Developing a Mobile Manipulation System to Handle Unknown and Unstructured Objects

by

Abdulrahman Al-Shanoon

A thesis submitted to the
School of Graduate and Postdoctoral Studies in partial
fulfillment of the requirements for the degree of

Doctor of Philosophy

In

Mechanical Engineering

Department of Automotive and Mechatronics Engineering

Faculty of Engineering and Applied Science

University of Ontario Institute of Technology

(Ontario Tech University)

Oshawa, Ontario, Canada

April 2021

THESIS EXAMINATION INFORMATION

Submitted by **Abdulrahman Al-Shanoon**

Doctor of Philosophy in **Mechanical Engineering**

Thesis title: **Developing a Mobile Manipulation System to Handle Unknown and Unstructured Objects**

An oral defense of this thesis took place on April 26th, 2021 in front of the following examining committee:

Examining Committee:

Chair of Examining Committee	Dr. Martin Agelin-Chaab
Research Supervisor	Dr. Haoxiang Lang
Examining Committee Member	Dr. Ibrahim Dincer
Examining Committee Member	Dr. Xianke Lin
University Examiner	Dr. Shahryar Rahnamayan
External Examiner	Dr. George Zhu, York University

The above committee determined that the thesis is acceptable in form and content and that a satisfactory knowledge of the field covered by the thesis was demonstrated by the candidate during an oral examination. A signed copy of the Certificate of Approval is available from the School of Graduate and Postdoctoral Studies.

ABSTRACT

The exceptional human's ability to interact with unknown objects based on minimal prior experience is a permanent inspiration to the field of robotic manipulation. The recent revolution in industrial and service robots demands high-autonomy and intelligent mobile-manipulators. The goal of the thesis is to develop an autonomous mobile robotic manipulation system that can handle unknown and unstructured objects with the least training and human involvement.

First, an end-to-end vision-based mobile manipulation architecture with minimal training using synthetic datasets is proposed in this thesis. The system includes: 1) effective training strategy of a perception network for object pose estimation, 2) the result is utilized as sensing feedback to integrate into a visual servoing system to achieve autonomous mobile manipulation. Experimental findings from simulations and real-world settings showed the efficiency of using computer-generated datasets, that can be generalized to the physical mobile-manipulator task. The model of the presented robot is experimentally verified and discussed.

Second, a challenging robotic manipulation scenario of unknown-adjacent objects is addressed in this thesis by using a scalable self-supervised system that can learn grasping control strategies for unknown objects based on limited knowledge and simple sample objects. The developed learning scheme can be beneficial to both generalization and transferability without requiring any additional training or prior object awareness.

Finally, an end-to-end self-learning framework is proposed to learn manipulating policies for challenging scenarios based on minimal training time and raw experience. The proposed model learns from scratch, from visual observations to sequential decision-making, manipulating actions and generalizes to unknown scenarios. The agent comprehends a sequence of manipulations that purposely lead to successful grasps. Results of the experiments demonstrated the effectiveness of the learning between manipulating actions, in which the grasping success rate has dramatically increased. The proposed system is successfully experimented and validated in simulations and real-world settings.

Keywords: autonomous system; mobile manipulation; robotic-object interaction; deep learning; and visual servoing.

AUTHOR'S DECLARATION

Abdulrahman Al-Shanoon

STATEMENT OF CONTRIBUTIONS

Parts of this thesis have been published or submitted for publication in the following:

Al-Shanoon Abdulrahman, Haoxiang Lang, Ying Wang, Yunfei Zhang, and Wenxin Hong. (2021) "Learn to Grasp Unknown Objects in Robotic Manipulation.“ *Intelligent Service Robotics,* under reviewing.

Al-Shanoon Abdulrahman and Haoxiang Lang. (2021) "Learn to Grasp Unknown-Adjacent Objects in Robotic Manipulation.“ *Journal of Intelligent & Robotic Systems,* under reviewing.

Al-Shanoon Abdulrahman, Haoxiang Lang, and Yanjun Wang. (2020) "DeepNet-based 3D Visual Serving for Long-range Mobile Manipulation." *International Journal of Robotics and Automation,* under reviewing.

Al-Shanoon Abdulrahman, and Haoxiang Lang. (2020) "Robotic Manipulation based on 3-D Visual Servoing and Deep Neural Networks." *Robotics and Autonomous Systems,* under reviewing.

Al-Shanoon Abdulrahman, Haoxiang Lang, and Ying Wang. "Vision-Based Hand Gesture Recognition With Deep Machine Learning for Visual Servoing." In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 51814, p. V05BT07A050. American Society of Mechanical Engineers, 2018.

Al-Shanoon Abdulrahman, Aaron Hao Tan, Haoxiang Lang, and Ying Wang. "Mobile Robot Regulation with Position Based Visual Servoing." In *2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pp. 1-6. IEEE, 2018.

Hao Tan Aaron, Abdulrahman Al-Shanoon, Haoxiang Lang, and Moustafa El-Gindy. "Mobile Robot Regulation With Image Based Visual Servoing." In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 51807, p. V05AT07A078. American Society of Mechanical Engineers, 2018.

The entire work presented in this thesis was conducted at the GRASP Laboratory, Ontario Tech University, Oshawa, Ontario, Canada, under the supervision of Dr. Haoxiang Lang. I was responsible for plant modeling, control algorithm design and experimentation with the guidance and advice of my supervisor, Dr. Haoxiang Lang.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Dr. Haoxiang Lang for the opportunity and assistance to carry out this thesis. This was a very valuable experience and would not have happened without his guidance and support.

I would also like to thank the alumni members of the lab that helped me through this journey. A special acknowledgment goes to Aaron Tan, Matthew Levins, Eric McCormick, and Taylor Gao. Their technical support was appreciated on every occasion, I wish them all the best in their careers.

Finally, I would like to express my great appreciation for my family for their assistance which enabled me to focus solely on my Ph.D. journey. This would not have happened without their support.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND SYMBOLS

ROM	Robotic Object Manipulation
AMM	Autonomous Mobile Manipulator
ROI	Robot Object Interaction
DNN	Deep Neural Network
CNN	Convolutional Neural Network
REL	Robotic Experiential Learning
DOF	Degree of Freedom
VS	Visual Servoing
DRL	Deep Reinforcement Learning
DVS	Deep Visual Servoing
DRGP	Deep Reinforcement Grasp Policy
DMP	Deep Manipulation Policy
AGV	Automatic Guided Vehicles
PBVS	Pose Based Visual Servoing
IBVS	Image Based Visual Servoing
PnP	Perspective-n-Point
CPM	Convolutional Pose Machines
RE	Reprojection Error
DH	Denavit-Hartenberg
ROS	Robot Operating System
DL	Deep Learning
TD	Temporal Difference
DP	Dynamic Programming
DQN	Deep Q-Network
FCN	Fully Convolutional Neural Network
StG	Shift to Grasp

MDP	Markov Decision Process
DMP	Deep Manipulation Policy
DMP-OG	Deep Manipulation Policy Only Grasping
CS	Confidence Score
DMP-NSR	Deep Manipulation Policy No Shift Rewards
CAD	Computer Aided Design
SIFT	Scale-Invariant Feature Transform
SURF	Speeded Up Robust Features
ConvNet	Convolutional Neural Network

Chapter 1. Introduction

1.1 Overview

The present research has explicit evidence about the importance of developing future robots that require human-like reasoning and understanding. The exceptional ability of humans to interact with objects based on minimal prior experience is an endless inspiration to the field of autonomous robotic manipulation. The advanced robotic ability to manipulate objects is pivotal to a broad spectrum of applications in industrial and service robots.

The latest industrial revolution requests robotic systems to have high level of autonomy and understanding. The intention is to execute independent robotic manipulation activities with the least human involvement. Robotic Object Manipulation (ROM) could be defined as the advanced robotic capabilities that can handle target objects skillfully. This robotic ability to manipulate objects is essential to many applications: from packaging in a manufacturing facility to assist the elderly at a homecare center; from transporting food cans at household to debris disposal after a disaster or arranging objects in grocery stores.

A mobile manipulation robot is a robotic system that includes a manipulator and a wheeled mobile platform. Those types of robotic systems combine the advantages of mobile platforms and robotic manipulators. For instance, the mobile platform extends the workspace of the manipulator, whereas an arm offers several operational functionalities.

In many industries, the use of Autonomous Mobile Manipulator (AMM) has increased, allowing these systems to transport, coordinate and process different assets independently. The logistic facilities and courier services are the two main industries that profit from this technology. To evolve the performance in autonomy and versatility, applied robots make use of several sensing technologies and robot motion control algorithms. There are typically two key steps required in order to execute autonomous mobile manipulation tasks. First, the perception step which is used to estimate target objects and perceive surroundings. This step is usually based on sensor-fusion network. Second, the robot motion-control-technique step which receives the inputs from a prior perception step and generates the required movements by controllers.

An alternative approach has been suggested to only utilize vision sensors to perceive target objects and understand the environments. Deep Neural Networks (DNNs), specifically Convolutional Neural Network (CNN), has become a promising method for handling object recognition issues. Traditional Deep-Learning (DL) methods have been successfully applied to 2D object detection problems. In addition, several researchers have recently been utilizing traditional deep learning to achieve 3D object detection and pose estimation. However, unlike 2D object detection, labeling 3D object is difficult and required experts. Using synthetic data for deep neural network training has solved this issue by proposing an unlimited amount of useful pre-labeled training dataset that is produced safely in a reasonable effort.

In order to achieve a completed robot-object-interaction task, such as autonomous grasping objects. DNN algorithms were applied in prior studies to handle grasping objects remarkably. Preparing datasets for training, however, could be laborious. Meanwhile, training CNN is computationally expensive and time consuming. In addition, empiric testing needs to be limited to familiar objects used during training which have similar colors and shape attributes.

Recent approaches have been documented to address the object-interaction task based on learning concepts. Robotic-Experiential-Learning (REL) techniques have been promisingly devoted to plan grasping through Deep Reinforcement Learning (DRL). Applied robots are able to learn gradually grasping tasks through trial-and-error patterns. Learning skills, however, aim to be extended to diverse situations without re-training or amendment requirements. Contemporary researchers have fairly examined the generalizable performance and succeeded in expanding the gained skills to grasp new objects. However, such approaches still demand a long-time of training on a vast dataset contains hundreds of different objects and requires massive amounts of grasps trials. In addition, learning-based methods have lacked to cover scenarios that should be close to real-life applications. In which applied robots could be learning the task from challenging situations and operating on novel objects, since it is almost impossible to train the robot on a specific object or certain scenarios.

1.2 Scope of the Research

To accomplish a skilled robotic manipulation task in the real-life environment. The robot should anticipate operating in unstructured surroundings and handling a wide range of unfamiliar objects in challenging situations. The crucial task is to physically interact with target objects and perform the required manipulations that are necessary to succeed in the task. This non-trivial task calls human-like perception and reasoning. It is noteworthy to mention that the only perceptual modality utilized in this thesis is visual observation. Moreover, the task should be learned using single robotic station, minimal training time and limited simulation datasets.

The main challenge for AMM-target-object interaction is to visually estimate the pose of the target object in a 3D space and combine it autonomously into a vision-based control scheme in mobile manipulation application. Several studies employed fiducial markers as points of reference for the pose of the target object in the workspace. Other works applied traditional computer-vision methods to understand the pose of the objects. For instance, model-based and feature-based methods were used for Robot Object Interaction (ROI) tasks. However, in addition to the empirical issues that lead to non-practical results, many drawbacks accompany those techniques, such as limited to structured background, poor performance with lighting and occlusion variations and required highly textured objects.

Another challenging task, in the field of autonomous robotic manipulation, is grasping an unknown object in real-life environment based on limited knowledge. The applied robot should be able to execute successful grasping task on unfamiliar objects which is equivalent to human level. Grasping unfamiliar objects (unknown during training) based on limited prior experience is a daunting task in robotic manipulation applications. Classical solutions usually require predefined information about target objects (such as object pose estimation, 3D CAD model, or object classifications). Various recent studies have investigated the grasping issue but still require task-specific training dataset, which makes it limited to extend the acquired knowledge and generalize to novel situations.

Further manipulating and reasoning abilities are also required to accomplish complicated robotic manipulation tasks, the used robot should be capable of manipulating unknown objects before performing grasping. For instance, arduous grasping scenario includes a clutter of objects that are located closely next to each other. In this case, those objects necessarily need to be moved apart to be more graspable. It is considerably harder to grasp unknown objects in such situation than grasping secluded objects. In order to solve this problem, recent solutions usually require pre-grasping (non-prehensible) intervention (e.g., pushing, toppling, squeezing or rolling) before performing grasping. Pre-grasping action assists to re-arrange the cluttered objects without executing evident grasping. However, such solutions play loose role and causing delays. The pre-grasping action should have intended utility and effect on the consecutive grasping action, because it is a sequential decision-making problem. The assistance of pre-grasping actions was studied separately and not combined as a consecutive issue with the potential post-action which is grasping. Sequential manipulation (pre-grasping and grasping action) based on limited knowledge is still an unexplored issue.

Even after the concern and the practical value of the research community, competent handling of novel objects in unstructured environments remains a broadly open challenge in robotic manipulation tasks. The outcome leads to the question about endowing the generalization: is a robot able to extend the gained knowledge of manipulation (from minimal experience) on to novel situations and operate on unknown objects? In this work, we mainly focus on the following key points that address the challenges and gaps existing in the field of autonomous robotic manipulation.

Firstly, an AMM system was presented, which is able to operate based on the deep-CNN model using a single camera. The proposed system includes perception network which estimates the full 6 Degree-of-Freedom (DOF) poses (position and orientation) of target objects. In addition, a vision-based control scheme was designed and developed to achieve the full movements of the long-range mobile manipulation. The continuous 3D detection and estimation (of target objects obtained by the perception network) were constantly sent to the autonomous vision-based robot control system to execute in the 3D space. The perception network was entirely trained on the computer-generated single RGB images. The proposed AMM system was, then, capable of effectively operating in the real-world environment. The kinematic model of the presented mobile manipulator robot was experimentally verified and discussed. The whole system was validated in the simulations and real-world experiments without the need to extra fine-tuning in the implementations. Moreover, the efficient perception network does not require post-refinements of the estimated target objects. Perception network was tested on several target objects in different real-life environments handling difficult backgrounds, various light conditions, and occlusion events. Empirical findings demonstrated a reliable vision-based AMM system that operates using a single camera.

Secondly, a scalable learning-based robotic agent was proposed to grasp unknown objects. The introduced framework was effective for an end-to-end learning concept based on self-supervised using raw experience. The agent utilizes visual observations to make decisions without requiring prior object-awareness nor model-based environment. It is worth mentioning that the agent learns with minimal training time in simulations using simple 3D objects. The acquired learning-based familiarity was then implemented in real-life settings without requiring extra re-training data and fine-tuning. The agent could successfully adapt the learning ability and validate the proof-of-concept by considering the physical experimentation purposes. The simulations and real-world experiments have demonstrated a robust visual robotic system for grasping unknown household objects (excluded from training sessions) based on simulated knowledge of ordinary objects and little experience.

Thirdly, we proposed a data-driven self-learning system for robotic manipulation. The proposed system was able to manipulate unknown objects in challenging situations based on minimal experience. The system works without requiring predefined object information and task-specific re-training datasets. The model was trained based on the trial-and-error manner and learned progressively manipulation actions. It should be noted that the agent learns end-to-end manipulation policies, from only visual observations and convert them to a sequential decision-making. Learning process was entirely carried out in simulation environment using simulated datasets. The obtained manipulation skills were, then, transferred effectively to the real-life generalization scenarios. It is interesting to note that the agent was able to learn combinations of manipulation sequential plans from simulated training phase. The limited training information from computer-environment was sufficient to be physically implemented on real robot. The proposed system was successfully verified and tested in simulations as well as real-world settings with taking into account generalization purposes.

1.3 Motivation

In order to achieve an advanced robotic manipulation application, there are three different features should be considered in this task. Figure 1.1 shows a chart composed of the main aspects required for skilled robotic manipulation applications. First, versatility that is known as the adaptability level of a robot to respond with different situations, instead of being limited to certain conditions. Autonomy is the second feature that defines robotic independency. For instance, a fully autonomous robot is able to execute a task without human interventions. The third feature is the dependability that is introduced into reliability and safety, including the robotic ability to perform effective tasks under modeling errors or inaccurate sensor data. The challenge point in an autonomous robotic manipulation application is how to successfully combine the previous features in a completed task.

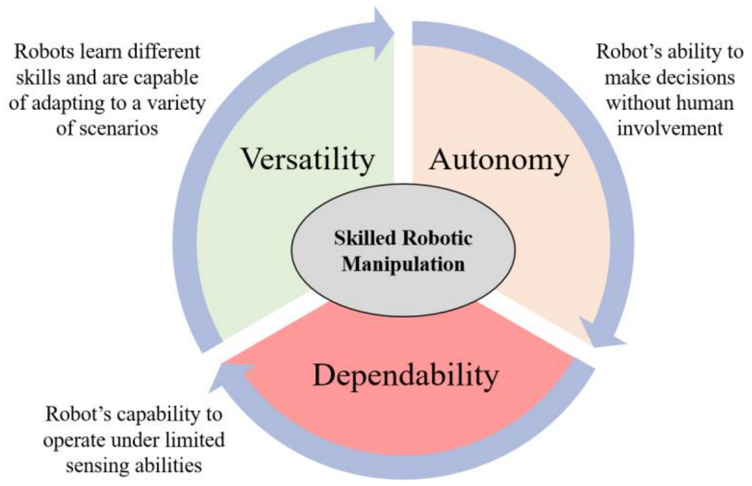

Robots learn different skills and are capable of adapting to a variety of scenarios
Robot's ability to make decisions without human involvement
Versatility
Autonomy
Skilled Robotic Manipulation
Dependability
Robot's capability to operate under limited sensing abilities

Figure 1.1: Main aspects chart for skilled robotic manipulation application.

Recently, the published studies on robot-object interaction system are mostly around navigating the robot end-effector to the pre-defined target object in a structured workspace. In addition, achieving robotic manipulation tasks by using remote control or wearable sensors and encoded the desired position as a target pose. However, perceiving the surrounding objects, estimating 6D pose parameters, and ultimately performing robotic interactive task without human interventions have paid less attention in literature.

Robotic manipulation studies the learning-based interaction between a robotic agent and target objects. There are two main challenges. First, detecting and understanding the unfamiliar objects presented in the real-life environment. Second, the ability of the robot to interact with unknown objects in challenging real-life situations. In addition, the critical destination is to combine the sub-tasks and accomplish a reasonable and effective response for the robotic-object interactive application. The challenges addressed in this research are as follows:

- Environment perception and understanding (e.g., vision-based object identification and pose estimation).
- Motion control based on visual perception.
- Robot-object interaction for unknown objects in challenging scenarios.
- Minimal training time, computer-generated datasets, and simulation experience.

Service robot is the robot that is able to execute beneficial services for human. Thus, the significance of developing service robots is growing rapidly towards fully autonomous tasks. Service robot could be divided into two main groups namely: professional service robots, used for performing professional service tasks, and personal service robots, used for education, and homecare companion robots. The latter one intends to perform assistance for elderly and impaired people in daily life activities.

The necessity of developing such robots is due to the highly increasing elderly population across the world. Statistics have shown and reported globally noticeable elevated percentages of people who 65+ aged. Figure 1.2 shows the percentage of population for people who are more than 65 years old, this is in five different regions in the world. In 2050, the percentage in North America will be more than 20%, which requires further health care allocations. Figure 1.3 illustrates the health care cost in 2013 in Canada only. An average of $6,000 was spent on a person aged 65 to 69. The cost increases to more than $11,000 for someone aged 75 to 79 and then jumps to $24,000 for someone aged 85 to 89. Therefore, assistive robots that are capable of carrying out seamless interaction activities and comprehending the home's surroundings have a huge demand, in order to provide potential assistance to elderly people in their daily life.

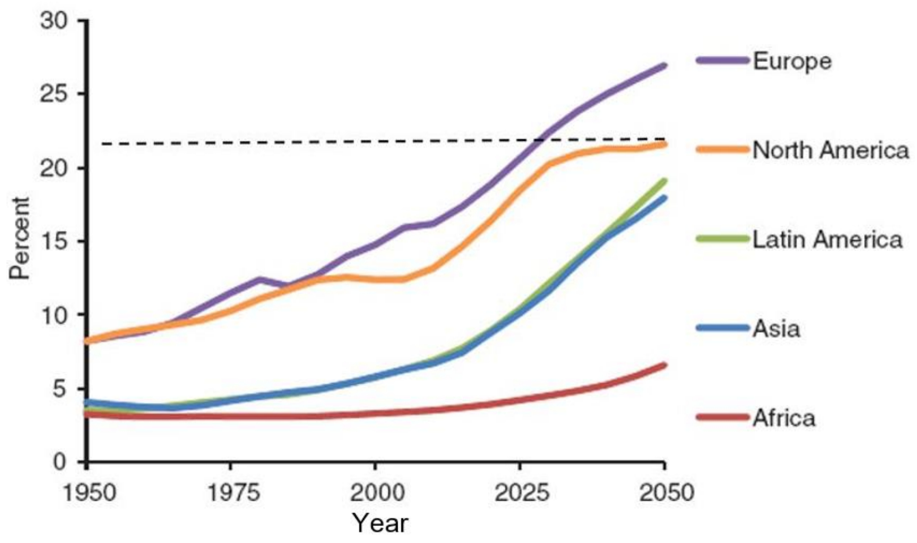


Figure 1.2: Percentage of population aged 65+ in five different regions in the world[1].

[1] Source: National Academies Press Washington, D.C. (2011)

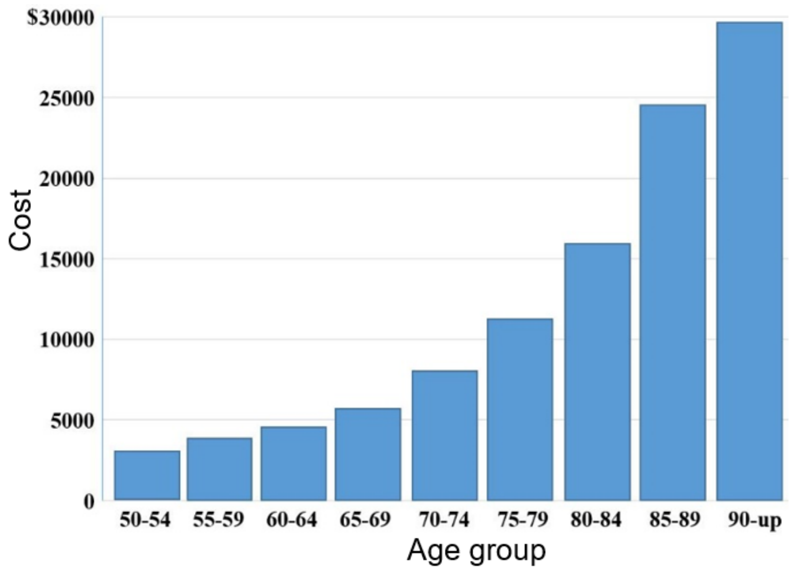


Figure 1.3: Healthcare expenditure per capita in 2013 in Canada[2].

The challenge of understanding the environment and interacting with objects in different situations is because of the perception limitations and short of robust robotic motion strategy. The robot is restricted with limited motions depending on the pre-defined desired pose and structured environment. Therefore, there is need for research and application to make the robot capable of interacting with target objects in an unpredictable environment and in various situations.

In this research, an intelligent physical interaction framework was proposed between robots and target objects and its modelling of vision-based robot motion control. Object identification and pose estimation of the target objects should be processed as a natural and smart reference in any objects in household environment. In addition, the thesis focuses on translating the required robotic manipulation with target objects for robot-object interaction and learning scheme. Our work presented a promising framework and algorithms, including the challenges of achieving learning-based seamless interactions in real-life.

[2] Source: Canadian Institute for Health Information (CIHI)

1.4 Robotic Manipulation (current state-of-the-art)

Robust robotic manipulation of unknown objects in cluttered scenario is unsolved problem because of the difficulties of prerequisites to interact with the real-life tasks and the required ability of human-like decision making. Despite the academic values and the practical novelty of the previous and current studies, research gaps and weaknesses were found in the literature. The followings are the main limitations of the existing research that the thesis will be trying to address.

- Complex Sensor-Fusion methods.

  Studies utilized multi-sensor approaches to cover wider perceptual modalities, which may provide usable feedback information. However, sensory fusion network results in complicated and expensive robotic system.

- Synthetic and real-world dataset collections.

  Combinations of synthetic and real-world datasets are typically utilized to train robotic systems. This method might reduce the efforts of preparing enough datasets. However, it creates practical issues called reality-gap, where systems do not hold in real-world settings and require extra fine-tuning. In addition, systems are limited to familiar target objects and pre-modeled environments.

- Limited to highly textured objects and structured environments.

  Model-based and feature-based methods seek to learn the features and pre-given models of target objects (such as 3D CAD models or objects categories). Robots usually struggle to perform better when they target less textured or mobile objects in a cluttered scene.

- Task-specific training or predefined information of target objects.

  Recent studies perform well on the known objects during the training phase and similar execution environments. However, adapting to different situations requires extra training efforts, which is impractical to train for numerous numbers of objects (for instance, household objects).

- Massive datasets and long training time.

  Preparing datasets for training sessions is a significant matter in the learning phase. Modern technologies overwhelmingly demand huge assortments of hundreds of different objects which consumes long-training time.

- Multiple robots or human data-collection.

  Robotic experiential approaches collect data from interactions that occur between robot and target objects. Several recent systems require multiple robots to cover a large scale of collecting data. In addition, manually collected training data could be required through human-based demonstrations.

- Performance on simple objects (such as cubes).

  Intelligent robots should be able to learn the task skillfully and be qualified to handle unknown comparable task. Generalization the acquired knowledge to different execution environment is a crucial matter in the field of skilled robotic manipulation.

1.5 Objectives

The main objective of this thesis is to develop a vision-based mobile manipulation system that can handle unknown and unstructured objects. The detailed objectives of this research are as follows:

- To design and train a perception network model that constantly detects and estimates the 6DOF pose of the target object in real-world environments, based on only synthetic single images.
- Examine the performance of the simulated-trained perception network in real-life environment during various situations included difficult backgrounds, illumination changings, and occlusion events.

- To model and develop a 3D continuous visual servoing system implemented on a long-range mobile manipulator robot.
- Integration of the estimated target object pose (from the perception network) autonomously with the 3D visual servoing control scheme.
- Design and train a robotic agent in simulation environments to learn grasping actions with minimal training time.
- To extend the agent's ability and generalize to real-life scenarios targeting unknown situations without the need for both (a) task-specific training data, and (b) prior object awareness.
- Developing and training a joint-self-supervised robotic agent for learning consecutive manipulation strategies.
- Generalization and transferring the limited acquired knowledge of the sequential decision-making policies in to real-life unknown scenarios.
- Proving that manipulation strategies can be jointly supervised by future action policies (which are self-supervised), both of which are simultaneously trained.
- To validate the proposed systems by testing the performance in simulations as well as physical experimentations.

1.6 Contributions

The main contributions of the thesis can be summarized below:

- Introduce an end-to-end visual mobile manipulation architecture with minimal training and synthetic datasets.
- Propose a self-supervised framework to learn grasping policies from simulated limited knowledge and simple objects.
- Propose a scalable self-learning system to learn manipulating control strategies for unknown objects in challenging scenarios based on minimal training time and raw experience.

- Introduce an end-to-end self-learning technology to learn from scratch, from visual observations to sequential decision-making, manipulating actions and generalize to unknown scenarios.

1.7 Thesis Outlines

- Chapter 1 introduces the research background, main interests, and current state-of-the-art in robotic manipulation. Meanwhile, the existing challenges and gaps are reviewed and discussed which become the motivations for this work. In addition, thesis objectives, motivations and contributions are covered and introduced.
- Chapter 2 demonstrates the detailed literature review of the prior and most recent research studies related to robot manipulation theory and real-world implementations. Meanwhile, the current research gaps have been determined and formed the main focus of the proposed thesis.
- Chapter 3 develops a vision-based algorithm for object detection and pose estimation in robotic manipulation applications. A novel framework of integration between the perception network and vision-based robot motion control scheme will be presented. Simulation and physical experimentations show that the synthetic-trained deep net model is able to be generalized to real-life environments and implemented in AMM system using a single camera. The system is named Deep Visual Servoing (DVS).
- Chapter 4 introduces a novel self-learning framework for grasping control policies based on limited information, using DRL. It is named Deep Reinforcement Grasp Policy (DRGP). The agent quickly learns grasping policies based on Q-learning concept by trial-and-error manner using depth sensor camera. It is noteworthy that the extension of the robot's skills was applied on different situations to fulfill generalization tests.
- Chapter 5 proposes a self-supervised learning framework for manipulation control strategies, named as Deep Manipulation Policy (DMP). A data-driven manipulation of pre-grasping and grasping as sequences of acts is introduced and implemented

on challenging scenarios. In this chapter, the system shows the effectiveness between the manipulation actions (pre-grasping and grasp policies). The success rate of grasping has greatly increased by the assistance of pre-grasping actions. In addition, the robot's ability to cover generalization purposes was also taken into account.

- Chapter 6 finally concludes the research thesis findings, limitations, and suggests recommendations and future work.

1.8 Summary

This chapter introduced the key background, current state-of-the-art and main interests of the proposed thesis. Future applications and the scope and motivations of the thesis are presented along with the research’s objectives. Finally, the major thesis contributions and research outlines are also demonstrated.

Chapter 2. Literature Review

2.1 Background

The related prior works presented in this chapter highlight the current gaps and challenges in the field of autonomous robotic manipulation. This chapter reviews the previous and recent studies of object detection and object pose estimation, and studies that developed for vision-based motion control in robotics. It reviews modern technologies that addressed the vision-based interaction between the robot and target object in order to achieve efficient manipulation applications. Furthermore, this chapter thoroughly reviews the methods and algorithms that examined learning-based strategies and training policies for robotic manipulation.

There are three essential challenges in the generic architecture of mobile manipulation systems. First, object identification and understanding as target reference in the real-world environment. The robot should utilize the provided perceptual modality (employed sensory network) to obtain sufficient localization information about the environment and objects.

Motion control of the robot is the second challenge to perform an interactive task between the used robot and the surroundings. It is a non-trivial task for the robot to reach a target object with respect to the correct position and orientation. Challenges such as inaccurate kinematics modelling and sensors feedback response in real-life mostly generate uncertainties will easily lead to failure of the manipulation task.

Finally, the necessary mechanism to physically interact with the object is the third challenge. There should be reasoning behind the interaction policies required from the robotic agent to target objects. Execution robotic manipulations in strategic manner will improve the learning progress and assist to achieve potential advanced robot-object interaction applications.

2.2 Review of Vision-Based Object Recognition

Traditional object recognition methods utilized different strategies to address the challenge of object detection in the real-life environments. Object detection methods concentrate on searching for specific features that could indicate particular classes. In image processing, the system deals with some interest points and their descriptors. It is important to know the definition of detector of point of interest and its descriptor. An interest point (key point/salient point) detector is an approach that selects particular points from the image based on certain criteria [1]. Generally, the interest point is the extreme of a function for instance, corner metric. A descriptor is a vector of values which can describe the patch which is located around the interest point. For example, histogram of the gradient orientation. The interest point and the corresponding descriptor are called local features. Local features could be used for object identification and tracking in computer vision. The common classic algorithms of object recognition are characterized based on the following fundamental strategies.

Appearance-based method. Several efforts have focused on this technique as a developed feature descriptors and pattern recognition approaches. The appearance of particular object is required to be used in the appearance-based models. The image of the object is usually captured by different 2D views of the object-of-interest. This strategy has been commonly used in face recognition and handwriting recognition. The method requires a set of correlated images which could be utilized as training references. Eigenspace approach is used at this process in order to reduce the dimension subspace and compress the dataset. Input images are displayed on the Eigenspace and the matching and correspondence process is occurred. The best two algorithms that work based on appearance-based approach are Principle Component Analysis PCA and Linear Discriminant Analysis LDA [2].

Feature-based method. The algorithms used in this approach focus on specific features present in the image. The features presented are expected to belong to a specific object; in most cases, one object may be represented by multiple attributes. These suggested features could be colors, contour lines, geometric forms or edges, and gradient of pixel intensity. The principle of feature-based method is to search and capture certain features in each input image and obtain features that should be compared and matched to reference dataset of models. The features and their descriptors could be obtained by considering the whole image, which is known as global features. Another approach is to perform by considering a defined part of the image, which is called local features. The feature vector that could describe the patch located around the interest point is denoted as a local descriptor. Each patch should be described by local descriptor and then carried out for geometrical transformation. Three stages should be conducted for building object recognition system, which are combinations of local interest detector, its descriptor, and geometrical verification. The invariant local interest concentrates on a certain area which is called a region of interest/key point. There are two types of detectors for region of interest (interest point), corner-based detector and blob-based detector. The following approaches are the famous algorithms for region of interest based on corner detectors and blob detectors [3].

- Moravec detector
- Harris detector and Shi and Tomasi detector
- HarrisLaplace detector
- HarrisAffine detector
- SUSAN and FAST detectors
- BRISK and ORB detectors
- Harris or Hessian point based detectors
- Difference of Gaussian Points (DoG) detector
- Harris- or Hessian affine invariant region detectors (Harris-Affine)
- Maximally Stable Extremal Regions (MSER)
- Entropy Based Salient Region detector (EBSR)
- Intensity Based Regions and Edge Based Regions (IBR, EBR)

The histogram of the pixel intensity could be stated as an example for global feature. This process is not always robust for considering the whole image as it is computationally expensive and requires long processing time. In addition, the changes in illumination, occlusion or rotation will dramatically generate different results; the effective recognition process is unreliable anymore. In order to avoid these issues, descriptors for local features need to be robust. Therefore, the local feature algorithms are more preferable than global algorithms. Scale-Invariant Feature Transform (SIFT) [4] and Speeded Up Robust Features (SURF) [5] are the two common algorithms that work based on local feature-based methods. They were reported good performance with treating images in different scales and orientations. These algorithms assist to recognize the pre-defined object based on determining feature points.

