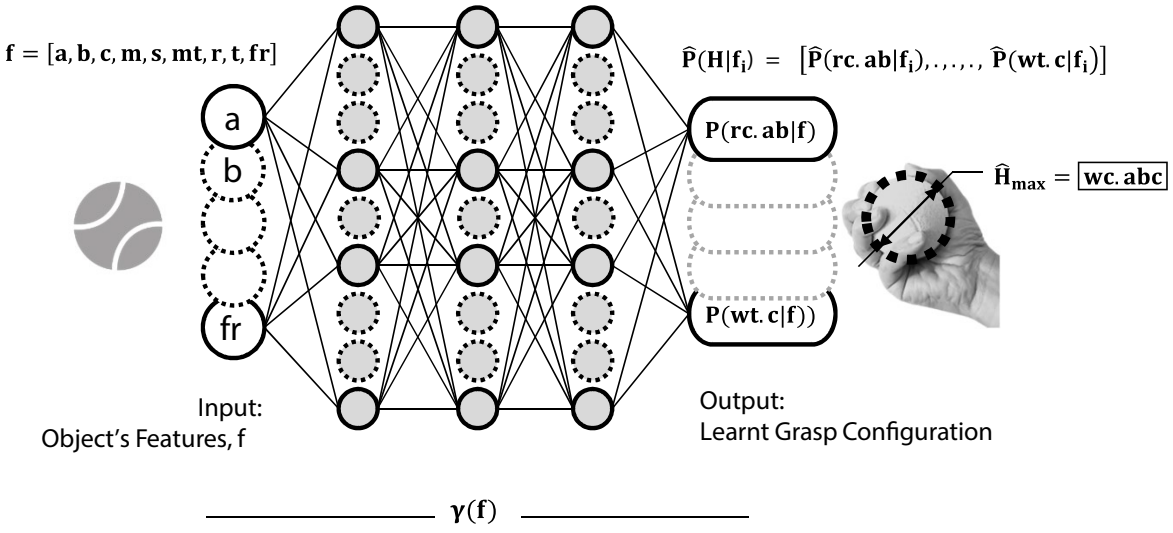

Figure 5.3: The Grasp Selection model.

The grasp with the maximum probability is chosen by

$$\hat{H}^*_{\max} = \arg\max_j \hat{P}(H|f) \tag{5.7}$$

the predicted grasp configuration $\hat{H}^*_{\max}$ contains information regarding the grasp type and object dimension along which the grasp can be executed, so $\hat{H}^*{}_{max}$ can be easily decomposed into grasp type h^* and grasp dimension d^*, which can be used subsequently to calculate robot hand configuration. The optimal grasp type is chosen as the one corresponding to the highest probability from the predicted probability distribution.

Because the model predicts probability distributions, we defined two scoring metrics for training and evaluation of the model. The predicted grasp choice is scored as a success if the same grasp type was chosen at least once in the evaluation dataset. The feasibility of

the grasp is scored as

$$F_l(P(H), \hat{P}(H)) = \begin{cases} 1 & P(\hat{H}_{\max}) > 0 \\ 0 & P(\hat{H}_{\max}) = 0 \end{cases} \tag{5.8}$$

where

$$\hat{H}_{\max} = \arg\max_j \hat{P}(H_j|f) \tag{5.9}$$

is the grasp topology with the maximal probability, and H is defined in (5.4). The feasibility score F_l is representative of the ability of the algorithm to pick a feasible grasp for a given object. The match score metric F_m is defined as

$$F_m(P(H), \hat{P}(H)) = \begin{cases} 1 & P(\hat{H}_{\max}) = P(H_{\max}) \\ 0 & P(\hat{H}_{\max}) \neq P(H_{\max}) \end{cases} \tag{5.10}$$

This match score is representative of the ability of algorithm to predict the most frequently applied human grasp as the grasp with the highest probability for a given object. In other words, F_m is akin to the accuracy. This metric is much more stringent and therefore we can expect the match score F_m to be always lower than feasibility score F_l

$$F_m(P(H), \hat{P}(H)) \leq F_l(P(H), \hat{P}(H)) \tag{5.11}$$

We used the feasibility score as the primary scoring metric, for the objective is to find one feasible grasp that can be successfully executed by a robot.

5.2.4 Task Planing

Traditional grasp deployment methods, relying on planning or learning approaches, require a deep understanding of the environment or extensive data for training which are not resource-efficient and sometimes impossible to achieve. In this section, we introduce a LLM-based Task Planing model, which leverages LLM's abilities of common sense and reasoning abilities, to deploy the learned grasp strategies with efficiency and flexibility.

Table 5.3: Prompt for Task Planing.

An object is placed on a table in front of the robot. The the width of the palm is 11 cm. The max length of the grasp aperture is 12 cm. The height of the object is along the vertical direction, width along horizontal direction, and the thickness along the forward-backward direction. Your task is to grasp the object and pick it up. To accomplish the task, you need to 1. decide which direction (side or top) would provide the most secure and stable grasp. 2. rotate hand and approach the object based on the selected direction. 3. move to the object. 4. grasp the object. 5. pick up for 10 cm. The following functions are available for you:
rotate_hand(direction): take a string input, rotate the robotic hand so that the hand can grasp the given direction
approach_object(direction): move to a safe distance based on the grasp direction
get_object_position(direction): return the current coordinates and orientation of the direction of the object.
move_hand([x,y,x]): move the hand to the given position [x,y,x]
grasp_object(): close the hand to grasp the object.
You need to learn the skill of picking up the object and holding it.The object is a water bottle with dimension 21x7.5x7.5 corresponding the height, the width, and the thickness. Use your common sense and reasoning skill to write python code to control the robot to pick up this specific object. You are allowed to create new functions using the available functions, but you are not allowed to use any other hypothetical functions. Keep the solutions simple and clear. Do not output code for each step. Output an overall code.
Additional points to consider when giving your answer:
1. Your responses should be informative, visual, logical and actionable.
2. Your logic and reasoning should be rigorous, intelligent, and defensible.
3. You can provide additional relevant details to respond thoroughly and comprehensively to cover multiple aspects in depth.

To generate appropriate grasping plans, we devised prompts that outline the task, elucidate the available functions, provide a description of the environment, and specify the goal. The model will analyze the given information, decide to grasp the object from the top or from the side, and generate a comprehensive and practical grasping deployment plan based on the decision. To simplify the problem, grasp pose is pre-adopted and the target object was placed in a manner that its grasp dimension can be grasped from the direction decided by the model. The final prompt, shown in Table 5.3, yields the grasping deployment code in Figure 5.4. Note that the prompt is task and environment specific. For different task and environment combinations, the prompt needs to be adjusted accordingly.

```
Python
# Step 1: Decide the most secure and stable grasp direction from the sides
grasp_direction = 'side'
# Step 2: Rotate hand and approach the object based on the selected
direction
rotate_hand(grasp_direction)
approach_object(grasp_direction)
# Step 3: Move to the object
object_position = get_object_position(grasp_direction)
target_position = [ object_position[0], object_position[1],
object_position[2]]
move_hand(target_position)
# Step 4: Grasp the object
grasp_object()
# Step 5: Pick up the object for 10 cm
target_position[2] += 10 # Move 10 cm upward from the current position
move_hand(target_position)
```

Figure 5.4: Generated grasping deployment plan.

5.2.5 High Level Robot Control Library

This High Level Robot Control Library (HLRC) is implemented to adapt to the code generated from the Task Planing model. HLRC is a versatile library developed, utilizing API from OpenCV, Sawyer and AR10 libraries, and ROS Python libraries, to provide precise robot control in real-world robot test beds. The "get_object_position" function locates the object and return its current coordinates and orientations in the environment through the MagicHand system [39]. The "rotate_hand" function rotates the robot hand so that the robot can grasp the object from either the left side or the top. The "approach_object" function ensures the object in the grip by calculating the relative position among the palm, the fingers and the object. The "grasp_object" function executes the grasping action until a specific contact force threshold is reached.

5.3 Experiments

Real-world experiments were conducted using an anthropomorphic robotic hand to evaluate the overall performance of the proposed Cognition Based Grasping System. We revised

our previous object dataset in [58] to test the accuracy of the Feature Complementation model and to train and evaluate the Grasp Selection network. The experiment results demonstrated the performance and improved the overall understanding of the proposed system.

5.3.1 Experiment Setup

Testing Environment

The MagicHand platform, as shown in Fig. 1.3, serves as the test bed for the proposed system. The robot includes an AR10 robotic hand, a Rethink Sawyer robot (with an in-arm camera), as well as an Intel RealSense RGB-D camera, Force Sensing Resistors (FSR), and an SCiO sensor installed on the wrist. The robotic hand has limited force sensing through the FSR attached to fingertips, so we focused on hand configurations and planning and used the force sensors to examine contact conditions.

Figure 5.5: Sample objects for experiments.

Testing Dataset

To align with the categorization strategy described in Section 2. B, we revise the Fragility, Rigidity, Shape, and Texture features of the object dataset [58]. The dataset was established

by nine non-expert college students who collected information from 100 everyday objects and labeled each object with all applicable grasp topologies. Object features, including name, dimensions, mass, shape, texture, fragility, material, and rigidity, were measured, estimated, and mapped with grasp topologies in 5.4. For each object, the label includes all applicable grasp topologies and their corresponding frequency. A selection of object samples is show in Figure 5.5.

5.3.2 Object Feature Estimation

The efficiency of the Feature Complementation (FC) model is evaluated with the revised object dataset. Perceived information from the Feature Perception model, including object name, dimension, and material, is enhanced and complemented. The performance of the FC model is shown in Figure 5.6.

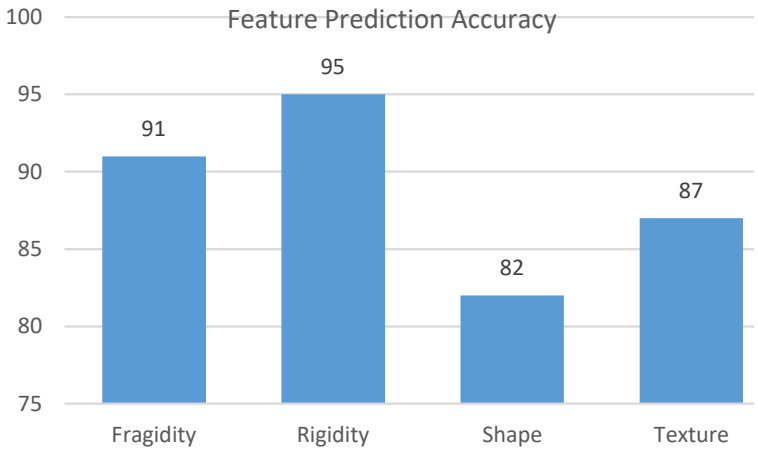

Figure 5.6: Feature estimation accuracy of the Feature Complementation model.

In the figure, we can see that the model has a lower accuracy in predicting shape and texture. This is due to the limitation of the given information. Objects with the same names and dimensions could have different shapes. For example, a 7x7x2 plate could either be square-shaped or disk-shaped. Additionally, differentiating cylinder and cuboid based on the dimensions of the object can be also challenging. By using diameter to describe the disk and cylinder-shaped object, the accuracy for predicting shape increased to 91%. The prediction of the texture faces the same problem, name and material of the object are not

enough for deciding whether its surface is smooth or rough. The surface of objects can vary in smoothness or roughness not only based on the material but also due to the surface structure. For instance, the surface of a metal cup can be either smooth or rough based on the surface finish or polishing.

5.3.3 Grasp Selection Network

	Labled Grasp Probability Distributions									Predicted Grasp Probability Distributions										
Object	rc.ab	rc.bc	rp.b	rp.c	wc.abc	wh.bc	wh.c	wp.bc	wt.c	rc.ab	rc.bc	rp.b	rp.c	wc.abc	wh.bc	wh.c	wp.bc	wt.c	Fl	Fm
calculator	0.00	0.00	0.56	0.22	0.00	0.00	0.00	0.00	0.22	0	0.22	0.41	0	0	0	0	0	0.37	1	1
water bottle	0.00	0.11	0.11	0.22	0.00	0.56	0.00	0.00	0.00	0	0	0.32	0.25	0	0.43	0	0	0	1	1
wood cylinder	0.00	0.22	0.56	0.22	0.00	0.00	0.00	0.00	0.00	0	0.18	0.31	0.29	0	0.22	0	0	0	1	1
cardboard box	0.00	0.00	0.56	0.22	0.00	0.00	0.00	0.00	0.22	0	0	0.37	0.19	0	0	0.22	0	0.22	1	1
mini rubix cube	0.11	0.44	0.33	0.11	0.00	0.00	0.00	0.00	0.00	0	0.45	0.31	0	0	0	0	0	0.24	1	1
wood wedge	0.00	0.22	0.44	0.22	0.00	0.00	0.00	0.00	0.11	0	0.12	0.25	0.43	0	0	0	0	0.2	1	0
wood disk	0.44	0.00	0.22	0.22	0.00	0.00	0.00	0.00	0.11	0.34	0	0.26	0.4	0	0	0	0	0	1	0
tennis ball	0.11	0.11	0.22	0.11	0.44	0.00	0.00	0.00	0.00	0.2	0.25	0	0.25	0.3	0	0	0	0	1	1
wood piece	0.22	0.44	0.22	0.11	0.00	0.00	0.00	0.00	0.00	0.11	0.39	0.26	0.24	0	0	0	0	0	1	1
plastic cap	0.56	0.00	0.22	0.11	0.00	0.00	0.00	0.00	0.11	0.15	0	0.25	0.2	0.2	0	0	0	0.2	1	0
medicine dispenser	0.00	0.00	0.67	0.22	0.00	0.00	0.00	0.11	0.00	0	0	0.35	0.18	0	0	0.22	0.25	0	1	1
screw driver	0.00	0.00	0.33	0.11	0.00	0.00	0.00	0.56	0.00	0	0	0.17	0.29	0	0.16	0	0.38	0	1	1
ball1	0.11	0.22	0.11	0.11	0.44	0.00	0.00	0.00	0.00	0.19	0.15	0.21	0	0.45	0	0	0	0	1	1
rubix cube	0.11	0.11	0.56	0.22	0.00	0.00	0.00	0.00	0.00	0.22	0.2	0.3	0.16	0.12	0	0	0	0	1	1
water bottle cap	0.56	0.00	0.33	0.11	0.00	0.00	0.00	0.00	0.00	0.35	0	0.23	0.23	0.19	0	0	0	0	1	1
sanitizer bottle	0.00	0.00	0.56	0.22	0.00	0.22	0.00	0.00	0.00	0	0.25	0.4	0.1	0	0.25	0	0	0	1	1
wallet	0.11	0.00	0.44	0.22	0.00	0.00	0.00	0.00	0.22	0.25	0	0.35	0.24	0	0	0	0	0.16	1	1
ball2	0.11	0.22	0.11	0.11	0.44	0.00	0.00	0.00	0.00	0.21	0.24	0	0	0.55	0	0	0	0	1	1
toy car	0.00	0.00	0.56	0.33	0.00	0.11	0.00	0.00	0.00	0	0	0.48	0.23	0	0.29	0	0	0	1	1
kitchen scale	0.00	0.00	0.00	0.33	0.00	0.00	0.00	0.00	0.67	0	0	0.18	0.27	0	0	0	0	0.55	1	1

Figure 5.7: Sample grasping strategy determination: ground-truth vs. predicted grasping.