However, in the empirical implementation of SIFT and SURF algorithms, several drawbacks have been encountered for recognizing the target objects. First, highly computational level is required for processing the input image, which leads to poor performance in real time object recognition and tracking. Second, since the algorithm requires to extract local features, the algorithm detects the features only with highly textured objects. Consequently, the computer vision duty has restrictions and limitations to objects with textured, otherwise it loses the tracking process. Third, the algorithms could relatively perform well with immobile objects. However, it confuses and fails with objects that changes their shapes such as human hand in hand-gesture applications. In addition, the stability of recognition process has shown poor real-time performance, including failure detection for multi-directional object. Moreover, the slight light reflections and changing the illumination will passively affect the task of feature-points detection.

Interpretation tree method. Another object recognition strategy which is called interpretation tree is used for model matching. The concept of this approach is to provide a methodology to determine various aspects and indicate n-dimensional geometric objects. The model database of known features is required. The features could contain distance, angle, and direction among points that located on the surface of the object. The procedure of this method starts with considering certain features of the model, then, compare them with the whole features of the proposed object. Number of iterations could be occurred until the combinatorial points are all processed [6].

Artificial neural network. The neural network commonly used for object recognition is called convolutional neural networks (CNNs). The architecture of the input image with certain number of pixels are convoluted with a filter in order to have a 3D feature map. The pooling process decreases the amount of data until one-dimensional vector is finally obtained. For object recognition, a CNN should be trained to adapt all the weights of the neurons. Different levels of features are obtained during the training phase. The color, lines and contrast are found in low-level features. Edges and corners are existed in the middle level features. Whereas, the high-level feature contains class specific forms and sections [7, 8]. The research in CNN has rapidly grown and evolved the object recognition mission in computer-vision [9]. In 2012 [10], a novel generation of CNN has introduced by Alex Krizhevsky in university of Toronto. After that, CNNs have been considered as the most promising approach for object recognition. Several studies have confirmed that CNN outperforms SIFT and SURF on descriptors matching [11, 12]. The study has compared the key-points features from different convolutional neural layers to the descriptors from SIFT algorithm.

Ever since, CNNs have become very interesting research field for image classification and object identification processes. CNN is the leading approach for high performance classification and object recognition systems. Several networks have been evolved from one to another implementation which can be deployed on autonomous systems. R-CNN [13], Fast R-CNN [14], Faster R-CNN [15], YOLO [16, 17], Mask R-CNN [18], and SSD [19] are well-known object recognition frameworks that have been built based on CNNs. The studies have provided noticeable well-developed performance comparing with the traditional object recognition concepts that depend on local feature descriptors [20-22].

2.3 Object Pose Estimation and Mobile Manipulator Robot

This section reviews the previous and existing studies conducted on object detection and 3D object pose estimation from classical techniques to trainable machine-learning methods. Meanwhile, vision-based robot motion control methods are also reviewed.

2.3.1 Object Pose Estimation for Visual Servoing

Conventional approaches utilized RGB images [23-25] with local interest-points and feature matching algorithms in order to achieve object recognition and pose estimation task [26]. The methods required certain local descriptor for different scale, rotation, and viewpoints. The drawbacks of such approaches are the requirement of highly textured objects and good lighting condition. It showed low performance in the conditions of varying lighting conditions and occlusion events. Employing RGB-D images [27] is another approach to achieve object pose estimation by introducing depth information [28-31]. This approach has been applied to indoor robotic tasks for grasping applications. However, complex structure and limited background were the issues that hinder the performance.

In recent years, researchers have considered CNN-based methods [32] as the most promising approach for object recognition [15, 33-35] and pose estimation [36] in different situations [37-40]. PoseCNN [41] suggests regressing 3D pose directly from the image by applying several CNN stages. Other studies documented decent performance and focused on challenges such as occlusions and various light conditions. However, the key matter of training data is how to generate an effective dataset that presents enough variations, where the deep net learns from variety of lighting and poses conditions. Labelling bounding boxes for 2D object detection is simple and effortless to annotate. However, 3D object detection requires high skills that makes it impossible to generate labeled training data manually. Some existing labelling tools such as LabelFusion [42] provides useful functions for labelling 3D objects. However, to our knowledge, there is no simple and effective tool which can help to generate real-training-data for 6 DOF object pose estimation that covers enough variations in terms of object poses and lighting conditions. Due to this difficulty, such methods consider real data for training purposes which are highly similar to the test data (e.g. same camera-parameters, object category, and similar lighting conditions). As a result, test data is always limited to any significant difference from the training data.

J. Tobin, R. Fong, et al. in [43] solved this issue by generating the data synthetically which can produce almost unlimited pre-labeled 3D training-data with little effort. A challenge of the current approach to synthetic data is called reality-gap. When the system trains on synthetic data, real data performance would always require extra fine-tuning. J. Tremblay, et al. in [44] conducted a recent solution to this issue, where synthetic dataset is randomized in non-realistic methods, in which the test data should seem merely another variance. The solution is called domain randomization which is a combination of non-photorealistic and photorealistic data. The domain randomization of synthetic data yields enough diversity for training dataset that covers variations in poses, lighting, and occlusion conditions. Thus, deep net is capable to perform well on real data with different light conditions without requiring extra fine-tuning. This combination is the state-of-the-art provided by FAT dataset [44] developed by Nvidia research-group and generated based on the YCB household objects [45]. The datasets are generated by the developed tool used for UnrealEngine4 so-called NDDS. The domain randomization method was validated for detecting objects in real-word environment [46]. Deep-learning methods require a vast number of datasets for training. Therefore, synthetic datasets were provided for training the deep network [47, 48]. In Chapter 3 of this thesis, our perception network is trained on only synthetically generated dataset (single RGB images) that was proved in [44] as the state-of-the-art. Which also covers sufficient possibilities of various poses in different environments, for instance, difficult backgrounds, extreme light conditions, and occlusions.

2.3.2 Visual Servoing Control System

Visual Servoing (VS) technique combines computer vision information and control law to achieve motion control of a robotic system. The practical concept of VS is to address how the robot can move based on visual data. In 1970, when the image processing ability was limited, the first visual servoing task has been proposed. Researchers have conducted many studies until a significant mathematical approach has introduced, the interaction matrix, which also known as image Jacobean. The relationship between the camera and the motion of the extracted image-features represented in interaction matrix equation. In vision-based control system, the measurement of the target is happening continuously by utilizing the vision feedback signal that controls the movement of the robot until the visual error become zero. The underlying idea of this research is how to control the pose of the robot with respect to the detected object by using visual features that obtained from the captured image [49, 50].

The formulation of visual servoing is divided into two major schemes, Image Based Visual Servoing (IBVS) and Position Based Visual Servoing (PBVS), which they determine the controlling model based on the vision features extraction. The essential working mechanism of VS is to eliminate the position error between the current position (*S (m(t), a*) and the desired position (S^d). This error could be minimized by controlling the motion of the robot's joints. The VS task representation is illustrated in equation 2.1. The control strategy of two fundamental visual servoing control methods are almost same with the difference of how the vision information is processed in the feedback loop of the control system. Figure 2.1 demonstrates the general block diagram of visual servoing.

$$e(t) = S\,(m(t), a) - S^d \quad \epsilon\, R^k \qquad 2.1$$

where (*m(t), a*) stands for the image measurement vector and intrinsic camera parameters, and ***k*** is the number of image features.

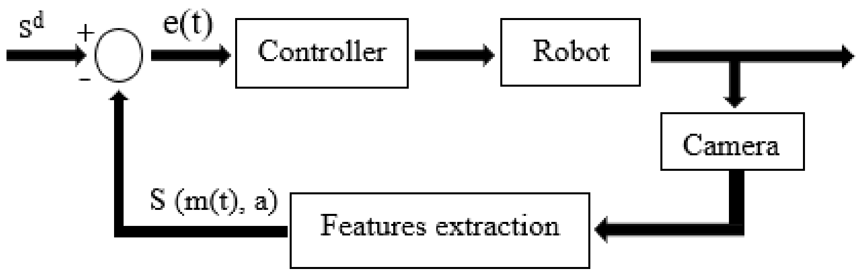


Figure 2.1: General visual servoing block diagram.

In the Figure, *S (m(t), a)* is the current object's measurement on the image plane or on 3D Cartesian space from target object pose estimation. Let assume ***S*** is the feature point in the image plane $\begin{bmatrix} u \\ v \end{bmatrix}$, the derivative **S'** has a velocity as $\begin{bmatrix} u^* \\ v^* \end{bmatrix}$ in the term of camera velocity ξ which contains linear and angular components $\begin{bmatrix} v \\ w \end{bmatrix}$. The relation between the velocities of the image feature and camera frame is defined by applying (***L***) interaction matrix, which relates the image features velocity to the camera velocity [51]. The following equation 2.2 shows the relationship between the feature points velocity and camera velocity relative to the camera frame.

$$\begin{bmatrix} u^* \\ v^* \end{bmatrix} = L\, \xi_c^c \qquad 2.2$$

where ξ_{c}^{c} represents the camera velocity with respect to the camera coordinates frame. $L \in R^{k\times m}$, (m) denotes as the number of DOF.

Position based visual servoing (PBVS) scheme [52] acquires the interest points of the target object from one/more cameras. The mathematical estimation of the desired camera pose is derived from the 3D pose of the target object. The control scheme requires 3D object model in Cartesian space, including the intrinsic and extrinsic camera calibrations. The main challenge in PBVS is the 3D pose estimation of the target object. The object's pose can be estimated based on predefined points or lines [53, 54]. The visual servoing system needs three coordinate frames which are the current frame f_c, desired frame f_d, and the reference frame f_r. The f_r represents the frame of the target object. The 3D Cartesian control duty is to compute the desired pose of the robot that eliminates the error measurement comparing with the current pose. The following equation 2.3 explains the PBVS parameters:

$$S = (T, \Theta U) \tag{2.3}$$

Where ***T*** stands for translational vector and ΘU denotes as a rotational vector. Equation 2.4 shows the error dynamic task.

$$e = (T_{o}^{c} - T_{o}^{c\,*}, \Theta U) \tag{2.4}$$

S expresses as $S = (T_{o}^{c}, \Theta U)$ as a current camera pose $\boldsymbol{F^{c}}$ and could be $S = (T_{o}^{c\,*}, \Theta U)$ as a desired camera pose $\boldsymbol{F^{c*}}$. The interaction matrix is given in equation 2.5.

$$L = \begin{bmatrix} -I_3 & \left[T_{o}^{c}\right]x \\ 0 & L\Theta u \end{bmatrix} \tag{2.5}$$

Where $\boldsymbol{I_3}$ stands for identity matrix, $\left[T_{o}^{c}\right]x$ is the skew matrix that related to the translation T_{o}^{c}. $L\Theta u$ is expressed in equation 2.6, where $sinc(\Theta) = sin(\Theta)/\Theta$.

$$L\Theta u = \boldsymbol{I_3} - \frac{\Theta}{2}[u]_x + \left(1 - \frac{sinc(\Theta)}{sinc^2\left(\frac{\Theta}{2}\right)}\right)[u]^2x \tag{2.6}$$

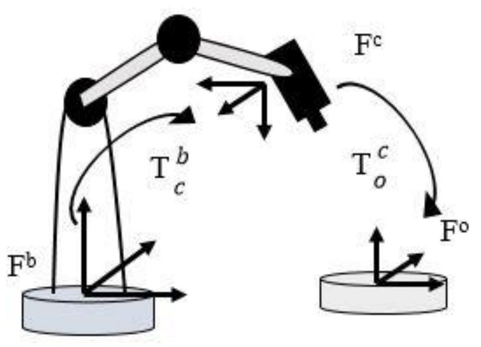
In image based visual servoing IBVS**,** pose estimation of the target object is not required, the control design uses the image features directly on the image plane. This approach controls the end-effector of the robot in the image space. Whereas, the control scheme in PBVD happens in Cartesian space. Therefore, IBVS is computationally less since the robot control tracks the feature points inside the image plane. The interaction matrix in IBVS implies the correlation between the camera's velocity in Cartesian space to the velocity of the interest points in image plane. Interaction matrix construction depends on the chosen feature points. In IBVS, depth estimation could be provided by considering particular interaction matrix that uses geometrical parameters for instance lines [55]. Laser, Ultrasonic and other range sensors are very accurate for estimating the depth directly [56]. In addition, stereo sensors use triangulation approaches for depth estimation [57]. The interaction matrix of typical feature point with classic camera projection is designed in equation 2.7.

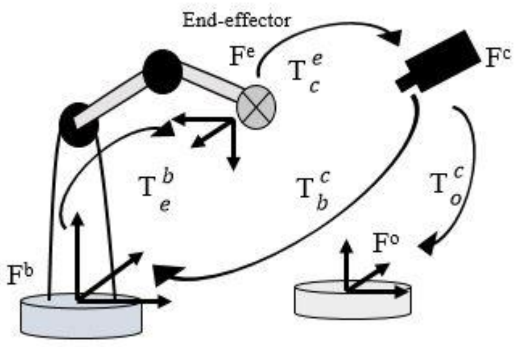
$$L = \begin{bmatrix} -\frac{f}{Z} & 0 & \frac{u}{Z} & \frac{uv}{f} & -(f^2+u^2)/f & v \\ 0 & -\frac{f}{Z} & \frac{v}{Z} & (f^2+v^2)/f & -\frac{uv}{f} & -u \end{bmatrix} \quad 2.7$$

where ***f*** and ***Z*** denote to the camera focal length and depth in Cartesian space relative to the camera frame, respectively. *u* and *v* are the 2D image point in image plane. The typical equation 2.8 illustrates the interaction matrix that connects the velocity of the feature point with camera frame velocity.

$$S' = L\,(u, v, z)\,\xi \quad 2.8$$

In the term of camera installation, there are two major methods to configure the camera place. The camera could be firmly attached at the end-effector which is known as eye-in-hand configuration (as shown in Figure 2.2 (a)). The second camera configuration is the eye-to-hand installation (as shown in Figure 2.2 (b)), the camera is mounted in the workspace and can observe the target object and the robot.





(a) eye-in-hand (b) eye-to-hand

Figure 2.2: Camera configurations.

Contemporary visual servoing studies underline the significance of considering the photometric details of the whole input image into VS system. This introduces such methods called direct/ photometric VS that were proposed in [58, 59]. However, a range of constraints on such approaches that preclude robust efficiency and other potential improvements. The direct VS methods are strongly affected by the image perturbations (such as pixel intensities, or occlusions) which cause instabilities. The bottleneck could be the different resolutions that highly impact the nature of the available information, various image quality (with various types of noises), or illumination changes that affect the pixel intensities. The main drawback here is the non-reliable convergence domain that happens due to the non-linearities of the cost function to be minimized. In addition, such method is restricted to the empirical setup. VS system only performs eye-in-hand configuration to reposition from starting to desired location. This makes it less feasible and limited to practical scenarios. Since the direct VS methods servo on the entire image features, extra fine-tuning is required to specify to certain location. In case of applying further developments by generalizing on to several target objects with various lightings and backgrounds, direct VS would not be the best choice due to the requirements of re-training datasets and task-specific fine-tuning. The photometric information of the entire input image is not required in the PBVS methods, but the geometrical pose of target object is needed. Recent works [36, 41, 46, 60, 61] have reported vision-based robot-object-interaction task without relying on the whole input image.

2.3.3 Vision-Based Mobile Manipulation

The configuration of mobile manipulator has the advantage of mobility achieved by the mobile base and dexterity executed by the manipulator. This type of robot is more flexible for manufacturing than any other traditional approach such as stationary manipulators or limited Automatic Guided Vehicles (AGV) [62]. Implementing visual servoing control scheme into manipulators could be challenging when the robot continuously operates in 3D space based on only visual features. In Pose-Based-Visual-Servoing, the measurement of the target is happening by utilizing the vision feedback signal that controls the movement of the robot until the visual error becomes zero [63]. Researchers in [64] used a hybrid control scheme which used the strengths of Image-Based and Pose-Based VS methods for aerial manipulation. The polar and cartesian parameters of a target object were used in conjunction with a Jacobian equation to compute the camera pose relative to the target object. This hybrid method also creates an effective system for a 6DOF manipulator by resolving the rotational and translational issues during object tracking [65]. Image Based Visual Servoing is the approach that operates without the need for pose estimation step of the target object [66, 67]. It uses the image features directly since the pose is computed implicitly [68]. It considers the vector which is a set of parameters that are present in image data. The main control loop architectures of both approaches are relatively similar; the only difference is the input type and the feedback loop. The input parameter in PBVS is the 3D position of an interesting point in the workspace. In [69], two methods of VS were tested to compare their stability and robustness if modeling errors were present in the computation. The authors concluded that both methods were asymptotically stable and locally robust.

The mobile manipulator configuration typically creates redundant joints which may inhibit its functionality. Several published studies have established the generic kinematic modelling of a mobile-manipulator configuration. Researchers in [70] proposed metaheuristic algorithms to enhance the kinematic solutions of mobile manipulator designs. A different configuration for a mobile-manipulator was introduced by Avilés S., et al in [71], where the mobile-base had two types of wheels including directional and fixed. The system was tested with simulation environments but still lacked the guided vision-based method. The controller law in [72] used fuzzy logic to present the path planning of a mobile manipulator using a vision-tracking system. However, soft computing concepts always propose approximation results which may provide a less accurate output. In [73], a multi-camera VS system architecture was modeled and presented to control a robotic system where a target object is covered by multiple views. The controller could make different decisions since the pose estimation system was processed based on two perspectives. However, the computational level required longer processing times in addition to the limited applicable purposes.

The reviewed literature documented studies that treated mobile-robot modeling problems, while other publications focused on improving stationary robot manipulators. However, combining the control techniques of a mobile platform and a manipulator, both are deep network-based, is the challenge for new technologies. This thesis is therefore aimed at developing and testing a complete 3D visual servoing system that operates based on the collected feedback from a single camera. Chapter 3 focuses on an alternative solution using deep-ConvNet and PBVS to control a long-range AMM. This type of control scheme requires the pose parameters related to a target object that defines the goal for the AMM robot's final pose. The underlying idea is how to control the pose of the robot with respect to the detected object by using visual features synthetically trained.

2.4 Grasping Unknown Objects for Robotic Manipulation

This section reviews the recent related works in objects-grasping for robotic manipulation application based on learning-methods. Extensive studies were found in literature for investigating robotic manipulation by deep-learning and iterative learning approaches.

2.4.1 Model-Oriented Grasping

Classical model-based methods [74], geometric grasping criteria [75], and pre-computed 3D-objects grasping used point cloud registration for estimating grasp planning

[76, 77]. The studies integrated semantic constraints to determine the regions of grasping. They attempted to accomplish a grasping task based on a geometrical perspective, involving analytic models of geometric constraints. Such methods are inapplicable to different scenarios, and typically need pre-defined information on target objects, making it non-practical in unstructured environments [78-80].

With the advent of deep learning, another set of studies has explored the approaches of training deep policies for grasping [81-83]. Recent model-based grasping task was documented in [84], where the system needed 5 million depth images of synthetic point clouds and grasp metrics computed from 3D objects to carry out training step. A common approach in [85, 86] estimates object poses before planning to perform grasping. Conventional model-based analytical methods assume access to the model of target objects and execution environments [87-89].

2.4.2 Model-Agnostic Grasping

Deep learning approaches have enabled the surge of research that uses data-driven methods to address grasping tasks [90-94]. The main concept of DL could be categorized mostly into two techniques depending on the style of supervision. Supervised learning methods that guide the system to learn from the annotated dataset that has been previously labeled. Training usually compares the annotated dataset to the pre-defined knowledge of the ground-truth. A number of data-driven approaches used known forms to achieve object grasping [95], included a human-supervised task that estimates grasping contact points [96]. In addition to the methods that predict proper grasping policy based on geometrical configuration and metrics [97], another set of studies trained the system in the form of standard servoing mechanism to compute the grasp upon pre-planned grasping pose through known grasping trajectories [98].

Other methods proposed an entirely different data-driven manner where the system does not require human annotation or prior knowledge of grasping points [84, 99-107]. Self-supervision methods empower the system to learn the task by trial-and-error concept, the corresponding grasp labels are generated during training phase by testing the given actions. Reference [108] used two components, including grasp success predictor and motor feedback after each successful grasping attempt. However, it requires large-scale data collection setup on multiple robotic stations. A study in [79] proved that it is possible to perform successful grasping based on object-agnostic grasping methods where the robot only requires local features of the target object. This technique has better evolved to achieve a higher grasping success rate on novel objects. The research team in [109] used ConvNet to predict grasp tasks and provided a full picking system tested in an unstructured environment. Even though the system used the idea of object-agnostic grasping, but it still requires forming the target objects with grasping proposals.

Recent works have investigated the model-free methods and proposed effective training policies on grasping tasks [110, 111]. Study in [112] explored a method to detect grasp pose in point clouds. Lerrel P. et al. in [104] presented a grasping framework for large-scale data collection. Grasping task with DRL problem was reported in [113] attempting to achieve real-world grasping. However, the learned system was able to test limited categories of target objects without considering generalization purposes.

Generalization based on little previous knowledge is a constant inspiration for self-learning concept. Numerous studies have directly used model-agnostic techniques to train policies that decide grasping without explicitly requiring pre-defined knowledge of the target object (such as object pose, dynamics, or shapes). Recent works have attempted to address this challenge and provided successful examples [114]. Others have investigated learning approaches by using algorithms like Monte Carlo estimation [115], adaptive Q-learning [116], guided policy gradients [117], trust-region policy optimization [118], and double deep Q-learning [119]. Systems performed successfully on individual tasks and reported efficient instances of handling novel scenarios that were not seen during the training step. However, the prior works did not consider the data-hungry nature of DL nor the training time-consuming. In order to apply the acquired training knowledge to new objects, a large amount of training datasets should be provided, which need to cover a big assortment of objects and scenarios. This requires thousands of grasping trials on hundreds of different objects. Generalizing the task of grasping based learning, on a reasonable dataset with limited knowledge, to novel objects was unclear and out of focus.

Recent works [113, 120-123] (have system structure similar to ours) utilized UR5 manipulator, parallel gripper, and RGB-D camera to achieve unknown objects in real-world situations. However, reference [122] requires segmentation step (included object contours) in order to break down the target objects. The system proposed in [123] is not end-to-end self-learning approach, the method requires thousands of manual data-collection trials that should be conducted by humans. Reference [121] demands a prior step to analyze the geometry around target objects in order to estimate a grasp pose. Study in [113] was unable to achieve grasping on completely new objects, it requires prior information of the general class of objects for which the system was trained. Another work [120] has presented a robotic grasping system that trains perception network and policy network separately instead of end-to-end approach.

On the contrary, our proposed scalable system in Chapter 4 is entirely end-to-end self-learning approach. It operates without requiring any type of prior knowledge of target objects. The agent learns grasping on a minimal dataset of grasp instances in simulations. The gained knowledge was then transferred to generalize on novel objects to fulfill the downstream manipulation tasks. DRL is utilized for training the agent on making grasping decisions without needing task-specific training data or 3D object CAD models. Perception network estimates grasping directly from RGB-D image using visual pixel-wise affordances, which map visual observations to grasping actions. Affordance is defined as the possibility of grasping success offered to the robot on the basis of the current state of the workspace. Primitive action is held by the robot at the corresponding highest affordance value.

2.5 Grasping Novel Objects in Challenging Situation

This section reviews the recent works related to object grasping for robotic manipulation challenge and non-prehensile manipulation planning. In addition, we highlight the learning-based approaches for grasping unfamiliar objects in cluttered scene combined with pre-grasping required actions.

2.5.1 Robotic Grasping for Manipulation

Grasping issues have been well documented in the literature. Several works employed model-driven methods of known objects with defined information and environment properties [95, 124]. Studies in [125] and [41] have used model-based reasoning methods to execute grasps based on previous modeling-forces. In addition, grasp points were computed from the predefined 3D objects models. These methods require poses and prior knowledge about dynamics that do not hold in real-world settings. Another group of works used data-driven approaches to achieve robotic grasping [126], and skill learning for precision assembly [127]. Such studies used deep-learning [128] methods to train grasping-models [129-132] for robotic manipulation and grasping tasks [133]. The methods used visual feedback to perform grasping actions without needing specific information like poses or dynamics. However, such methods did not consider a cluttered scenario in adjacent situations where pre-grasping actions are necessarily needed. Grasping objects in challenging scenarios without non-prehensile assistance remains intractable.

2.5.2 Non-Prehensile Manipulation Planning

Non-prehensile manipulation action is an essential issue that was found in the literature to facilitate grasping objects in hard scenarios. For example, multiple objects in a cluttered situation where it is difficult for a target object to be directly grasped [134, 135]. Early studies utilized conventional solutions that make assumptions and model the dynamics of non-prehensile with frictional forces. Such works also investigated the frictional distributions findings that happen while pushing the surfaces of the objects. However, inaccurate predictions from friction modeling were found in real-world experiments [136, 137].

Prior recent researches proposed to use data-driven approaches to study the dynamic-modeling of pregrasps. These studies, however, concentrated on performing stable pregrasp action on an individual object. They most likely struggled with sequences of pre-grasping actions that should benefit to potential grasps [89]. Adaptive grasping for a large range of objects was successfully held in [138]. However, the approach was carried out without considering grasping in hard adjacent scenarios. Study in [139] proposed a framework that utilizes pushing-grasping system to handle object pose uncertainty during grasping attempts. Other approaches in [140, 141] used model-free planning of pre-grasping actions that, for example, push target objects to a specific position to be more graspable. Much of the existing works in [77, 142] explored pre-grasping and grasping actions to achieve grasps with no predefined object knowledge needed. However, prior systems need to cover generalization on novel hard scenarios that have entirely different geometrical shapes. Results were reported in [143] which investigated the approach of using reinforcement learning to train strategies of pushing and grasping. The framework segments objects and estimates pushing and grasping that result with the highest expected rewards. The system framework, however, is not end-to-end where the observation step is separated from execution. Simon M. in [144] addresses sequential learning problem for robotic manipulation by human demonstrations. The study in [145] proposed linear push policy based on manual metrics to achieve pre-grasping which is defined by using point clusters as objects.

Modern studies [145-148] have formed motivations for this thesis to explore the joint self-learning of pregrasp-and-grasp manipulations. The sequential decision-making problem based on minimal raw experience is an inspiring challenge in autonomous robotic manipulation, which also has been obtained less attention.

2.6 Summary

This chapter reviewed the related prior studies for the field of robotic manipulation strategies and object identification methods. In addition, we highlighted the current gaps and challenges in the robot-object interaction community. Which have motivated this thesis to focus on the requirements in order to develop an intelligent robotic system for the autonomous manipulation application.

The determined gaps in the existing studies start with complex sensory-fusion network and costly perceptual modality methods. Furthermore, combinations of computer-generated training data and real-world information to train a robotic agent and gain the required knowledge, as well as restricted to highly textured objects and predefined environments.

Learning-based methods typically demand model-based or feature-based information (such as 3D CAD model, pose estimation, or objects categories). Task-specific training session or predetermined data of target object could also be required. It should be noted that object-agnostic systems often demand large assortments of datasets with long training time. Furthermore, besides being unaffordable, it would be impractical to provide multi-station of robots to train on certain tasks and collect data. Advanced technologies should fulfill the adequate learning-based abilities utilizing only robot's experience and without the need for human data-collection assistance.

Finally, generalization the acquired (training-based) knowledge to novel execution environment is a crucial matter in the field of advanced robotic manipulation application. The trained robot should be capable to handle more than simple target objects (for instance cubes). It is unreasonable to require extra training efforts when the robot generalizes to novel comparable task.

Chapter 3. Visual Servoing in Autonomous Mobile Manipulation

3.1 Introduction

The fourth industrial revolution (industry 4.0) demands high-autonomy and intelligence robotic mobile manipulators. The goal is to accomplish autonomous mobile-manipulation tasks with least human interventions. The critical challenge for robot-object interaction is to estimate visually the pose of the target object in a 3D space and apply it into a vision-based control scheme in autonomous mobile-manipulation applications.

In this chapter, a perception network algorithm is developed for object pose estimation. The information is then utilized as the feedback loop in 3D Visual Servoing (VS) to achieve mobile manipulation using a single RGB camera. An extensive system called Deep-Visual-Servoing (DVS) is introduced with its experimentations. The proposed system addresses the integration of: 1) training of deep-CNNs using synthetic (single RGB images) datasets, 2) continuous 6DOF object pose estimation utilized as sensing feedback, 3) to control mobile manipulator in 3-D space based on the vision information. The developed DVS consists of two main steps. First, a perception network that trains based on only synthetic datasets and generalizes efficiently on real-world experiments without post-refinements. Second, the controller takes the estimated target pose and generates continuous translational and orientational joints velocities in 3D space. The full system is presented along with the developed design of a proportional controller based on Lyapunov's theory. Experimental findings from simulations and real-world settings showed the efficiency of using synthetic datasets, which can be generalized to the physical mobile-manipulator task. The kinematic model of the presented robot is experimentally verified and discussed using the Husky mobile-base and 6DOF UR5 manipulator.

The rest of the chapter is organized as follows: Section 3.2 briefly presents the state-of-the-art and chapter contribution. Section 3.3 illustrates the modeling of the perception network for 3D object detection and pose estimation, camera, robot and visual servoing model development. Section 3.4 shows experimentation results with discussions. Lastly, the conclusions are set out in Section 3.5.

3.2 Autonomous Mobile Manipulation

The use of Autonomous Mobile Manipulators (AMM) has grown in many industries with developments of enabling systems to autonomously transport, organize and process various assets. Two main industries that utilize this technology are manufacturing facilities and courier services that maintain a large inventory and benefit from an efficient autonomous robot. To evolve the performance in autonomy and versatility, applied robots make use of several sensing technologies and control algorithms. Two key steps are typically required in order to carry out autonomous mobile manipulation task. Firstly, the perception step based on sensor-fusion method which is used to estimate objects and perceive surroundings. Secondly, sensing-based robot motion control technique. This sensory network, however, leads to complex and expensive robotic systems [149, 150].

Recent studies have aimed to use only vision sensors to perceive the robot's environment and gain adequate information about target objects [151]. However, the preparation of training datasets requires effort and skills. Training of a deep convolutional network is computationally expensive and time consuming [41]. Unlike 2D object detection, labeling 3D object is difficult and required experts. Thus, the use of synthetic datasets for deep neural network training has addressed this problem by proposing an endless amount of valuable pre-labelled training datasets that are safely generated in a fair effort [44]. Studies in [152] trained on synthetic and real-world dataset collections including fine-tuning requirements. As a result, implementing these methods in new real-world environments is restricted and needs structured background. Another set of visual perceptions studies [36, 60, 61] requires complex vision-based setup to obtain pose estimation with limited background. Stereo vision algorithm was proposed and tested in [151] using pointcloud data from multiple stereo systems and utilizing iterative closest points. Vision-based mobile-manipulator control was attempted in [153] using evaluation policy and requiring off-line training step. M. V. Minniti, et al. in [154] addressed the whole-body control of mobile manipulator without considering tracking a target object. Much of the latest studies [155] developed a mobile manipulation system included kinematic model where a path planner, however, is still required.

Nevertheless, none of the prior works have demonstrated a complete continuous framework for achieving 3D visual-servoing in mobile-manipulator based on synthetic-trained deep neural network. This chapter proposes and verifies a novel long-range mobile manipulation system that combines visual perceptions with the robot motion control mechanism, named Deep-Visual-Servoing (DVS). The system presents an end-to-end framework of perception network combined with 3D visual servoing for a sophisticated robotic manipulation task. The perception network constantly estimates the 6DOF of target object, this network entirely trained on computer-generated (single RGB) images and successfully generalized on real-world experiments.

The proposed pipeline is shown in Figure 3.1. There are two main phases explained in the architecture: 1) perception network based on deep-learning to estimate the pose of the object; 2) execution, which achieves the desired pose between the end-effector and the object by utilizing the estimated pose and visual servoing control. The main idea is to utilize CNN to identify the 2D target object in the single RGB input image, followed by pose estimation methods to retrieve the 3D translation and 3D orientation of the target object. The results of 6 DOF object pose information will be utilized as a feedback into vision-based control system so-called pose-based visual servoing. Since the aim of the DVS is to achieve robotic-visual-servoing task of household objects, we evaluate our system based on YCB images [45].

The outputs of the perception network are fed to visual-servoing scheme to autonomously control the movements of a long-range mobile manipulator, (6DOF manipulator arm mounted on 2DOF differential drive mobile-base). The proposed system was successfully implemented in simulation environment as well as real-world settings. The use of synthetic datasets was applied in the context of 6DOF object pose estimation from single image and executed in 3D continuous AMM task. Our findings have shown the physical capabilities of generalization to novel environments for the handling of light and occlusion variations.

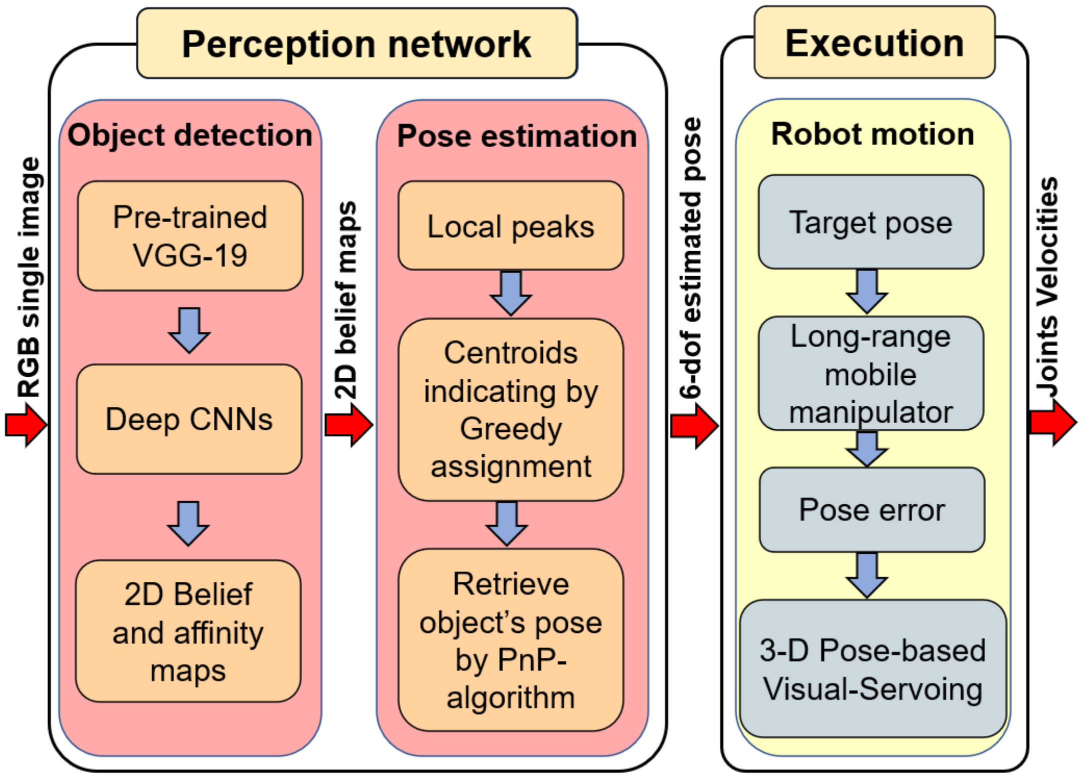


Figure 3.1: The proposed architecture of the DVS for robotic manipulation systems.

3.3 System Modelling and Visual Servoing

The proposed system architecture consists essentially of three interconnected phases. Firstly, deep-ConvNet identifies 2D objects using a single RGB camera. A synthetic dataset was utilized in training. It covers varieties of lighting, occlusions, and background conditions. The results yield information of 2D belief maps that represent multiple 2D objects in the image. Secondly, pose estimation algorithms utilize belief maps to retrieve the 3D translational and 3D orientational pose of the object in the workspace without post-alignment step. Lastly, the desired pose of the target object is directly sent, without the need to post-refinements of the estimated pose, to the visual servoing controller which is in charge of achieving robot pose with respect to the camera frame.

Figure 3.2 demonstrates the overall system architecture, the robotic environment representation was taken from the simulation settings during the experiments. Single RGB image of the target object is sent to the perception network which returns full pose of the object. This step is followed directly by the execution step that produces the required translational and orientational joints velocities. The used robot should expectedly move from the initial to the desired pose with respect to the target object. The modelling of the proposed system is illustrated in this section, which includes perception network and training, mobile manipulator robot kinematics, and the designing of VS control law based on the principle of Lyapunov.

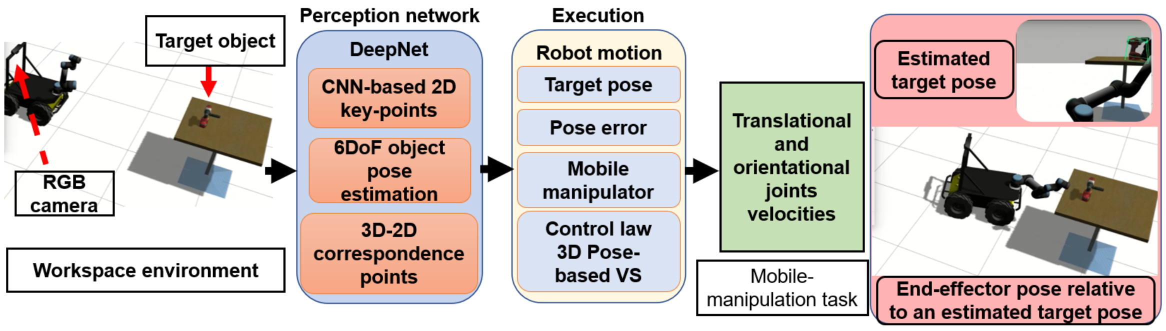


Figure 3.2: Overall system architecture.

3.3.1 Perception Network and Training Protocol

The proposed perception network consists of two main steps namely: CNN-based and pose-estimation step. The network infers the location of the object with pose estimation in the workspace. Inspired by Convolutional Pose Machines (CPMs) [156], our deep-ConvNet infers the target key-points in a single-shot, fully convoluted deep network and through multi-stage architecture. Each stage of the convolutional neural network generates 2D belief maps of the target object. The input of the subsequent stage takes advantage of both, image features and belief maps of the previous stage. These stages are the sequence of predictors that make better predictions and increasingly refine the estimate of the object location. Since the design of the network is convolutional, the early stages (of generating

2D belief map) might have ambiguities that will be resolved by the later stages, as shown in Figure 3.3.

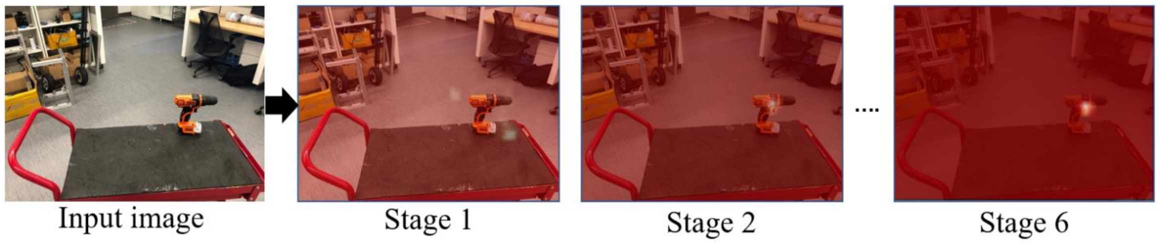


Figure 3.3: An input image proceeded in multiple stages (from stage 1 to stage 6) with one vertex.

The training network utilizes a larger receptive field on both the image features and belief maps, this manner improves the network accuracy. During the training, transform-learning is applied as a feature extraction step, followed by multiple-stages architecture. The final goal of the feedforward network is to detect 2D key-points of the observed objects in the image. The network outputs two types of maps, which are belief maps and vector fields, that help indicating multiple objects of the same category. Each stage in the training network generates 9 belief maps, each one represents a vertex which will be at the end 8 projected vertices of the 3D bounding box, and one belief map for the centroids.

In the same manner, each stage produces 8 vector fields that indicate the direction to the centroid from each 8 projected vertices. To detect the object's centroid, the network always seeks for the local peaks from the belief maps. It uses the greedy algorithm to associate the projected vertices to the indicated centroids, similar to [36, 61]. Then, the object's projected vertices of the 3D bounding box are utilized by Perspective-n-Point (PnP) algorithm to retrieve the 6 DOF object's pose [157]. PnP algorithm at least requires four vertices to obtain the pose. Intrinsic camera parameters and object dimensions are needed to estimate the translation and orientation of the target object relative to the camera frame. Figure 3.4 demonstrates the steps of object pose estimation operated on multiple target objects without the need to the post-alignment of the estimated poses.

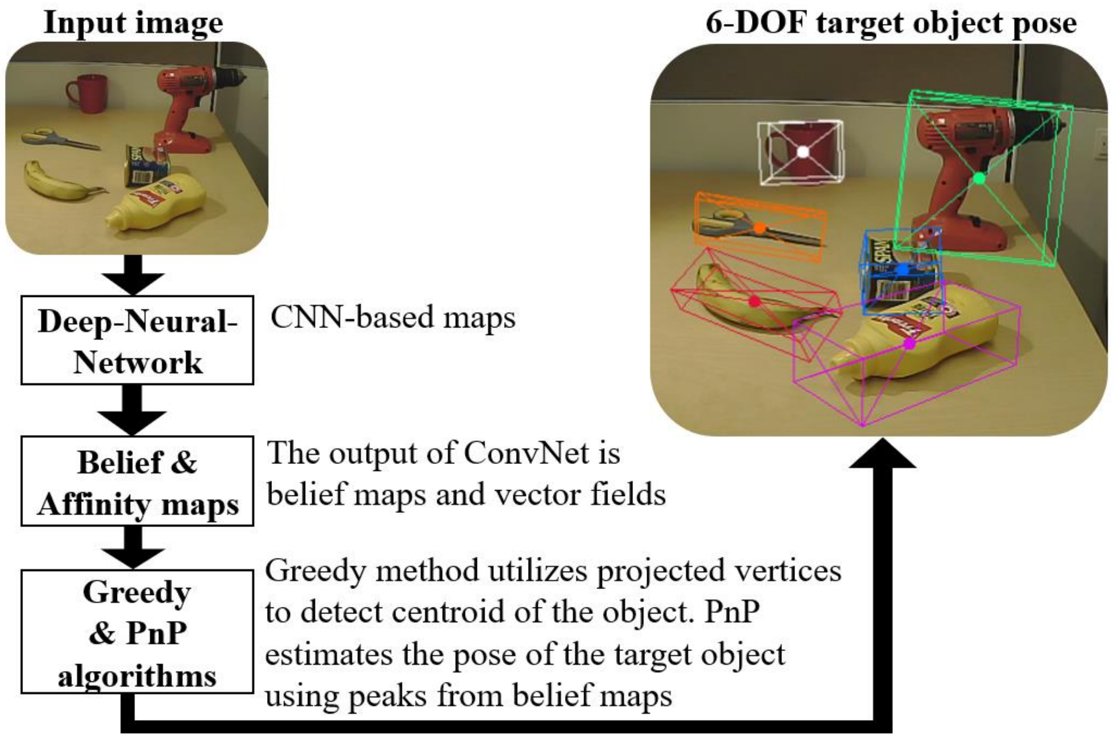

Input image
6-DOF target object pose
Deep-Neural-Network
CNN-based maps
Belief & Affinity maps
The output of ConvNet is belief maps and vector fields
Greedy & PnP algorithms
Greedy method utilizes projected vertices to detect centroid of the object. PnP estimates the pose of the target object using peaks from belief maps

Figure 3.4: Object identification and pose estimation of the multiple objects.

In Figure 3.5, the network trains directly on the input image (400×400×3) and the first 10 layers of the network compute the image features based on VGG-19 model [158], pre-trained on ImageNet [159]. Then, features dimensions minimized from 512 to 256 and from 256 to 128 by using two 3×3 convolutional neural layers. The feature map dimension-128 is the input to the first stage that contains 3 layers of (50×50×128) and one layer of (50×50×512), as shown in Figure 3.6.

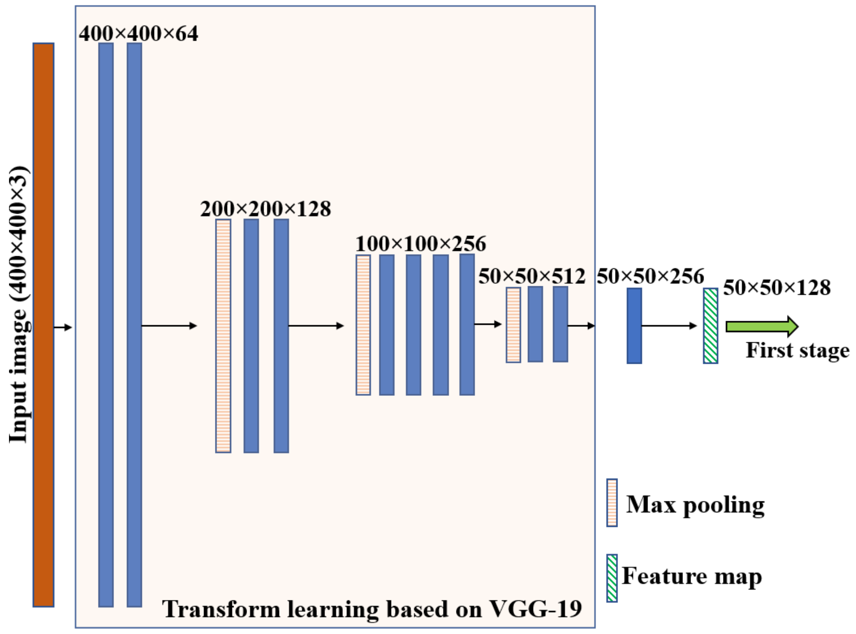


Figure 3.5: Network architecture and features extractions.

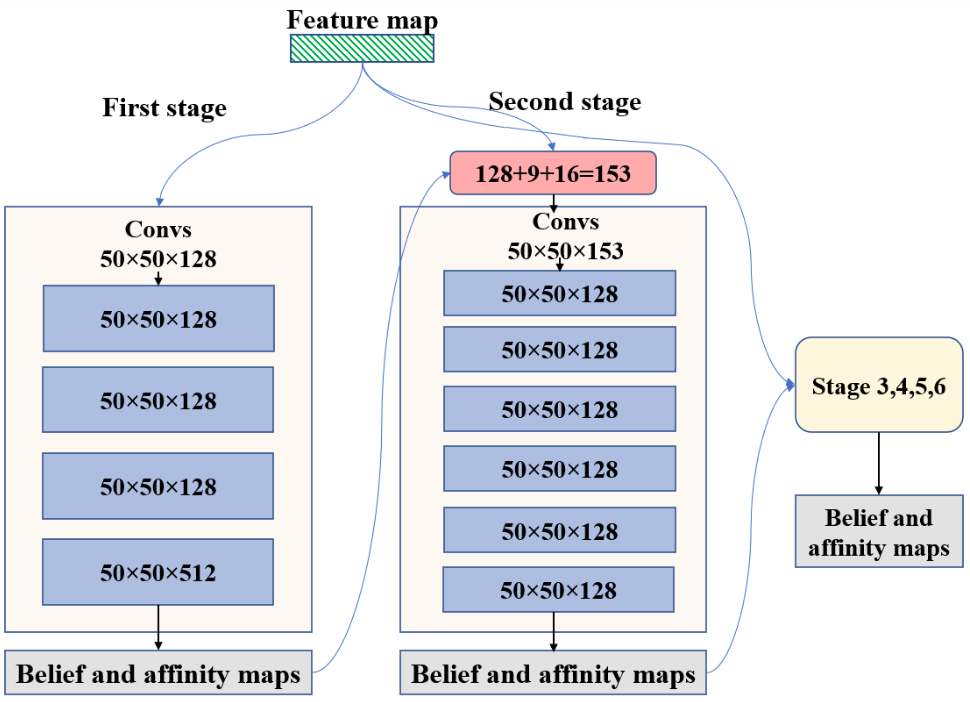


Figure 3.6: Proposed CNN-based architecture for 2D belief maps.

This stage followed by belief map (50×50×9) and vector field (50×50×16) that both are fed to the next stage, including the output of the feature map. Similarly, the remaining stages (from 2 to 6) should have same structure as the first stage. The receiving dimension input (128+9+16=153) is an output of image features map, as well as the belief map and vector field of the immediately preceding stage. In the remaining stages, there are 6 layers of (50×50×128) and the output is belief map and vector field. The output of a certain layer maps to the subsequent input layer by Rectified Linear Unit (ReLU) activation functions which always keep the positive value.

For the training dataset, more than 60k of synthetic images (RGB image only) were considered based on YCB object-dataset. The image features extractor for first layers-set learns from VGG-19 pre-trained model. The system has been implemented on PyTorch platform [160]. Training was carried out on 8 GPUs (each one is Nvidia Tesla K-80), different sessions used 4 GPUs (each one is GEFORCE RTX 2080 Ti) for 70-80 epochs with a batchsize of 128, the results have been tested on CPU Intel. Core i7-7700k. Learning rate is 0.0001 based on network's optimizer, similar to Adam [161]. During the training step, the regularization method of the Loss function (L_2) is used to calculate the loss error between the predicted output and the true value (ground-truth) for each input. At the end of each epoch, error value is accumulated by applying Mean Square Error (MSE) of each input. This is applied for the entire training phase which shows how the model learns by minimizing the square of the differences between the true and estimated values. Loss function will build and label belief and affinity maps. The used L_2 loss function calculates the loss error between the predicted belief maps and the true value (ground-truth) of the training data. The total loss in the stage i is the sum of losses $L = \sum_i^n L_i$, where L_i is defined below:

$$L_i = \frac{1}{n} \sum_{i=1}^{n} \sum_{v=1}^{m} \left(B_v^i - \dot{B}_v\right)^2 \qquad 3.1$$

where B_v^i is the CNNs output for a belief map at stage $i \in (1\dots n)$ for vertex v $\in (1\dots m)$, n is the stage number and m stands for the number of vertices, $\dot{B}_v$ is the ground truth. This approach called intermediate supervision where the gradient is stored at the end of each stage, this avoids the well-known issue vanishing gradient.

To obtain the 3D bounding box, the 2D object vertices output from belief maps followed by PnP iterative approach, as shown in Figure 3.7. The purpose of PnP problem is to estimate the translation and orientation of the calibrated camera from the known 3D points (the object dimensions) to the corresponding 2D image projections. This method helps to find the object pose from 3D-2D correspondences point, which is based on Levenberg-Marquardt optimization approach [162], where the proper object pose should be found by minimizing the reprojection error (RE). RE is the sum of the squared distances between the observed 2D projections image-points and projected 3D object-points, as shown in equation 3.2.

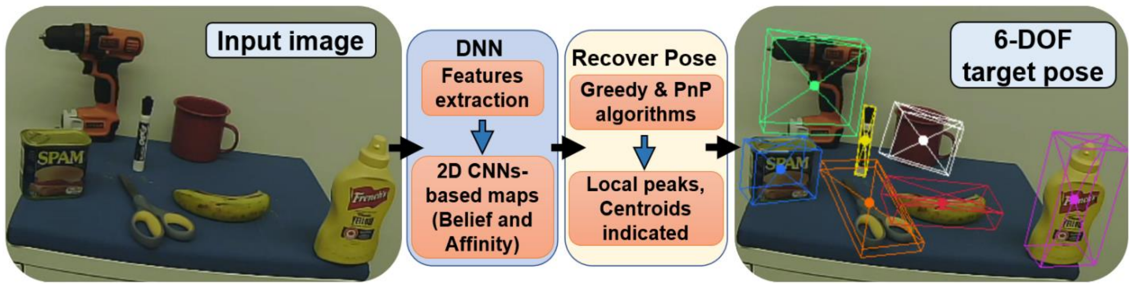


Figure 3.7: Object pose estimation for multiple target objects.

$$x^2(m,b) = \varDelta y_1^2 + \varDelta y_2^2 + \varDelta y_3^2 + \cdots \qquad 3.2$$

The PnP inputs are the intrinsic camera parameters, target object dimensions, and 2D observed points. The output results retrieve the rotational and translational vectors of the 3D object into 2D image plane. There are three coordinate frames namely: world, camera, and the image plane. If the orientation and translation of a 3D point are known in the world coordinate, the corresponding points of the objects could be transformed into camera coordinate, using equation 3.3. Then, the 3D point can be projected into the image plane, by utilizing the camera intrinsic parameters. Let P_i^w be the 3D point in the workspace coordinates, which is shown in equation 3.3 relative to C_i^C camera coordinate system. The

3D point $[x_i^w \quad y_i^w \quad z_i^w]^T$ is projected into image plane m_i as $[u_i \quad v_i]^T$. The perspective transformation for the pinhole camera model is shown in equation 3.4.

$$C_i^C = \mathrm{R}\, P_i^w + t$$

$$\begin{bmatrix} x_i^c \\ y_i^c \\ z_i^c \end{bmatrix} = \mathrm{R} \begin{bmatrix} x_i^w \\ y_i^w \\ z_i^w \\ 1 \end{bmatrix} + \mathrm{t} \qquad 3.3$$

$$\mathrm{S}\, m_i = K\, [\mathrm{R}|\mathrm{t}]\, P_i^w$$

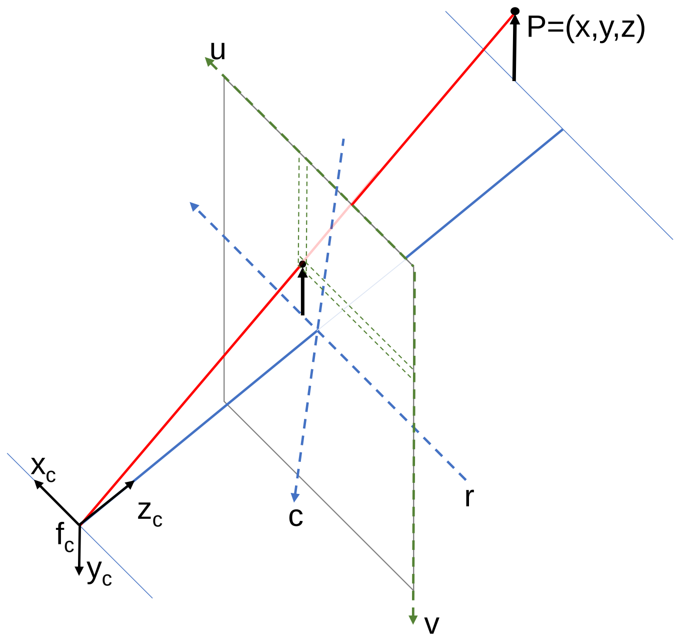
$$\mathrm{S} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_r \\ 0 & f_y & c_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x_i^w \\ y_i^w \\ z_i^w \\ 1 \end{bmatrix} \qquad 3.4$$

Where K is the intrinsic camera matrix, [R|t] is the extrinsic joint matrix which represents rotation and translation vectors respectively. It is used to describe the homogeneous rigid motion of the object point with respect to the camera coordinates and translates the coordinates of each 3D point into the coordinate system relative to the camera frame. S denotes as a scalar projective factor. f_x and f_y are the focal length expressed in pixel coordinates. c_r and c_c are the principal point that is image center in pixel frame.

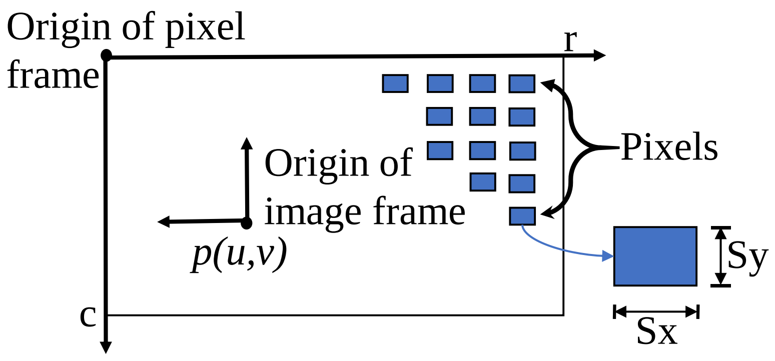
If P_i^w is known and the camera intrinsic is also known, therefore, the projected image point m_i could be determined. However, [R|t] is unknown, PnP algorithm is used for non-linear system to determine such a pose that minimizes RE, an approximated estimation of [R|t] could be obtained by iteratively changing the estimated [R|t] and reducing RE. The process requires the 3D object points and 2D observed projection points proposed by deep-ConvNet and vertices.

3.3.2 Single Camera and Mobile Manipulator Model

Single Camera. Investigating the geometry of the image formation process can demonstrate the relationship between the interest feature points in the world frame and their projections on the image plane. A common image formation model is the Pinhole lens approximation [163]. As shown in Figure 3.8 (a), *P* is the object point in the workspace with coordinate *P= (x, y, z)*.



(a)



(b)

Figure 3.8: Single camera model, (a) pinhole model, (b) pixel coordinate frame.

The point *P* is projected as *p (u, v)* on the image plane with coordinate *(u, v, f)*. The perspective projection of the point *P* in the camera coordinate frame is (x_c, y_c, z_c), and the same point is represented as (*r, c*) into pixel coordinate frame. The intersection of the z-axis with the image plane is the principle point (c_r, c_c), which is measured with respect to the coordinates of the pixel frame. *f* is the distance between the image plane and camera frame (camera focal length). In case where *f* is known and the coordinate of the point *P* in the camera frame (x_c, y_c, z_c) is also known. Then, the 3D point of *P* in the workspace can be determined by indicating the projected point (*u*, *v*) in the image plane [164]. The perspective projections result in equation 3.5.

$$u = \frac{fx}{z}, v = \frac{fy}{z} \qquad 3.5$$

This helps to find the object position in the workspace by introducing (*u*, *v*). However, the sensor of the digital camera (CCD/CMOS), which is a 2D array, is measured in pixels. This is shown in Figure 3.8 (b) where (*u*, *v*) is the origin of the image plane and (*r, c*) is the origin of the pixel frame. The pixel shapes are usually rectangular, and the pixel's width and height are given as *Sx* and *Sy* respectively. The centers of the image plane and pixel coordinate frame are not the same. Therefore, the coordinate transformation between the image plane and the pixel frame are expressed below:

$$r = -f_x \frac{X_c}{Z_c} + Cr \quad , \quad c = -f_y \frac{Y_c}{Z_c} + Cc \qquad 3.6$$

In order to obtain the position of the projected object from the world into the image plane, pixels are used as shown in the previous equations. However, the various camera parameters such as *Sx, Sy, u, v,* camera focal length (*fx, fy*), and (c_r, c_c) are unknown. These parameters are known as intrinsic camera parameters and they are obtained through camera calibration and they are constant for a given camera. A chess board of known dimensions and number of squares was availed to calibrate the used camera and obtain the necessary parameters.

Mobile manipulator. The robotic system in this work consists of; 1) a diffrential drive mobile robot (Husky A200) and 2) a 6DOF manipulator arm (UR5) mounted on the top of the mobile base. Figure 3.9 demonstrates the kinematics frames of 6DOF manipulator arm when the joints angles are zero. Homogeneous representation is used to define the relationship between the frames of the manipulator and the mobile base.

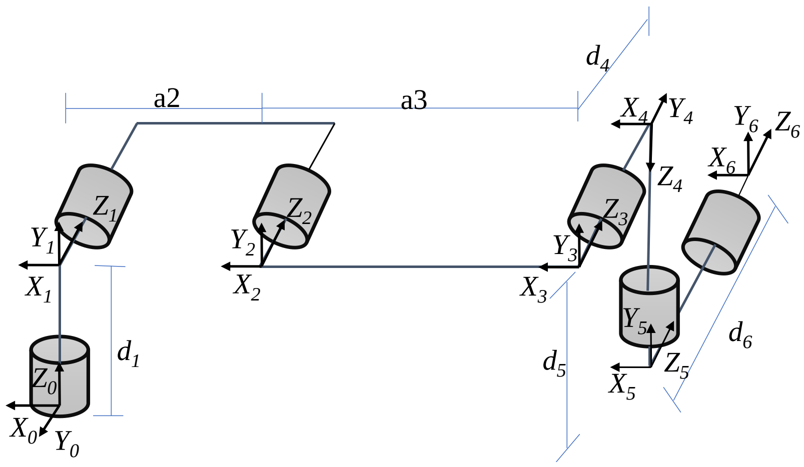


Figure 3.9: Manipulator arm at zero position.

The relationships between the wheel speeds of the mobile platform and 6-joints velocities of the manipulator arm are presented in this section. The robot's rigid motion is demonstrated by the homogeneous transformation matrix (*H*) as shown in equation 3.7.

$$H_n^{n-1} = \begin{bmatrix} R^{3\times3} & d^{3\times1} \\ 0 & 1 \end{bmatrix} \qquad 3.7$$

Equation 3.7 includes the $R^{3\times3}$ which is a rotational matrix expressed in terms of Euler's angles (α, β, γ) and a ($d^{3\times1}$) displacement vector. These both stand for defining the pose relationship between two coordinate frames. A Denavit-Hartenberg (DH) convention is used to define the coordinate frames of our model, as shown in Table 3.1. By using the DH table, we can generate our homogeneous transformation matrix and determine the movement of the end-effector relative to the base-frame.

Table 3.1: DH table for 8DOF mobile manipulator.

n	θ_n	α_n	d_n	a_n
1	θ_1	0	0	0
2	0	90	d_2	0
3	θ_2	90	d_1	0
4	θ_3	0	0	a_2
5	θ_4	0	0	a_3
6	θ_5	90	d_4	0
7	θ_6	-90	d_5	0
8	θ_7	0	d_6	0

Figure 3.10 demonstrates the primary frames of interest including world frame, base frame, arm base frame, and end-effector frame. The modelling of the differential mobile robot is analogous to the two-joint manipulator that comprises of prismatic and revolute joints. The linear and angular velocities are the interpretations of the movements of prismatic and revolute joints of the mobile-robot relative to the world frame.

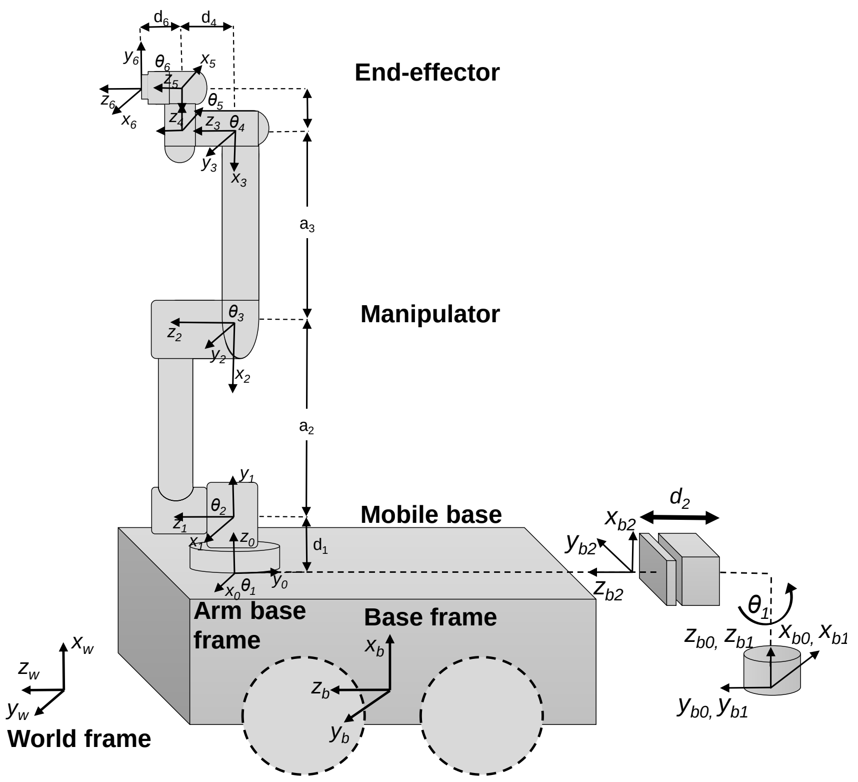


Figure 3.10: Schematic diagram and frames of interest for mobile manipulator robot.

The following equation illustrates the velocity transformation of the base-frame relative to itself.

$$ {}^{b}_{b}V = \begin{bmatrix} {}^{b}_{ab}R & s(d^{b}_{ab})\,{}^{b}_{ab}R \\ 0_{3x3} & {}^{b}_{ab}R \end{bmatrix} {}^{ab}_{ab}V \qquad 3.8 $$

The transformation matrix below shows the base velocity relative to the world, where ${}^{w}_{b}R$ is the rotational matrix between the world and base frame.

$$ {}^{w}_{b}V = \begin{bmatrix} {}^{w}_{b}R & 0_{3x3} \\ 0_{3x3} & {}^{w}_{b}R \end{bmatrix} {}^{b}_{b}V \qquad 3.9 $$

Equations 3.10 and 3.11 explain the velocity transformation that is required to obtain the velocity of the arm-base and end-effector frame respectively.

$$ {}^{ab}_{ab}V = \begin{bmatrix} {}^{ab}_{ee}R & 0_{3x3} \\ 0_{3x3} & {}^{ab}_{ee}R \end{bmatrix} {}^{ee}_{ee}V \qquad 3.10 $$

$$ {}^{ee}_{ee}V = \begin{bmatrix} {}^{ee}_{c}R & s(d^{ee}_{c})\,{}^{ee}_{c}R \\ 0_{3x3} & {}^{ee}_{c}R \end{bmatrix} {}^{c}_{c}V \qquad 3.11 $$

The relationship between the arm-base frame and the robot-base-frame is fixed. In addition, the transformation between the camera frame to the end-effector frame is also fixed. Thus, the skew symmetry matrix ($s(d)$) is required to define the location of the camera relative to the end-effector and the location of the robot manipulator's base frame relative to the base-link of the arm. These parameters are in terms of physical measurements from the robot, as stated in the previous equations. Skew symmetry matrix is mentioned below.

$$ s(d) = \begin{bmatrix} 0 & -r_Z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \qquad 3.12 $$

The velocity of the base relative to the world is further expressed below by substituting equations 3.8, 3.10 and 3.11 in equation 3.9:

$$ {}^{w}_{b}V = E \, {}^{c}_{c}V \tag{3.13} $$

where $E = \begin{bmatrix} {}^{w}_{b}R & 0_{3x3} \\ 0_{3x3} & {}^{w}_{b}R \end{bmatrix} \begin{bmatrix} {}^{b}_{ab}R & s\left(d^{b}_{ab}\right){}^{b}_{ab}R \\ 0_{3x3} & {}^{b}_{ab}R \end{bmatrix} \begin{bmatrix} {}^{ab}_{ee}R & 0_{3x3} \\ 0_{3x3} & {}^{ab}_{ee}R \end{bmatrix} \begin{bmatrix} {}^{ee}_{c}R & s(d^{ee}_{c}){}^{ee}_{c}R \\ 0_{3x3} & {}^{ee}_{c}R \end{bmatrix}$.

Along with the velocity of the base-frame relative to the world-frame, the angular and linear velocities of the mobile-robot are obtained through forward kinematics. This is shown in the equation below where J is the Jacobian matrix that relates ${}^{w}_{b}V$ with $[\dot{\theta n} \quad \dot{dn}]^{T}$.

$$ {}^{w}_{b}V = J^{6\times 8} \begin{bmatrix} \dot{\theta}_{b} \\ \dot{d}_{b} \end{bmatrix} \tag{3.14} $$

Substituting equation 3.13 in 3.14 yields equations 3.15 and 3.16 which determine the camera velocity relative to itself.

$$ E \, {}^{c}_{c}V = J \begin{bmatrix} \dot{\theta}_{b} \\ \dot{d}_{b} \end{bmatrix} \tag{3.15} $$

$$ {}^{c}_{c}V = E^{-1} J \begin{bmatrix} \dot{\theta}_{b} \\ \dot{d}_{b} \end{bmatrix} \tag{3.16} $$

In the term of the left and right wheel velocities, equation 3.16 is further examined and shown in equations 3.17 using kinematics of a differential drive mobile robot (D). In the presented work, the wheel diameter (d) and axle length (l) are 0.33 and 0.545 meters, respectively.

$$ {}^{c}_{c}V = E^{-1} J \, D \, [\theta_{1} \; d_{2} \; \theta_{2} \ldots \theta_{n}]^{T} \tag{3.17} $$

Where $D = \begin{bmatrix} -d/l & d/l & 0 & 0 & 0 & 0 & 0 & 0 \\ d/2 & d/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$.