We developed the neural-network model to learn grasping strategies as proposed in Section 2. C, and optimized the model in terms of cross-entropy. The input layer includes nine nodes activated by ReLu function. The next four layers are hidden layers which contains 2^7, 2^9, 2^7 and 2^5neurons which are also activated by ReLu function. The output layers has nine nodes activated by Sigmoid activation function. The input of the model is the acquired object features, and the output is the grasping strategies corresponding to human preference and knowledge. The grasping strategies were represented by normalized probability distributions. The results of grasping strategy determination with scores are reported and compared to the ground truth in Fig. 5.7. The feasibility score F_l of the model is 100%, which was defined as the hit rate of the predicted grasp strategy in all human preferred grasps. The

experiment shows that the model's capability in picking feasible (human validated) grasping. The match-score F_m, on the other hand, measures the accuracy of the prediction considering only the most preferred human grasping. The experiment demonstrated that the max-match rate was around 85% for the test objects. The objects with complemented features were also tested by the network. The testing result shows a feasibility score of 100% and a match-score of 83% which is similar to the object with human-decided features, indicating the grasp topology predicted by the grasp selection network based on complemented features are comparable to those made by human decision-makers.

5.3.4 Robotic Grasping

To further examine the performance of the proposed system, we performed grasping experiments using the MagicHand platform. Test objects were placed on the table with a default initial orientation, and located and identified by the Object Perception model. The perceived information was then complemented by the Feature Complementation model. The robot autonomously chose the grasp strategies according to the completed object affordance. The success of a grasp was determined by the stability of grasping after brief maneuvers including grasping, lifting, and holding. A grasping task was considered a success if the object stays secure throughout the maneuvers.

We conducted ten grasping tasks on different objects and the overall success rate of grasping was 80%. One failure case involved the grasping of a large-sized metal cup, where the model predicted the most preferred human grasping strategy (wh) and grasp direction (side). However, the hand failed to wrap the cup tightly and securely, leading to the failure of the grasping task. This was attributed to the limitations of hand dexterity and the usage of the pre-defined grasp pose. The other failure case happened while trying to grasp a small bottle cap. Even though the model predicted the most preferred human grasp (rp), it decided to grasp the object from the side, we terminated the task to prevent potential collision between the hand and the table.

We designed a sequential system to emulate human decision-making processes. In such a

system, errors from one model could permeate or propagate to subsequent models. However, in the experiments, the final grasp execution on the robot was minimally affected by the errors generated in each model. The primary reason could be that, while human grasping is complex, it is also highly resilient to external perturbations of contextual variables. When modeling robot grasping using human grasp primitives, this resilience behavior was emulated as well. For example, there are multiple ways to grasp an object, so it is less likely to choose a wrong grasp as we have seen from the results of our Grasp Selection model. The other reason is that the experiment objects were designed with the intent of handling by five-fingered hands, so when there is a miscalculation, e.g., grasp dimensions, the fingers conform to the object shape and still result in a secure grasp.

5.4 Summary

This paper has presented a Cognition Based Grasping System to applying a proper strategy to grasp an object without complete sensing of object affordance. The framework of the proposed system emulates the human grasping process, including object affordance acquisition, strategy determination, and grasping deployment, by combining common sense, intuitive, reasoning, and machine learning approach. The accuracy of the feature complementation process are between 82% to 95% depending on the feature. The grasp selection model achieves a 100% feasibility score and an 85% match score in predicting human grasping knowledge. Experiment on the humanoid robot achieved 80% success rate, demonstrating the practicality and efficiency of the proposed system in dexterous grasping task in unfamiliar environments. In summary, the experiments show that the proposed system is efficient and suitable for real-world grasping applications.

CHAPTER 6

GRASP INTENTION INTERPRETATION IN OBJECT HANDOVER

6.1 Introduction

Human-robot collaboration, particularly in the context of object handovers, is essential for ensuring smooth and effective interactions in both structured and unstructured environments like healthcare and industrial settings [2, 7, 70]. For example, a robot handing tools to a nurse or parts to a factory worker must adapt to the human's grasping habits and workspace, improving safety and efficiency. Extensive research has been conducted to address the handover task in human-robot collaboration. Studies have analyzed the trajectory and velocity of approach movements to ensure smooth transitions [30, 48]. Additionally, object orientation and affordances have been optimized to make it easier for the receiver to grasp the object [9, 67]. Some approaches also involve learning from human behavior to improve the naturalness and effectiveness of handovers [9]. However, less work has been done to adapt to the habits of the receiver, particularly the variability in human grasping behaviors, which are influenced by individual preferences and situational factors. Developing systems that can accurately recognize and adapt to these diverse human behaviors is crucial for making robots more intuitive and practical in real-world applications.

Grasp topology and taxonomy are key to understanding human grasping behaviors and robotic adaptation. Grasp topology describes the geometric configuration of the fingers or contact points with an object, such as pinching or cupping. Grasp taxonomy categorizes these topologies into structured classes based on factors like contact points and object shape. This classification aids in designing robots that can effectively recognize and adapt to human grasping behaviors in various tasks and environments. Previous research has shown that human grasp choices tend to cluster over a large set of objects, leading to the development of grasp taxonomies to simplify grasping choices. For example, Cutkosky's taxonomy identified 16 grasp types used by machinists [14], and Feix's taxonomy expanded this to 33 different grasp types [8, 22].

The FreiHAND dataset provides a large collection of annotated 3D hand poses, high-resolution RGB images, and key points annotations, making it highly valuable for hand

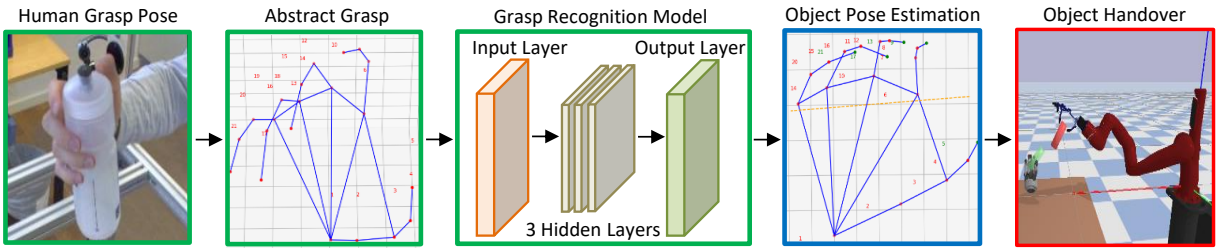

Figure 6.1: Structure of the grasp adaptation system.

tracking and grasp recognition research [85]. In this paper, we extend the FreiHAND dataset to map individual grasping habits to a standard set of grasp topologies.

Reinforcement learning (RL) is a powerful approach for robot control, where robots learn to perform tasks through trial and error [71, 68, 27]. By receiving feedback in the form of rewards or penalties based on their actions, robots improve their behavior over time [69]. This method enables robots to develop adaptive and optimized control strategies for complex and dynamic environments, enhancing their ability to perform a wide range of tasks autonomously. In this paper, a RL model is designed to conduct the object handover task in a simulation environment which is identical to the MagicHand system [40, 39].

We propose a grasp adaptation algorithm, as illustrated in Fig. 6.1, that processes an RGB image of a human grasping pose, converts it into an abstract grasp representation, and classifies it into one of six standard grasp topologies. Key points are then selected from the abstract grasp based in the identified grasp topology to estimate the appropriate object pose, and a reinforcement learning model is used to optimize the object handover task. The contributions of this research include:

- We designed and developed a deep learning network to classify grasp poses into six standard topologies, enabling the robot to determine the appropriate object pose.
- We created a reinforcement learning model to optimize the object-handover task.

6.2 Human Grasping Habit Adaption

The proposed system comprises three models: grasp recognition, object pose estimation, and a reinforcement learning model for object handover tasks. This section provides detailed descriptions of each model.

6.2.1 Recognition of Human Grasp Pose

Standard Grasp Topology

In this paper, we classify grasp poses into six distinct grasp topologies [57], as illustrated in Fig. 6.2. The figure categorizes grasps into two main types: power grasps and precision grasps, based on object shapes and the involvement of virtual fingers (VF). Power grasps, such as those for circular and prismatic objects, prioritize security and stability, utilizing more virtual fingers and 3D wrapping. In contrast, precision grasps focus on dexterity and sensitivity, typically involving fewer virtual fingers and 2D wrapping.

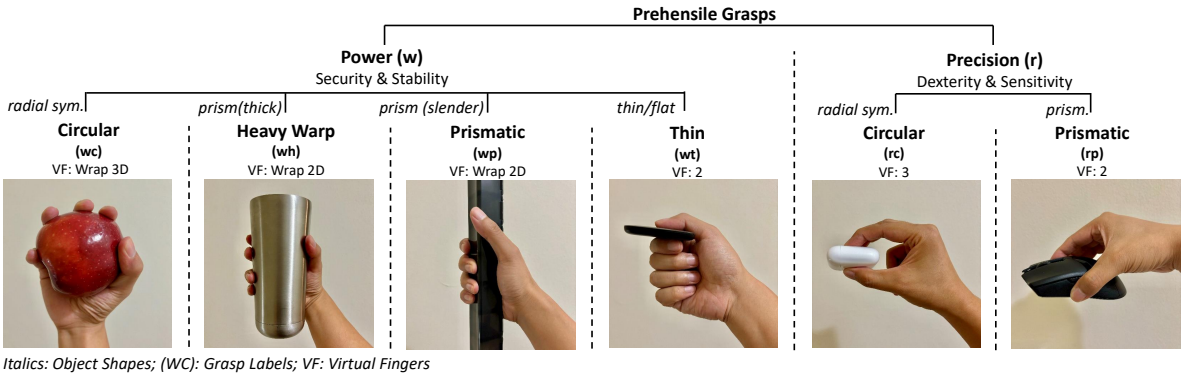

Figure 6.2: The predefined grasp topology.

After establishing the grasp taxonomy, a standard grasp pose was selected for each topology by evaluating grasp poses from the FreiHAND dataset. The evaluation was based on how well each pose matched the defined topologies, and the pose that best fit each topology was chosen as the standard grasp pose. The final results are shown in Fig. 6.3, which displays both the RGB image of the standard grasp pose and the corresponding grasping information. The grasping information includes the abstract grasp and the key points (highlighted

in green) of that specific grasp topology. These key points are used to estimate the object pose. The orange dotted axis in the figure represents the estimated object pose, derived from the selected key points.

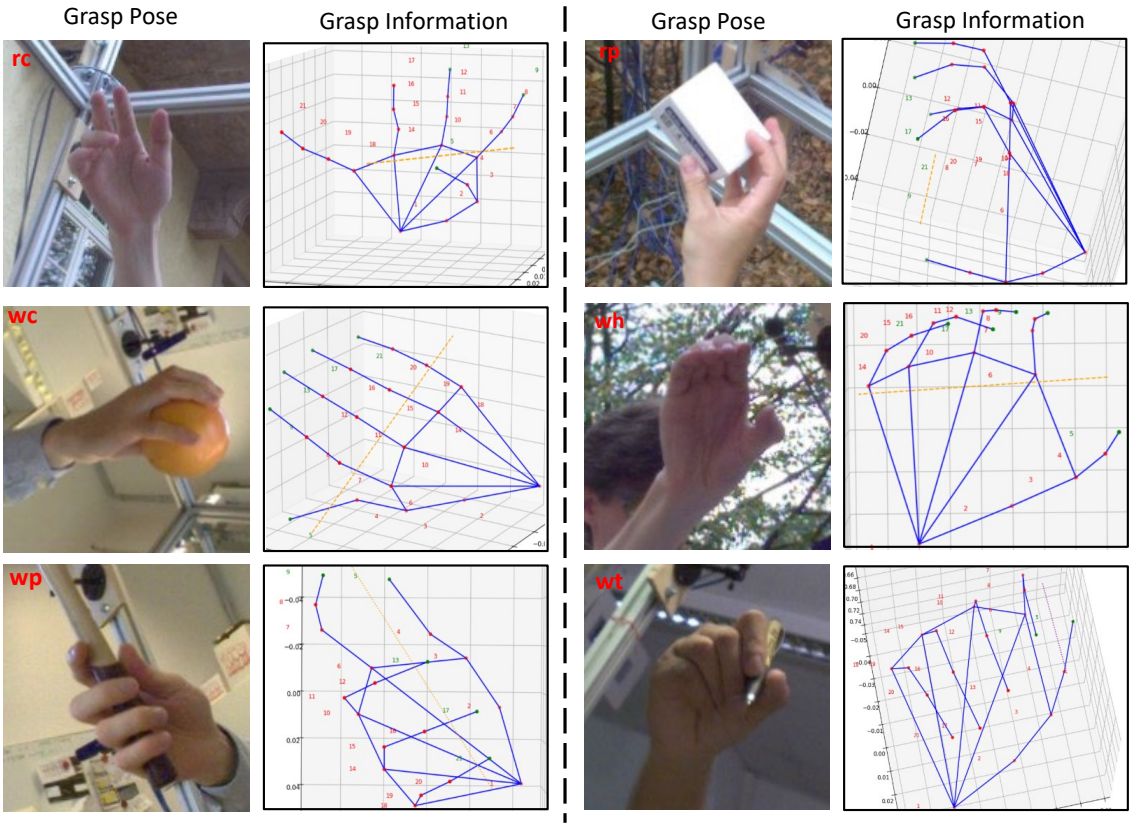

Figure 6.3: Standard grasp topology: This figure displays images of the grasp topology, including the skeletons of each standard grasp topology and their corresponding key points, highlighted in green.

Grasp Topology Recognition

A multi-layer perceptron (MLP) deep neural network was developed to map abstract grasps to standard grasp topologies. Rectified Linear Units (ReLU) were used as activation functions in the input and hidden layers. The input layer has 63 neurons, corresponding to the 21 3D points in each abstract grasp. The network features three hidden layers with 1,024, 256, and 32 nodes, respectively, to refine the model. The output layer consists of six neurons with Softmax activation functions to classify the grasps. The model was trained on a dataset

refined from the FreiHAND dataset.