Equation 3.18 demonstrates the relationship between the camera velocity relative to the joints velocities (including two-joints for the mobile-robot and 6-joints for the manipulator arm). The final velocity equation of the system model is designed and shown in equation 3.18, where $E\, J^{\dagger}\, D^{-1}$ relates camera velocity ${}^{c}_{c}V$ to joint velocities $[\theta_n \quad d_n]^T$. This equation will be combined later with the derived control-law of the visual servoing control scheme PBVS.

$$[\dot{\theta n} \quad \dot{dn}]^T = E\, J^{\dagger}\, D^{-1}\, {}^{c}_{c}V \qquad 3.18$$

Where $J^{\dagger}$ is the pseudoinverse method of Jacobian matrix that used for solving inverse kinematics with J.

3.3.3 Visual Servoing Controller Design

Figure 3.11 illustrates the generic block diagram of the proposed visual servoing system, where the perception network constantly detects and estimates the current object's pose. The generated pose error stimulates the control law to send joints velocities and reduce the error by achieving robot pose relative to the target object.

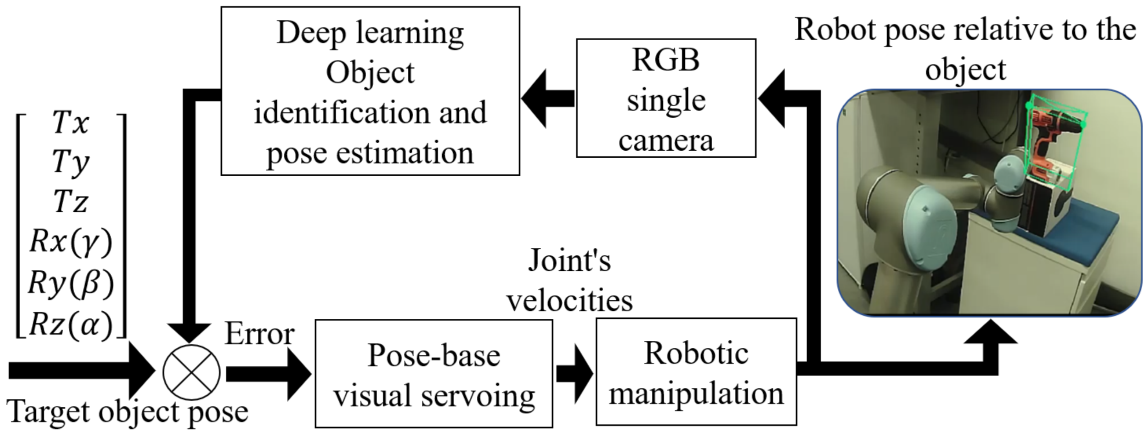

Deep learning Object identification and pose estimation
RGB single camera
Robot pose relative to the object
$\begin{bmatrix} Tx \\ Ty \\ Tz \\ Rx(\gamma) \\ Ry(\beta) \\ Rz(\alpha) \end{bmatrix}$
Joint's velocities
Error
Pose-base visual servoing
Robotic manipulation
Target object pose

Figure 3.11: Deep-Visual-Servoing control diagram.

The PBVS control aims to minimise the error between the desired and current end-effector pose by regulating the movement of the robot through the determination of the necessary translational and rotational commands. Control law in PBVS formulated from the perspective of the desired end-effector frame. The basis of the control law begins with Lyapunov's proportional control scheme, $\dot{e}(t) = -ke(t)$. The proportional gain is denoted as k and the solution yields an exponential decrease of error. Visual servoing error is defined by the difference between the current image and camera parameters, s, and the desired image and camera parameters, s_d, $e(t) = s - s_d$.

The used robotic system operates in three-dimensional cartesian space therefore the error parameters must be defined by two vectors representing the position and orientation of the end-effector. This data is obtained by means of pose estimation algorithms, which use a 3D model of a target object to refer to the image data for defining the pose end effector. Equation 3.19 defines the results from the pose estimation algorithm.

$$s = \left(T_e^{e_d}, \phi_e^{ee_d}\right),\ s_d = (0, 0) \qquad 3.19$$

The first vector, $T_e^{e_d}$, is the position vector of the current end-effector frame related to the desired end-effector frame. The second vector, $\phi_e^{ee_d}$, is the orientation vector, in terms of Euler's angles, of the current end-effector frame related to the desired end-effector frame. The desired pose vector is zero, since all vectors are represented in relation to this desired frame. The main error equation is further expressed in equation 3.20.

$$e(t) = \left(T_e^{e_d}, \phi_e^{ee_d}\right) \qquad 3.20$$

The overall control scheme is derived from both the position and orientation vectors introduced earlier. The change in position vector can be expressed in terms of a rotation matrix between the current and desired end-effector pose, $R_e^{e_d}$, as seen in equation 3.21.

$$T_e^{ed} = \left(R_e^{e_d}\right)^T T_e^{\dot{e}e_d} \qquad 3.21$$

Repeating the general process for the orientation vector seen in equation 3.22.

$$w_e^{ed} = \left(R_e^{e_d}\right)^T w_e^{\dot{e}e_d} \qquad 3.22$$

Angular velocities are determined by the rate of change in Euler angles found in $\phi_e^{ee_d}$ which are transformed using $T(\phi)$ defined in equation 3.23.

$$w_e^{\dot{e}e_d} = T(\phi)\, \phi_e^{ee_d} \qquad 3.23$$

Where $T(\phi) = \begin{bmatrix} 0 & -sin\varphi & cos\varphi * cos\theta \\ 0 & cos\varphi & sin\varphi * cos\theta \\ 1 & 0 & -sin\theta \end{bmatrix}$.

Substituting Equation 3.23 into 3.22 determines the velocity of the end-effector in terms of angular velocities.

$$w_e^{ed} = \left(R_e^{e_d}\right)^T T(\phi)\, \phi_e^{ee_d} \qquad 3.24$$

Equation 3.21 also defines $T_e^{\dot{e}e_d}$ as the rate of change in translational error. This definition can produce the following formula.

$$T_e^{ed} = \left(R_e^{e_d}\right)^T T_e^{\dot{e}e_d} = \left(R_e^{e_d}\right)^T e(\dot{t})_t \qquad 3.25$$

Like the translational error, the rate of change in orientation error is defined by $\phi_e^{ee_d}$ when considering equation 3.20 to generate the following equation.

$$w_e^{ed} = \left(R_e^{e_d}\right)^T T(\phi) * e(\dot{t})_W \qquad 3.26$$

The control law for the velocities of the end-effector is derived, as shown below, by applying Lyapunov's proportional control scheme from proportional error to equations 3.25 and 3.26.

$$\xi_{ee}^{ee} = \begin{bmatrix} T_e^{ed} \\ w_e^{ed} \end{bmatrix} = -k\, L\, e(t) \qquad 3.27$$

where L is known as the interaction matrix that relates the error value to the end-effector velocity.

The estimated pose error is the input to the controller, while end-effector velocity is the output. By combining this with the previous system model developed in equation 3.18, the completed control law is designed and determined in equation 3.28. This law controls the joints speeds in proportion to the error that happens between the current and desired robot end-effector pose. In order to analyze the performance characteristics of the robot model, this control is implemented physically by experimentations in the following section.

$$[\theta_1 \quad \dots \quad \theta_n]^T = -k\, E\, J^{\dagger}\, D^{-1}\, L\, e(t) \qquad 3.28$$

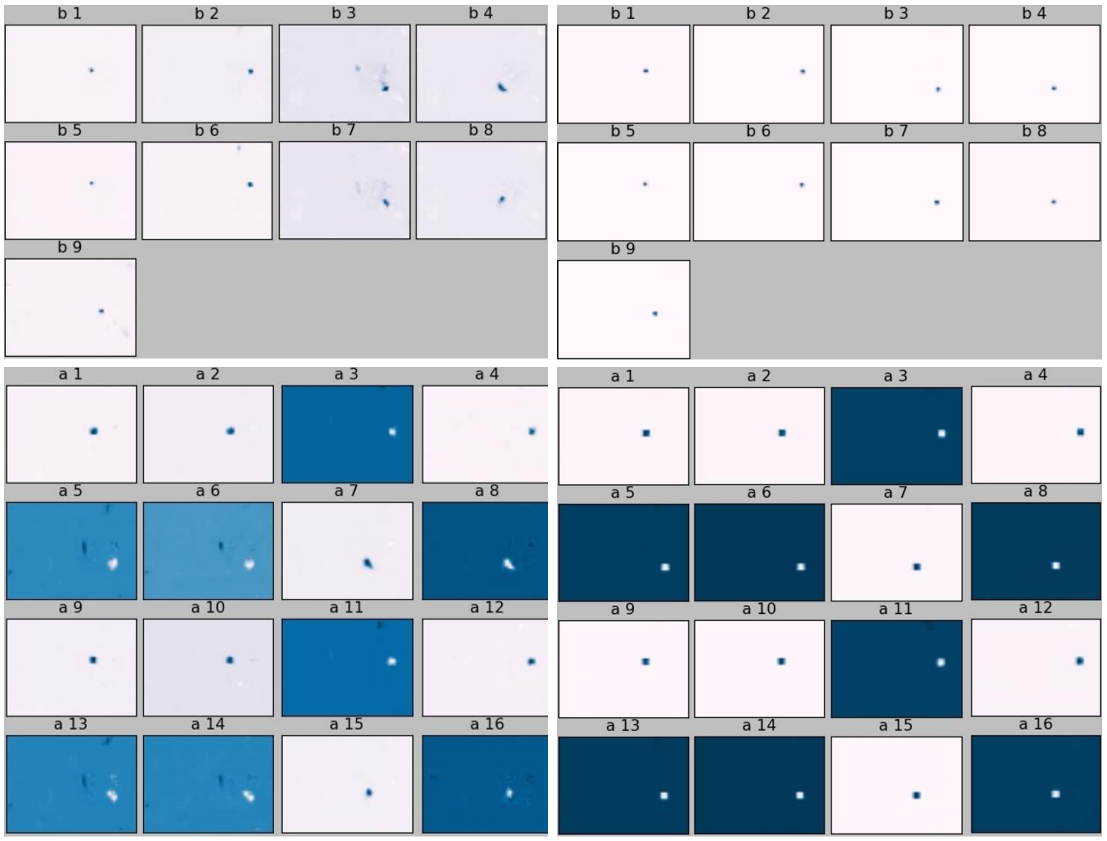
3.4 Experimentations and Results

In this section, the experiments are carried out in simulation and real-world settings. The goal is to execute a real-time tracking test of autonomous mobile manipulation in the real-life environments. Several tests of autonomous 3D visual-servoing conducted to show the performance of the entire system. The proposed perception network showed effective and robust findings that are reliably enough to be implemented in long-range mobile manipulator model designed in the previous section. The behavior anticipated during the experiments involves navigating the robot to a desired destination and orienting towards a target object. The target is constantly detected, included its pose estimation information. The controller law (designed in the previous section) is implemented in the experimentations. Training step is not required in the implementation with VS of mobile manipulator. The mobile manipulator robot consists of 6DOF manipulator arm (UR5 developed by *UniversalRobot*) mounted on top of differential mobile base (A200 Husky built by *ClearpathRobotics*). An ordinary single camera was used to capture target object in the workspace.

3.4.1 Deep-Based 6DOF Object Pose Estimation

The single input image is processed, by the deep-ConvNet model that was previously trained, to produce belief maps which hold probability values for each pixel of an image. Higher values represent the target object location (target-keypoints). Perception network detects the 2D keypoints associated with target object defining a bounding box. The detected keypoints include the centroid of target object with vertices of bounding box. A 3D pose of target object is retrieved by utilizing the estimated 2D keypoints, camera parameters, and target object dimensions.

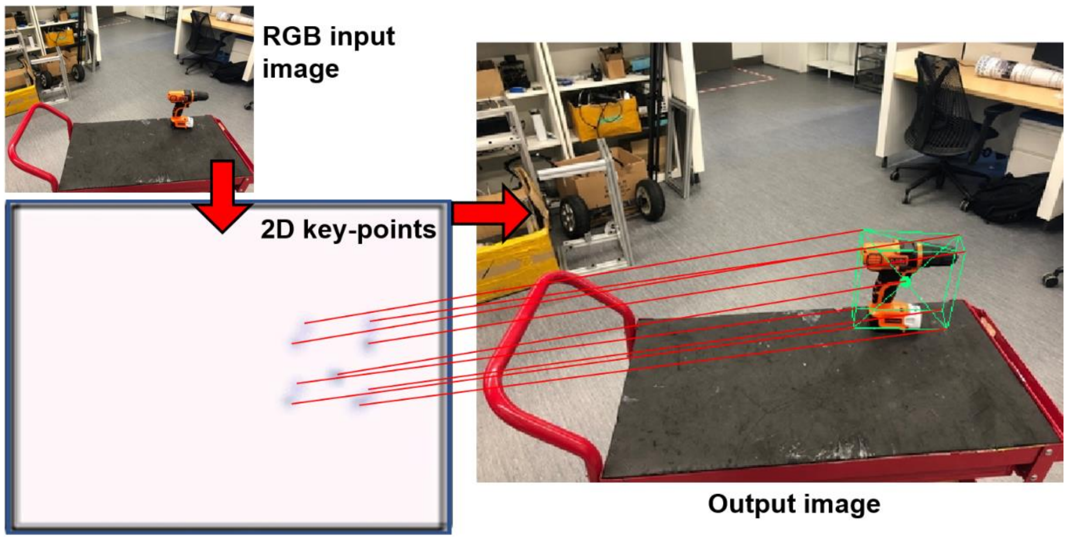
The output of the series of the CNNs indicates 9 vertices of belief maps. In order to extract individual objects from the belief maps, greedy algorithm is used to select the peaks and determine centroids. In addition, affinity map is utilized to find multiple instances of the same object in the image. Belief and affinity maps show how the model improves throughout multiple stages, Figure 3.12 (a) and (b) illustrate the difference between the results of the first and sixth stages respectively. Figure 3.12 (a) (top and bottom) shows the output of 9 vertices of belief maps and output of affinity maps respectively for stage 1. Figure 3.12 (b) (top and bottom) shows how the belief maps are improved after 6 stages. As mentioned earlier, increasingly larger receptive fields cover more context and resolve the problem of ambiguity that clearly occurs in the early stages.



(a) (b)

Figure 3.12: Belief maps (top) affinity maps (bottom) for: (a) stage 1; (b) stage 6.

As a result of feature extractions step, feature map is fed to a series of CNNs that output belief map tensor. Each belief map tensor represents one of each 9 vertices of the 3D bounding boxes. Figure 3.13 (a) illustrates the combination of the 9 vertices (of the final stage) that can form bounding cuboid as well as one belief map for the centroid, key-points are produced from a single input image. Similarly, a process works simultaneously to infer multiple instances of the object. Figure 3.13 (b) is the output final image with the combined vertices and bounding box.



(a) (b)

Figure 3.13: Single input image and 2D key-points: (a) Nine vertices of the final stage. (b) Output image with bounding box.

The entire execution time of the perception network is reported in Table 3.2, including object detection and pose estimation for different number of stages operating on seven different objects (drill, mug, banana, scissors, meat can, marker, and mustard). The average of the execution-time of the entire network is about 0.24 sec.

Table 3.2: Speed network performance (sec.) for different stages on seven different objects.

Target objects	S-1	S-2	S-3	S-4	S-5	S-6
Drill	0.019	0.0424	0.062	0.079	0.098	0.182
Mug	0.05	0.0901	0.124	0.166	0.208	0.248
Banana	0.052	0.089	0.13	0.168	0.209	0.249
Scissors	0.05	0.094	0.125	0.167	0.21	0.261
Meat can	0.063	0.092	0.174	0.167	0.224	0.25
Marker	0.054	0.094	0.136	0.17	0.217	0.258
Mustard	0.048	0.089	0.125	0.169	0.209	0.239

Table 3.3 illustrates the processing time to find 2D projected points (CNN-based), retrieving target pose, and total perception network time for 6 different target objects. Total processing time varies from 0.18 to 0.25 sec to detect and estimate 6DOF pose of target object.

Evaluation metric during multi-stages was calculated to find the accuracy performance. Average-distance is the difference between the ground truth and the estimated key-points from the perception network. Table 3.4 documented the findings of the average-distance of seven different target objects. Table 3.4 shows how the accuracy improves throughout the stages. Where later stages resolve the ambiguities and result accurate performance with lower distances between the centroids of ground truth and estimated 2D key-points. The rate of the average-distance (after the final stage) is around 1.66 mm.

The accuracy-threshold for the robotic manipulation was measured experimentally to find the necessary level of accuracy for grasping purposes. Accuracy-threshold was found around 15 mm, by calculating the difference of centroids between ground truth and estimated points, using our robotic system (UR5 manipulator, Husky mobile-base and 2-finger gripper from RobotiQ).

Table 3.3: Processing time of the perception network, including 2D estimated key-points and retrieving target object pose for six different objects.

Target objects	Processing time (sec) to find 2D key-points CNN-based.	Processing time (sec) to retrieve target object pose.	Network elapsed time (sec) after 6 stages.
Drill	0.0032057	0.00031208	0.18202
Banana	0.0045201	0.0003559	0.25114
Scissors	0.004116	0.0004518	0.25149
Mug	0.004416	0.000319	0.27156
Mustard	0.004464	0.00033807	0.25168
Meat can	0.004755	0.00033497	0.25134

The early stages of generating 2D key-points produce ambiguities and unstable predictions that will be resolved by the later stages. Figure 3.14 illustrates the process of generating 2D key-points throughout multi-stages. Poor performance of the first 3 stages is clearly indicated. However, the last 2 stages provide robust predictions where all target objects are well estimated by the final stage image.

The poses estimation of the target objects is further examined in difficult unstructured backgrounds. Figure 3.15 demonstrates the performance of pose estimation operating on seven target objects with multiple poses in unprepared lab environment and difficult backgrounds.

Another round of testing was necessarily carried out to show the network performance in various of light conditions. Figure 3.16 displays the estimated poses of seven different objects in various illuminations. As seen in the figure, a light source was used closely to the target objects to disturb the image view and examine the perception capability in such lighting situations. The perception network performs robust and stable predictions of estimated poses of multi target objects.

Table 3.4: Average-distance (mm) between the projected points and ground truth of seven different objects during different processing stages.

Target objects	S-1	S-2	S-3	S-4	S-5	S-6
Drill	51.21	22.49	9.366	6.66	3.587	1.462
Mug	91.15	18.79	11.243	3.933	1.934	0.949
Banana	92.04	26.93	17.129	7.725	4.525	1.841
Scissors	88.61	37.48	8.852	3.94	1.716	1.148
Meat can	117.7	33.56	14.298	9.722	6.084	2.61
Mustard	82.51	23.29	12.879	9.784	4.7147	1.907
Marker	94.52	37.85	22.98	13.01	5.58	1.717

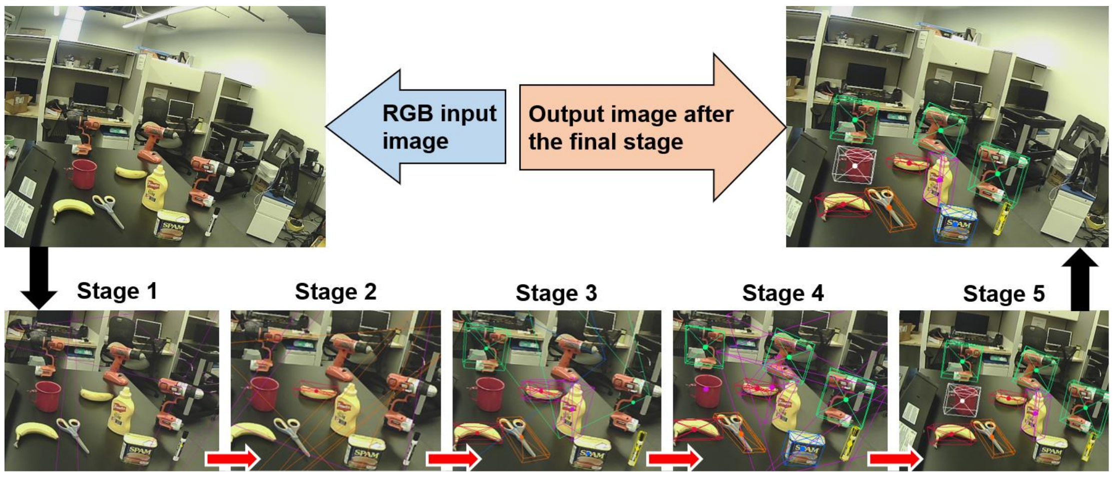


Figure 3.14: The process of 2D key-points predictions throughout multi-stages.

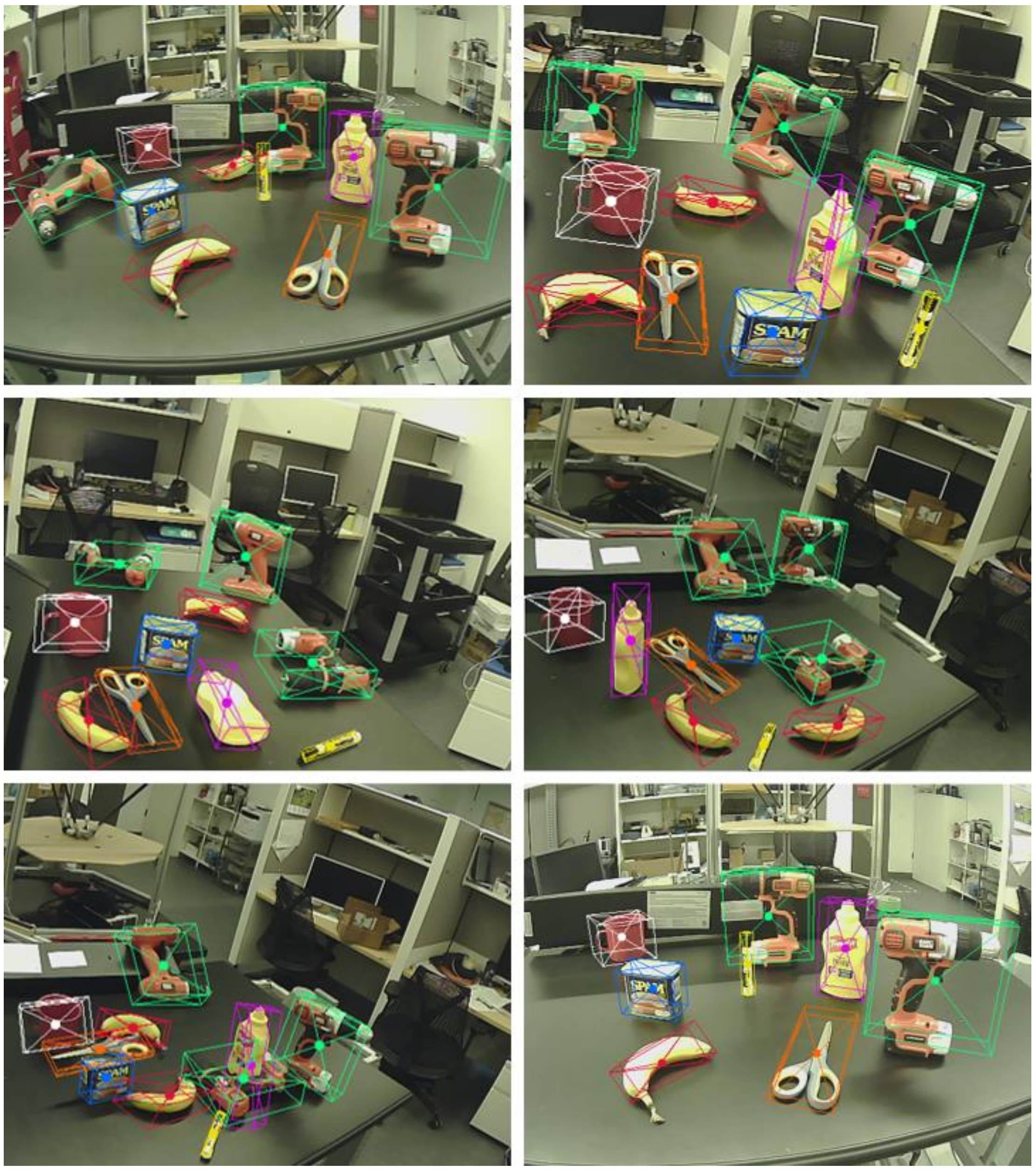


Figure 3.15: Real-world performance of the perception network in difficult backgrounds operating on seven different objects.

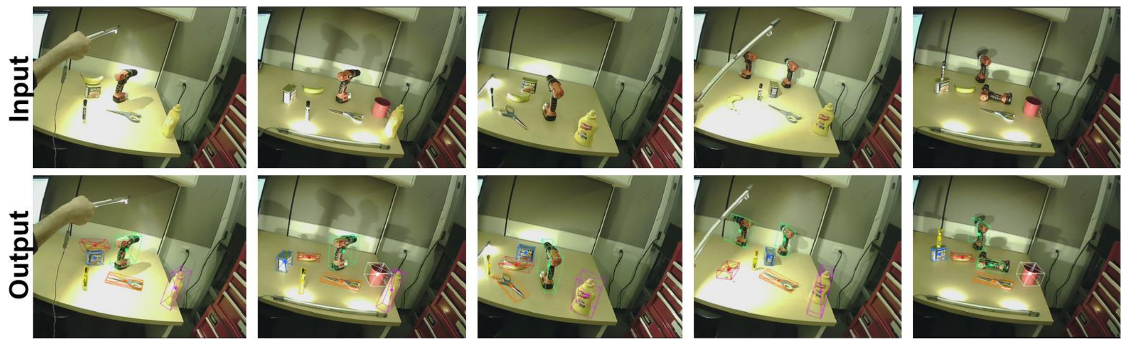


Figure 3.16: Perception network performance during different lighting conditions.

The perception network is able to estimate the object's pose even when part of the object is invisible. Figure 3.17 shows instances of occlusions on different target objects. Object poses are well estimated even though objects obstruct each other randomly.

The ultimate tests of the proposed perception network have presented a sufficient model accuracy that is able to achieve 3D visual servoing task implemented in long-range mobile-manipulator robotic system. Next section presents the incorporation of the findings from the perception network model to the visual servoing system (designed in the previous section).

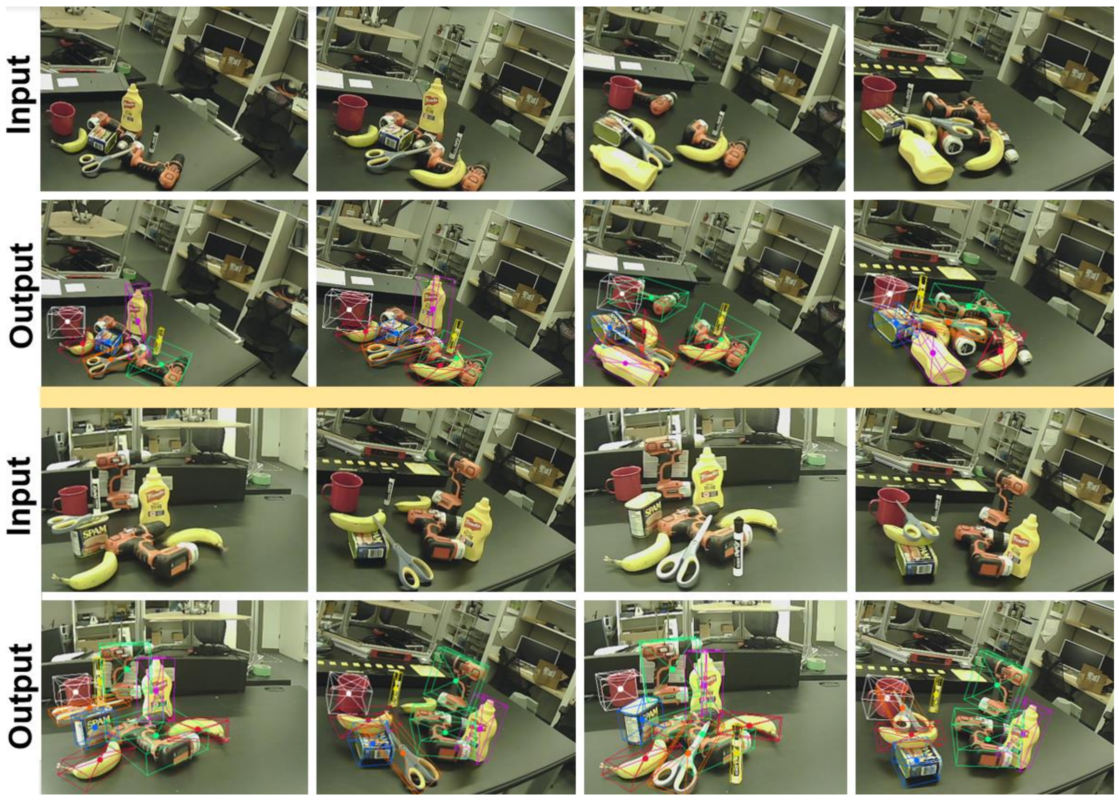


Figure 3.17: Perception network performance with occlusion events, seven different objects are tested at the same time.

3.4.2 3-D Visual Servoing and Deep-ConvNet

After examined the perception network, a completed autonomous mobile manipulation system should be implemented to extract the performance characteristics. The experiments of an entire AMM were performed in simulation environments and real-world settings utilizing the visual information from the perception network model.

i. **Simulation environment**

Figure 3.18 demonstrates the experimental setup prepared in Gazebo (a 3D robotics simulator), an object placed on the table is the desired pose of the robot end-effector. Frames of target object, camera, world-base, and robot end-effector are processed by the robotics middleware ROS (Robot Operating System), and all frames are exhibited in 3D visualization tool called Rviz.

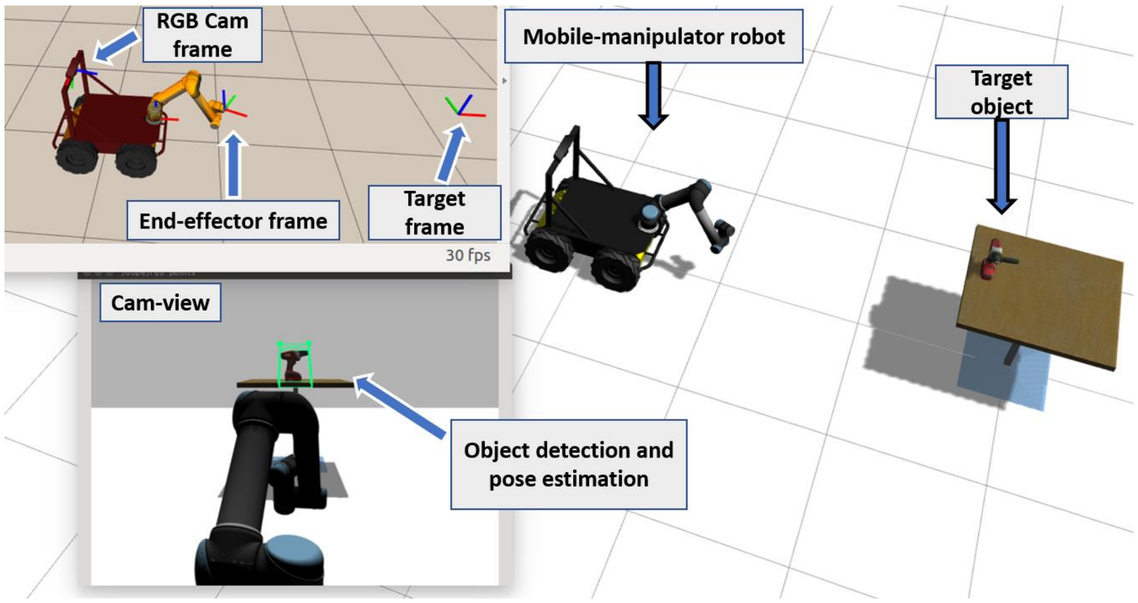


Figure 3.18: Experimental setup in simulation environment.