6.2.2 Object Pose Estimation

The pose of the object, including both position and orientation, is determined based on the key points associated with the standard grasp topology. For grasp topologies such as "wc", "wh", "wp", "rc", and "rp", the object should be positioned within the grasp's aperture, which is the space between the fingertips of the thumb and the fingers. The object position is defined as the midpoint between p_t, the point representing the tip of the thumb, and p_c, the closest fingertip to p_t. The object position is expressed as $p_m = 0.5\,(x_t + x_c, y_t + y_c, z_t + z_c)$. Since the orientation is typically aligned with the palm or fingers, two key points, p_s and p_e, are pre-selected from the abstract grasp to define the object's orientation. These points are usually located on the palm. The orientation of the object is expressed as

$$\mathbf{L}(t) = (x_m, y_m, z_m) + t \cdot (x_e - x_s, y_e - y_s, z_e - z_s) \tag{6.1}$$

where x, y, and z are coordinates of the point and t is a scalar parameter. The points p_s, p_e, p_c, and p_t, are specific to each grasp topology and serve as key points provided for analysis.

For the grasp topology "wt" the object should be positioned between the fingertip of the thumb and the side of the index finger, specifically at the key point p_{pip}, which corresponds to the PIP joint (Proximal Interphalangeal Joint) of the index finger. The object's position for this grasp topology is expressed as $p_m = 0.5\,(x_t + x_{pip}, y_t + y_{pip}, z_t + z_{pip})$. The orientation of the object should be roughly parallel to the index finger. In this case, the key points p_s and p_e in (6.1) represent the PIP and MCP (Metacarpophalangeal) joints of the index finger, respectively.

6.2.3 Object Handover Using Reinforcement Learning

Once the object's pose is determined, the robot must adjust the object to the specified position and orientation. We designed and developed an reinforcement learning model to achieve this goal. A simulation environment mirroring the MagicHand platform, which

supports a variety of manipulation tasks, has been established [39, 41].

The task is to handover the object to a human hand, simulated by a Schunk anthropomorphic robotic hand in a simulation environment, and positioning it at the target pose, represented by a green area. The action space of the proposed model is a six-dimensional vector, including movements and rotations of the robotic hand along the x, y, and z axes. The observation space consists of seven dimensions: the relative position and orientation between the object and the target, as well as the distance between them.

In the task, our goal is to position the object as close to the target as possible, rewarding smaller distances between the object and the target. Additionally, we aim to align the object's orientation with the target's orientation, rewarding smaller differences in orientation along the x, y, and z axes. To ensure safe interaction, we impose penalties for collisions with the human hand. The reward function is expressed as

$$r = e^{-|d|} + e^{-|h|} + e^{-|l|} + e^{-|k|} - \alpha n_{\mathrm{c}} \tag{6.2}$$

where d is the distance between the object and the target location, and h, l, and k represent the differences in orientation between the object and the target along the x, y, and z axes, respectively. The coefficient α is a constant, and n_{c} represents the number of contact points with the human hand. We chose an exponential function because its value changes more rapidly when the exponent is large and more slowly when the exponent is small. This approach encourages the robot to make larger adjustments when the current pose is far from the target, while allowing for more precise, gradual adjustments as the pose approaches the target.

The proximal policy optimization (PPO) algorithm is employed to train the model. PPO is a reinforcement learning algorithm designed to improve policy stability by limiting the size of policy updates. It uses a clipped objective function to ensure that the new policy

does not deviate excessively from the old policy. The objective function is given by

$$J(\theta) = \mathbb{E}\left[\min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}\hat{A}(s,a), \text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}, 1-\epsilon, 1+\epsilon\right)\hat{A}(s,a)\right)\right]$$

where $\pi_\theta(a|s)$ and $\pi_{\theta_{\text{old}}}(a|s)$ are the probabilities of taking action a in state s under the new and old policies, respectively, and $\hat{A}(s,a)$ is the advantage function. The clipping function clip restricts the ratio of the new to old policy probabilities, with ϵ controlling the extent of the allowed change, thereby balancing exploration with stability.

6.3 Experiments

The proposed system was evaluated under different thresholds. The simulation environment was set up using PyBullet and Gym, and task simulations were conducted to test the final performance of the system.

6.3.1 Data Preparation

The FreiHAND dataset provides high-resolution RGB images with detailed 3D hand poses and key points, supporting research in hand tracking and grasp recognition. Its comprehensive annotations are valuable for developing and evaluating hand tracking algorithms and applications in robotics and augmented reality. We revised the dataset by labeling each grasp with a specific topology. A sample of the revised dataset is shown in Fig. 6.4. Each grasp pose has been meticulously reviewed and classified into one of six predefined topologies or marked as "other". The updated dataset now includes 100 grasp poses for each of the six topologies, totaling 600 annotated grasp poses.

6.3.2 Grasp Recognition

The revised dataset is divided into two parts: 540 grasp poses for training and validation, and 60 new grasp poses for testing. The proposed algorithm was evaluated using 4-fold cross-validation with hyperparameters of a batch size of 64, 500 epochs, and the Adam optimizer with a learning rate of 0.001. The training accuracy achieved 93.3% while the accuracy on

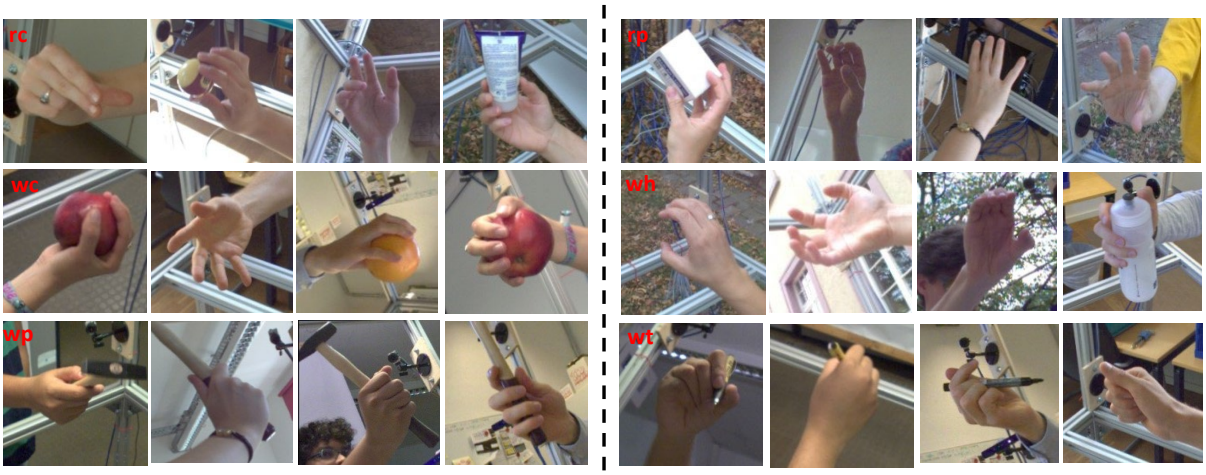

Figure 6.4: The revised dataset consists of 600 grasp poses, each paired with a corresponding abstract grasp representation for each grasp topology.

testing set achieved 87.2%. The testing accuracy is relatively low because some of the grasp topologies are difficult to distinguish. For example, the "wh" and "rc" grasps have similar configurations, making them harder to recognize accurately.

6.3.3 Object Pose Estimation and Handover

Determining the effectiveness and accuracy of an object pose can be challenging, so we evaluate it through an object-handover task in a simulation environment. An object pose is considered effective if the robot can successfully pass the object to the human hand, which should then be able to securely grasp it. For this evaluation, we used PyBullet [13] to simulate an AR10 robotic hand mounted on a Sawyer robot holding the object. The human hand, simulated by a Schunk robotic hand, positioned in front of the robot, performs a variant of one of the six grasp topologies. The system estimates the object pose based on the grasp pose of the simulated human hand and highlights the estimated pose as a green area. The handover task for each grasp topology is illustrated in Fig. 6.5, where the robot's goal is to position the object to align with the green area while avoiding collisions with the human hand.

The model was trained for 20,000 episodes with a learning rate of 1.6×10^{-6} and a batch size of 32, with varying target positions and orientations in each episode. Each grasp

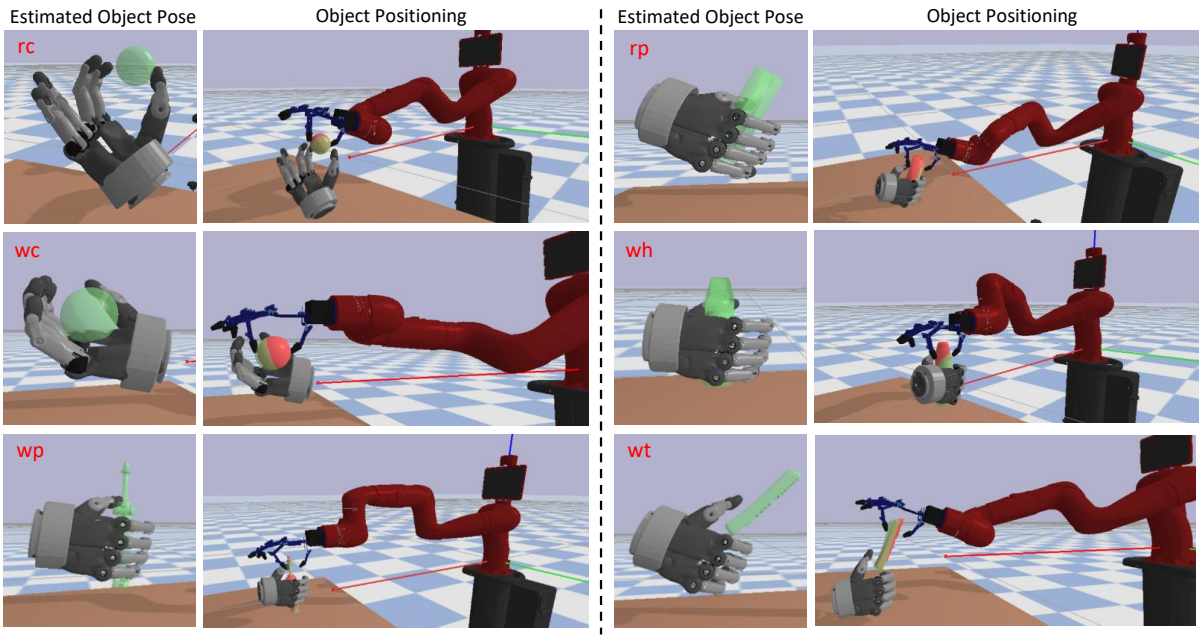

Figure 6.5: Handover task for each grasp topology: In each task, the system first estimates the object pose based on the grasp pose of the simulated human hand. The model then attempts to place the object at the target pose.

topology was tested 100 times achieving an overall success rate of 83%.

6.4 Conclusions

In conclusion, this paper presents a robust approach to enhancing human-robot cooperation through the development of a grasp adaptation system. By accurately recognizing diverse human grasping habits and classifying them into standard grasp topologies, the system determines optimal object handover strategies for smooth handovers. The use of deep learning for grasp pose recognition and reinforcement learning for strategy optimization demonstrated strong performance in experimental settings. Although challenges remain in distinguishing similar grasp configurations, the results underscore the potential of the proposed system to improve human-robot interaction by making robots more adaptable to varied human behaviors.

CHAPTER 7

TASK-ORIENTED GRASPING USING REINFORCEMENT LEARNING WITH A CONTEXTUAL REWARD MACHINE

7.1 Introduction

Robotic dexterity, the ability of a robot to manipulate objects with precision, adaptability, and control, akin to human hand dexterity, is essential for performing complex tasks across diverse applications, including aerospace, automotive, manufacturing, and warehousing [19]. While robots excel in structured environments and repetitive tasks, they remain constrained in unstructured and dynamic scenarios. Advancing robotic dexterity has the potential to bridge this gap and allow robots to handle complex tasks in uncertain environments [32]. The current methods struggle with dexterous manipulation, especially when handling objects of varying shapes, sizes, and materials.

Grasping an object represents the initial and foundational step of dexterous manipulation. A key factor in grasping tasks is the selection of an appropriate grasp topology, which defines the specific configuration of a robotic hand when interacting with an object. The grasp topology plays a pivotal role in minimizing redundant finger movements and streamlining the overall manipulation process. Selecting a suitable topology is a critical prerequisite for effective manipulation, as it ensures stable object acquisition and simplifies downstream control [76]. A firm or adaptive grasp stabilizes the object and enhances task efficiency and execution success.

Research in this area has explored various strategies for determining effective grasp topologies, including learning from human demonstrations to replicate natural grasp pat-

terns [47], designing soft robotic hands for flexible and compliant interactions [15], and leveraging advanced learning-based techniques for data-driven optimization [18]. Selecting a grasp topology that aligns with object properties and task requirements reduces manipulation complexity and improves performance across diverse scenarios [66].

While numerous grasp types exist, they generally cluster into a limited set that is suitable for manipulating everyday objects and tools [23, 61]. This insight has led to the development of grasp taxonomy that categorizes and simplifies grasp poses [14, 8]. The taxonomy provides a systematic way to map object characteristics and task requirements to suitable grasp types, thereby improving planning and control.

The choice of grasp topology is influenced by both object features, such as size, shape, surface texture, and mass, as well as task-specific requirements, including the intended action, applied forces, and environmental constraints [58, 20]. For instance, precision tasks, like picking up small or fragile objects, typically utilize pinch or tripod grasps. In contrast, tasks requiring greater stability or force, such as lifting heavy items, benefit from power grasps. Tool-use scenarios introduce additional complexity and often necessitate specialized topologies, such as cylindrical grasps for tool handles or lateral grasps for flat items. Building on these insights, our previous work [41] successfully integrated object features and task requirements into grasp strategies and established a structured framework for the grasping process.

Deploying grasping tasks presents significant challenges due to environmental uncertainties, particularly in unstructured environments. The perception of target objects is often incomplete or inaccurate, which negatively impacts grasping performance. Traditional planning methods struggle to manage these complexities because they rely on precise context modeling, which is rarely feasible in real-world scenarios.

Task-oriented grasping can be formulated as a sequential decision-making problem in which an agent learns optimal behaviors through trial and error to maximize cumulative rewards. Reinforcement learning (RL) has shown strong potential in addressing such problems

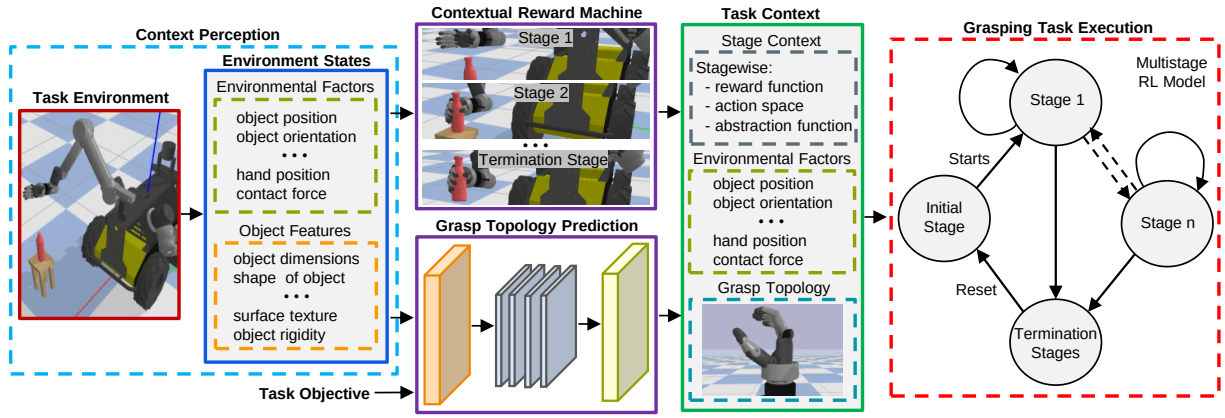

Figure 7.1: Context-aware task-oriented grasping framework with a contextual reward machine.

effectively [69, 56, 46].

RL has demonstrated particular success in solving complex control problems in robotics, especially when integrated with Proximal Policy Optimization (PPO) [71, 68, 27]. PPO, a widely used RL algorithm, addresses many limitations of RL by improving training stability and sample efficiency [64]. Its simplicity and robustness make it well suited for high-dimensional and continuous action spaces, which enable effective policy optimization.

Despite these advantages, RL methods for grasping still face several challenges, including high computational demands and difficulties in achieving stable convergence. These limitations highlight the need for further development of RL techniques that can meet the demands of real-world applications.

To address these challenges, researchers have proposed multistage reinforcement learning approaches where each stage of a task is trained separately using specialized sub-networks that collaborate to determine an overall optimal policy [80]. Although this method achieves stable convergence, it remains computationally expensive. The reward machine framework has been introduced to solve the problems by organizing complex tasks into modular subtasks, each associated with a distinct reward function. This structure improves learning efficiency and reduces computation cost [31]. Although reward machines offer a structured approach to task decomposition, traditional implementations often lack the flexibility to adapt

to dynamic environments or incorporate detailed contextual information. These constraints limit their applicability to real-world scenarios, where adaptability and context-awareness are essential for reliable robotic performance.

In this paper, we propose a context-aware task-oriented grasping approach that leverages a Contextual Reward Machine (CRM) to enhance efficiency and adaptability. The CRM decomposes grasping tasks into sequential stages with each stage defined by a stage-specific context including a reward function, an action space, and abstracted states. This structure guides intra-stage task progression and improves learning efficiency. Additionally, a transition reward mechanism is designed to facilitate smooth transitions between stages.

The general structure of the proposed method is illustrated in Fig. 7.1. In this approach, the context of the environment for a grasping task is continuously perceived and analyzed. The environmental context comprises object features, such as dimensions, shape, and texture. It also includes environmental factors, such as object pose, contact forces, obstacle positions, and object affordances. The object features and the task objective are processed by a pretrained grasp selection network to determine the appropriate grasp topology. Meanwhile, the environmental factors and object affordances are used by the CRM to identify the grasp location, determine the current stage, and retrieve the corresponding stage-specific context. They are also utilized by the RL agent to learn and optimize its policy. Task execution is carried out by an RL model, which integrates the grasp topology, grasp location, and stage-specific contexts under CRM framework. The model dynamically adapts to the changing conditions and performs robust, precise, and efficient grasps.

The main contributions of the paper include:

- We implemented a context-aware task-oriented dexterous grasping approach that enables adaptive and efficient grasping in unstructured environments.
- We designed and developed a Contextual Reward Machine that decomposes grasping tasks into sequential stages. It utilizes stage-specific contexts and transition rewards to enhance learning efficiency and adaptability.

7.2 Framework of Task-Oriented Grasping

Task-oriented grasping is inherently a context-aware process involving the perception of environmental context, the generation of grasp strategies based on that context, and the efficient, adaptive execution of those strategies. To address these challenges, environmental context was obtained through sensor fusion by integrating inputs from multiple sensors. A deep learning network was developed to generate grasp strategies, while a reinforcement learning model under CRM framework was designed and implemented to enable efficient and adaptive execution of grasping tasks.

7.2.1 Context Perception

Task-oriented grasping tasks require detailed environmental context. An RGB-D camera is used to capture object dimensions, shape, and surface characteristics. These object features guide the selection of an appropriate grasp topology. Environmental factors, such as the object's position, orientation, and nearby obstacles, are also captured with the same RGB-D camera. In parallel, force-sensing resistors (FSRs) are used to record contact forces and provide precise pressure data to improve task accuracy.

7.2.2 Grasp Topology Determination

A proper grasp topology facilitates smoother and more efficient manipulation. To simplify the selection process, we defined a grasp taxonomy comprising six primary topologies and developed a grasp selection network [41] that maps object features and task demands to the most suitable grasp topology.

The adopted taxonomy is illustrated in Fig. 7.2: (1) the platform grasp for holding, pushing, or pressing; (2) the power grasp (poPmAb25) for securely gripping objects; (3) precision grasps (pPdAb2, pPdAb23, pPdAb25) for tasks requiring fine dexterity; and (4) the intermediate grasp (InSiAd2) for levering or twisting actions.

The grasp selection network is a multi-class, multi-label Multilayer Perceptron (MLP) neural network that takes object features and task objectives as input and predicts the

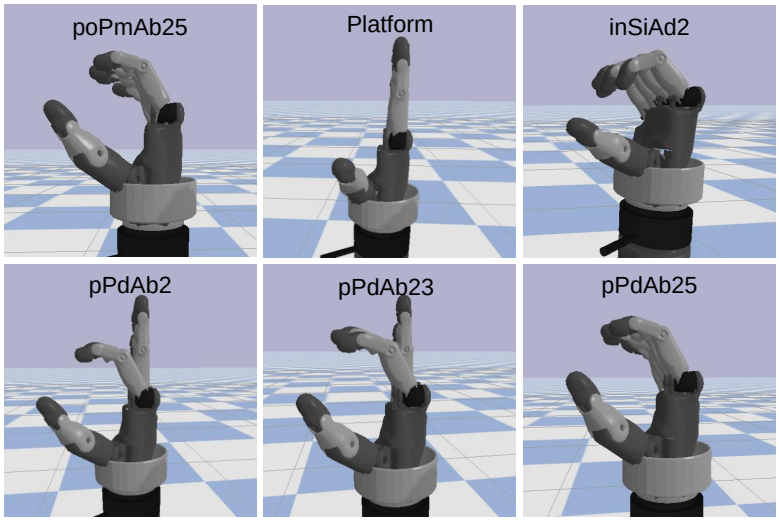

Figure 7.2: Grasp taxonomy with six grasp topology: the grasp topology names represent grasp attributes where "In," "po," and "p" indicate intermediate, power, and precision grasps. "Si," "Pm," and "Pd" refer to side, palm, and pad opposition. "Ab" and "Ad" signify abduction and adduction, and numbers define virtual finger groups.

probability of each grasp topology in the predefined taxonomy. The topology with the highest probability is selected as the target grasp pose.

7.2.3 Execution of Grasping Tasks

We used a reinforcement learning approach for grasping task execution. Due to the inherent complexity of grasping, such tasks often exhibit limited flexibility and present optimization challenges. To reduce task complexity, we decomposed the grasping process into a sequence of manageable stages. Although grasping tasks can be decomposed in various ways, Fig. 7.3 shows a general decomposition framework. In the framework, the initial state represents the initial configuration of the task environment. In the approach stage, the robot hand moves to an appropriate position and orientation (grasp location) for grasping. During the grasping stage, the fingers adjust to establish a stable or adaptive grasp based on the task's requirements. Finally, the termination stages reflect task outcomes, such as grasp success, grasp failure, or the object being out of reach, and signify the completion of the task.

This method requires stage-specific learning mechanisms as each stage operates within a

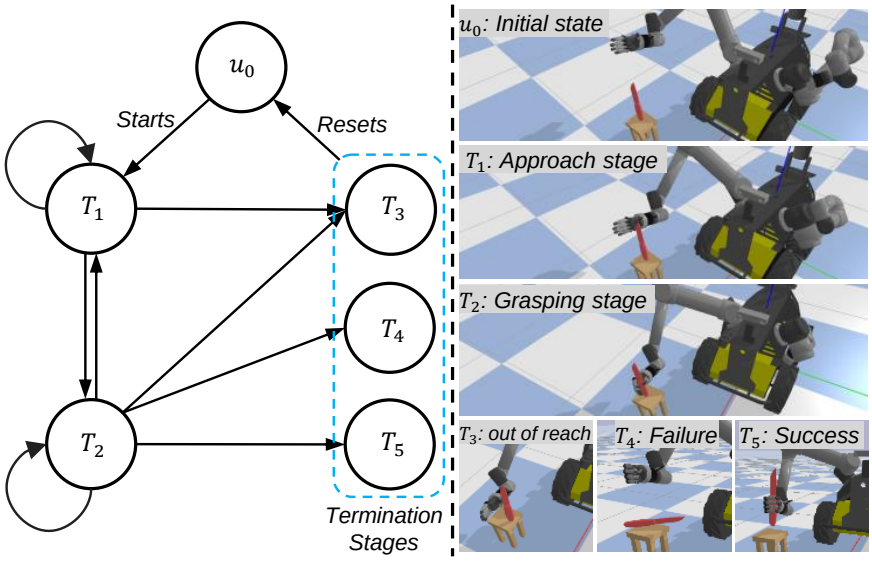

Figure 7.3: A general example of grasping task decomposition.

distinct context. To address this problem, we propose a Contextual Reward Machine that explicitly defines and manages these contexts.

7.3 Contextual Reward Machine

The CRM provides a structured and interpretable framework for addressing complex tasks by encoding task-specific knowledge into a hierarchical representation. This structure enables adaptive rewards based on task progress and stage transitions toward the desired goal. By guiding the agent through task-relevant stages, this approach improves learning efficiency and supports effective, goal-directed behaviors. Further details are provided in this section.

7.3.1 The General Framework of CRM

The general framework of the CRM extends the standard reward machine [31] by incorporating task context and a stage transition function, formally defined as

$$\mathcal{M} = (U, u_0, \Sigma, \delta, \mathcal{T}, R_{\mathrm{T}}) \tag{7.1}$$

where $\mathcal{T} = \{(A_i, r^{(i)}, \phi_i)\}_{i=1}^{k}$ represents a set of k stages (or sub-tasks), each characterized by task-specific knowledge. Each stage T_i consists of an action set $A_i \subseteq \mathcal{A}$, a state abstraction function $\phi_i : U \to U_i'$, which maps the global state space U to a stage-relevant abstract state space U_i' to simplify stage representation, and a stage reward function $r^{(i)} : U_i' \times A_i \to \mathbb{R}$ which defines the rewards for actions within the stage T_i.

The transition function $\delta : U \times \Sigma \to \mathcal{T}$ determines whether a stage transition occurs and, if so, to which stage, based on the current state and Σ, the set of events triggering stage transitions. A transition occurs when the current state satisfies one of these events. Upon a stage transition, the stage transition history set $\mathcal{H} \subseteq \mathcal{T} \times \mathcal{T}$ is updated as

$$\mathcal{H} = \{(T_i, T_j) \mid i, j = 1, 2, \ldots, k\} \tag{7.2}$$

where each transition $(T_i, T_j) \in \mathcal{H}$ is associated with a reward $R_{\mathrm{T}} : \mathcal{H} \to \mathbb{R}$, which quantifies the desirability of the transition.

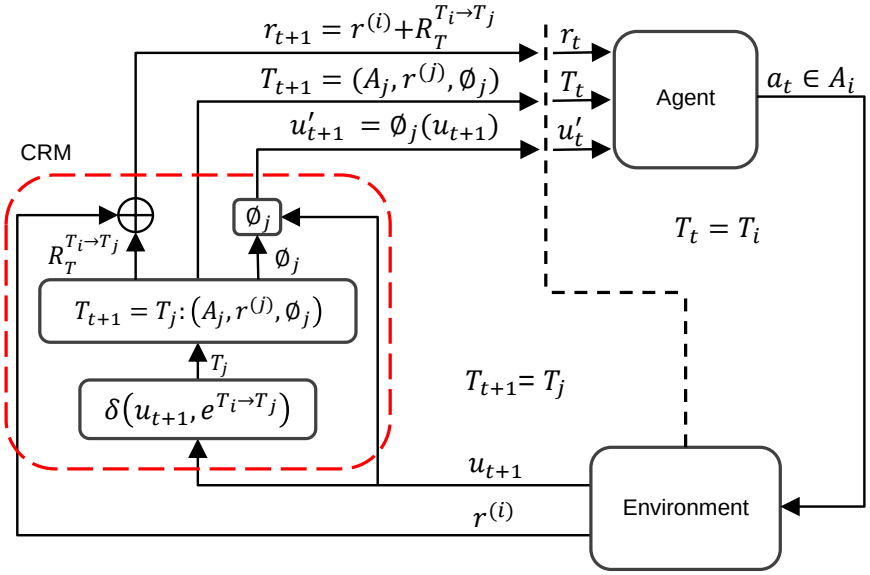

Figure 7.4: The Contextual Reward Machine: The dotted line separates the RL processes at timesteps t and $t+1$, illustrating the sequential interaction between the agent, environment, and the CRM.

The CRM within the RL framework operates iteratively and allows the agent to interact

with the environment and adapt dynamically, as illustrated in Fig. 7.4. At timestep t, the agent operates within stage $T_t = T_i$. Based on the abstract state u'_t and action space A_i, the agent selects an action $a_t \in A_i$. The environment responds with the next global state u_{t+1} and an intra-stage reward $r^{(i)}(u'_t, a_t)$ which reflects the action's outcome.

The CRM processes the feedback to determine the next stage $T_{t+1} = T_j$ using the transition function $\delta(u_{t+1}, e^{T_i \to T_j})$, where $e^{T_i \to T_j} \in \Sigma$ represents the triggering event. If no transition occurs ($i = j$), the global state u_{t+1} is abstracted to u'_{t+1} using ϕ_i. If a transition occurs ($i \neq j$), the global state u_{t+1} is abstracted to the corresponding state u'_{t+1} of stage T_j using ϕ_j. These abstraction functions extract task-relevant features and simplify state representation.