Multiple rounds of tests were carried out to show the performance of the entire system. A straight test is where the target object was placed in front of the robot. The end-effector of the robot moved as expected towards the detected target. Likewise, another test was performed but deliberately positioned target object at an angle to cause a curved trajectory for the end-effector.

ii. Real-World Settings

Figure 3.19 shows the real robotic system used for experimentations carried out in the lab environment. Husky mobile-base and 6DOF UR5 manipulator, mounted on the top of the mobile robot, were used with uncostly single camera placed at the end-effector. In terms of camera installations, a single RGB camera was used to test two different configurations. First, the eye-in-hand configuration illustrated in Figure 3.19, where target object and camera are both moving simultaneously. Second, eye-to-hand configuration

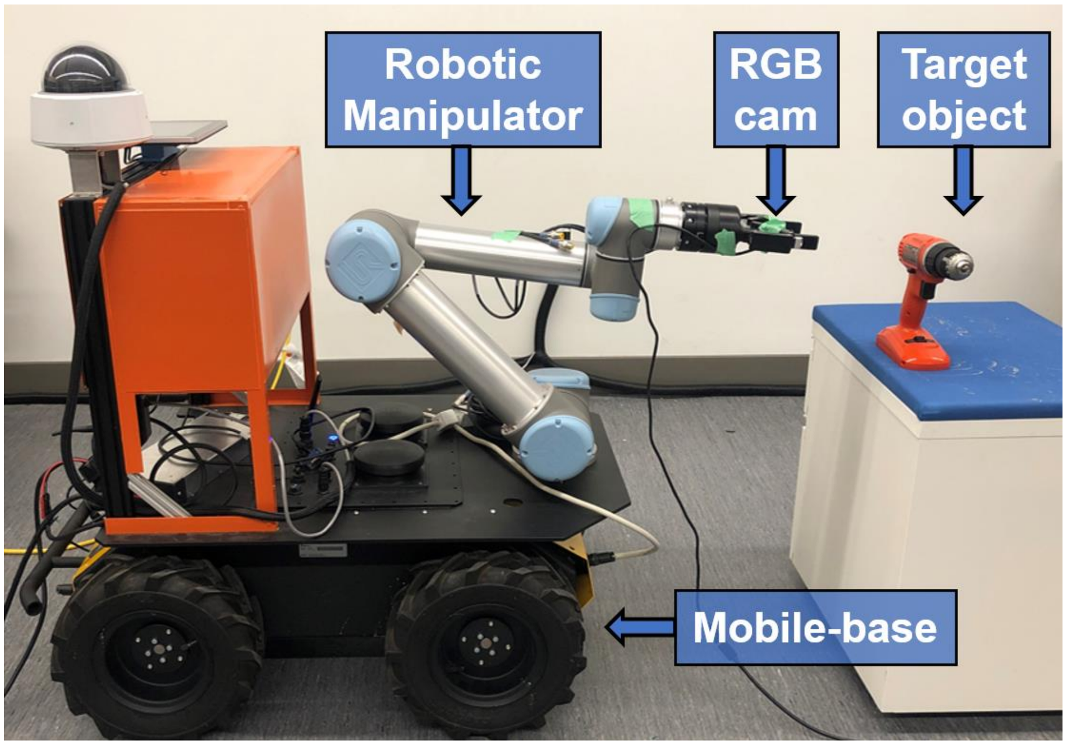
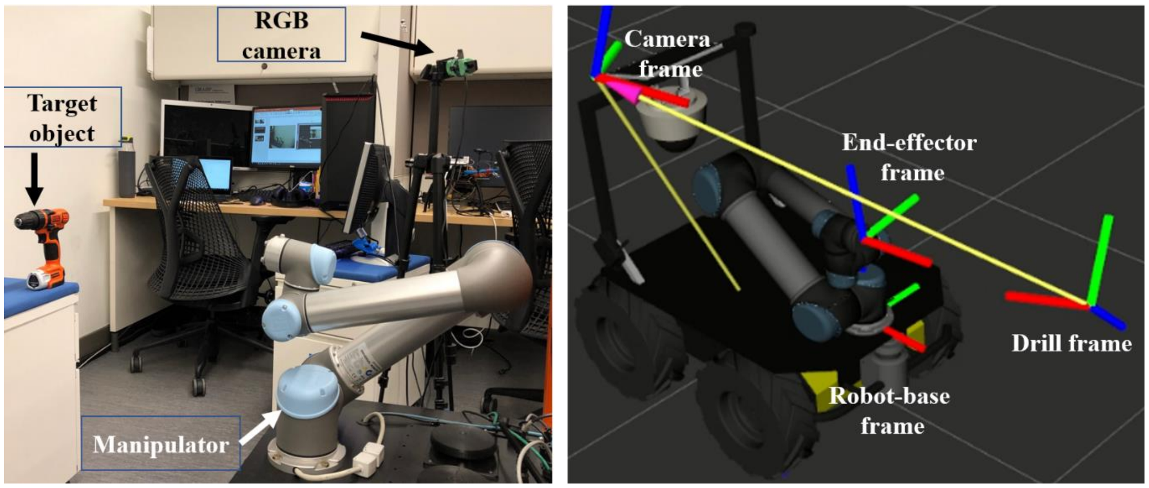


Figure 3.19: Real robotic system used for experiments in unprepared lab environments.

where the camera is fixedly placed in the workspace in which can observe both the target object and the robot, as shown in Figure 3.20.

In Figure 3.20 (a), the physical implementations, and the testing setup that were occurred in unprepared lab environment, are represented in the Robot Operating System (ROS) framework to accomplish an autonomous operation. Figure 3.20 (b) demonstrates the frames of interest, namely robot-base, camera, end-effector, and target-object frame, all are exhibited in ROS 3D visualization tool (Rviz).



(a) (b)

Figure 3.20: (a) Experimental setup (eye-to-hand); (b) ROS 3D visualizer (Rviz).

In order to test the visual-servoing controller, designed in the previous section, we first implemented the PBVS control law on the mobile platform. Figure 3.21 describes two separate tests (straight and steering) conducted to evaluate the performance characteristics of the system.

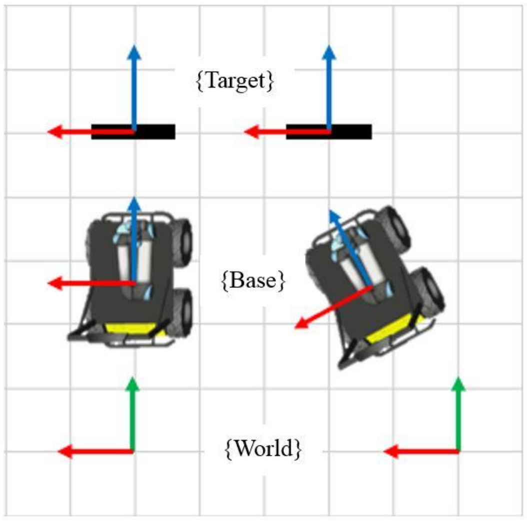
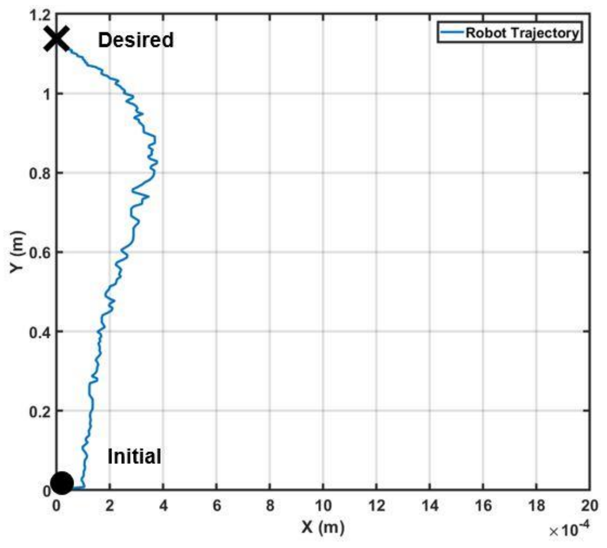


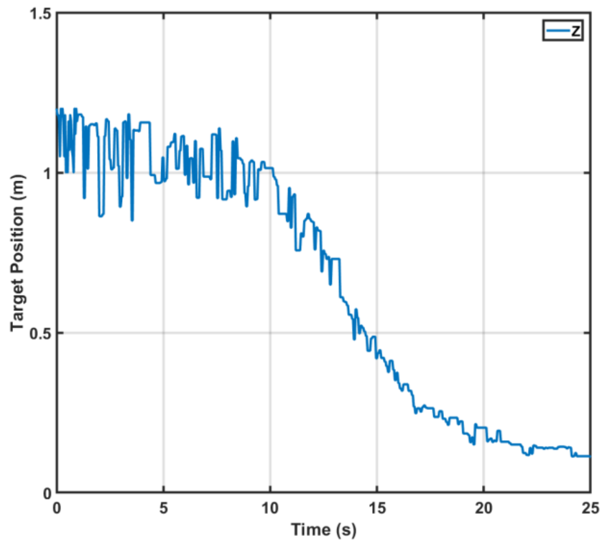
Figure 3.21: Autonomous mobile regulation test. Straight Test (left), Steering Test (right).

In the straight test, the target object is placed at approximately 1.2 meters in front of the mobile robot. Figure 3.22 (a) illustrates the robot's trajectory as it expectedly moves towards the target object in the world frame. From this figure, the robot deviates to the right by a negligible magnitude. This insignificant deviation is due to the result of the target object being handheld at a non-rigid position.

From the acquired image data, the distance between the target object and the camera is estimated through the object pose estimation step. This is shown in Figure 3.22 (b) where the depth value decreases from approximately 1 to 0.2 meters over a total 25 second operation time. The results in Figure 3.22 (b) also illustrates the control system's ability to



(a)



(b)

Figure 3.22: Real-time performance during the straight test, (a) robot trajectory and (b) target object position.

decrease the error between initial and desire pose of the target object as it was able to move the robot towards its goal.

The linear and angular velocities of the mobile robot are represented in Figure 3.23. The maximum achieved linear velocity of the robot is approximately 0.09 m/s while the angular velocity is negligible in this scenario.

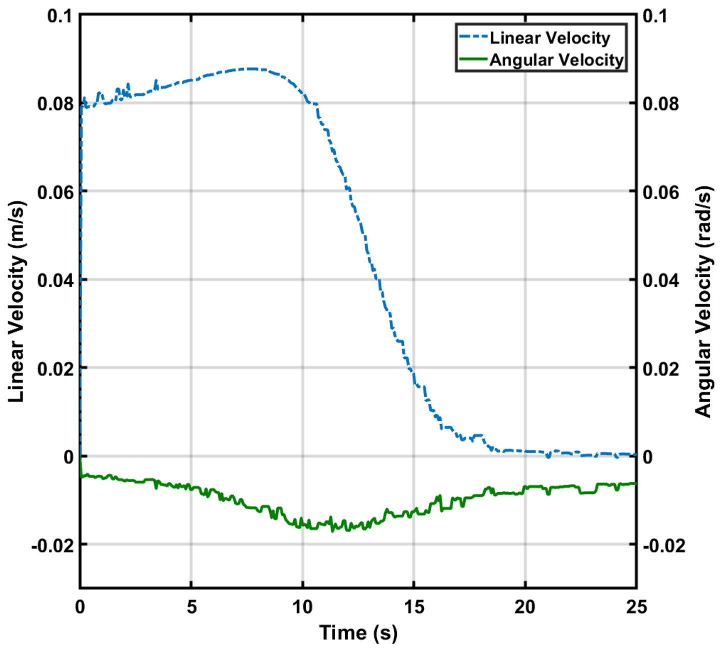
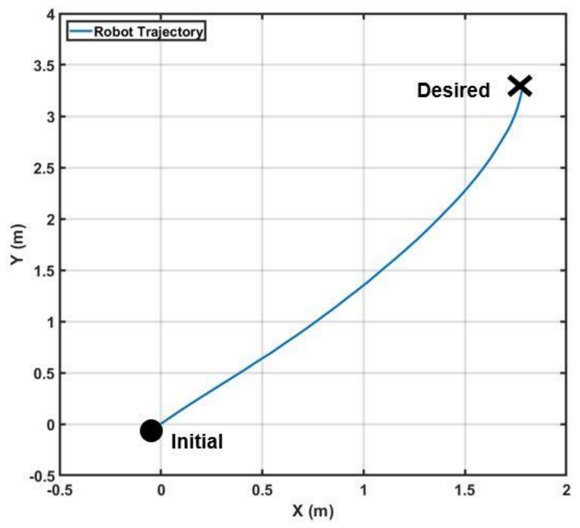
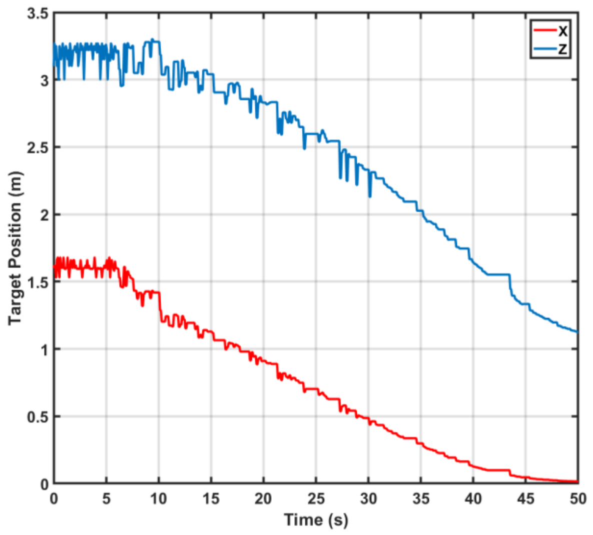


Figure 3.23: Linear and angular velocities during the real-life performance, straight test.

For the steering test, the initial pose of the mobile base origin coincides with that of the origin of the world like the Straight test. The target object is placed at approximately 1.75 and 3.5 meters in the x and y direction; respectively, in the world frame. The robot’s trajectory in the world frame is illustrated in Figure 3.24 (a). Unlike the previous test, there is a significant change in robot orientation as it steered towards the target.



(a)



(b)

Figure 3.24: Real-time performance during the streering test, (a) robot trajectory, (b) target object position.

The target object position relative to the camera frame is shown in Figure 3.24 (b). To start, the object is estimated to be approximately 3.5 meters in the positive z direction relative to the camera. This distance decreased over the span of the navigation time to approximately 1 meter. Additionally, the object also moved towards the middle of the frame as it decreased from 1.5 to 0 meter.

Figure 3.25 illustrates the linear and angular velocities of the mobile robot over the total operation time of 50 seconds. The maximum linear velocity reached is comparable to the Straight test while the angular velocity is more significant.

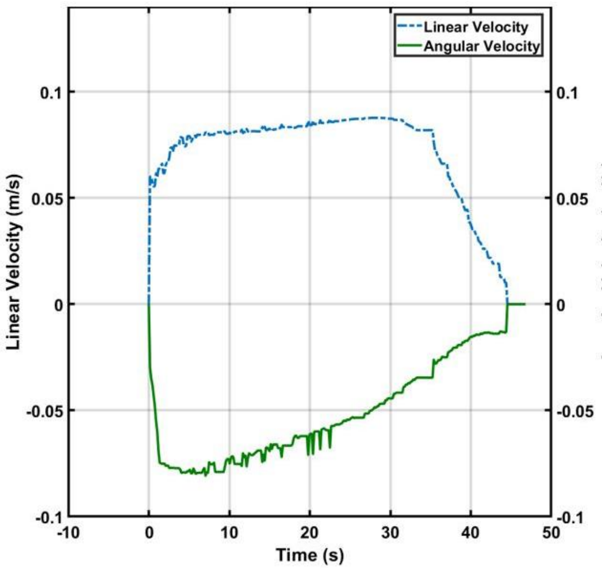
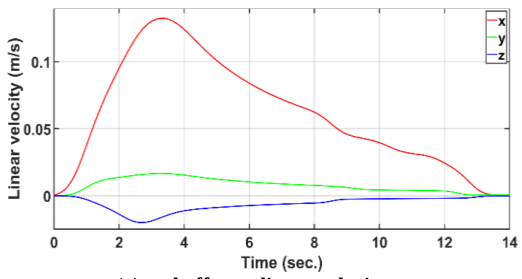
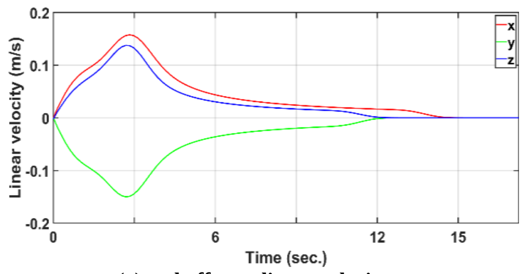


Figure 3.25: Real-time performance, linear and angular velocity during steering test.

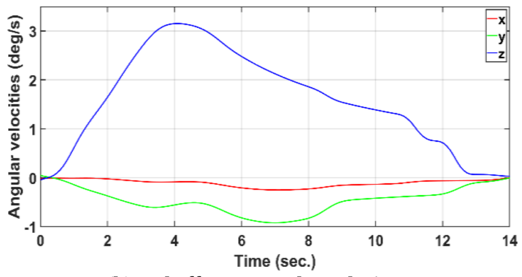
After estimating the 6DOF pose of the target object, the execution network operates autonomously to achieve a complete task of deep-based and long-range autonomous mobile manipulation system (manipulator arm and mobile platform). Without the need to post-refinements of the estimated pose of a target object. The performance of the autonomous manipulation system was demonstrated by two rounds of tests. First, a straight test in which the target object was located in front of the robot. The robot's end effector moved towards the detected object as anticipated. A second test was also conducted, but the target object was purposely placed at an angle to allow the end effector to have a curved trajectory. More measurements are extracted during the test experiments (straight and curved) and shown in Figure 3.26. The linear and angular end-effector velocities of both experiments are shown at (a) and (b) of both sides of Figure 3.26, correspondingly. Similarly, on both sides, errors of position and orientation are shown in (c) and (d), respectively, representing the measured errors between the end-effector and the target object during the time of operation. The error minimizes when the end-effector moves close to the object and becomes within the user-defined safety distance. The 3D trajectory of the end-effector is drawn in part (e) of both sides of Figure 3.26 .



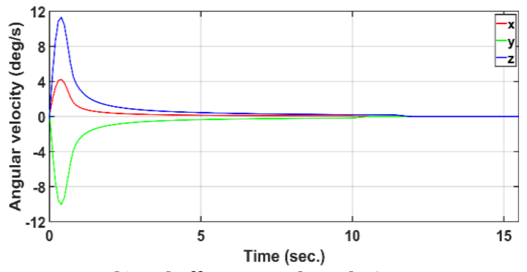
(a) end-effector linear velocity



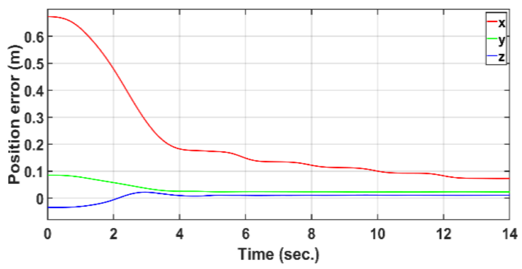
(a) end-effector linear velocity



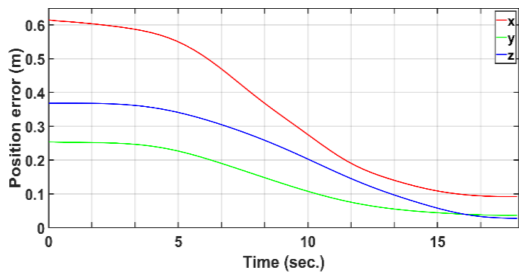
(b) end-effector angular velocity



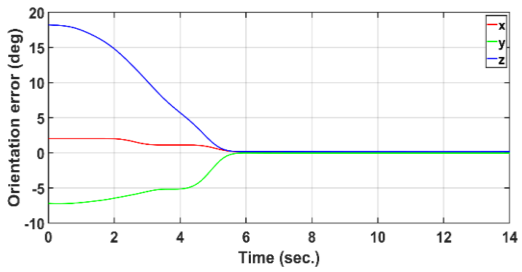
(b) end-effector angular velocity



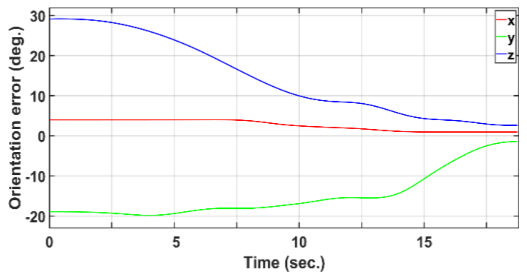
(c) position error



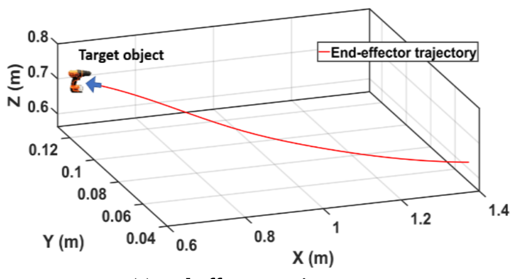
(c) position error



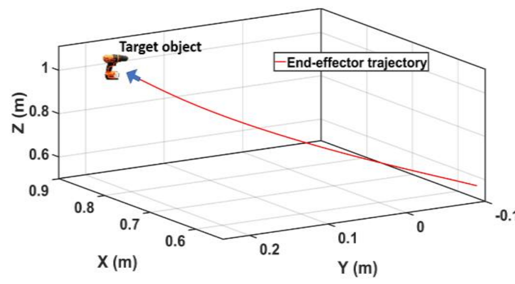
(d) orientation error



(d) orientation error



(e) end-effector trajectory



(e) end-effector trajectory

Figure 3.26: Results of real-world experiments demonstrate velocities, pose error, and 3d end-effector trajectory during straight (left-side) and curved tests (right-side).

Tracking test in the autonomous mobile manipulation. The proposed system does not require any type of synthetic marker or quick response tag assistance at any stage of experimentation. Tracking test was required to invistigate the physical capabilities of the proposed AMM system. Figure 3.27 illustrates an instance of the tracking test scene where Drill object was used as a target object. Figure 3.27 (a) shows the frames of interest presented in Rviz visualizer, (b) indicates the estimated pose of the target object, (c) is the third person view covers the used robot with the target object. Eye-in-hand camera installation was considered during the tracking test. This shows a stable and robust performance of the entire system even though the target object and camera are simultaneously moving. Video recordings of the experiments have been provided to show the performance of the proposed AMM system.

Prior visual servoing studies [36, 41, 46, 60, 61, 152] requiring i) complex vision-based setup to obtain pose estimation with limited background structure, ii) or training on synthetic and real-world dataset collections including fine-tuning efforts. Thus, after training on synthetic datasets, implementing these methods in new real-world environments is difficult and limited. Comparing to such methods, our system estimates object poses competitively, which are trained only on synthetic data and generalized to physical 3-D pose-based visual servoing implemented in long-range mobile manipulator. The use of domain randomized computer-generated dataset was i) applied in the context of 6DOF object pose estimation from single image ii) and operated in 3-D continuous VS task implemented in AMM. Our results have demonstrated the physical capabilities of generalizing to diverse situations and dealing with the differences in light changings and occlusions.

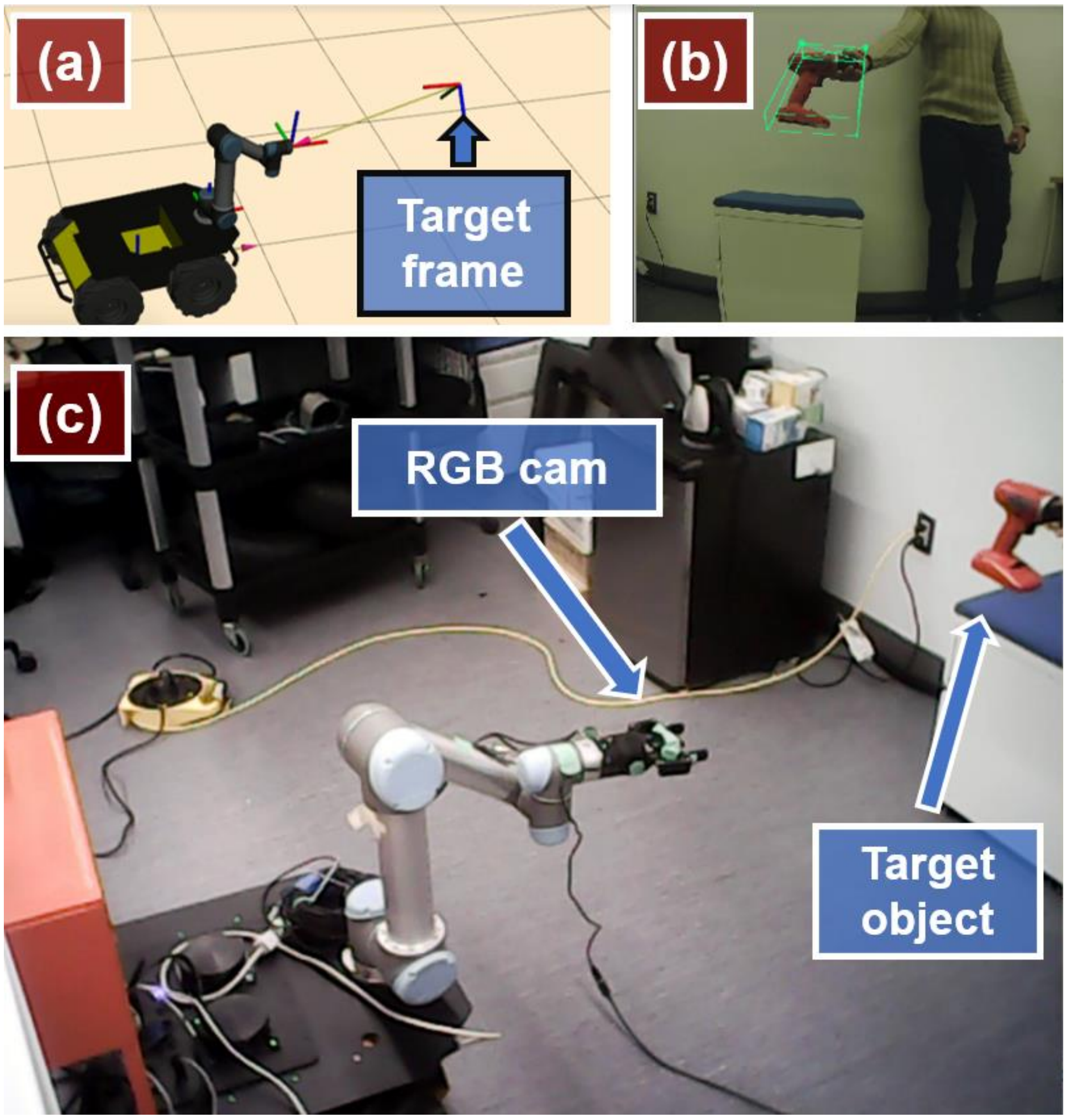


Figure 3.27: Real-world tracking experiments (AMM): (a) frames of interest exhibited in the 3D visualizer (ROS-Rviz), (b) detected target object with its pose, (c) mobile-manipulator robot.

3.5 Summary

This chapter set out to develop an autonomous 3D visual servoing system based on deepConvNet, implemented in a sophisticated mobile manipulator system and utilizing single RGB image. Two main steps construct the entire system: first, perception network to detect and estimate the pose of objects in 3D space. Using an effective (deep-CNN and pose estimation algorithms) model architecture. Second, the pose estimation data was then used in a 3D visual servoing scheme to control the motion of AMM system.

Perception network was entirely trained using computer-generated RGB images, depth images and segmentations are not required. The system was, then, generalized successfully into real-world environment without fine-tuning or extra re-training. The results of deep ConvNet showed that it was sufficient to be trained using synthetic datasets, then implemented effectively for real-world robotic manipulation purposes. Besides simulation experiments, the proposed system was physically tested (on 6 DOF manipulator arm mounted on differential robot-base) to extract the performance characteristics of the robot model. The findings of experimentations have resulted in a robust and continuous 3D visual-servoing operation of AMM with handling occlusion and light variations. In terms of future work, it would be important to carry out further studies with the goal of applying object grasping and targeting transparent objects.

Chapter 4. Learn to Grasp Unknown Objects in Robotic Manipulation

4.1 Introduction

Autonomous mobile manipulation task was demonstrated in Chapter 3 along with implementation on long-range mobile manipulator robot. We showed that it is possible to execute a real-time fully autonomous mobile manipulator task based on only synthetic-trained perception network model. The proposed AMM system was oriented to the target objects. Nevertheless, skilled robotic manipulation applications still demand grasping unknown objects.

Grasping unfamiliar objects (unknown during training) based on limited prior knowledge is a challenging task in robotic manipulation. Recent solutions typically require pre-defined information of target objects (e.g., pose estimation, 3D CAD models, or object classification), task-specific training data, or a huge experience data with training time consuming to achieve usable generalization ability. This makes the developed systems difficult to scale up for generalization on novel objects. This chapter introduces a robotic grasping strategy based on the model-free deep reinforcement learning, named Deep Reinforcement Grasp Policy (DRGP). The developed system uses simple geometric objects in training and generalizes efficiently on novel objects. Without requiring any type of prior object awareness or task-specific training data. Our scalable visual grasping system is entirely self-learning approach. It emphasizes off-policy learning method and learns quickly through trial-and-error manner. The model trains end-to-end policies (from only visual observations to decisions making) to seek optimal grasp strategy. A perception network utilizes a convolutional neural network that maps visual observation to grasp-action as dense pixel-wise $\boldsymbol{Q}$-values that represent the location and orientation of a primitive action executed by a robot. After training on limited simulated objects, the gained knowledge is transferred successfully to real-life scenarios with generalization to unknown objects. In the experiments, a 6DOF robot manipulator with a 2-finger gripper is utilized to validate the developed method. The empirical results demonstrated successfully based only on minimal previous knowledge of a few hours of simulated training and simple objects.

This chapter is organized in the following way. The limitations of the related methods, that attempted to address the task with academic value and practical novelty, are introduced in section 4.2. Meanwhile, the focus and contributions of this chapter are demonstrated based on the main challenge presented in the task. The problem statement and agent's objective are formulated primarily in the third section 4.3. The system modelling is illustrated in the fourth section 4.4. Finally, the fifth and last sections (4.5 and 4.6) demonstrate the experimental findings, discussion, and conclusions, respectively.

4.2 Grasping Unknown Objects

Robotic manipulation in a real-world environment is a difficult activity, since the robot should be able to perform an effective grasping task on unfamiliar objects. This calls for human-like perception and reasoning. Previous works in robot object manipulation have specifically sought to address the challenge of grasping in a wide spectrum of methods, starting from analytic task metrics until learning-based concepts. Traditional studies have investigated different methods to recognize target objects and achieve grasping tasks. Researches attempted employing the fiducial marker as a point of reference for the target object in the workspace. Other studies applied traditional computer-vision methods to identify objects using their edges and corners and plan potential grasping points as the controller reference. For instance, model-based and feature-based methods were used for robot-object-interaction tasks. However, there are many empirical issues accompanied by those techniques which can eventually lead to limit the applicability, such as limited to a structured scenario, poor performance with light and occlusion variations and required highly textured objects. In addition, model-based grasp planning requires pre-defined information about target objects (e.g. pose estimation or 3D object CAD models), which makes it limited to address novel objects.

Understanding unknown objects and planning robust grasp strategies based on previous knowledge is a necessary skill for a robotic manipulation system in succeeding the task. Deep neural-network algorithms reported successful results in grasping objects [36, 46, 165]. However, the preparation of training datasets requires effort and skills, as well as deep convolutional network training is computationally expensive and time consuming. In addition, empiric performance should be limited to objects used during training or similar objects with same color and shape attributes.

Robotic-experiential-learning approaches have been promisingly devoted to plan grasping [88, 108, 166] through DRL [167]. Robots could learn tasks progressively by trial-and-error and perform in high precision with least human involvement. The main challenge in learning-based robotic grasping is generalization. Is robot able to learn grasping policies (from little experience) and handle new objects (were not seen before) that have entirely different geometrical shapes? Recent studies in [108, 168, 169] have attempted to improve generalizable perception tests, where they succeeded at grasping new objects. However, it still requires long-time training on a huge dataset of hundreds of different objects in thousands of grasps trials. Contemporary works have addressed learning-based methods with self-supervision for robotic grasping with maximize affordance metrics or grasp stability metrics. However, such methods have been limited to objects variation and simple geometrical shapes, such as cubes. It remains unclear how to explore the generalizing issue for robotic manipulation. Diversity has been widely concentrated [170, 171] on videogame applications [172] and simple simulated robots [173], but rarely applied to sophisticated robotic grasping maneuvers.

In this chapter, a scalable self-supervised framework is proposed to learn grasping control policies from limited raw experience, which is named Deep Reinforcement Grasp Policy (DRGP). The contributions are recapped in two points.

First. DRGP is an end-to-end entirely self-learning approach based on model-free DRL. Our agent (i.e. the robotic manipulator) learns from scratch, from only visual observations to decisions-making, and trains in the form of off-policy Q-learning framework by trial-and-error manner. DRGP is unlike the conventional techniques that require human demonstration and annotation, heuristics, or hard-coded parameters. Our visual grasping system consists of visual network (to extract features from visual observations) and policy network (to map the features into the action space). Both of which train simultaneously. This is different than classic models where perception and policy modules are trained separately.

Second. Our learning scheme can be beneficial to both generalization and transferability without requiring any additional training or fine-tuning. The agent efficiently learns from minimal training time and limited ordinary object in simulation. The gained knowledge transfers to real-world scenarios and generalizes to novel objects. Without the need for both (a) task-specific re-training data or (b) pre-defined object information. This is unlike traditional methods that typically demand massive dataset and training time-consuming. It is hard for classic methods to handle unknown objects in different execution environments without requiring shape primitives or attempting to model object geometry in any way. DRGP is in contrast with prior grasping systems that are restricted to prerequisite or any type of knowledge of objects beforehand (e.g. pose estimation, segmentation, or class categories).

Q-learning is utilized to solve the agent's task by presenting the environment state as pixel-wise Q-maps where each pixel represents a primitive action held on 3D location (included image orientation) captured from the depth sensor. The robot interacts repeatedly with the environment by executing the defined actions. Based on changes that may occur in the environment, the response is given as rewards to the agent. It learns progressively by maximizing future rewards until the environment is resolved. Different action combinations are learned through trial-and-error process. The agent seeks the best sequence until the environment reaches a terminal state.

In this chapter, training phase was achieved in V-REP [174] (3D robot simulation software) simulator using UR5 manipulator and two-finger parallel jaw gripper. The acquired knowledge was then transferred and carried out in physical experiments with novel target objects. Figure 4.1 demonstrates the real-world grasping experiments operated by a robot manipulator with a two-finger gripper mounted at the top of the end-effector. The RGB-D camera captures the workspace image and sends it to the perception network, which infers pixel-wise Q-values visualized as heat-maps. Model-free DRL computes the grasp action with the highest Q-value quality executed by a robot. The following sections of the chapter discuss the formulation, simulation and validation of the proposed architecture need for generalized robotic manipulation applications.

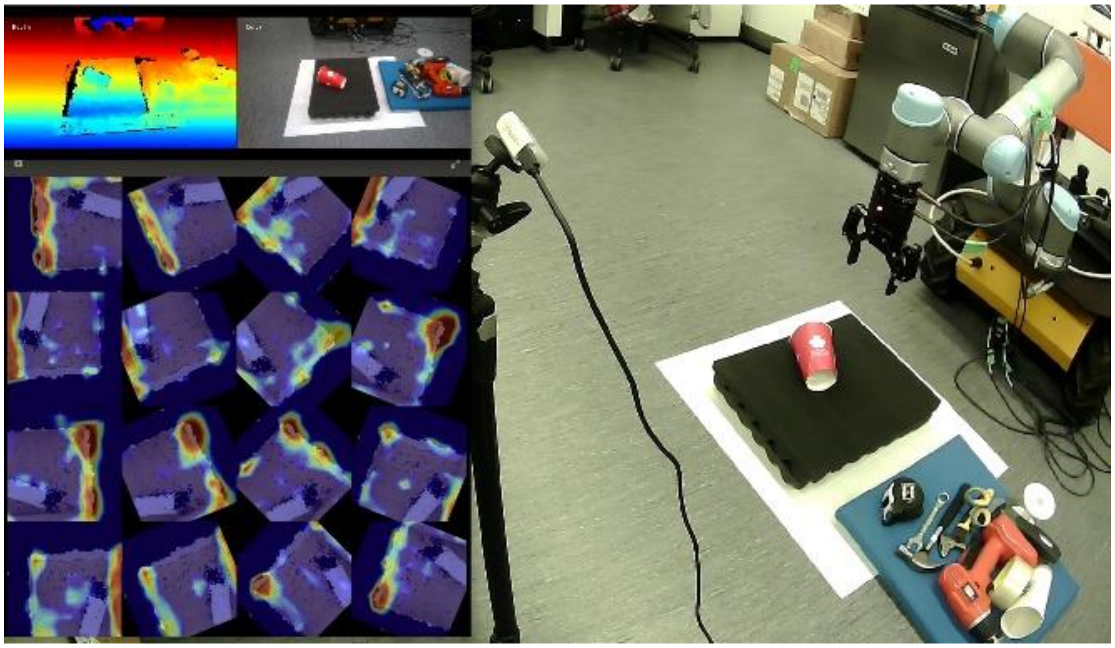


Figure 4.1: Physical experimentation setup in the lab scenario.