The reward at timestep $t + 1$ is computed as

$$r_{t+1} = r^{(i)}(u'_t, a_t) + R_{\mathrm{T}}^{T_i \to T_j} \tag{7.3}$$

and the cumulative reward is expressed as

$$R = \sum_{t=1}^{T} r_t(u'_t, a_t) + \sum_{(i,j) \in \mathcal{H}} R_{\mathrm{T}}^{T_i \to T_j} \tag{7.4}$$

where r_t represents the stage-specific reward at timestep $t \in [1, \ldots, T]$, and T is the total number of timesteps. The first term captures intra-stage rewards, while the second accounts for transition rewards. The stage knowledge in T_{t+1}, the abstract state u'_{t+1}, and the reward r_{t+1} define the context for the next timestep which supports efficient task decomposition and reward optimization for solving complex tasks.

7.3.2 PPO with CRM

To adapt PPO to the CRM framework, the objective function is revised to align with the hierarchical structure of CRM. For a stage T_i at timestep t, the clipped surrogate objective

is

$$\mathcal{L}_{\mathcal{M}}^{\text{CLIP}}(\theta) = \mathbb{E}_{(u',a)\sim\pi_{\text{old}}}\Big[\min\big(r_t^{\mathcal{M}}(\theta)A_t^{\mathcal{M}}, \text{clip}(r_t^{\mathcal{M}}, 1-\epsilon, 1+\epsilon)A_t^{\mathcal{M}}\big)\Big] \tag{7.5}$$

where $r_t^{\mathcal{M}}(\theta) = \frac{\pi_\theta(a_t|u'_t)}{\pi_{\text{old}}(a_t|u'_t)}$ represents the probability ratio between the current and old policies based on the abstract state $u'_t = \phi_i(u_t)$.

The advantage function $A_t^{\mathcal{M}}$ is defined as

$$A_t^{\mathcal{M}} = r_t(u'_t, a_t) + \gamma V(u'_{t+1}) + R_{\text{T}}^{T_i \to T_j} - V(u'_t) \tag{7.6}$$

where $r_t(u'_t, a_t) = r^{(i)}(u'_t, a_t)$ is the intra-stage reward, $R_{\text{T}}^{T_i \to T_j}$ is the transition reward, and $V(u'_t)$ and $V(u'_{t+1})$ are value estimates for the current and next stages, respectively.

7.4 CRM-PPO for Grasping Tasks

The CRM-PPO model divides grasping tasks into distinct stages, each defined by a specific context and transition mechanism. To optimize task performance, it is crucial to clearly specify these contexts and mechanisms for each stage. This section outlines the stage contexts and corresponding transition mechanisms for each stage of the grasping task.

7.4.1 Task Decomposition

We followed the task decomposition strategy shown in Fig. 7.3 to decompose grasping tasks using the general CRM framework (Eq. 7.1). The global state U is defined as the set of all possible states in the grasping environment, while $u_{\text{initial}} = u_0$ represents the initial state corresponding to the environment's default configuration. Each grasping task begins from the initial state u_0, and upon task completion, the environment resets to u_0 to prepare for the next task.

The system transitions directly from the initial state to the approach stage without receiving any reward. During the approach stage, the robot moves its hand toward the object. If the object becomes out of reach in this stage (e^{aor}), the task transitions to the

out-of-reach stage with a penalty $R_{\mathrm{T}}^{\mathrm{aor}}$. Arriving at the grasp location (e^{arrive}) transitions the task to the grasping stage, with a transition reward $R_{\mathrm{T}}^{\mathrm{arrive}}$. The approach stage cannot transition directly to the grasp-failure or grasp-success stages, as a failed or successful grasp requires hand-object interaction, which occurs only in the grasping stage.

In the grasping stage, the robot manipulates its fingers to attempt a grasp. Knocking the object out of reach in the grasping stage (e^{gor}) results in a transition to the out-of-reach stage with a penalty $R_{\mathrm{T}}^{\mathrm{gor}}$. Failure to grasp the object (e^{fail}) results in a transition to the grasp-failure stage with a penalty $R_{\mathrm{T}}^{\mathrm{fail}}$. Successfully grasping the object (e^{succ}) leads to the grasp-success stage with a reward $R_{\mathrm{T}}^{\mathrm{succ}}$.

The out-of-reach, grasp-success, and grasp-failure stages serve as termination stages, which conclude the current task and reset the environment to the initial state u_0 in preparation for the next one.

7.4.2 Action Space

The designed stages of a task-oriented grasping include the approach stage, the grasping stage, and the termination stages.

The Approach Stage

In the approach stage, the objective is to move the hand toward the grasp location while avoiding collisions with the object. Finger movements are disabled in the approach stage to reduce collision risks and improve sample efficiency. The action space is defined as

$$A_{\mathrm{approach}} = [\Delta x, \Delta y, \Delta z]$$

where Δx, Δy, and Δz represent incremental changes along the x, y, and z axes, respectively.

The Grasping Stage

The action space for the grasping stage is defined as

$$A_{\text{grasp}} = [\Delta x, \Delta y, \Delta z, \theta_{\text{thumb}}, \theta_{\text{index}}, \theta_{\text{middle}}, \theta_{\text{ring}}, \theta_{\text{little}}]$$

where θ_{thumb}, θ_{index}, θ_{middle}, θ_{ring}, and θ_{little} represent the proximal interphalangeal (PIP) joint angles of the thumb, index, middle, ring, and little fingers, respectively. In the grasping stage, fine adjustments to the hand position are crucial for achieving an optimal grasp, even when the hand is close to the grasp location. To enable these adjustments, the hand's movements remain constrained by Δx, Δy, and Δz, but with reduced step sizes for finer control.

To enhance grasp quality, we developed a simplified hand model inspired by human hand kinematics and anatomy. The model captures key biomechanical features such as joint articulation and finger linkage and enables more natural and effective grasping strategies. This model focuses on finger flexion and extension while excluding finger spreading. Joint angles are constrained based on the PIP joint angle, θ_{PIP}, with the following relationships $\theta_{\text{DIP}} = \alpha_{\text{DIP}} \cdot \theta_{\text{PIP}}$ and $\theta_{\text{MCP}} = \alpha_{\text{MCP}} \cdot \theta_{\text{PIP}}$ where θ_{DIP} and θ_{MCP} represent the joint angles of the distal interphalangeal (DIP) joint and the metacarpophalangeal (MCP) joint, respectively. The constants α_{DIP} and α_{MCP} define proportional joint coordination. For the thumb, the relationship between the interphalangeal (IP) joint angle θ_{IP} and the trapeziometacarpal (TMCP) joint angle θ_{TMCP} is given by $\theta_{\text{IP}} = \alpha_{\text{TMCP}} \cdot \theta_{\text{TMCP}}$. Here, the constant α_{TMCP} defines the proportional coupling between the two joints, it reflects the biomechanical constraints of human thumb movement. The corresponding values for α_{DIP} are 0.77, 0.75, 0.75, and 0.57 for the index, middle, ring, and little fingers, respectively. The value of α_{MCP} is 0.67 for all fingers, while $\alpha_{\text{TMCP}} = 0.5$, as reported in [49, 60, 28].

The out-of-reach, grasp-success, and grasp-failure stages are termination stages and therefore have no associated action sets.

7.4.3 Observation Space

The observation space is defined as

$$\mathcal{O} = [n_{\mathrm{c}}, o_{\mathrm{dist}}, \boldsymbol{o}_{\mathrm{object}}, o_{\mathrm{cone}}, \boldsymbol{o}_{\mathrm{relative}}, \boldsymbol{o}_{\mathrm{force}}, \boldsymbol{o}_{\mathrm{torque}}] \tag{7.7}$$

where each component represents a critical aspect of the grasping task. The variable n_{c} indicates the number of contact points between the robot hand and the object which reflects contact extent and contributes to grasp stability. The distance o_{dist} measures the proximity of the robot hand to the grasp location. The vector $\boldsymbol{o}_{\mathrm{object}}$ represents the object position, which supports object out-of-range detection and task success or failure evaluation.

The vector $\boldsymbol{o}_{\mathrm{relative}}$ describes the spatial relationship between the robot hand and the object which provides essential grasp configuration details. The vectors $\boldsymbol{o}_{\mathrm{force}}$ and $\boldsymbol{o}_{\mathrm{torque}}$ represent the summed magnitudes of contact forces and torques at all contact points along the x, y, and z axes. They enable the evaluation of force and torque equilibrium. Together, these components comprehensively describe the grasping task which covers grasp position, configuration, and stability.

The Boolean variable o_{cone} indicates whether all contact forces lie within the friction cone and ensures stability through frictional constraints. The friction cone is defined by the coefficient of friction μ and the normal force $\mathbf{F}_n$, which satisfies the condition: $\|\mathbf{F}_c\| \leq \mu \cdot \mathbf{F}_n$ where $\mathbf{F}_c$ is the contact force vector. Grasp stability is determined as $o_{\mathrm{cone}} = 1$ if the above condition is satisfied for all contact points, and $o_{\mathrm{cone}} = 0$ otherwise. This ensures that all contact forces remain within the friction cone which prevents slippage and enhances grasp stability.

Each stage of the grasping task has specific goals and therefore requires specific information. The state abstraction function ϕ_i extracts the task-relevant information from the observation space which reduces computational complexity and improves efficiency.

7.4.4 Reward Function

The Approach Stage

The objective of this stage is to move the robot hand as close as possible to the grasp location while avoiding collisions and ensuring the object remains within the workspace. The reward function for this stage is defined as

$$r_{\text{appr}} = r^{\text{dist}} - \rho_{\text{appr}} n_{\text{c}} \tag{7.8}$$

where ρ_{appr} is a constant coefficient, and the distance-based reward is defined as $r^{\text{dist}} = -\text{e}^{|o_{\text{dist}}|}$, which is inversely proportional to the distance between the hand and the grasp location. The exponential form ensures rapid changes when the hand is far from the target and more gradual changes as it approaches the object. It encourages larger adjustments at greater distances and finer movements when closer. This enhances both the efficiency and accuracy of the task.

To reduce the risk of unintended collisions, which could cause the object to be knocked out of the workspace, a penalty proportional to the number of contact points n_{c} is applied. This reward design encourages precise and collision-free hand movements during the approach stage.

The Grasping Stage

The reward function for the grasping stage is designed to encourage stable and efficient grasping behaviors. The model is rewarded based on the number of contact points n_{c}, as a higher number of contact points leads to increased grasp stability. Additionally, the reward function evaluates equilibrium by minimizing net forces $\boldsymbol{o}_{\text{force}}$ and torques $\boldsymbol{o}_{\text{torque}}$ along the three axes at all contact points. The closer these values are to zero, the higher the reward, which indicates a more stable and secure grip. The reward function for the grasping stage

is defined as

$$r_{\text{grasp}} = r^{\text{equil}} + \rho_{\text{grasp}} n_{\text{c}} + R_{\text{T}}^{\text{arrive}} \tag{7.9}$$

where the equilibrium reward is expressed as

$$r^{\text{equil}} = -\mathrm{e}^{|\boldsymbol{o}_{\text{force}}|} - \mathrm{e}^{|\boldsymbol{o}_{\text{torque}}|} \tag{7.10}$$

The coefficient ρ_{grasp} adjusts the reward's sensitivity to the number of contact points. The stage transition reward $R_{\text{T}}^{\text{arrive}} = R_{\text{T}}^{T_1 \to T_2}$ encourages the transition from the approach stage to the grasping stage. Its value is set higher than the maximum achievable reward in the approach stage r_{appr} to ensure that transitioning to the grasping stage is prioritized, while still maintaining a stable and controlled grip.

The Termination Stages

The out-of-reach, the grasp-success, and the grasp-failure stages monitor task outcomes, where the reward is only related to the position of the object $\boldsymbol{o}_{\text{object}}$. The reward function for the out-of-reach stage is defined as

$$r_{\text{oor}} = R_{\text{T}}^{\text{aor}} \cdot \mathbf{1}_{\{e^{\text{aor}}\}} + R_{\text{T}}^{\text{gor}} \cdot \mathbf{1}_{\{e^{\text{gor}}\}} \tag{7.11}$$

where both transition rewards are negative penalties. The condition $R_{\text{T}}^{\text{aor}} = R_{\text{T}}^{T_1 \to T_3} < R_{\text{T}}^{\text{gor}} = R_{\text{T}}^{T_2 \to T_3}$ reflects greater task progress when transitioning from the grasping stage compared to the approach stage. The indicator function $\mathbf{1}_{\{e\}}$ equals 1 if the event e occurs and 0 otherwise. It ensures the right penalty is applied only when specific transitions occur. Here, $e^{\text{aor}} = e^{T_1 \to T_3}$ and $e^{\text{gor}} = e^{T_2 \to T_3}$.

The reward function for the grasp-failure stage is

$$r_{\text{fail}} = R_{\text{T}}^{\text{fail}} \tag{7.12}$$

where $R_{\mathrm{T}}^{\mathrm{fail}} = R_{\mathrm{T}}^{T_2 \to T_4}$ is a smaller penalty compared to the out-of-reach stage, as it acknowledges partial task completion.

In addition to monitoring task outcomes, the grasp-success stage evaluates grasp quality through a friction cone analysis. The reward function of this stage is defined as

$$r_{\mathrm{succ}} = R_{\mathrm{T}}^{\mathrm{succ}} + R_{\mathrm{cone}} \cdot \mathbf{1}_{\{o_{\mathrm{cone}}\}} \tag{7.13}$$

where $R_{\mathrm{T}}^{\mathrm{succ}} = R_{\mathrm{T}}^{T_2 \to T_5}$ is a large reward for successfully transitioning to this stage, and R_{cone} is an additional reward given only when all contact points satisfy the friction cone condition. This reward design encourages both task completion and a stable grasp.

7.5 Experiments

The proposed method is evaluated in a simulated environment and compared with the state-of-the-art methods. To validate its real-world applicability, the method is transferred to a physical robot for performance testing. The evaluation results are analyzed and discussed in this section.

7.5.1 Experiment Setup

Environment Setup

The task environment for the grasping task is illustrated in Fig. 7.5. The target object is placed on a table in front of a dual-arm mobile robot. This robot comprises a Husky UGV (Unmanned Ground Vehicle) for mobility, two UR5e robotic arms, a Schunk SVH robotic hand (right), and a PSYONIC Ability Hand (left). Each robotic hand is mounted on a UR5e arm, and both arms are attached to the Husky UGV. This configuration enables coordinated manipulation.