4.3 Problem Formulation and Learning

The grasping task in this chapter is formulated as a Markov Decision Process (MDP), where s_t is the state at specific time t, a_t denotes as an action at t and $\pi(s_t)$ is the policy according to $\pi(s|a)$. The robot makes decisions upon a state s_t and executes actions a_t relative to policy $\pi(s_t)$. Thereafter, a robot obtains an instant corresponding reward defined as $R_{at}(s_t\ ,\ s_{t+1})$ and moves to a new transition state s_{t+1}. The return rewards are necessary to improve agent’s understanding by informing which action-state pairs are good. G_t as shown below in equation 4.1 is the total expected rewards gained across all states sequentially.

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \qquad 4.1$$

where $\gamma \in [0, 1]$ is the discount factor.

Policy π is the strategy that the agent could decide which action, upon a current state and a policy, maps the state s_t to an action a_t as shown in equation 4.2.

$$\pi(s|a) = \mathbb{P}[A_t = a \mid S_t = s] \qquad 4.2$$

The agent's objective is to seek the optimal policy π^* which is the decision that selects the best action with the highest quality to maximize the action-value function and sum of the expected return of future rewards, expressed in equation 4.3. Maximization could be achieved by choosing an action a_t (among all possible actions) which includes the highest value in $Q\pi(s, a)$. The deep neural network used to solve the action-value function $Q\pi(s, a) = [G_t|S_t]$ which calculates the quality of any possible action a_t at given state s_t. Figure 4.2 illustrates the schematic diagram of learning interactions between the agent and the environment in a standard RL problem. Representation scene of the environment is observed as a state s_t which is the input to the deep network. The output is the action with the highest quality, which then causes to obtain an immediate reward.

$$Q\pi^*(s, a) = \max_{\pi} Q\pi(s, a) \qquad 4.3$$

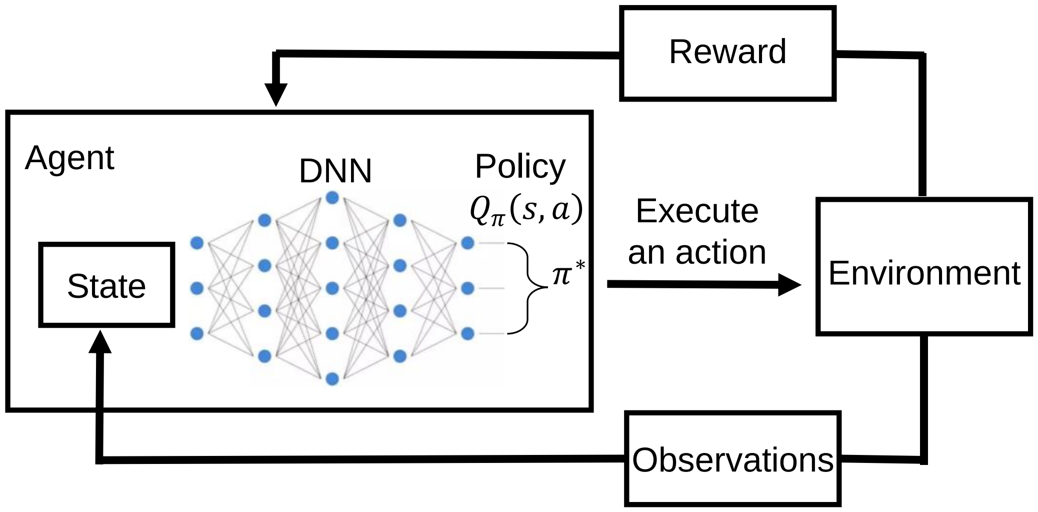


Figure 4.2: Schematic diagram of learning interactions between the agent and the environment.

In Q-learning, the target policy is greedy according to $Q(s, a)$ which selects the highest Q-value, the target policy is shown below:

$$\pi^*(s,a) = \begin{cases} 1, & if \;\; a = arg \;\; \max_{a} \;\; Q\pi^*(s,a) \\ 0, & otherwise \end{cases} \qquad 4.4$$

For the learning strategy of computing optimal policy, we implement off-policy Temporal Difference (TD) method to update action-value function for each action in a given state towards the estimated return TD-target. The ε-greedy policy is utilized here for behavior action to fulfill the need for balancing between exploration and exploitation. Unlike the Greedy-Deterministic learning method that precludes beneficial states whose values are unavailable to be discovered. Similar to Deep Q-Network (DQN) implementation [175], deep Q-learning composes the task into action selection and action evaluation. Target-network uses a greedy policy to calculate $Q(s, a_{-i})$ for each possible action a_{-i} at given state s_t and find the highest value of $Q(s, a_{-i})$. The right side of equation 4.5 is the TD-Target. Which is the sum of the instant reward $r = R_{at}(\, s_t\,,\; s')$ given to the agent at current state and the discounted Q-value, where $a' = a_{t+1}$ is the action for the next state $s' = s_{t+1}$.

$$y_i^{DQN} \;= r + \gamma Q\left(s', arg \max_{a'} \;\; \left(Q(s', a')\right)\right) \qquad 4.5$$

Learning objective is designed in equation 4.6 as a minimization of the distance between $Q(s_t, a_t)$ and TD-Target. This objective iteratively minimizes the temporal difference error L_i of $Q(s_t, a_t)$ to the target y_i^{DQN}.

$$L_i = |\; Q(s_t, a_t) - \; y_i^{DQN} \;| \qquad 4.6$$

The temporal differences for any possible action-value $Q(s_t, a_t)$ are computed during the TD-learning method (i.e. differences of two values of $Q(s_t, a_t)$, before and after executing an action a_t at state s_t). Then, TD is used to update the value of $Q(s_t, a_t)$, until the action-value $Q(s_t, a_t)$ converges to its true value. The proposed learning algorithm is summarized in the pseudocode below.

TD-learning method Pseudocode

```
1:Initialize $Q(s_t, a_t), \forall \in \mathcal{S}, a \in \mathcal{A}(s), arbitrarily,$
    and $Q = 0$
2:Repeat (for each episode):
3:      Initialize $s_t$
4:      Repeat (for each step of episode):
5:          Select $a_t$ at given $s_t$ according to $\pi^*$:
                (greedy deterministic policy for TD-target, and
             $\varepsilon$-greedy for behavior action)
6:          Execute $a_t$, and observe: $r, s_{t+1}$
7:          Calculate TD-learning:
8:          $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r + \max_{\alpha}, Q(s', a') - Q(s_t, a_t)]$
9:          Apply $y_i^{DQN}$ and minimize the TD-error $L_i$
10:     $s_t \leftarrow s'$;
11:  Until $s_t$ is terminal
```

4.4 System Modelling

This section illustrates modelling of the proposed grasping system, including state representation, perception network, and action rewards and training protocols. Figure 4.3 demonstrates the overall self-learning framework starting from visual observations functioning as the state representation. The perception network calculates Q-functions at the pixel-level and finds the grasp-action with the highest quality, which then executed by a robot manipulator.

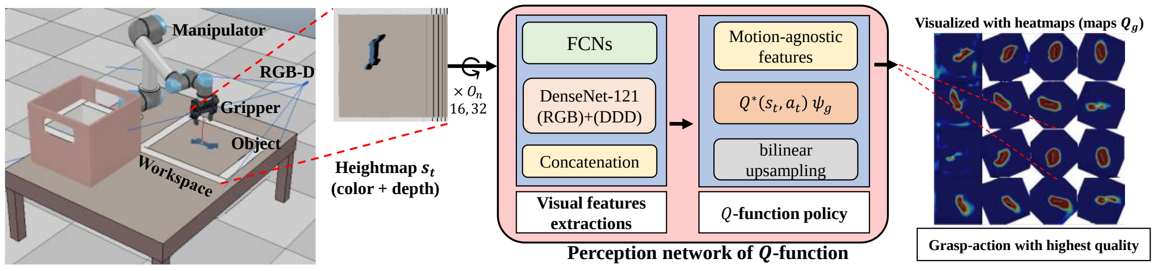


Figure 4.3: Overview system architecture of the proposed DRGP.

4.4.1 Visual Observations

Each state s_t of the environment (at a specific time t) is modelled as a heightmap representation image, which is constructed from RGB-D image by projecting the data onto 3D point cloud [176]. Next, the heightmap representation image (color and depth heightmap) is generated through orthographic re-projection the point cloud data along with the gravity direction with a known extrinsic parameter of the camera. Robot's workspace ($0.4^2 m$) is defined as a pixel resolution of 200×200. Therefore, the spatial input pixel represents $0.002^2 m$ vertical column of heightmap in the 3D workspace.

4.4.2 Perception Network

State representation is fed to the perception network of Q-function, which is designed as a feed-forward fully convolutional neural network (FCN) [177] that extracts the visual features. We train FCN to predict dense pixel-wise Q-maps of future expected returns for all pixels of a state representation. FCN takes the heightmap image as an input and predicts visual affordances of dense-pixel-wise Q-values as outputs. FCN and Q-function policy visualize the heat-maps at different pixels, each one parameterizes a primitive action. Each pixel represents a confidence value of location on which to hold grasping.

FCN architecture has two parallel 121-layer DenseNet [178] pre-trained on ImageNet [159]. One DenseNet takes as input the RGB color image and the second one takes the depth channel DDD, both from the heightmap image representation. The features extraction step is concatenated to produce motion-agnostic features as the inputs to a Q-function policy network. Which contains 3-layer convolutional network and bilinear upsampling. For each episodic epoch during training, the weight and bias model of the entire training network are saved with reasonable memory cost (less than 150 MB).

4.4.3 Primitive Action

The expected robot's behavior known as primitive actions ψ_g, grasping. The motion primitive behavior defines the action a_t at specific state s_t. As shown in equation 4.7, p determines the 3D location of the executed action a_t.

$$(a_t) = (\psi_g, p) | \in \{grasp\}, p \twoheadrightarrow pixel \in s_t \qquad 4.7$$

To learn oriented action primitive, the process of providing state representation to FCN iterates with different orientations of heightmap at O_n. This determines the orientation of grasping. Various rotated heightmaps are provided to the FCN to encompass varying degrees of grasping. The middle point of the top-down tip of the gripper is represented by p at O_n. We tested O_n as 16 or 32 for better performance, the rotated image results multiples of oriented action at degree $22.5°$ or $11.25°$, respectively.

The Q-function network concatenates the output from the visual features network (x, y, O_n) (target object coordinates) and the execution parameters p (on which to execute the action) to form the state of the policy $s_t = (x, y, O_n, p)$. By utilizing fully connected layers, the Q-function policy maps the action distribution over current state representation to sample the optimal action, $a_t \sim \pi(s_t|a)$. Candidate primitive action is the pixel-wise with highest Q-value comparing to other pixel-wise predictions at a current state, $\arg\max_{a'} \left(Q(s_t, a')\right) = \arg\max_{\psi_g, p} \left(grasp(s_t)\right)$.

4.4.4 Q-Function

The policy is a deep reinforcement agent that takes the extracted visual features and infers the optimal action. We implemented Q-function policy which is favorable to a high dimensional robotic control problem. The goal of the agent is to learn the action-value function which estimates the expected returns for grasping action. The agent is represented as Q-function approximator of FCN which is effective for pixel-wise computations. Each state representation s_t enables Q-values for all possible actions $(200 \times 200 \times 16 =$

640×10^3) for ($O_n = 16$). Q-values prediction map combines self-learning approach with visual affordances for grasping actions. Q-value predicts the future expected reward given to the agent for specific grasping ψ_g at p upon a state s_t, $p \twoheadrightarrow pixel \in s_t$. Primitive action ψ_g executes (p, O_n) at s_t and receives a corresponding immediate reward R_{at}.

The figure below demonstrates the steps to begin learning the task. The agent first initializes a Q-table with the dimensions defined by the n columns, n denoted as expected actions, and m rows which are the number of the states. Each pixel of the state representation input image holds confidence data of Q-value that represents primitive action. Which is eventually performed by the robot. Each executed action may result changes (to the current state) that produce rewards. Returned reward values should be utilized to update the Q-table. This on-line learning style continuously upgrades and revises the Q-table based on previous experience. In this way, the robotic agent is capable to adapt slight changes and comprehend different scenarios.

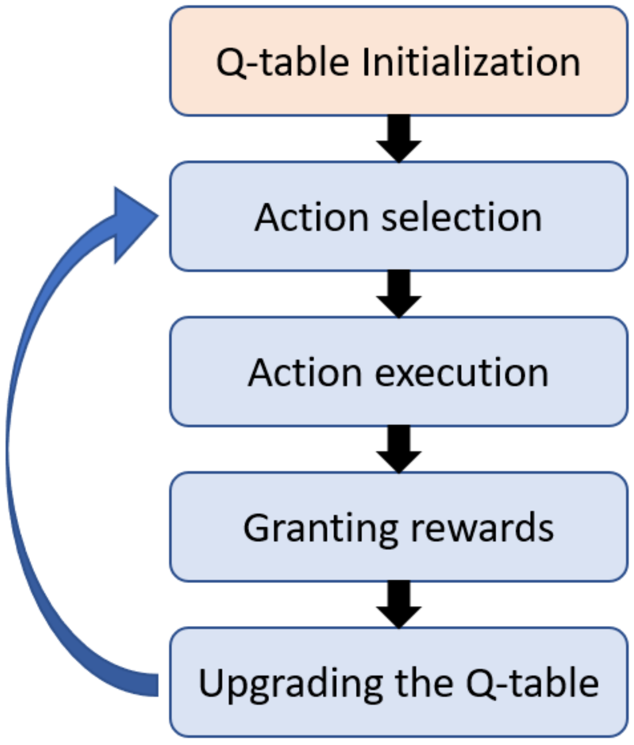


Figure 4.4: Agent's learning process.

4.4.5 Rewards and Training

Rewarding policy for our model-free DRL is simply designed as $R_{\psi_g}(s_t ,\ s_{t+1}) = 1$, when a grasping attempt is completed successfully (defined by thresholding the distance between the robot fingers after grasping).

For the training phase, the iterative optimizing method by the stochastic gradient descent used here to train FCN with learning rate 0.0001 and weight decay 2^{-5}. The agent was trained by minimizing the temporal difference error L_i (as given in equation 4.6) based on Huber loss function at each iteration i, as shown below:

$$\mathcal{L}_{Li} = \begin{cases} \frac{1}{2}\left(Q^{Qi}(s_i, a_i) - y_i Q^{Q_i^-}\right)^2, for \ \left|Q^{Qi}(s_i, a_i) - y_i Q^{Q_i^-}\right| < 1, \\ \left|Q^{Qi}(s_i, a_i) - y_i Q^{Q_i^-}\right| - \frac{1}{2}, otherwise \end{cases} \qquad 4.8$$

where Q_i is the network parameters at iteration i, the target parameters denote as Q_i^-. The system has been implemented on PyTorch platform [160] and trained on GPU support (Nvidia RTX 2080 Ti) for a few thousands of attempts, as seen in section 4.5. The training style prioritizes the experience replay and stochastic rank-based prioritization [179].

4.4.6 Data Collection

The robot collects data during training to learn grasping tasks by interacting with objects iteratively. There are multiple (2-10) simple objects randomly arrange in front of the robot and within the workspace. The robot is tasked to grasp one object per one grasping attempt. If the object is successfully grasped, then the grasp attempt counts as true and robot moves to the next state, trying to grasp the rest of the objects. The agent follows the ε-greedy policy to learn exploration where ε starts at 0.5 and decreases over training time. The learning performance of grasping objects is progressively improving over the number of training attempts, as shown in section 4.5.

4.5 Experiments, Results and Discussion

A series of experiments has been carried out (in simulations and real-world environments) to examine the proposed grasping system. The experimentations investigate the feasibility of using MDP to train a robot to learn grasping strategies for robotic manipulation. Meanwhile, it serves to study the training capabilities of DRGP and proof-of-concept held directly from visual observations on non-trivial grasping task. Finally, it also examines the performance of generalization targeting novel objects (excluded from training).

4.5.1 Simulation Environment

To address the aforementioned objectives, simulated experiments were prepared (as shown in Figure 4.5) in V-REP using UR5 6DOF manipulator robot developed by *UniversalRobot*, and 2-finger gripper made by *RobotiQ Inc.*. A depth sensor was fixedly placed in the workspace, and 3D ordinary geometrical objects (randomly arranged in front of the robot) used for training purposes. Current state s_t representation is fed to the perception network as a heightmap scene taken from RGB-D image. FCN DenseNet infers pixel-wise Q-values (visualized as heat-maps) for primitive-grasping-actions. The candidate grasping-action with the highest Q-value will be selected and executed.

Figure 4.6 demonstrates examples of random arrangements of target objects during data collection and training session. Color and depth images are captured by the depth-sensor and the heatmaps are the output predictions of the Q-pixel wise maps.

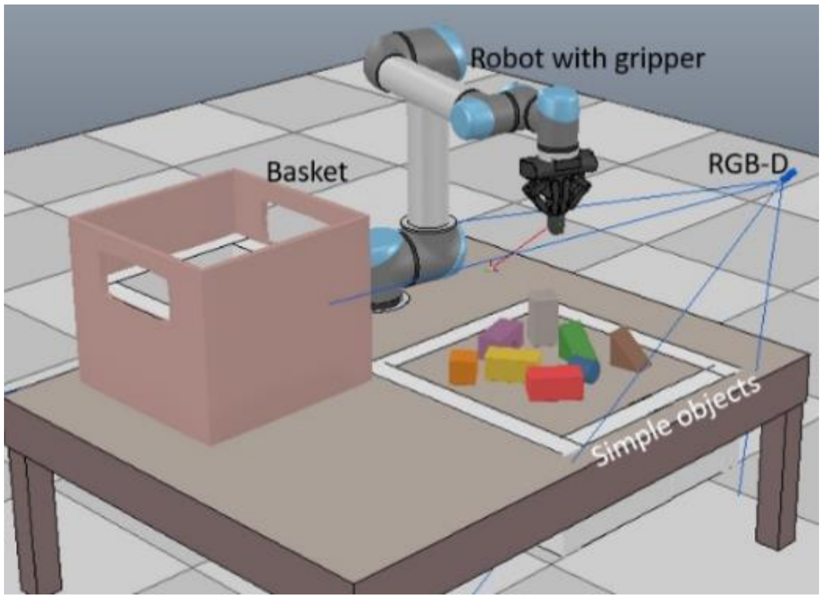


Figure 4.5: Setup of training scenario with simple geometric objects.

We ran multiple training sessions with different number of objects and heightmaps rotations, in which the latter one feeds FCN different grasping angles O_n. The more objects randomly placed in the workspace the more clutter scene and challenge will be created. For evaluating the performance, n is the number of runs that should be executed for each round of test. For each successful run, the robot should grasp and lift the object and leave it into the basket. The performance is evaluated with two metrics. First, the average clearance rate over n test attempts, which measures the model's ability to complete the round of test by picking up all objects per number of runs. Second, grasping efficiency rate which is the grasp success rate per clearance, where the grasp-action efficiency is the percentage of objects number over the number of grasping attempts ($\frac{no.\ of\ objects}{grasping\ attempts} \times 100\%$). This presents how efficiently the policy can complete the given task.

Figure 4.7 shows how grasp-success-rate raises over time where more grasping attempts were executed. Four of simple objects are randomly arranged (within the workspace) for each test-round. The grasping success rate already reaches $\sim 60\%$ in less

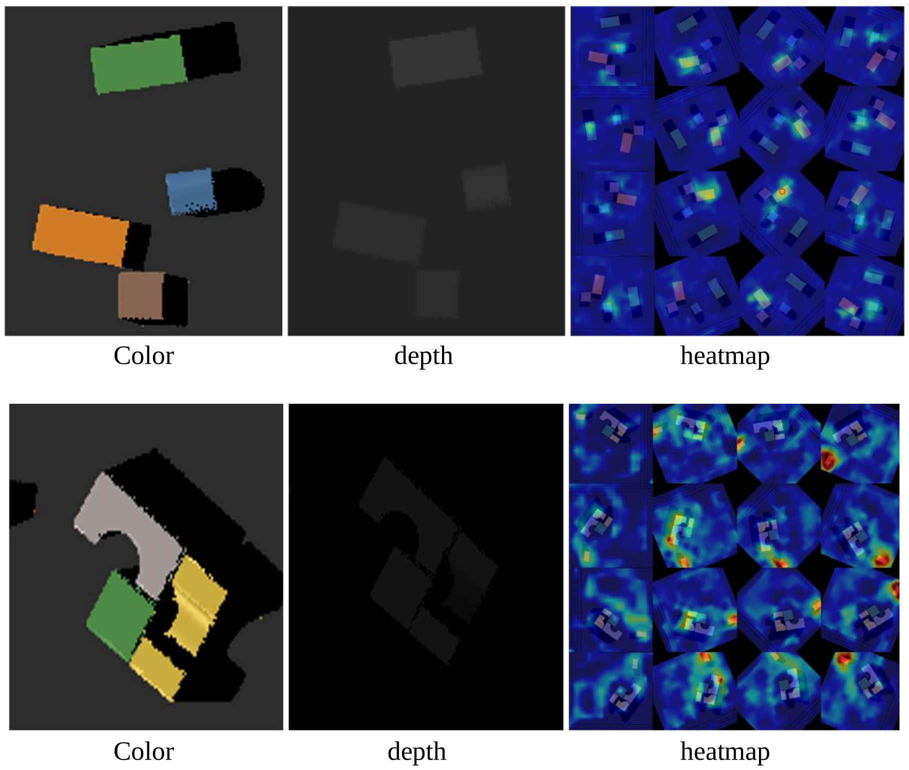


Figure 4.6: Examples of target objects randomly arranged in the workspace during the training and data collection session.

than 1000 grasping attempts. The total trials are 2500 and rotated angles O_n are 16. Each grasping-attempt counts as true if the robot is able to grasp the object properly, lift it, and leave it above the basket. Otherwise, grasping attempt is false (such as a robot fails to grasp the object, or non-firm grasping attempt where the object falls on the way going to the basket). Thus, a robot was trained to learn combinations of grasping policies and which strategy leads to successful and complete grasping attempt.

Figure 4.8 shows another training session where grasping-orientation is doubled $O_n = 32$, and 10 objects randomly placed (for each test-round) in front of the robot. Grasping performance remains interestingly stable and less erratic even when there are more target objects, which means more cluttered scene. The system efficiently learns grasping in less than 2000 transitions. This happens in about 5.5 hours when the robot needs 10 seconds to execute each trial.

Table 4.1 summarizes all training sessions (2500 attempts for each session), the

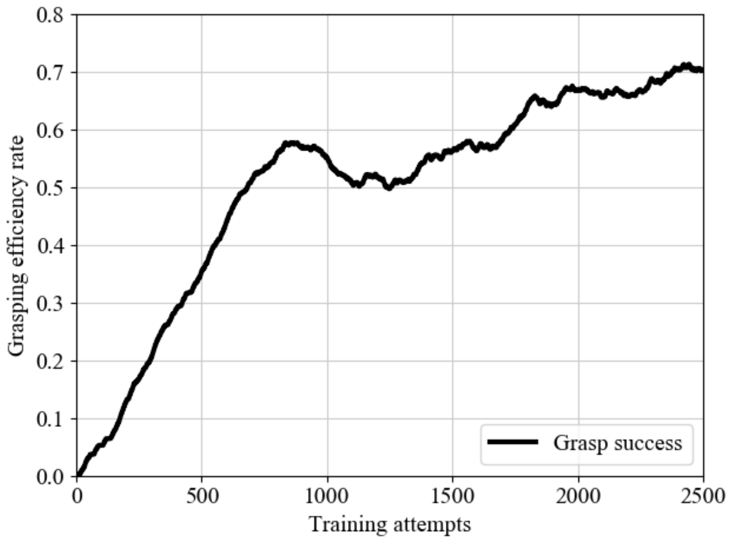


Figure 4.7: Training performance: 4 simulated objects with random arrangements and $O_n = 16$.

clearance rate % and grasping efficiency % remain acceptable even when we apply more challenging random arrangements of target objects. The reason for increasing the repeated rotations O_n is to account a wider range of grasping angles for a more cluttered scene.

Next, we compared the training performance between two different objects-arrangements. The target objects in both arrangements are randomly placed within the workspace. However, the first arrangement includes two objects which are usually placed in different positions in front of the robot, they are slightly spaced out. In the second arrangement, six target objects are placed in one spot which creates crowd and cluttered scene. The training performance of the two cases were represented in the Figure 4.9. Grasping success rate has reached to 95% for the first 1000 attempts (as seen in Figure 4.9 (a)), this is because the facility provided to the robot to learn the task in less challenging situation. However, grasp success rate has only reach 60% for the same number of attempts, as seen in Figure 4.9 (b). The robot has encountered relatively harder situation where crowd of target objects was created.

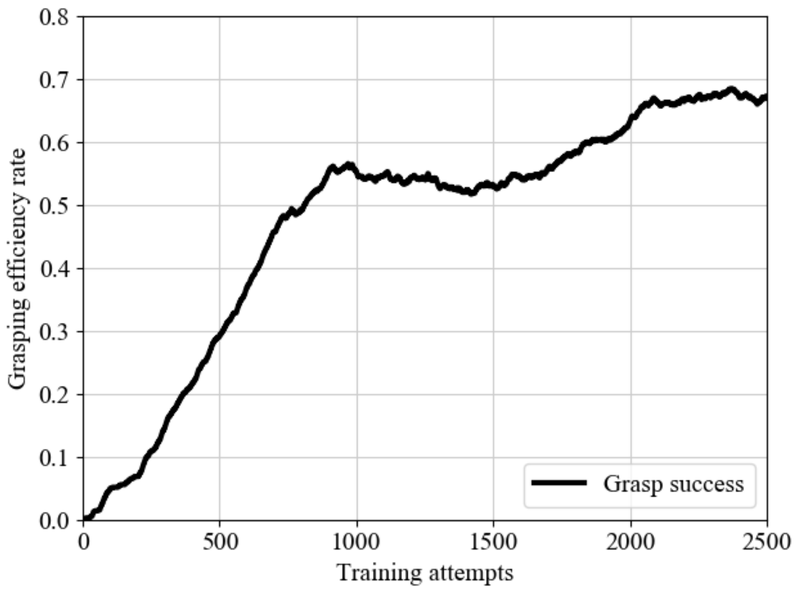
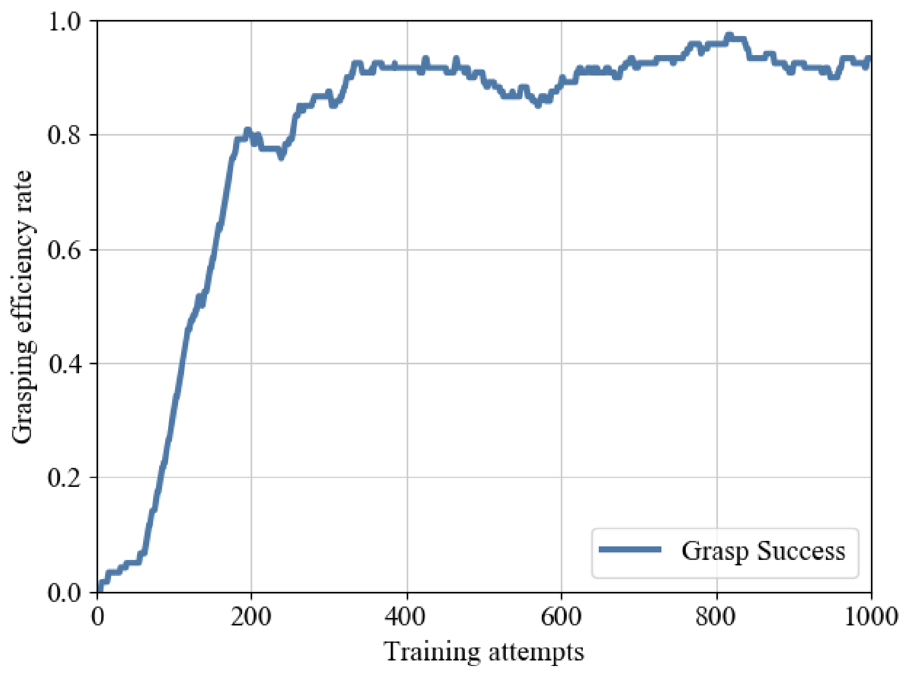


Figure 4.8: Training performance: 10 objects and $O_n = 32$.

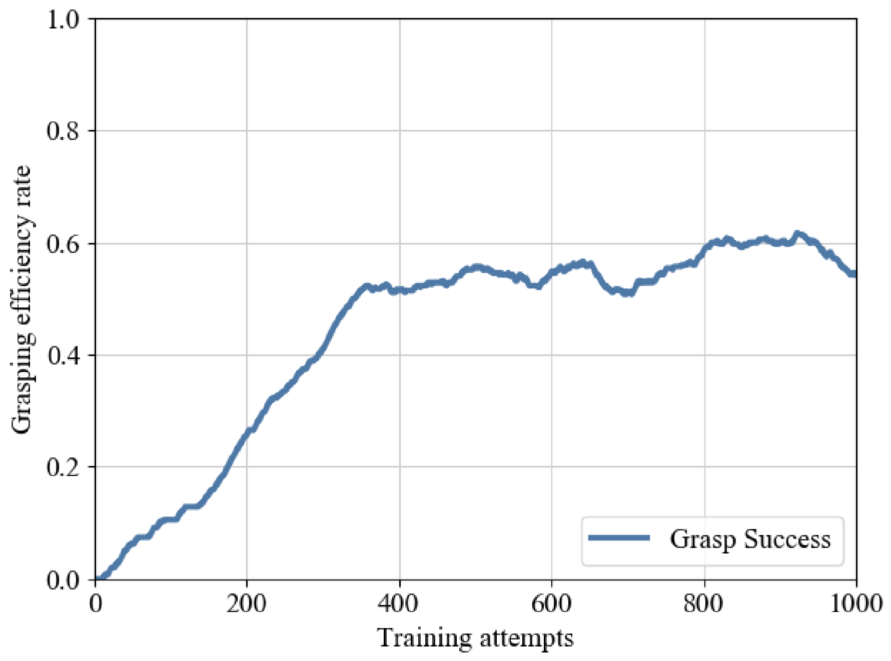
Table 4.1:A series of training sessions with different number of objects and rotations O_n.

NO.	OBJECTS NO.	CLEARANCE RATE %	GRASP EFFICIENCY%	O_n
1	2	95.5	83.1	16
2	4	89.8	74.1	16
3	5	72.8	72	16
4	8	63.3	58.3	16
5	10	53.1	60.6	16
6	5	82.8	68.7	32
7	8	67.7	63.6	32
8	10	48.1	65.4	32

Based on limited simulated training sessions, and without specific-training or pre-defined object information, the robot can successfully achieve generalization scenarios and grasp new objects. Figure 4.10 demonstrates the grasp success rate of novel objects in a simulator such as a watch, wrench, drill, and tape (as seen in Figure 4.10 (a)). Grasping



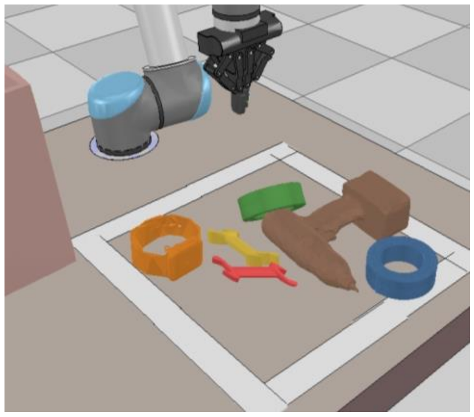
(a)



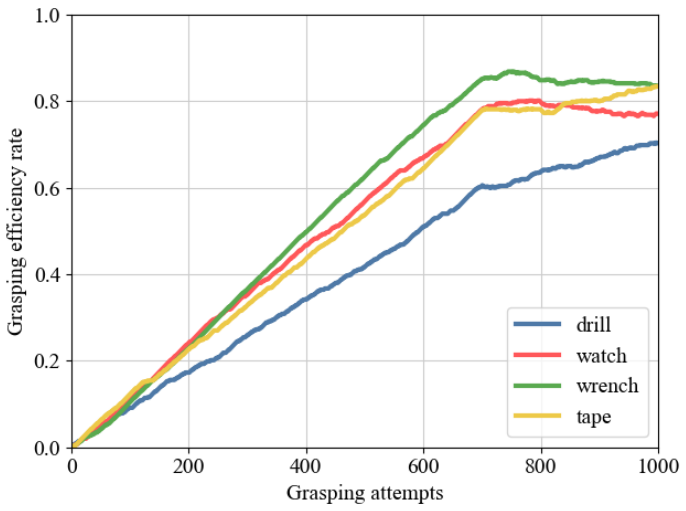
(b)

Figure 4.9 Training performance of two different random arrangements of target objects, (a) two separated objects, (b) six gathered objects.

success rate for novel objects is computed in Figure 4.10 (b). Grasping performance rate quickly reaches around 80% in less than 800 grasping-attempts.



(a)



(b)

Figure 4.10: Generalization performance. (a) Examples of novel objects in simulation experiments. (b) Grasp success rate for novel objects.

4.5.2 Comparative and Ablation Studies

In order to demonstrate the proposed learning method is suitable for the grasping task, we further carried out comparative and ablation studies. Because the system presented in [113] was unable to achieve grasping on full novel objects, we designed a supervised learning method (named Supervised-method). This method is analogous to our DRGP architecture with the same perception state and action space (as explained in the previous section). However, Supervised-method utilizes a binary-based classification with the annotations in greedy deterministic policy that predicts the pixel-wise map with values between 0 and 1. Figure 4.11 compares the performance between our DRGP and the Supervised-method. The learning scenario is similar to the training method in simulation. There are 3 objects are randomly arranged withing the workspace and O_n is 16. Our DRGP has performed better than the Supervised-method, this is probably because of the lack of information that leads to incapability of optimizing the learning progress.

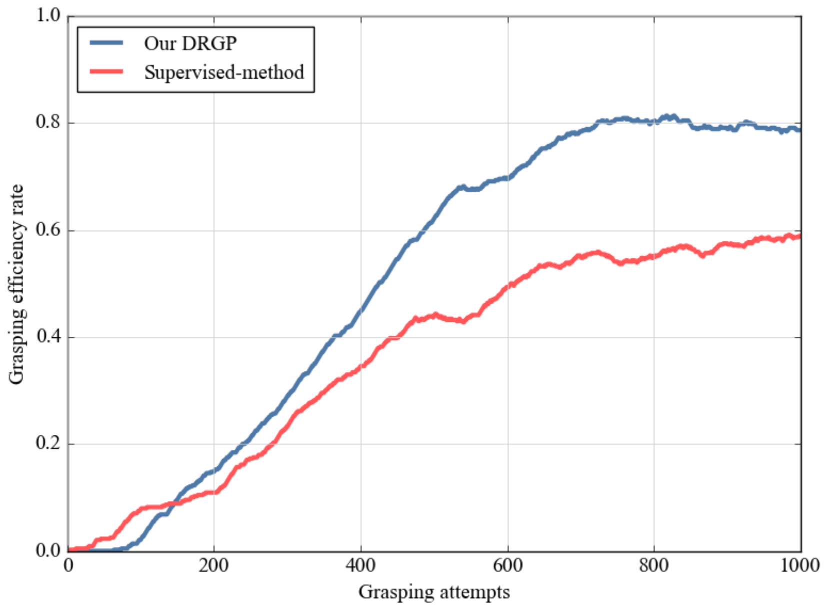


Figure 4.11: The performance of our DRGP with a Supervised-method.

To present the benefit of the DRGP, we conducted ablation study by comparing our DRGP with two ablated versions. First, No-pretrain-onDepthChannel is the version that starts with random initialization (on depth channel) and trains without the assistance of pre-trained model (ImageNet). Second, ablated version that trains without the height predictions, only information of color channel is available. In this case, FCN works without the depth channel (DDD) from the RGB-D image, this ablated version named No-DepthChannel. Figure 4.12 indicates that our proposed DRGP outperforms the two ablated versions. It is worth mentioning that the No-pretrain-on-DepthChannel version has no significant impact on the efficiency of our DRGP. However, it assists to accelerate the learning progress and slightly improve the final performance. Whilst, the existence of the depth information (height from bottom) is essential for accomplishing successful grasping task. This is shown in Figure 4.12 as the DRGP trajectory obviously performs better than the No-DepthChannel version.

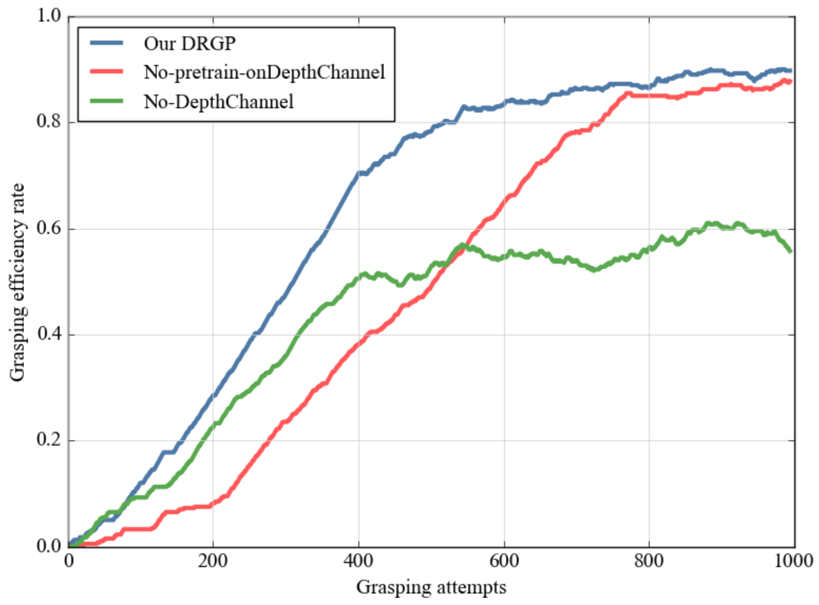


Figure 4.12: The performance of our DRGP compared to two different ablated versions. 1) Without pretrain model on depth channel, 2) and without depth channel.

4.5.3 Short-Term Pivot

The agent focuses more on future rewards when the discounted factor of future returns adjusted larger at $\gamma \cong 1$. However, we investigated the ability of the agent to plan short-term strategies and focus on near future rewards where long-term rewards worth less. It is named as DRGP-Short-Sighted where the discounted factor is smaller at two values ($\gamma = 0.25$, and $\gamma = 0.125$). Grasping performance rate has improved at faster progress (as shown in Figure 4.13), requiring only few batches of initial trials of training. This illustrates how important the agent tends to focus on early trials. The agent impressively comprehends the shortsighted policies and initiates learning from the early training range ($50 - 150$ trials). Comparing to Figure 4.7 (where $\gamma = 0.5$), the DRGP-Short-Sighted versions clearly

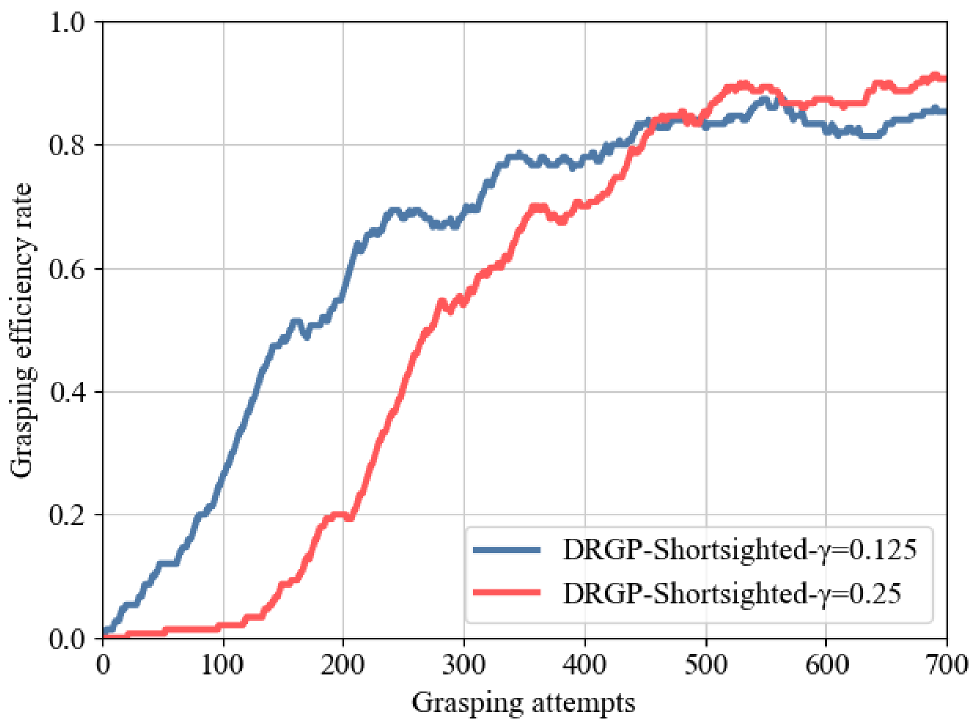


Figure 4.13: Shortsighted policies, Grasp success rate for short-term performance.

learn at the faster pace where the agent could approach 85% over only 700 trials. The experiment findings were conducted in simulation settings where $n = 30$, objects = 4 (random arrangements), and the grasp success rate was calculated for the last ($tr = 150$) training steps.

4.5.4 Real-Robot Experiments

The agent has learned grasping strategies from random arrangements of limited simulated objects. The gained knowledge during simulation-training sessions was implemented in real-life scenarios targeting unfamiliar objects. For the real-generalization test, a robot is capable of planning grasping on novel objects which have different shape attributes from the training objects. Figure 4.14 shows the experimental lab setup for grasping novel objects (for instance, household objects) used for generalization purposes with random arrangements. Realsense D435 camera was fixedly mounted in the workspace. Camera calibration process was availed using a known checkboard with

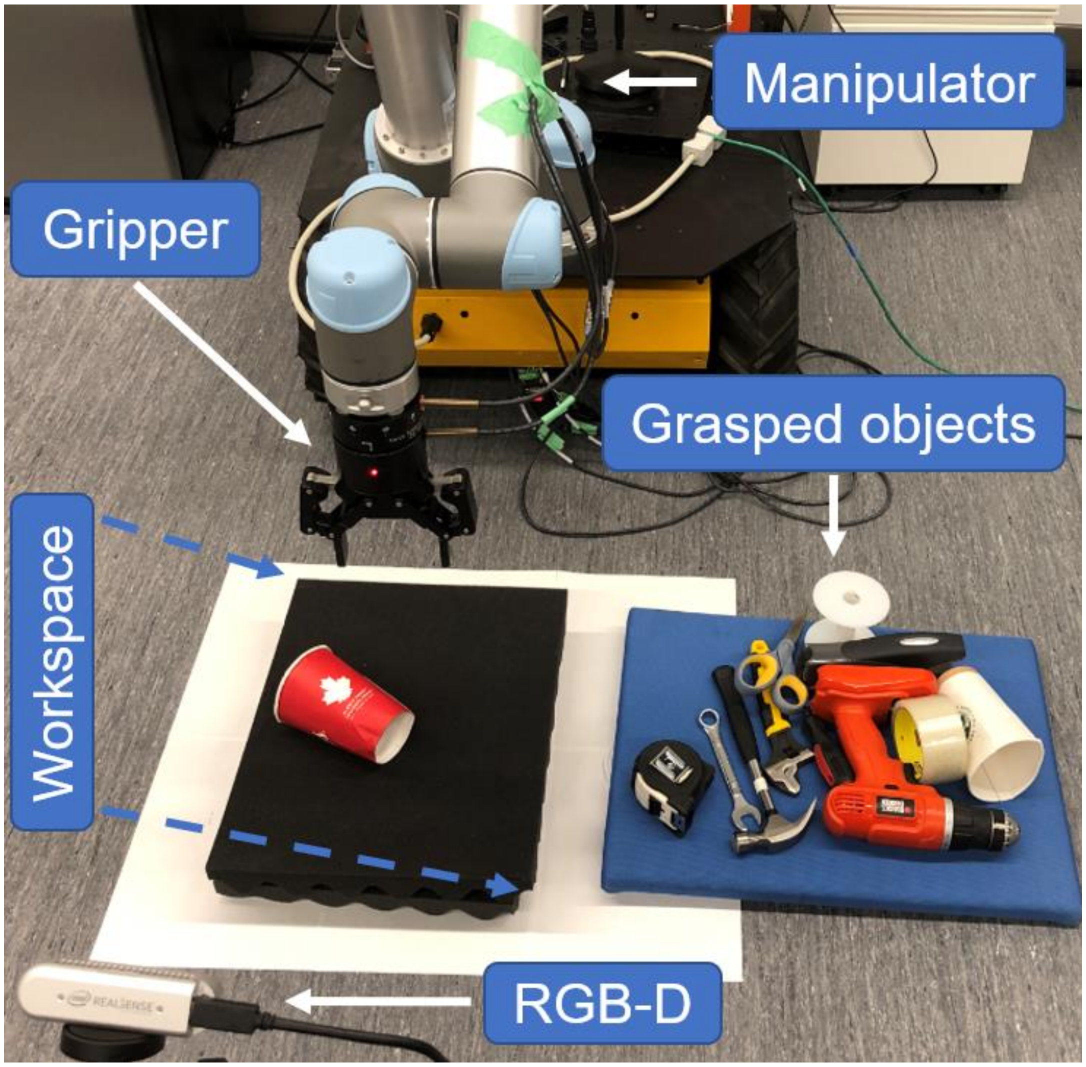
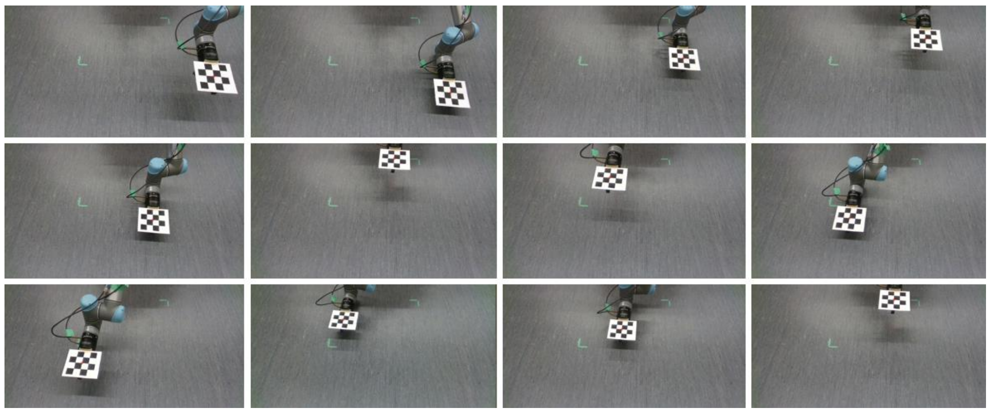


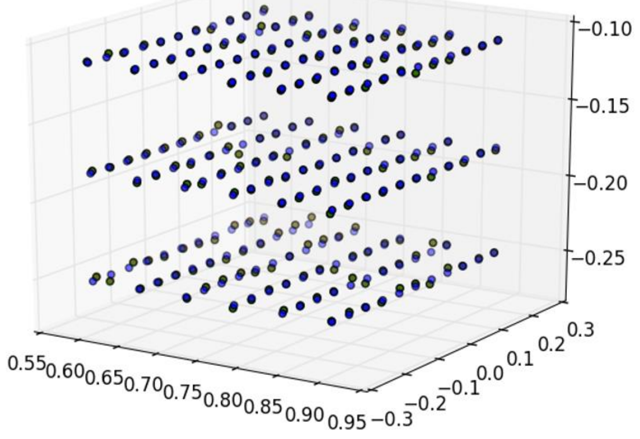
Figure 4.14: Generalizing experiments implemented physically using multiple novel household objects.

multiple points to construct 3D grid across the workspace, as shown in Figure 4.15. Fine-tuning the trained model in the real-world settings is not required. The findings showed that our system was able to generalize to new environments. The robot could successfully grasp all novel objects based on minimal simulated experience.

Three different scenarios were carried out to try various objects arrangements. Included single scenario for single object available in the workspace and cluttered scenarios which involve multiple objects (crowded scene). Table 4.2 reported the real-life experiments used for generalization purposes with random arrangements. Grasp success rate reaches 70% with six unknown objects.



(a)



(b)

Figure 4.15: Camera calibration process, (a) multiple known points, (b) construction of 3D calibration grid of the workspace.

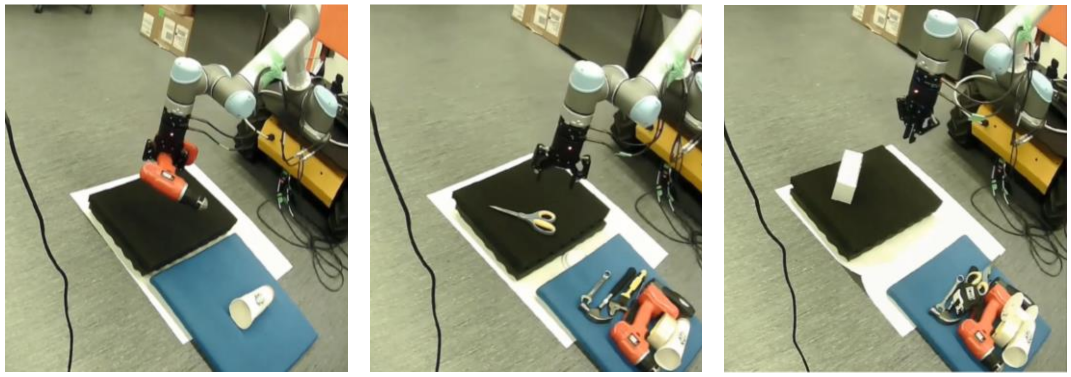
Table 4.2: Grasp success rate for real-world experiments on different objects arrangements.

No. of Objects	Object arrangements	Grasp success rate
10	Single	93%
7	Crowded	62%
6	Crowded	70%

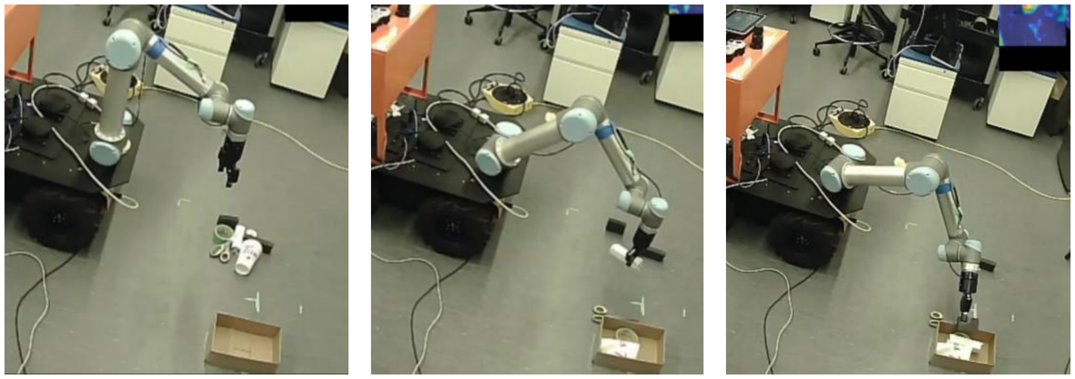
Different grasping scenarios carried out targeting various novel objects. Figure 4.16 demonstrates instances of three grasping scenarios. Which are; Figure 4.16 (a) is single unknown object, Figure 4.16 (b) and (c) are cluttered scene (multiple unknown objects). Comparing to prior recent grasping systems [108, 168, 180] which require at least one of the following requisites that hinder the self-learning performance. 1) Millions of grasping attempts, 2) massive training datasets, 3) consuming long hours of training time, 4) multiple robots at the same time to collect grasping data for manipulation tasks. 5) Finally, performed on simple objects (such as cubes) without exploring generalization task on real novel objects. Our model learns grasping policies quickly using one robot and limited simple simulated objects. Training findings have been successfully transferred to real-world with novel objects to test generalization capability of the proposed model. Experimental findings showed our model performed high grasping success rate for multiple unknown household objects. Video recording of the experiments is provided to show the training session and generalizing grasping tasks.

4.6 Summary

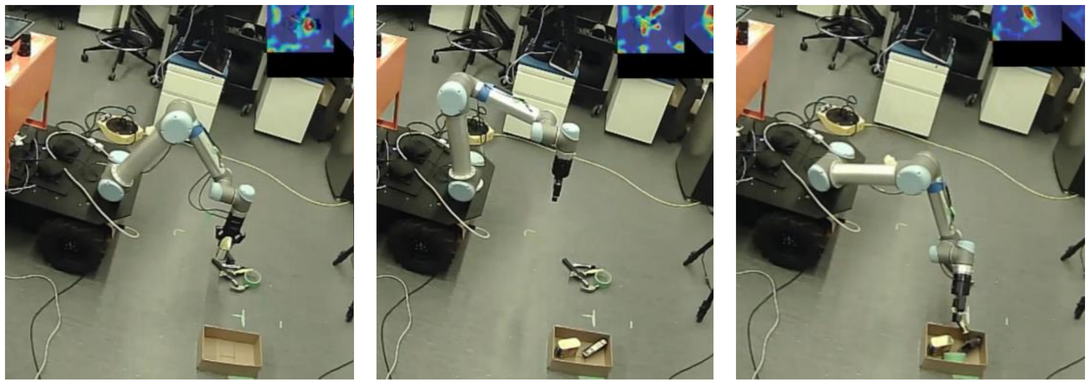
This chapter proposed an architecture to grasp unknown objects based on limited knowledge and information. Our scalable framework, named as Deep Reinforcement Grasp Policy (DRGP), was specifically designed to handle a wide range of novel objects. Perception network utilized FCN that takes visual observation of the current state and maps it into dense predictions pixel-wise Q-values. FCN infers the utility of primitive grasping-action that eventually executed by the robot. The agent learned from scratch (starting from input pixels to joint velocities), grasping strategies through trial-and-error manner. The grasping system was formulated as self-learning framework that operates with off-policy model-free deep reinforcement learning. The robot iteratively interacts with the environment with discrete action spaces (i.e. defined primitive actions) and learns progressively until the environment reaches the terminal state. The agent was well-versed in handling different scenarios of novel objects through experimentations. A demo-video was provided to show training sessions as well as successful real-world grasping novel different objects. The model was agnostic to object identity where it operates without object segmentation or classification. Because our DRGP is scalable, it would be interesting to address further studies in manipulation skills or object placement challenge as future works.



(a)



(b)



(c)

Figure 4.16: Instances of real-life experiments of the proposed DRGP system. (a) single unknown objects, (b) and (c) multiple unknown objects.

Chapter 5. Learn to Grasp Unknown-Adjacent Objects for Sequential Robotic Manipulation

5.1 Introduction

Previous Chapter has presented visual robotic system for grasping unknown objects in real-world. Notwithstanding the grasping system succeeded at handling various objects, unknown objects in real-life are often placed in challenging situations which require more than direct grasping action. In this case, it is difficult for the robot to execute successful grasps without prior manipulations.

Grasping unfamiliar-adjacent objects based on limited previous information is a daunting task in robotic manipulation. It is substantially more difficult to grasp an object in such a scenario than grasping secluded objects. For this reason, recent solutions typically require non-prehensile actions prior to grasping (e.g., pushing, toppling, squeezing, or rolling). However, these solutions play a loose role and causing delays. The non-prehensile action should have intended utility and effect on the consecutive grasping action because it is a sequential decision-making problem.

This chapter takes a step towards solving the issue by introducing a self-learning strategy to manipulate unknown objects in challenging scenarios based on minimal prior knowledge. The developed system (named Deep-Manipulation-Policy (DMP)) learns jointly pre-grasping (non-prehensile shifting) and grasping (prehensile) actions using model-free deep reinforcement learning. The agent comprehends sequences of pregrasp manipulations that purposely lead to successful potential grasps. The system is object-agnostic, which operates with needing neither task-specific training data nor predefined object information (e.g., pose estimation or 3D CAD models). The proposed model trains end-to-end policies (from only visual observations to sequential decisions-making) to seek optimal manipulating strategies. Perception network maps visual inputs to actions, as dense pixel-wise Q-values, and learns quickly through trial-and-error manner. Experimentation findings have demonstrated the effectiveness of the joint learning between pregrasp manipulation and grasp policies, in which the success rate of grasping has greatly increased. The proposed system has been experimentally tested and validated in simulations and real-world settings using 6DOF manipulator robot with two-finger gripper.

This chapter is organized in the following way: section 5.2 introduced the current state of art in robotic manipulation. Meanwhile, the main idea and contributions of the proposed method were illustrated. The proposed learning strategy is formulated primarily in the third section 5.3. The system methods are illustrated in the fourth section 5.4. The fifth 5.5 and last 5.6 sections demonstrate the experimental findings, discussion, and conclusion, respectively.

5.2 Grasping Unknown-Adjacent Objects in Challenging Scenario

Human intuitively use trivial strategies to push apart the clutter of objects and achieve grasping. This seemingly simple task is one of the most bottlenecks in the field of autonomous robotics. The recent demands in practical robotic applications have made this problem even more relevant and challenging. Skilled robotic manipulation includes grasping unfamiliar objects (exclusive from training) in a cluttered scenario where objects are adjacent and in close contact with each other. Grasping isolated objects is more doable than adjacent objects because of the ability of the robot to reach target objects. Prior studies attempted to solve the problem by providing pre-grasping (non-prehensile) action to separate objects and make the necessary space for arm and fingers. Pre-grasping action helps to re-adjust clutter objects without explicitly performing grasps. Classic solutions have tried pushing, rolling, or toppling as pre-grasping action to break up the objects that are too cluttered to be grasped. However, the pre-grasping actions were studied separately and not combined as a consecutive issue with the potential post-action which is grasping. Another set of works attempted to include actions of agnostic-pregrasps to facilitate grasping, but detailed generalization to novel situations was not taken into account. Moreover, such methods required costly preparedness of datasets or model-based [181-183]. Considering pregrasp manipulation and grasping policies as a sequential decision-making problem for robotic manipulation based on minimal knowledge is still an unexplored issue.

In this chapter, data-driven manipulation of shifting and grasping as sequences of acts was proposed and implemented, which synergizes to provide better grasp outcomes in such challenging scenarios. A self-learning control strategy is proposed for a rich and diverse task such as grasping, using model-free deep reinforcement learning. Figure 5.1 demonstrates the proposed grasping approach (Shift-to-Grasp (StG)) of picking objects which are in an adversarial situation (e.g., adjacent objects in (a)). In this scenario, the robot needs to move objects apart by executing pre-grasping action to make a target more graspable before perform grasping. Arrows in Figure 5.1 (b) and Figure 5.1 (c) represent the directions of the moves, where a target object will be shifted by the tip of a gripper. Next, the robotic arm can perform the grasping and successfully grasp and lift the object as shown in Figure 5.1 (d) and Figure 5.1 (e). Our framework was built on the basis of Q-formulation utilized for the purpose of high-dimensional discrete action spaces. An agent benefits from the sophisticated co-actions between non-prehensile (shifting) and prehensile (grasping) policies where pre-grasp assists to rearrange cluttered objects and generate grasps.

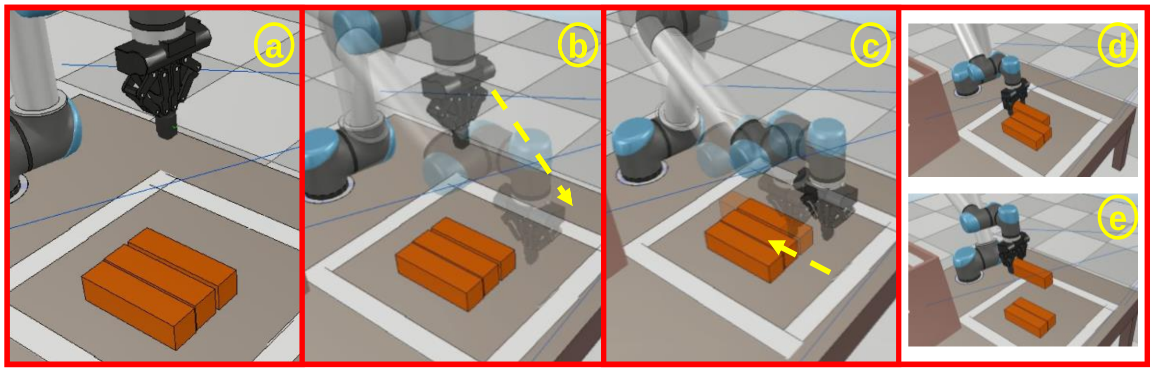


Figure 5.1: An example of adjacent objects problem and the proposed Shift-to-Grasp (StG) manipulation approach for robotic manipulation.

This chapter proposes a joint self-supervised framework to learn manipulating control strategies for unknown objects in challenging scenarios based on minimal raw experience. The contributions are summarized in two points. First, an agent concurrently learns pre-grasping and grasping policies with model-free DRL. Learning happens from scratch, from only visual observations to sequential decision-making. Pregrasp manipulations seek beneficial policies to intentionally enable grasping to increase the grasp success rate. This is unlike prior systems which required a hard-coded method or estimating pregrasps heuristically. Second, the proposed system trains in the form of deep Q-learning utilizing ordinary scenarios for a few hours. The acquired knowledge is, then, transferred to generalize on novel challenging scenarios. The model neither requires extra task-specific re-training data nor predefined object information (e.g., pose estimation or 3D CAD models). This is different than other systems that only operate known objects or have limited capacity to generalize.

In this chapter, the agent (robotic manipulator) interacts iteratively with the environment by executing the primitive actions. Rewards are granted to the robot upon the changes happening to the environment responsively. Shifting action obtains reward if it leads to a successful grasp, which also earns a reward. In this way, the robot learns progressively by maximizing future rewards. This procedure repeats continuously until the environment is resolved. Different action combinations are carried out through the trial-and-error manner. DRL seeks the best sequential plan that leads the environment to reach the terminated state. Our system uses the off-policy Q-learning concept to train on ordinary objects for a few hours. Then, generalizes efficiently on novel challenging scenarios, where even an optimal grasping system would struggle to achieve grasps without decluttering first. The training, evaluation and testing were carried out in simulation environment using V-REP (3D robot simulation software) as well as physical experimentations, employing UR5 manipulator and two-finger parallel jaw gripper. Our findings I) proved that it is possible to learn the sequential policies between pregrasps and grasping actions using self-supervised model-free DRL, and II) operated on novel challenging scenarios emphasizing the real-world generalization purposes based on limited simulated training knowledge.

5.3 Spatial Learning Formulation

The formulation of the sequent task (StG) is modelled as a Markov Decision Process, which is formally used to describe the stochastic environment to the agent (i.e., a controller). Representation image of the scene defined as a state s_t at a given time t. The action space, denoted by $a_t \in \mathcal{A}$ $(\text{shift}, \text{grasp})$, holds parameter vectors of shifting and grasping actions.

Action type a_t is executed according to $\pi(s_t)$ which is the policy related to $\pi(s|a)$. A policy π is a function that returns an action a_t upon a state s_t, as shown below.

$$\pi(s|a) = \mathbb{P}[A_t = a \mid S_t = s] \tag{5.1}$$

Decisions should be made by the agent (based on a given state s_t) which executes an action a_t relatives to the policy $\pi(s_t)$. Consequently, an immediate corresponding reward defined as $R_{at}(s_t\ ,\ s_{t+1})$, $R(s,a) \in \{0,1\}$ given to the agent and proceeds to the next transition state s_{t+1}. The expected return rewards improve the agent's understanding by indicating which pairs of action-state are good. G_t is the total discounted rewards from time step t, as expressed below, the goal is to maximize this return across all states sequentially.

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{5.2}$$

where $\gamma \in [0,1]$ is the discount factor that deduces how the agent should care about rewards now to rewards in the future which are simply worth less when $\gamma < 1$.

The agent seeks to solve the action-value function $Q_\pi(s_t, a_t) = [G_t|S_t]$ which estimates the quality of any possible action a_t in a given state s_t, shown in equation 5.3.

$$Q_\pi(s,a) = \mathbb{E}_\pi\, [S_t = s, A_t = a] \tag{5.3}$$

In Q-learning, the target policy is greedy according to $Q(s,a)$ which selects action that leads to the highest value of Q, as shown below:

$$\pi(s,a) = \begin{cases} 1\,, & if\ \ a = argmax_a\, Q(s,a) \\ 0\,, & \quad otherwise \end{cases} \tag{5.4}$$

The agent objective is to determine an optimal policy π^* (or controller actions) which is the decision that selects the best actions with the highest quality that maximizes the action-value function, expressed in the equation 5.5.

$$Q\pi^*(s,a) = max_\pi\, Q\pi(s,a) \qquad 5.5$$

While policy π is a distribution over actions in given states, π^* maximizes the sum of the expected return of future rewards and it should be better or equal to all other policies. π*≥ any possible π if and only if $Q\pi^*(s,a) \geq Q\pi(s,a)$ for $s \in S, a \in \mathcal{A}$. Deep neural network used here to calculate $Q\pi^*(s,a)$ of any possible actions in each state.

In terms of the learning strategy, an off-policy Temporal Difference (TD) control method is implemented to solve the sequential control problem and find the optimal policy of StG approach. TD learns from raw experience which is unlike Dynamic Programming (DP) [184] that requires a predefined model of environment. Learning in TD happens from each transition, regardless of the subsequent actions, and guarantees convergence to the correct predictions. This contrasts with Monte Carlo method [115] that only learns by the end of the episode and shows slow convergence with no bootstrapping. Unlike conventional DRL which requires double network for action selection and evaluation, TD-learning meant to reduce the overestimation gap, which needs the unnecessary burden of computations. The right side of the equation 5.6 below called the TD-Target, which is the sum of the immediate reward $r = R_{at}(\,s_t\,,\ s')$ given to the agent at the current state and the discounted value $Q_\pi(s',a')$, where $a' = a_{t+1}$ is the action for the next state $s' = s_{t+1}$.

$$Q_\pi(s,a) = \mathbb{E}_\pi\,[r + \gamma Q_\pi(s',a') | S_t = s, A_t = a|] \qquad 5.6$$

In TD-learning, we update action-value function for each action in a given state towards the estimated return TD-target. We use ϵ-greedy policy as behavior action (given in equation 5.7) to fulfill the need for balancing between exploration and exploitation. Unlike the greedy deterministic policy that might preclude beneficial states whose values are unavailable to be discovered. The epsilon-greedy is often exploitable with limited chances of exploring as epsilon refers to the probability of exploration.

$$\mu(s,a) = \begin{cases} random\,action\,,\ if\ \ p \leq \epsilon \\ argmax_a\, Q(s,a)\,,\ otherwise \end{cases} \qquad 5.7$$

Similar to DQN implementation [175], deep Q-learning composes the task into action selection and action evaluation. Target-greedy policy uses the next state s' to calculate $Q(s, a_{-i})$ for each possible action a_{-i} at a given state s_t and finds the highest value of target $Q(s, a_{-i})$. TD-learning update rule is expressed below:

$$y_i^{DQN} = r + \gamma Q\big(s', argmax_{a'}(Q(s', a'))\big) \qquad 5.8$$

$argmax_{a'}(Q(s', a'))$ is more than the instant reward from an action; it is the maximum expected return into the future. Therefore, the value of the state/action pair is the reward that the agent just received, as well as how much reward the agent expects to collect going forward.