The robot integrates various sensors to enhance perception and interaction. The Schunk SVH hand is equipped with an ErgoGLOVE Force Sensing System for contact force detection, while the PSYONIC Ability Hand features built-in force sensors. A RealSense D435 depth

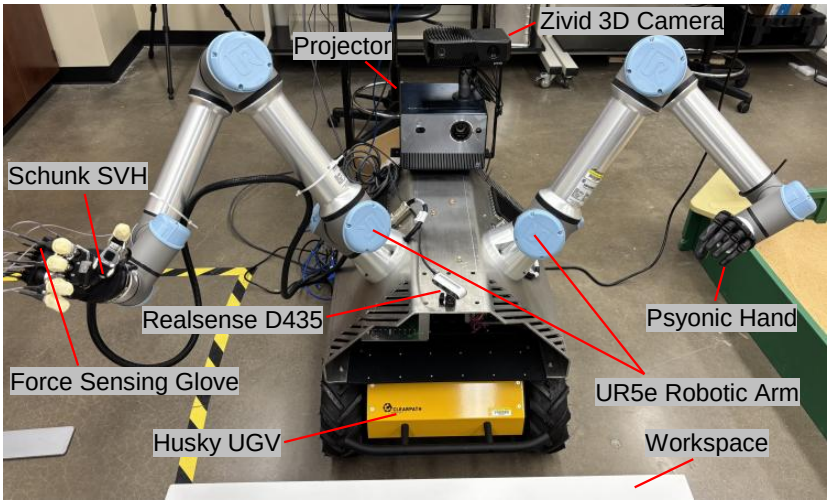

Figure 7.5: Real-world experiment setup for grasping tasks.

camera provides visual input which enables precise object recognition, object pose estimation, and environmental awareness. Additionally, a Zivid One 3D camera and a projector are also part of the robot but are not utilized in this study.

A simulation environment was developed using PyBullet and OpenAI Gym to train and test the proposed model. It replicates the real-world setup of the robot, which enables seamless transfer of learned policies to the physical robot for performance evaluation and practical applications.

Dataset

The AffordPose dataset serves as a benchmark for robotic grasping tasks which emphasizes affordance-based pose estimation [33]. It provides 3D object models, annotated grasp poses, and corresponding affordance labels.

For this work, we refined the AffordPose dataset by removing redundant entries, classifying grasp poses based on the grasp taxonomy defined in Fig. 7.2, and computing grasp locations to support the proposed grasping task framework. The revised dataset comprises six unique grasp poses, seven distinct grasping objectives, and 20 diverse objects. As a result, the dataset contains a total of 26 different grasping tasks across various objects, purposes,

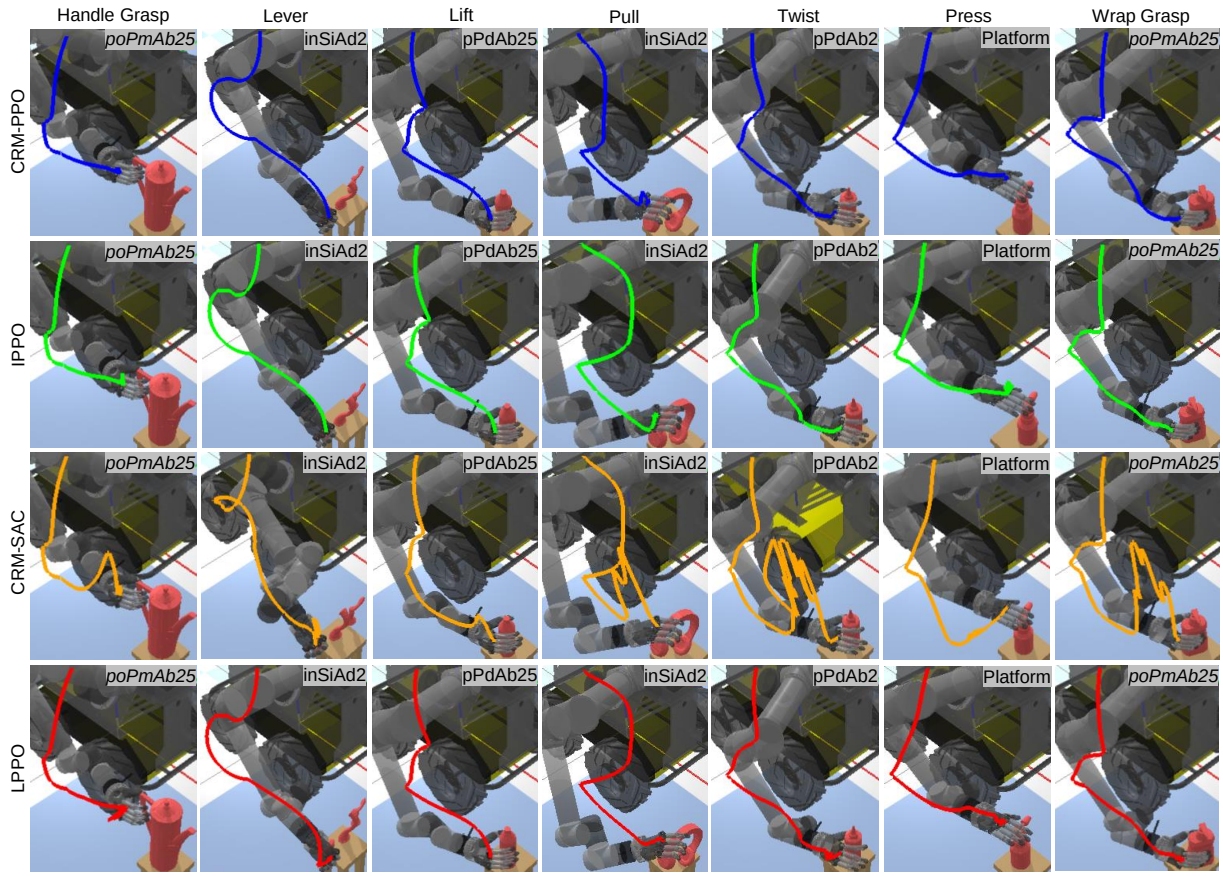

Figure 7.6: Trajectory visualization of grasping tasks performed by the proposed model and baseline models across different task objectives and grasp topologies.

and grasp configurations.

7.5.2 Performance Evaluation in the Simulation Environment

Evaluation Metrics

The proposed CRM-PPO model was evaluated against three baseline models using the benchmark dataset. The evaluation is divided into two parts: (1) performance assessment within the CRM framework and (2) comparison with state-of-the-art methods.

The first baseline, CRM-SAC, integrates the CRM framework with the Soft Actor-Critic (SAC) algorithm. This setup enables a direct comparison with CRM-PPO under identical conditions, which demonstrates the superior performance of PPO within the CRM framework. The second and third baselines, LPPO (Learning-based PPO) and IPPO (Im-

proved PPO), represent state-of-the-art approaches for grasping tasks based on stage-wise mechanisms. LPPO employs hierarchical dense rewards to enhance training efficiency and generalization across diverse object configurations [65]. IPPO employs a stage-wise sparse reward structure, which improves convergence speed and grasping accuracy [83]. The comparison with these baselines confirms the performance advantages introduced by the CRM framework.

The evaluation metrics include task success rate and average task completion time in timesteps.

Training Setup

The proposed model[1] is trained for approximately 12,000 episodes (equivalent to 5 million timesteps) using a discount factor of 0.99, a GAE lambda of 0.95, and a batch size of 64. A dynamic learning rate is applied, starting at 3×10^{-5} for the first 40% of training progress. Between 40% and 70% progress, the learning rate is reduced to 90% of its initial value. During the final 30% of training, it is further reduced to 80%.

During training, the robot hand performs grasping tasks with different task objectives and grasp topologies on a variety of objects. To simulate real-world uncertainties, random noise of $\pm 3\,\text{mm}$ in object position, $\pm 11.5^\circ$ in object orientation, and $0.02\,\text{rad}$ in joint positions was applied. This domain randomization method enhances the model's robustness and generalization to real-world scenarios by introducing randomized variations in object pose and joint configurations during training.

For task objectives such as handle grasp, lift, lever, pull, and wrap grasp, a task is considered successful if the robot picks up the object and holds it steadily for 5 seconds without dropping it. For the twist objective, success is defined as applying sufficient torque in the twisting direction after grasping the object. In the press objective, success is achieved if the robot applies enough force in the pressing direction.

[1]This model is available at https://github.com/hhelium/DexMobile

Table 7.1: Testing results for the proposed and baseline models.

Models	IPPO [83]		LPPO [65]		CRM-SAC		CRM-PPO	
	Success Rate	**Episode Length**	**Success Rate**	**Episode Length**	**Success Rate**	**Episode Length**	**Success Rate**	**Episode Length**
Lift	0.85	412.63	0.88	456.33	0.63	534.68	**0.88**	**342.43**
Pull	0.86	433.60	0.76	315.82	**1.00**	276.22	**1.00**	**217.34**
Press	0.76	415.82	0.29	793.22	0.33	787.64	**0.96**	**167.30**
Twist	0.74	436.20	0.67	531.34	0.49	599.17	**0.91**	**335.52**
Lever	0.70	466.80	0.90	367.67	**0.95**	307.24	0.93	**246.15**
Wrap-Grasp	0.89	348.88	0.67	372.49	0.60	377.41	**1.00**	**252.17**
Handle-Grasp	**0.98**	303.70	0.97	369.71	0.75	338.43	0.97	**281.09**
Overall	0.84	390.86	0.71	461.75	0.61	479.36	**0.95**	**273.07**

An early stopping mechanism is implemented to improve training efficiency. The training process is terminated when the average success rate of the most recent 100 episodes reaches or exceeds 99%.

Examples of trajectories generated by the proposed and baseline methods are shown in Fig. 7.6. The figure illustrates that the trajectory generated by the PPO-based method is smoother and more optimized compared to that of the SAC-based method. This difference may be due to SAC's lower efficiency in tasks with well-shaped rewards.

For the PPO-based method, both the proposed and baseline approaches exhibit similar trajectories. The proposed method completes the trajectory significantly faster and avoids overlapping or revisiting previous paths. This improvement is due to the stage-specific context and transition rewards, which effectively guide the model to complete the task efficiently and without redundancy.

Result Analysis

The proposed CRM-PPO model and the three baselines were trained five times each. For each model, the average success rate and average episode length were calculated as evaluation metrics, along with their standard deviations to reflect performance variability across runs. The training results are presented in Fig. 7.7, where solid curves represent average values and shaded areas indicate standard deviations.

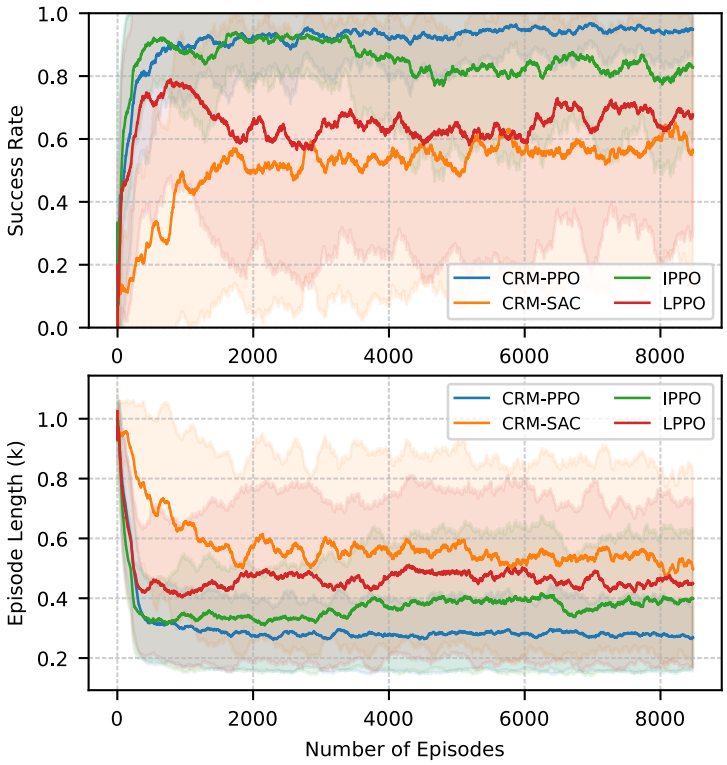

Figure 7.7: Comparison of different models in grasping tasks based on success rate and episode length.

The proposed CRM-PPO model outperforms all baselines across all evaluation metrics. It records the highest average success rate and the shortest average episode length. The model reaches the early termination criterion after approximately 8,200 episodes on average, while none of the baseline models meet this criterion within the full training period. All five runs of CRM-PPO reach early termination and show the lowest standard deviation. These results reflect superior robustness, consistency, and sample efficiency. This strong performance results from its hierarchical reward structure, which provides clear stage-specific guidance and supports effective transitions between stages through transition rewards.

The performance of IPPO surpasses that of LPPO, primarily due to differences in reward design. IPPO assigns sparse rewards based on stage transitions, while LPPO defines continuous intra-stage rewards without explicitly encouraging transitions. In summary, IPPO provides only transition rewards, whereas LPPO offers only intra-stage rewards. IPPO outperforms LPPO because its sparse reward structure effectively guides the agent toward

target stages. In contrast, LPPO focuses more on intra-stage optimization and lacks incentives for transitioning between stages, which results in lower overall performance despite its intra-stage guidance.

During early training, IPPO's performance approximates that of the proposed CRM-PPO model, as its sparse reward structure provides implicit transition-based guidance. Due to the absence of intra-stage rewards, IPPO does not support fine-grained exploration and adaptation, which reduces data efficiency. As training progresses, this limitation causes IPPO's performance to lag behind CRM-PPO.

The CRM-PPO model effectively integrates both approaches by combining stage-specific guidance and transition rewards, and it outperforms the baseline models across evaluation metrics. CRM-SAC performs the worst across all evaluation metrics, which indicates that PPO-based methods are better suited for hierarchical grasping tasks.

The proposed model and baseline models were evaluated using a benchmark dataset, with each model tested 1,000 times on randomly selected tasks. The results are summarized in Table 7.1. The proposed model consistently outperformed baseline models in most tasks. Notably, it excelled in the challenging Twist task, where baseline models showed relatively low success rates. This task demands precise and coordinated actions, it demonstrates CRM-PPO's ability to handle complex scenarios due to its task-specific structured design, which provides effective execution guidance.

In the Lever and Handle-Grasp tasks, the proposed model achieved slightly lower success rates, trailing the best-performing baseline by 2% and 1%, respectively. Despite this, it surpassed all baselines in episode length and completed tasks more efficiently. These results confirm the effectiveness of the CRM-PPO model in task-oriented grasping tasks.

7.5.3 Real-World Evaluation

The robots are integrated through ROS2 Humble on Ubuntu 22.04 with a low-latency kernel to support real-time performance. Inverse kinematics and collision avoidance are managed through MoveIt2. An Intel RealSense D435 depth camera is spatially aligned

with the robot's coordinate frame via an eye-to-hand calibration, which enables accurate perception and interaction within the shared workspace. The depth images align with the color images, both with a resolution of 640 $\times$ 480 pixels. The image streams and force feedback data from the ErgoGLOVE Force Sensing System are synchronized and published to the proposed model through ROS2 topics. This setup ensures coherent sensory input for perception and interaction tasks.

Even though the simulation and the real robot are identical in design, a sim-to-real gap persists due to discrepancies in dynamic properties, environmental conditions, and sensor noise. To bridge this gap, domain randomization is employed [72]. The policy is trained across numerous variations of simulation parameters, such as joint position error, object pose error, and sensor noise. By randomizing these parameters over a wide range, the likelihood of the policy generalizing to real-world conditions increases significantly.

The robot perceives the object's pose and the contact forces between the Schunk hand and the object. The object pose is estimated using FoundationPose [77], which provides millimeter-level accuracy in pose estimation. Based on the object pose, the observations o_{dist}, $\boldsymbol{o}_{\text{object}}$, and $\boldsymbol{o}_{\text{relative}}$ can be calculated. The ErgoGLOVE Force Sensing System detects the contact force between the Schunk hand and the object and provides n_{c}. Since the force-sensing glove measures force along only one axis, $\boldsymbol{o}_{\text{force}}$, $\boldsymbol{o}_{\text{torque}}$, and o_{cone} cannot be detected and are set to zero.

In the simulation environment, the grasping force cannot be determined properly due to the agnostic nature of the target object's material properties, as it is represented by a 3D model. As a result, the grasping force cannot be accurately calibrated, and the object remains unaffected by any applied force. In contrast, in the real world, the contact force must be carefully controlled to avoid damaging the object or the robotic hand. To address this discrepancy and enable the transfer of the simulation model to a physical robot, we manually defined grasping force thresholds. For fragile objects, the contact force was constrained to a maximum of 1 N, with the robotic fingers halting motion upon reaching this threshold. For

sturdy objects, the threshold was set between 1 N and 3 N, depending on factors such as the object's texture and weight. Some representative feedback from the system is illustrated in Fig. 7.8.

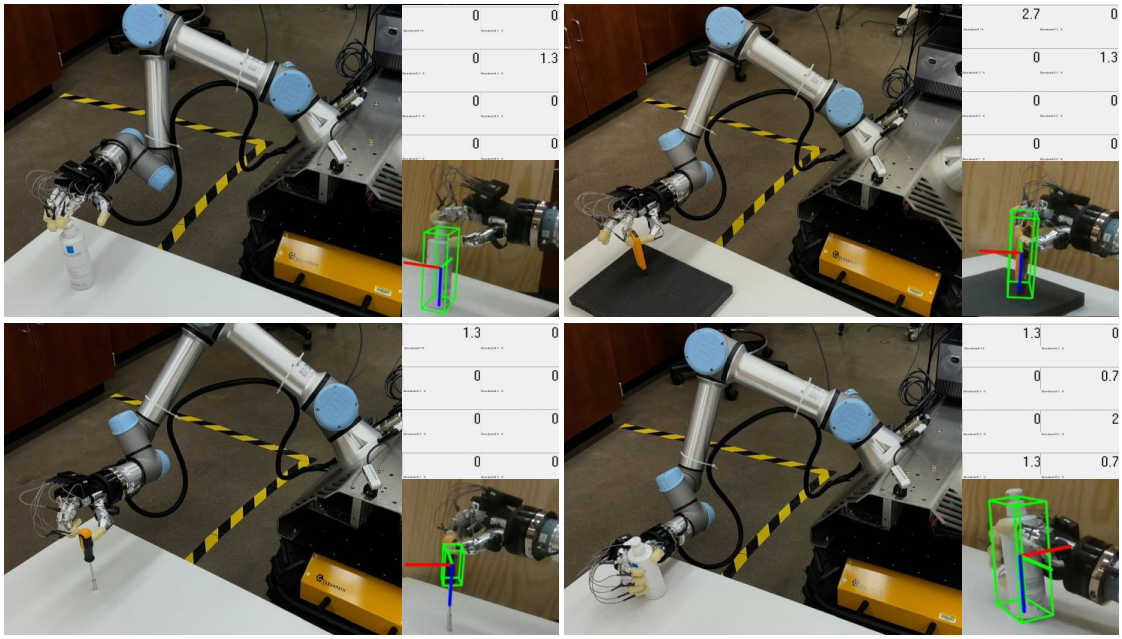

Figure 7.8: Representative feedback of force and object pose collected from the robotic testbed.

The proposed CRM-PPO model is fine-tuned for 300 episodes across six affordances, including Handle Grasp, Lift, Press, Pull, Twist, and Wrap Grasp, each utilizing the corresponding grasp topology, as shown in Fig. 7.9. The achieved testing success rates for these tasks are 100%, 90%, 60%, 80%, 70%, and 100%, respectively. In the experiment, the grasping tasks for press and twist exhibited lower success rates of 60% and 70%, respectively. The low success rate for the twist task is attributed to the small, round-shaped handle of the screwdriver, which is prone to slipping when grasped using the topology inSiAd2. Additionally, the screwdriver, when placed on the table, can be easily knocked over with even light contact from the hand. The press task had the lowest success rate due to the limitations of the sensing glove. The eight force sensors on the glove cover only a limited area of the hand, and during pressing, the actual contact area often lacks sensor coverage, which can result in task failure. After sensor repositioning, the success rate improved significantly, while success

rates for other tasks decreased. The overall success rate achieved was 83.3%, which can be further improved by adding more force sensors and further fine-tuning.

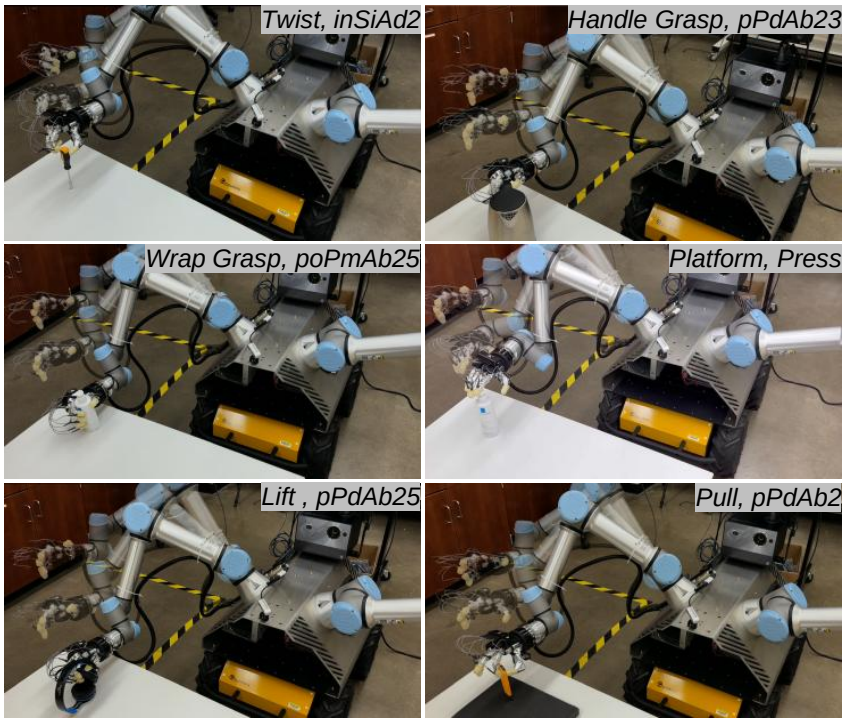

Figure 7.9: Real-world grasping tasks across different affordances and grasp topologies.

7.6 Conclusion

This paper presents a context-aware task-oriented dexterous grasping approach leveraging a Contextual Reward Machine framework. The CRM decomposes complex grasping tasks into modular sub-tasks with stage-specific contexts, which enables efficient learning and execution. By integrating Proximal Policy Optimization, the proposed method achieves significant improvements in learning efficiency, task performance, and adaptability. Extensive experiments in simulated and real-world environments validated the effectiveness and robustness of the proposed approach. The CRM-PPO model achieved a 95% success rate in simulation across 1,000 grasping tasks. When transferred to a real robot, it attained an 83.3% success rate over 60 real-world tasks. The proposed model exceeded the performance of state-of-the-art models in success rate and task completion time. Its ability to adapt to

diverse grasping objectives, various grasp topologies, and dynamic conditions highlights its practical applicability in unstructured environments. The results demonstrate the potential of the CRM-PPO framework to advance robotic dexterity and manipulation.

CHAPTER 8

CONCLUSION

8.1 Summary

This research presents an efficient approach to complex task-oriented grasping using an anthropomorphic robotic hand. The proposed method leverages advanced sensing and learning technologies to enable adaptive grasping based on object affordances and task requirements. Extensive experiments validate the effectiveness and practical applicability of the system, demonstrating its potential to advance the field of robotic manipulation.

8.2 Contributions

The primary contributions of this work are summarized as follows:

1. Implemented a novel method for real-time, non-destructive identification of an object's material composition with high accuracy using near-infrared spectroscopy, eliminating the need for time-consuming chemical analysis.

2. Developed a context-aware dexterous manipulation system that integrates multiple sensors to precisely capture critical object attributes, including position, orientation, and shape.

3. Implemented a method that leverages a large language model to augment object knowledge through commonsense reasoning, particularly for challenging properties such as rigidity, texture, and fragility.

4. Designed and trained a multi-class, multi-label deep learning model to learn grasp strategies from human demonstrations.

5. Proposed an innovative Contextual Reward Machine (CRM) within a reinforcement learning framework that decomposes complex grasping tasks into manageable subtasks, each characterized by stage-specific reward functions, action spaces, and abstracted state representations. Transition rewards are introduced to guide progression between stages. When integrated with Proximal Policy Optimization (PPO), the CRM framework significantly improves sample efficiency and model robustness.

6. Developed multiple robotic systems and their corresponding digital twins to validate the proposed approach. The CRM-based reinforcement learning model was trained in simulation using benchmark datasets and transferred to real-world robots via domain randomization.

7. Established multiple datasets, including one for material recognition using near-infrared spectra, one for learning from human demonstrations, and another for contextual grasp planning, to support training and benchmarking of the proposed models.

8.3 Future Work

The proposed method uses reinforcement learning to adaptively deploy grasping strategies. However, the contact positions and orientations where the fingers make contact with the object are not explicitly planned, which is critical for enabling subsequent dexterous manipulation. My future work will address this limitation using a contact-aware optimization approach that incorporates tactile feedback and object geometry to predict and control finger placement. While this study focuses solely on the grasping task, the full scope of dexterous manipulation such as in-hand reorientation, tool use, or fine manipulation has not been considered. Building on a reliable grasp, future work will aim to plan and execute the post-grasp manipulation process to optimize task execution. Ultimately, the model will

be extended to enable dual-arm cooperative manipulation, allowing for more complex and coordinated tasks that require bimanual interaction.

8.4 Research Recognition and Publications

This study has received the NSF I-Corps Grant, the Shocker Innovation Award, the Excellent PhD Award, and the Outstanding Research Output Award. It is supported by the following publications:

1. Hui Li, Akhlak Uz Zaman, Fujian Yan, Hongsheng He. "Task-Oriented Grasping Using Reinforcement Learning with a Contextual Reward Machine". IEEE Transactions on SMC: Systems (Under review)

2. Yun Chen, Xinyu Zhang, Hui Li, Hongsheng He, Qiang Zhang. "Towards Neurorobotic Interface for Finger Joint Angle Estimation: A Multi-Stage CNN-LSTM Network with Transfer Learning". 2025 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2025.

3. Hui Li, Akhlak Uz Zaman, and Hongsheng He "Grasp Intention Interpretation in Object Handover for Human-Robot Teaming". International Conference on Social Robotics. Springer Nature Singapore 2024

4. Akhlak Uz Zaman, Hui Li, Fujian Yan, Yinlong Zhang, Hongsheng He "Omnisurface: Common Reality for Intuitive Human-Robot Collaboration". International Conference on Social Robotics. Springer Nature Singapore 2024

5. Li, Hui, et al. "Knowledge Augmentation and Task Planning in Large Language Models for Dexterous Grasping". 2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids). IEEE, 2023.

6. Tran, Dang, Hui Li, and Hongsheng He. "AI Planning from Natural-Language Instructions for Trustworthy Human-Robot Communication". International Conference on Social Robotics. Singapore: Springer Nature Singapore, 2023.

7. Li, Hui, et al. "Learning task-oriented dexterous grasping from human knowledge". 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021.

8. Rao, Bharath, Hui, Li, et al. "Knowledge-augmented dexterous grasping with incomplete sensing". arXiv preprint arXiv:2011.08361 (2020).

9. Li, Hui, Jindong Tan, and Hongsheng He. "Magichand: Context-aware dexterous grasping using an anthropomorphic robotic hand". 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2020.

10. Rao, Achyutha Bharath, Hui Li, and Hongsheng He. "Object recall from natural-language descriptions for autonomous robotic grasping". 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO). IEEE, 2019.

11. Li, Hui, Yimesker Yihun, and Hongsheng He. "Magichand: In-hand perception of object characteristics for dexterous manipulation". Social Robotics: 10th International Conference, ICSR 2018, Qingdao, China, November 28-30, 2018, Proceedings 10. Springer International Publishing, 2018.

REFERENCES

[1] Alaa Awad Abdellatif, Amr Mohamed, Carla Fabiana Chiasserini, Mounira Tlili, and Aiman Erbad. Edge computing for smart health: Context-aware approaches, opportunities, and challenges. *IEEE Network*, 33(3):196–203, 2019.

[2] Arash Ajoudani, Andrea Maria Zanchettin, Serena Ivaldi, Alin Albu-Schäffer, Kazuhiro Kosuge, and Oussama Khatib. Progress and prospects of the human–robot collaboration. *Autonomous robots*, 42:957–975, 2018.

[3] Edgar Batista, M Angels Moncusi, Pablo López-Aguilar, Antoni Martínez-Ballesté, and Agusti Solanas. Sensors for context-aware smart healthcare: A security perspective. *Sensors*, 21(20):6886, 2021.

[4] Stefan Hein Bengtson, Thomas Bak, Lotte NS Andreasen Struijk, and Thomas Baltzer Moeslund. A review of computer vision for semi-autonomous control of assistive robotic manipulators (arms). *Disability and Rehabilitation: Assistive Technology*, 15(7):731–745, 2020.

[5] Ning Bian, Xianpei Han, Le Sun, Hongyu Lin, Yaojie Lu, and Ben He. Chatgpt is a knowledgeable but inexperienced solver: An investigation of commonsense problem in large language models. *arXiv preprint arXiv:2303.16421*, 2023.

[6] Antonio Bicchi and Vijay Kumar. Robotic grasping and contact: A review. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 348–353. IEEE, 2000.

[7] Aude Billard and Danica Kragic. Trends and challenges in robot manipulation. *Science*, 364(6446):eaat8414, 2019.

[8] Ian M Bullock, Thomas Feix, and Aaron M Dollar. The yale human grasping dataset: Grasp, object, and task data in household and machine shop environments. *The International Journal of Robotics Research*, 34(3):251–255, 2015.

[9] Maya Cakmak, Siddhartha S Srinivasa, Min Kyung Lee, Jodi Forlizzi, and Sara Kiesler. Human preferences for robot-human hand-over configurations. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1986–1993. IEEE, 2011.

[10] Roberto Calandra, Andrew Owens, Dinesh Jayaraman, Justin Lin, Wenzhen Yuan, Jitendra Malik, Edward H Adelson, and Sergey Levine. More than a feeling: Learning to grasp and regrasp using vision and touch. *IEEE Robotics and Automation Letters*, 3(4):3300–3307, 2018.

[11] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5556–5565, 2015.

[12] Matei Ciocarlie, Corey Goldfeder, and Peter Allen. Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem. In *Robotics: Science and systems manipulation workshop-sensing and adapting to the real world*. Citeseer, 2007.

[13] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2019.

[14] Mark R Cutkosky et al. On grasp choice, grasp models, and the design of hands for manufacturing tasks. *IEEE Transactions on robotics and automation*, 5(3):269–279, 1989.

[15] Raphael Deimel and Oliver Brock. A novel type of compliant and underactuated robotic hand for dexterous grasping. *The International Journal of Robotics Research*, 35(1-3):161–185, 2016.

[16] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013.

[17] Anind K Dey. Understanding and using context. *Personal and ubiquitous computing*, 5:4–7, 2001.

[18] Haonan Duan, Peng Wang, Yayu Huang, Guangyun Xu, Wei Wei, and Xiaofei Shen. Robotics dexterous grasping: The methods based on point cloud and deep learning. *Frontiers in Neurorobotics*, 15:658280, 2021.

[19] Yongxiang Fan, Liting Sun, Minghui Zheng, Wei Gao, and Masayoshi Tomizuka. Robust dexterous manipulation under object dynamics uncertainties. In *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 613–619. IEEE, 2017.

[20] Thomas Feix, Ian M Bullock, and Aaron M Dollar. Analysis of human grasping behavior: Correlating tasks, objects and grasps. *IEEE transactions on haptics*, 7(4):430–441, 2014.

[21] Thomas Feix, Roland Pawlik, Heinz-Bodo Schmiedmayer, Javier Romero, and Danica Kragic. A comprehensive grasp taxonomy. In *Robotics, science and systems: workshop on understanding the human hand for advancing robotic manipulation*, volume 2, pages 2–3. Seattle, WA, USA;, 2009.

[22] Thomas Feix, Javier Romero, Heinz-Bodo Schmiedmayer, Aaron M Dollar, and Danica Kragic. The grasp taxonomy of human grasp types. *IEEE Transactions on Human-Machine Systems*, 46(1):66–77, 2015.

[23] Thomas Feix, Javier Romero, Heinz-Bodo Schmiedmayer, Aaron M Dollar, and Danica Kragic. The grasp taxonomy of human grasp types. *IEEE Transactions on Human-Machine Systems*, 46(1):66–77, 2016.

[24] JA George, DT Kluger, TS Davis, SM Wendelken, EV Okorokova, Q He, CC Duncan, DT Hutchinson, ZC Thumser, DT Beckler, et al. Biomimetic sensory feedback through peripheral nerve stimulation improves dexterous use of a bionic hand. *Science Robotics*, 4(32):eaax2352, 2019.

[25] James J Gibson. *The ecological approach to visual perception: classic edition*. Psychology press, 2014.

[26] Abhishek Gupta, Clemens Eppner, Sergey Levine, and Pieter Abbeel. Learning dexterous manipulation for a soft robotic hand from human demonstrations. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 3786–3793. IEEE, 2016.

[27] Wei He, Hejia Gao, Chen Zhou, Chenguang Yang, and Zhijun Li. Reinforcement learning control of a flexible two-link manipulator: an experimental investigation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(12):7326–7336, 2020.

[28] Christopher-Eyk Hrabia, Katrin Wolf, and Mathias Wilhelm. Whole hand modeling using 8 wearable sensors: Biomechanics for hand pose prediction. In *Proceedings of the 4th Augmented Human International Conference*, pages 21–28, 2013.

[29] Yingbai Hu, Zhijun Li, Guanglin Li, Peijiang Yuan, Chenguang Yang, and Rong Song. Development of sensory-motor fusion-based manipulation and grasping control for a robotic hand-eye system. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(7):1169–1180, 2016.

[30] Markus Huber, Markus Rickert, Alois Knoll, Thomas Brandt, and Stefan Glasauer. Human-robot interaction in handing-over tasks. In *RO-MAN 2008-the 17th IEEE international symposium on robot and human interactive communication*, pages 107–112. IEEE, 2008.

[31] Rodrigo Toro Icarte, Toryn Q Klassen, Richard Valenzano, and Sheila A McIlraith. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73:173–208, 2022.

[32] Jasper Wollaston James and Nathan F Lepora. Slip detection for grasp stabilization with a multifingered tactile robot hand. *IEEE Transactions on Robotics*, 37(2):506–519, 2020.

[33] Juntao Jian, Xiuping Liu, Manyi Li, Ruizhen Hu, and Jian Liu. Affordpose: A large-scale dataset of hand-object interactions with affordance-driven hand pose. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14713–14724, 2023.

[34] Roland S Johansson and J Randall Flanagan. Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nature Reviews Neuroscience*, 10(5):345, 2009.

[35] M Humayun Kabir, M Robiul Hoque, Hyungyu Seo, and Sung-Hyun Yang. Machine learning based adaptive context-aware system for smart home environment. *International Journal of Smart Home*, 9(11):55–62, 2015.

[36] Yara Khaluf and Marco Dorigo. Modeling robot swarms using integrals of birth-death processes. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 11(2):8, 2016.

[37] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000.

[38] Gerard Lacey and Shane MacNamara. Context-aware shared control of a robot mobility aid for the elderly blind. *The International Journal of Robotics Research*, 19(11):1054–1065, 2000.

[39] Hui Li, Jindong Tan, and Hongsheng He. Magichand: Context-aware dexterous grasping using an anthropomorphic robotic hand. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9895–9901. IEEE, 2020.

[40] Hui Li, Yimesker Yihun, and Hongsheng He. Magichand: In-hand perception of object characteristics for dexterous manipulation. In *Social Robotics: 10th International Conference, ICSR 2018, Qingdao, China, November 28-30, 2018, Proceedings 10*, pages 523–532. Springer, 2018.

[41] Hui Li, Yinlong Zhang, Yanan Li, and Hongsheng He. Learning task-oriented dexterous grasping from human knowledge. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6192–6198. IEEE, 2021.

[42] Yu Liu, Xi Liu, Song Gao, Li Gong, Chaogui Kang, Ye Zhi, Guanghua Chi, and Li Shi. Social sensing: A new approach to understanding our socioeconomic environments. *Annals of the Association of American Geographers*, 105(3):512–530, 2015.

[43] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[44] Jeffrey Mahler, Matthew Matl, Vishal Satish, Michael Danielczuk, Bill DeRose, Stephen McKinley, and Ken Goldberg. Learning ambidextrous robot grasping policies. *Science Robotics*, 4(26):eaau4984, 2019.

[45] Marco Mamei and Franco Zambonelli. Pervasive pheromone-based interaction with rfid tags. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 2(2):4, 2007.

[46] Priyanka Mandikal and Kristen Grauman. Learning dexterous grasping with object-centric visual affordances. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 6169–6176. IEEE, 2021.

[47] Priyanka Mandikal and Kristen Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661. PMLR, 2022.

[48] Andrea H Mason and Christine L MacKenzie. Grip forces when passing an object to a partner. *Experimental brain research*, 163:173–187, 2005.

[49] M Mentzel, A Benlic, NJ Wachter, D Gulkin, S Bauknecht, and J Gülke. The dynamics of motion sequences of the finger joints during fist closure. *Handchirurgie, Mikrochirurgie, Plastische Chirurgie: Organ der Deutschsprachigen Arbeitsgemeinschaft fur Handchirurgie: Organ der Deutschsprachigen Arbeitsgemeinschaft fur Mikrochirurgie der Peripheren Nerven und Gefasse: Organ der V...*, 43(3):147–154, 2011.

[50] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[51] Wei-Hao Mou, Ming-Fang Chang, Chien-Ke Liao, Yuan-Han Hsu, Shih-Huan Tseng, and Li-Chen Fu. Context-aware assisted interactive robotic walker for parkinson's disease patients. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 329–334. IEEE, 2012.

[52] John R Napier. The prehensile movements of the human hand. *The Journal of bone and joint surgery. British volume*, 38(4):902–913, 1956.

[53] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 143–152, 2017.

[54] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[55] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE communications surveys & tutorials*, 16(1):414–454, 2013.

[56] Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Hao Su, and Xiaolong Wang. Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *Conference on Robot Learning*, pages 594–605. PMLR, 2023.

[57] Achyutha Bharath Rao, Krishna Krishnan, and Hongsheng He. Learning robotic grasping strategy based on natural-language object descriptions. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 882–887. IEEE, 2018.

[58] Achyutha Bharath Rao, Hui Li, and Hongsheng He. Object recall from natural-language descriptions for autonomous robotic grasping. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1368–1373. IEEE, 2019.

[59] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[60] Alba Roda-Sales, Joaquín L Sancho-Bru, and Margarita Vergara. Studying kinematic linkage of finger joints: estimation of kinematics of distal interphalangeal joints during manipulation. *PeerJ*, 10:e14051, 2022.

[61] Marco Santello, Martha Flanders, and John F Soechting. Postural hand synergies for tool use. *Journal of Neuroscience*, 18(23):10105–10115, 1998.

[62] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *wmcsa*, pages 85–90. IEEE, 1899.

[63] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[64] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[65] Asad Ali Shahid, Loris Roveda, Dario Piga, and Francesco Braghin. Learning continuous control actions for robotic grasping with reinforcement learning. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 4066–4072. IEEE, 2020.

[66] Weiwei Shang, Fangjing Song, Zengzhi Zhao, Hongbo Gao, Shuang Cong, and Zhijun Li. Deep learning method for grasping novel objects using dexterous hands. *IEEE Transactions on Cybernetics*, 52(5):2750–2762, 2020.

[67] Kyle Strabala, Min Kyung Lee, Anca Dragan, Jodi Forlizzi, Siddhartha S Srinivasa, Maya Cakmak, and Vincenzo Micelli. Toward seamless human-robot handovers. *Journal of Human-Robot Interaction*, 2(1):112–132, 2013.

[68] Freek Stulp, Evangelos A Theodorou, and Stefan Schaal. Reinforcement learning with sequences of motion primitives for robust manipulation. *IEEE Transactions on robotics*, 28(6):1360–1370, 2012.

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

[70] Mahdi Tavakoli, Jay Carriere, and Ali Torabi. Robotics, smart wearable technologies, and autonomous intelligent systems for healthcare during the covid-19 pandemic: An analysis of the state of the art and future vision. *Advanced intelligent systems*, 2(7):2000071, 2020.

[71] Thomas George Thuruthel, Egidio Falotico, Federico Renda, and Cecilia Laschi. Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators. *IEEE Transactions on Robotics*, 35(1):124–134, 2018.

[72] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[73] Dang Tran, Fujian Yan, Yimesker Yihun, Jindong Tan, and Hongsheng He. A framework of controlled robot language for reliable human-robot collaboration. In *International Conference on Social Robotics*, pages 339–349. Springer, 2021.

[74] Elio Tuci, Roderich Groß, Vito Trianni, Francesco Mondada, Michael Bonani, and Marco Dorigo. Cooperation through self-assembly in multi-robot systems. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 1(2):115–150, 2006.

[75] Hamed Vahdat-Nejad, Zahra Abbasi-Moud, Seyed Abolfazl Eslami, and Wathiq Mansoor. Survey on context-aware healthcare systems. In *2021 IEEE 11th annual computing and communication workshop and conference (CCWC)*, pages 1190–1196. IEEE, 2021.

[76] Ruicheng Wang, Jialiang Zhang, Jiayi Chen, Yinzhen Xu, Puhao Li, Tengyu Liu, and He Wang. Dexgraspnet: A large-scale robotic dexterous grasp dataset for general ob-

jects based on simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11359–11366. IEEE, 2023.

[77] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17868–17879, 2024.

[78] Bohan Wu, Suraj Nair, Li Fei-Fei, and Chelsea Finn. Example-driven model-based reinforcement learning for solving long-horizon visuomotor tasks. *arXiv preprint arXiv:2109.10312*, 2021.

[79] Natsuki Yamanobe, Weiwei Wan, Ixchel G Ramirez-Alpizar, Damien Petit, Tokuo Tsuji, Shuichi Akizuki, Manabu Hashimoto, Kazuyuki Nagata, and Kensuke Harada. A brief review of affordance in robotic manipulation research. *Advanced Robotics*, 31(19-20):1086–1101, 2017.

[80] Yuguang Yang. A deep reinforcement learning architecture for multi-stage optimal control. *arXiv preprint arXiv:1911.10684*, 2019.

[81] Hanna Yousef, Mehdi Boukallel, and Kaspar Althoefer. Tactile sensing for dexterous in-hand manipulation in robotics—a review. *Sensors and Actuators A: physical*, 167(2):171–187, 2011.

[82] Hanbo Zhang, Jian Tang, Shiguang Sun, and Xuguang Lan. Robotic grasping from classical to modern: A survey. *arXiv preprint arXiv:2202.03631*, 2022.

[83] Zhizhuo Zhang. Simulation of robotic arm grasping control based on proximal policy optimization algorithm. In *Journal of Physics: Conference Series*, volume 2203, page 012065. IOP Publishing, 2022.

[84] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.

[85] Christian Zimmermann, Duygu Ceylan, Jimei Yang, Bryan Russell, Max Argus, and Thomas Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 813–822, 2019.

[86] Theodor Zingg. *Beitrag zur schotteranalyse*. PhD thesis, ETH Zurich, 1935.