The learning objective is the difference between two versions of an action-value function, one before an action and one after that (TD-target). Learning policy is designed in equation 5.9 as a minimization of the distance between $Q(s_t, a_t)$ and TD-Target (this is called temporal difference L_i). Which is iteratively minimizes the error of $Q(s_t, a_t)$ to the fixed target y_i^{DQN}.

$$L_i = Q(\theta_t; s_t, a_t) - [r + \gamma \max_a Q(\theta_t^-; s_{t+1}, a_t)] \qquad 5.9$$

where θ_t are the parameters of the network at time t, and θ_t^- are the target network parameters.

Algorithm 1 simply summarizes the policy execution of StG approach. The algorithm is repeated until the robot grasps all objects or exceeds the maximum number of motions. For each iteration, one of the two actions (shift or grasp) is effective upon the given state. Rewards are granted following each action and upon state changes to improve understanding of the progress.

Algorithm 1. Joint self-learning policy to StG

Input: RGB-D image

Output: action decision a_t upon a state s_t at time t

```
1. $Q\pi^*(s,a) \geq Q\pi(s,a)$ for $s \in S, a \in \mathcal{A}$
2.     if   $p \leq \epsilon$
3.         $a_t(\psi,p)$ = random $Q\pi(s,a)$
4.         obtain $R_{at}(s_t, s_{t+1})$
5.     else
6.         $a_t(\psi,p) = \max_\pi Q\pi(s,a)$
7.         obtain $R_{at}(s_t, s_{t+1})$
8.     end if
```

Training in entire self-supervision manner, the system was trained by minimizing the TD error (as shown in L_i) based on Huber loss function at each iteration i, as shown below:

$$\mathcal{L}_{Li} = \begin{cases} \frac{1}{2}{L_i}^2, & if \left|Q^{Qi}(s_i,a_i) - y_i Q^{Q_i^-}\right| < 1 \\ \left|Q^{Qi}(s_i,a_i) - y_i Q^{Q_i^-}\right| - \frac{1}{2}, & otherwise \end{cases} \quad 5.10$$

During the training phase, the iterative optimizing method by the stochastic gradient descent used to train FCN with learning rate 0.0001 and weight decay 2^{-5}. The system has been implemented on the PyTorch platform [160] and trained on GPU support (Nvidia GTX 1080) for the number of attempts, as seen in section 5.5. The training style prioritizes the experience replay and stochastic rank-based prioritization.

5.4 Shift Objects for Grasping

This section illustrates the modelling of the proposed manipulating system, including visual observations and perception network, primitive StG actions, rewards and co-action learning. Figure 5.2 demonstrates the overall framework implemented in the simulation environment, starting from the state representation of the current scene. The perception network computes the Q-maps for manipulation actions as dense pixel-level of Q-values. The action-value function is then executed by the robot manipulator.

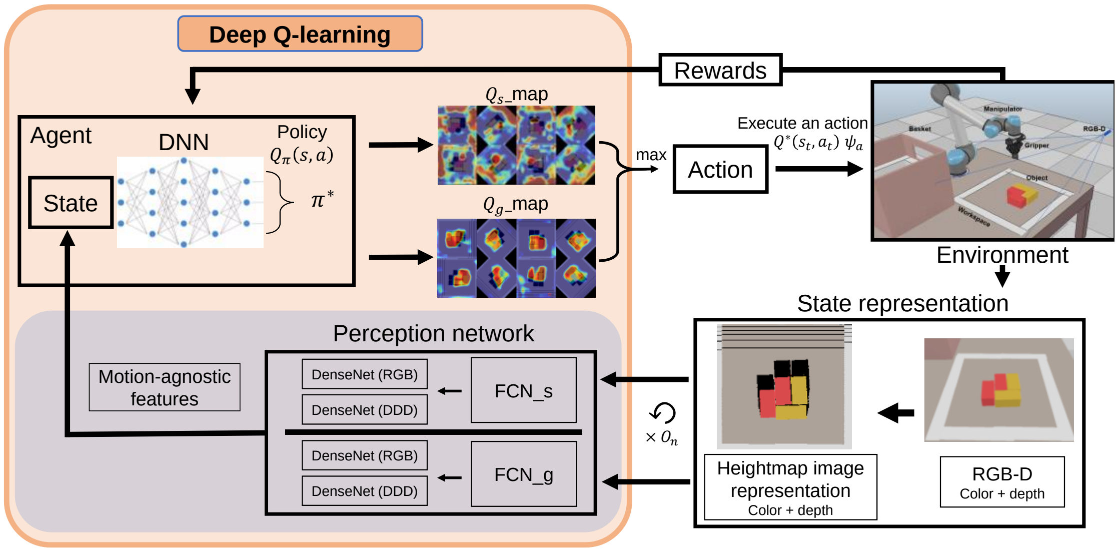


Figure 5.2: Overview of the model and system architecture.

5.4.1 State Representation

To start with visual observations, a RGB-D sensor is fixedly placed in the workstation to observe the workspace and capture the RGB and depth images simultaneously. This carries depth details at the pixel level which makes it possible to construct a visual map called heightmap [185]. The RGB-D data is projected onto a 3D point cloud. Then, the heightmap representation image is generated through orthographic re-projection of the point cloud along the gravity direction with the known camera parameters that were obtained by chessboard calibration. Each state s_t at the time t is modelled as a heightmap representation scene. Dimensions of the heightmap image (pixel resolution of 200×200) are defined according to the robot workspace which is $(0.4^2 m)$ flat surface. Each pixel of the heightmap scene represents $0.002 \times 0.002\, m$ vertical column of 3D space. State representation of the heightmap image is next fed into the perception network to extract features of objects placed in the workspace.

5.4.2 Perception Network

We modelled the learning of a fully convolutional action-value function similar to deep Q-networks [175]. Perception network of Q-function is designed as two feed-forward fully convolutional neural networks (FCN) [177], each FCN processes one type of action (shift or grasp). FCNs work as features extractions that take heightmap image representations as inputs, followed by two layers of concatenated DenseNet-121 [178] (pre-trained on ImageNet [159]) which produces motion-agnostic features. One DenseNet layer input is RGB color image, and the second one is the depth channel DDD, both from the heightmap image representation. Then, DNN takes the features as input and predicts two Q-maps, shift Q_s_map and grasp Q_g_map. The output is the inferring visual affordances map of dense-pixel-wise Q-values, which represented with the same state resolution. Dense-pixel-wise is visualized as heatmaps at different pixels. Each one represents a confidence value of location on which to hold the defined action a_t. Each estimation of Q-value at pixel p announces the future expected reward of taking defined action a_t. This iterates for the different heightmap orientations $(O_n = 8)$, which decide the shifting direction and grasping orientation. Each FCN is provided with different rotated heightmaps in order to cover differing degrees of action a_t.

5.4.3 Action Decision

The expected robot's behavior defined as primitive actions ψ (which holds parameter vectors of shift and grasp) at a given state s_t. As shown in equation 5.11, p

determines the 3D location of the executed action a_t projected from a pixel of a heightmap scene.

$$(a_t) = (\psi, p)| \in \{shift, grasp\}, p \twoheadrightarrow pixel \in s_t \qquad 5.11$$

The middle point of the top-down tip of the gripper, for grasping action, is represented by p at one of the O_n. For shifting action, p is the position where the closed gripper starts the shifting move at specific O_n. The objective of pregrasp rearrangement planning is to separate objects intentionally to be more graspable. Shifting is executed (in straight trajectory) by the end of the closed-finger gripper headed down.

Q-values pixel-wise map combines a self-learning approach with visual affordances for the defined actions. Q-value predicts the future expected reward given to the agent for specific action ψ at p upon a state s_t, $p \twoheadrightarrow pixel \in s_t$. Primitive action ψ executes (p, O_n) at s_t and receives an immediate corresponding reward R_{at}. Similar types of parameters defined shifting and grasping actions, except that shifting requires the closed fingers movement. The direction of shifting is determined by the FCN_s which infers how an object moves in order to successfully generate possible grasp. FCNs are effective for pixel-wise computations of each state representation. Q-function approximator calculates Q-values of all possible actions ($200 \times 200 \times 16 = 640 \times 10^3$), where $O_n = 8$ for each Q-map.

5.4.4 Rewarding Policies and Joint-Supervised Learning

The rewarding scheme is simply designed as follows. First, $R_{\psi g}(s_t\ ,\ s_{t+1}) = 1$ for a complete and successful grasping attempt. Which is not only grasp the target object but also lift and carry it to the basket without failing throughout the way. Second, efficient shifting action is assigned as $R_{\psi sh}(s_t\ ,\ s_{t+1}) = 0.5$. Which is the action that makes noticeable changes in the workspace measured by the difference between two consecutive RGB-D scenes (defined by threshold T, $(s_{t+1} - s_t > T)$.

Pre-grasping reward $R_{\psi sh}$ is given to the agent for efficient individual pregrasp. This rewarding style has no explicit effects on sequential teamwork actions. In other words, the pregrasp might not intend to produce certain grasps. It is, however, more about stimulating the system to declutter beneficially and make changes. For this reason, another rewarding policy for shifting action was also carried out to prove the concept of joint-supervised learning (which is based on self-learning). In which, rewards are granted only for grasping action, shifting obtains nothing, $R_{\psi sh}(s_t \, , \, s_{t+1}) = 0$. In this case, the agent learns the efficient pregrasp that intends to make useful changes and lead to successful grasps which ultimately have the rewards. In addition, the agent would be learning to prioritize the action order. When the target object is graspable, pregrasp assistance is unnecessary thus grasping could be executed directly without shifting. (More experimental details are explained in the next section).

5.5 Experiments, Results and Discussions

A series of experiments has been carried out to accomplish our system named Deep-Manipulation-Policy (DMP). The experiments addressed to the following points:

- Examine how MDP can be employed to train an agent on learning sequential strategies for robotic manipulation.
- Explore if incorporating pregrasp manipulation will increase the probability of grasp success.
- Proving that pregrasp policies can be jointly supervised by future grasping policies (which are self-supervised), both of which are simultaneously trained.
- Study the capabilities of the proposed DMP system learns directly from only visual observations on non-trivial sequential manipulation problems, predefined object information and task-specific training data are both not required in implementation.
- Test the performance of real-world generalization on adversarial scenarios based on minimal simulated training knowledge.

5.5.1 Sparse Scenario and Training

To address these objectives, the environment was simulated in V-REP [174] using UR5 6DOF manipulator robot (an agent) developed by *UniversalRobot* and 2-finger gripper made by *RobotiQ*. Figure 5.3 demonstrates multiple training sessions. 3D simple objects are utilized for the training purposes in sparse scenario, where objects are randomly placed in front of the robot and within the workspace.

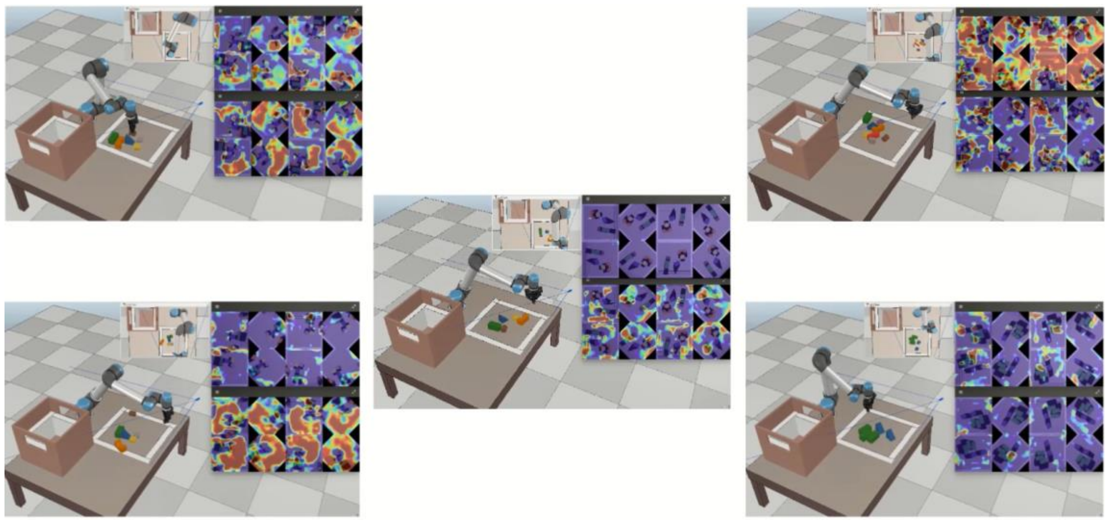
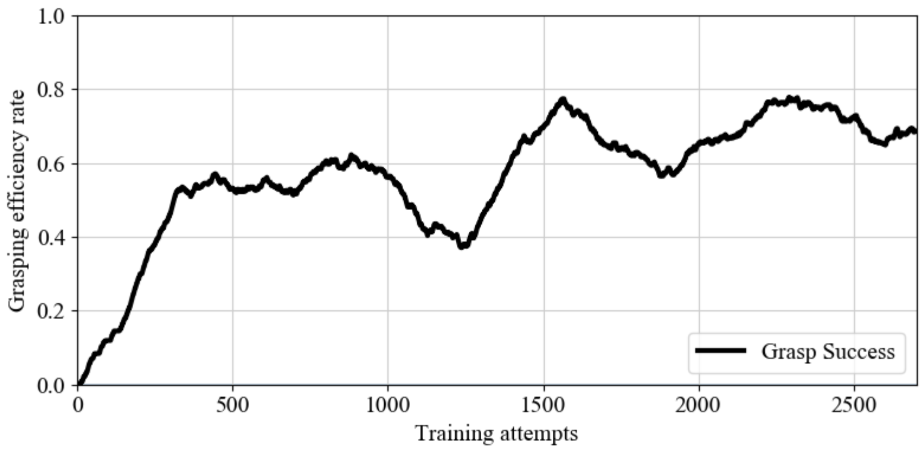


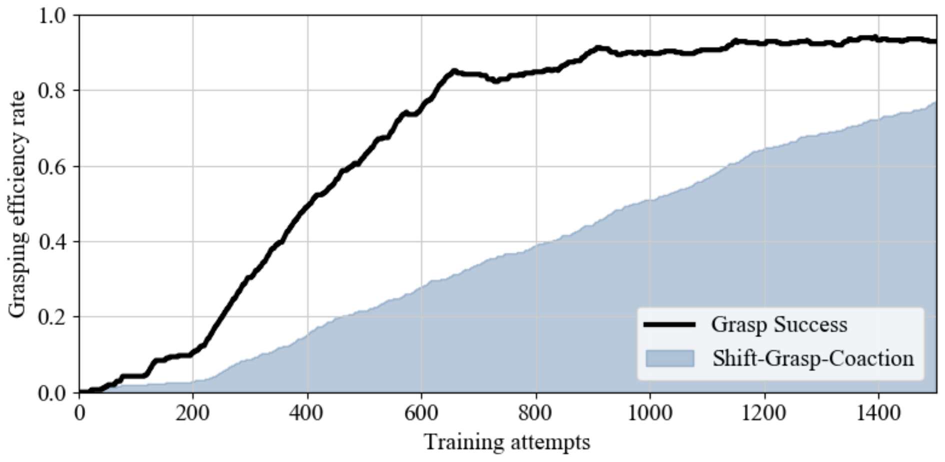
Figure 5.3: Training sessions in simulation environment (sparse scenarios).

In data collection and training, a robot collects data and learns actions by interacting with objects iteratively, which explores sequences of strategies to pick all objects with least executed actions. The robot is tasked to perform one primitive action for one object. If the executed attempt is effective, whether valid shifting or successful grasp, then the attempt counts as true and robot moves to the next state.

Figure 5.4 shows the performance of the training session with simple simulated five objects in a sparse scenario. Learning of pre-grasping and grasping actions is progressively improved over the number of training attempts. Training progress with only-grasping action is shown in Figure 5.4 (a) (training without pregrasps). A robot here struggles to achieve grasping seamlessly as objects are often ungraspable. Another training session, however, demonstrates great improvements in grasp success rate (shown in Figure 5.4 (b)).



(a)



(b)

Figure 5.4: Training performance in sparse scenario: (a) primitive action is only-grasping. (b) primitive actions are shifting and grasping.

This time pregrasp manipulation was considered which makes the progress certainly less erratic and more effective. This is showing how the existence of pre-grasping action facilitates the grasping task. Part (b) of Figure 5.4 obviously outperforms part (a) in less training attempts. The shaded area (Figure 5.4 (b) Shift-Grasp-Coaction) presents the shifting action that leads to successful grasps (StG). The agent learns which pregrasp manipulation would generate potential grasps. During training, the agent follows the ϵ-greedy policy to learn exploration where ϵ starts at 0.5 and decreases over training time.

5.5.2 Challenging Scenario

After training on a sparse scenario, the acquired knowledge was transferred to challenging scenarios where unknown objects are manually engineered to shape the cluttered scene. In which even optimal grasping model would struggle to directly perform grasping, declutter action is necessarily needed here. Figure 5.5 exhibits different challenging scenarios, all objects should be adjacent and in close contact with each other's, on which a robot executes multiple sessions of testing.

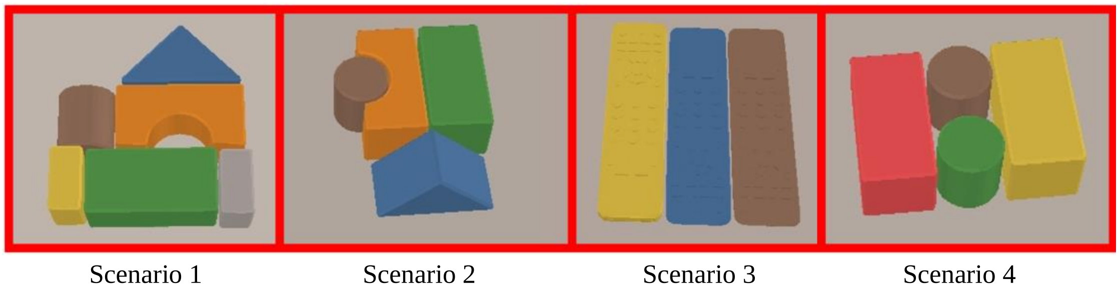
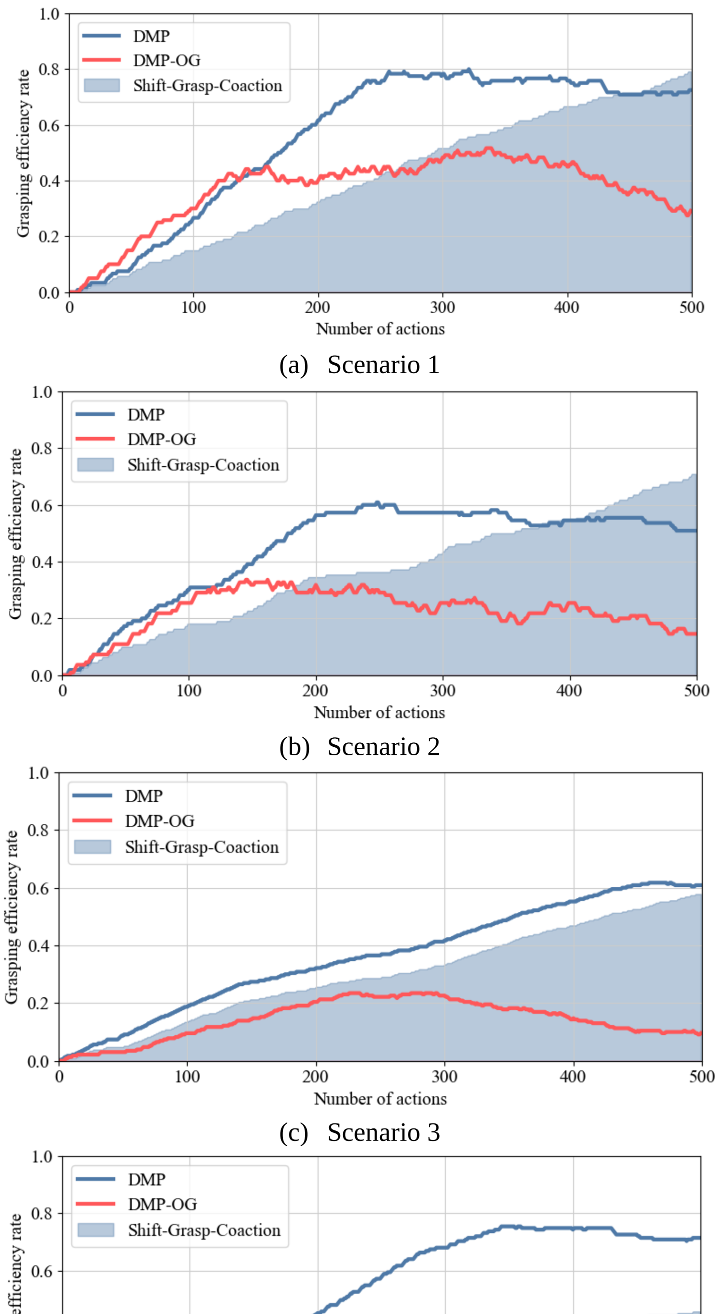


Figure 5.5: Adversarial scenarios: provide arduous grasping task in which robot needs first to de-clutter objects intentionally, then perform grasping.

Figure 5.6 illustrates the importance of performing pregrasp manipulations, in addition to grasping action, on multiple unknown challenging scenarios (that are shown in Figure 5.5). Grasp efficiency rate performs better when pre-grasping action is executed over a few hundreds of actions. DMP (Blue trajectories) represents the grasp-success rate that has been assisted by pregrasp manipulations. Whereas, grasp-success rate with Only-Grasping action (without preceded by pregrasps) is denoted as DMP-OG. The DMP expectedly surpasses DMP-OG performance as pregrasps are significant in such scenarios. Shaded areas are the progress of Shift-to-Grasp (StG) rate, which are the pre-grasping actions that generate immediate successful grasps. In scenarios 3 and 4, fewer pregrasps are sufficiently effective to achieve all required grasps. The number of the needed pregrasps are less comparing to pregrasps in scenarios 1 and 2. This shows that the robot executes mostly grasping in comparatively modest scenarios. Scenarios 1 and 2 are relatively harder, thus more pregrasps are required to generate better grasps. Figure 5.6 shows how pregrasp actions are necessarily needed on such challenging scenarios. Grasping target objects directly without pregrasps assistance could result to erratic and unstable grasp-success rate. As shown in scenario 4 where the DMP-OG trajectory decreases right before 200 attempts, then slightly increases around 300 attempts to continue reducing again.



(a) Scenario 1

(b) Scenario 2

(c) Scenario 3

(d) Scenario 4

Figure 5.6: Performance of two versions of DMP (with and without pregrasps assistance) on challenging scenarios.

Figure 5.7 demonstrates examples of depth images captured in a challenging scenario (same scenario shown in Figure 5.1). White arrows represent the direction of shift move and the point of grasping. The robot decided to execute pregrasps (shown in a, c, d, e, and f), and grasps (in b, g, and h) as confidence values of taking action are different at each state.

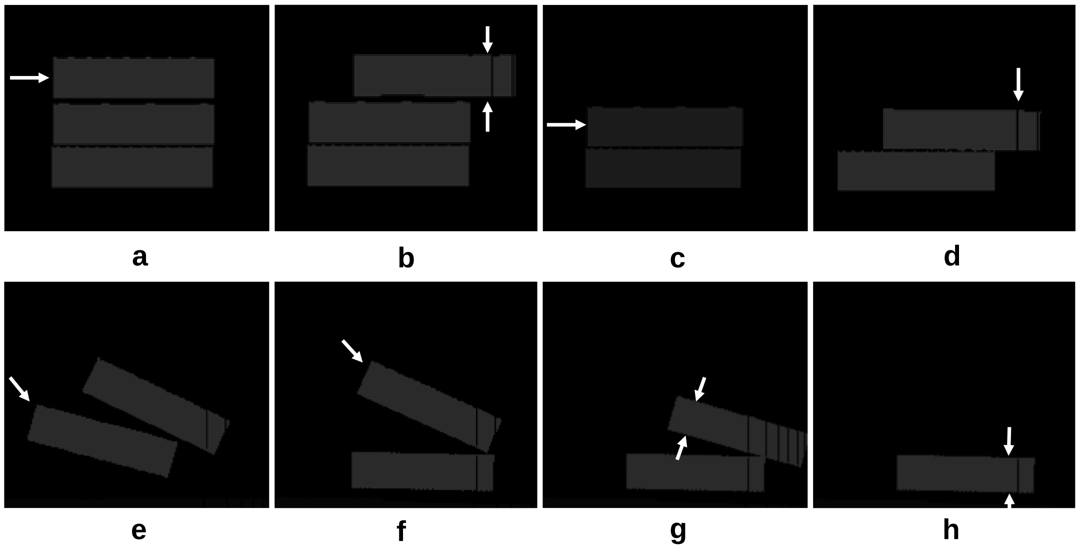


Figure 5.7: Examples of depth images presented in a challenging situation.

Table 5.1 reports the confidence score (CS) of shifting ψ_{sh} and grasping ψ_g in all states illustrated in Figure 5.7. The robot was tasked to execute actions with highest values, (except for (e) and (f) random action was chosen as exploration strategy is ϵ-greedy). The difference between two successive actions is calculated as |d|. Which shows that only one possible action should be taken (for example a, c, and h) as CS is clearly larger than other states (such as in b, d, g). (Note, we used sparse scenario and ϵ-greedy policy for training mode. Figure 5.7 is just an illustrative example with ϵ-greedy policy and challenging scenario).

Table 5.1: Confidence scores of shifting and grasping actions which are shown in Figure 5.7.

CS	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
ψ_{sh}	1.08	1.07	1.30	1.14	0.96	1.08	0.91	0.81
ψ_g	0.74	1.09	0.74	0.92	1.32	1.36	1.04	1.13
$\lvert d \rvert$	0.34	0.01	0.56	0.21	0.35	0.28	0.13	0.33

Further generalization testing was carried out on different challenging unknown scenarios in simulation setup, as shown in Figure 5.8. This round of testing operated successfully without fine-tuning or extra-training requirements.

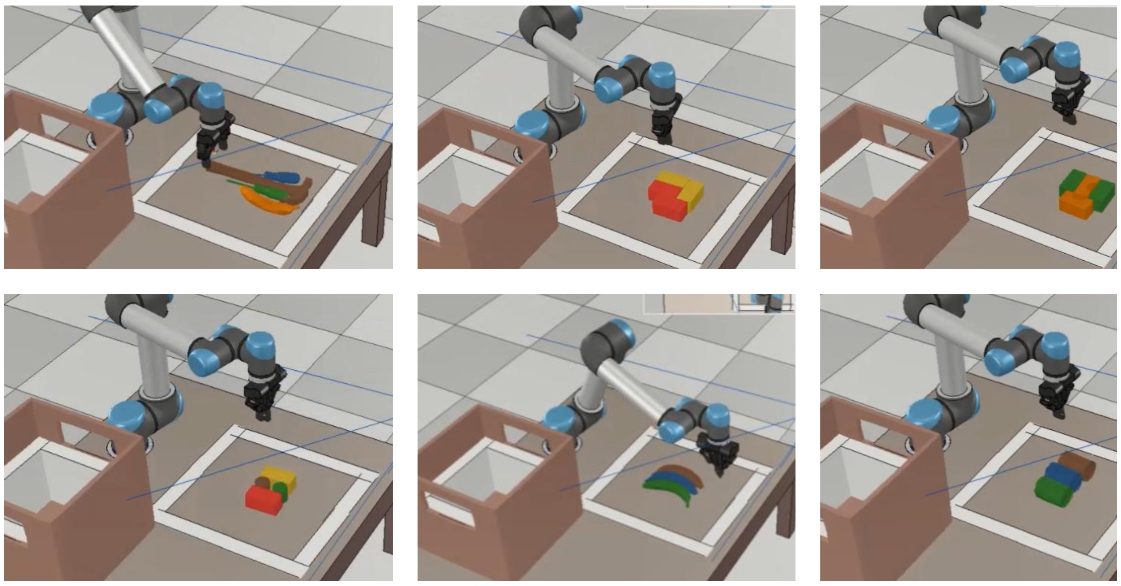


Figure 5.8: Generalization performance of the proposed DMP system on different challenging unknown scenarios in simulation settings.

For evaluating the performance, n is the number of runs that should be executed for each round of tests. The performance is evaluated after each execution of n with four metrics measured on the challenging scenarios (which are shown in Figure 5.5). First, the average clearance rate, which measures the agent's ability to complete the round of tests by clearing the workspace per number of runs. Second, grasp the success rate, which shows the average of grasps per clearance. Third, an action efficiency presents how efficiently the policy can complete the given task. Where how many actions the robot needs to successfully grasp all objects, $\frac{no.\ of\ objects}{no.\ of\ actions} \times 100\%$. Finally, the action ratio which is the grasp to actions ratio (number of grasps over executed actions). These metrics are expressed in Figure 5.9, grasp success rate shows at an average of 80% for all challenging scenarios.

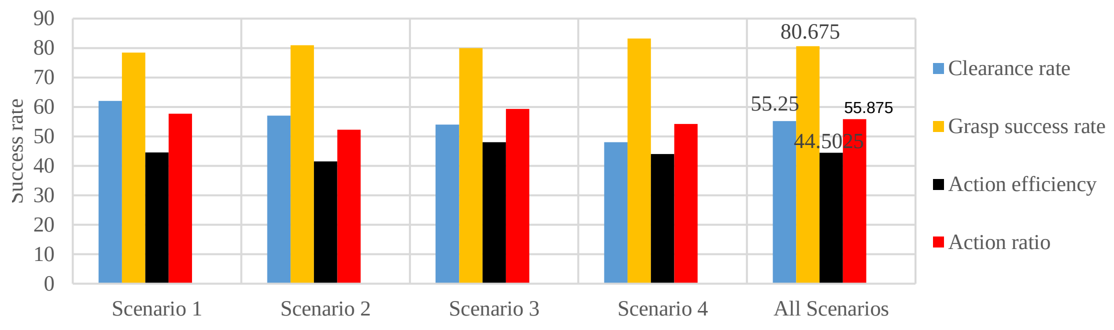
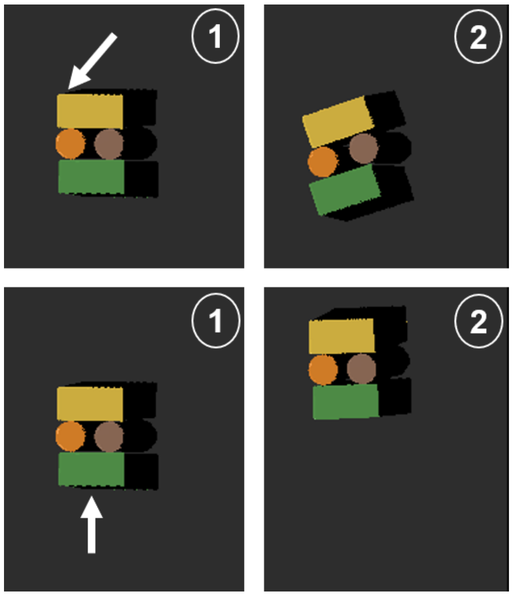


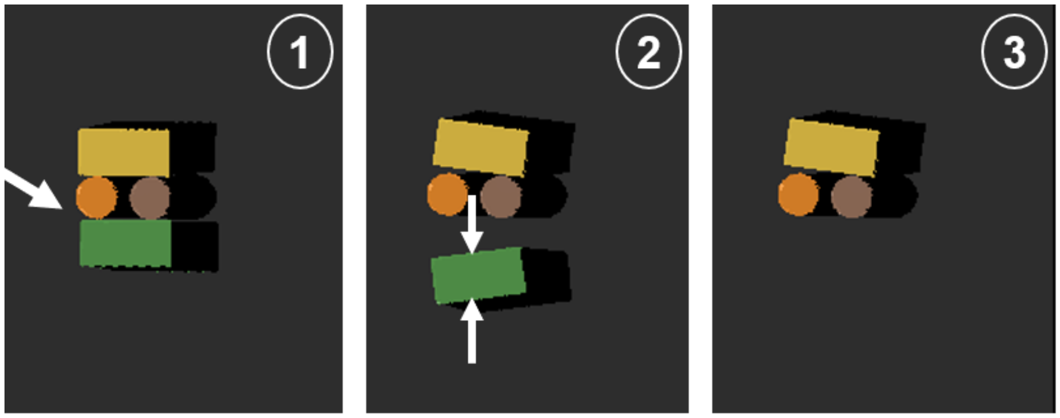
Figure 5.9: Evaluation performance metrics on different challenging scenarios. Number of executed actions =500.

5.5.3 Joint-Supervised in Challenging Scenario

Efficient pre-grasping action produces noticeable changes in the environment state. However, in the early learning stages, the generated status (after efficient pregrasp) is not necessarily useful for further grasping actions. The agent needs to determine a strategic planning of pregrasp that, in addition to produce noticeable changes, explicitly generates useable grasps. Figure 5.10 demonstrates examples of shifting actions that produced noticeable changes. Figure 5.10 (a) illustrates two instances of shifting actions represented as white arrows. Despite the objects have been changed their locations, potential grasp is distinctly unavailable. However, in Figure 5.10 (b)(1), a strategic shifting action has designedly selected the direction that breaks the cluttered and generates usable grasp. White arrows in Figure 5.10 (b)(2) indicated the grasping point.



(a)

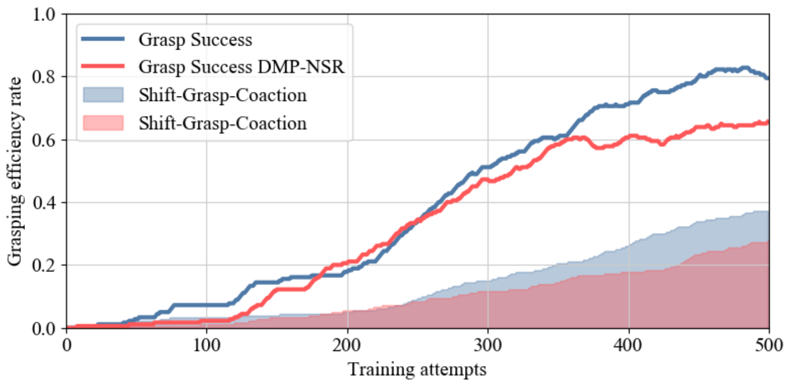


(b)

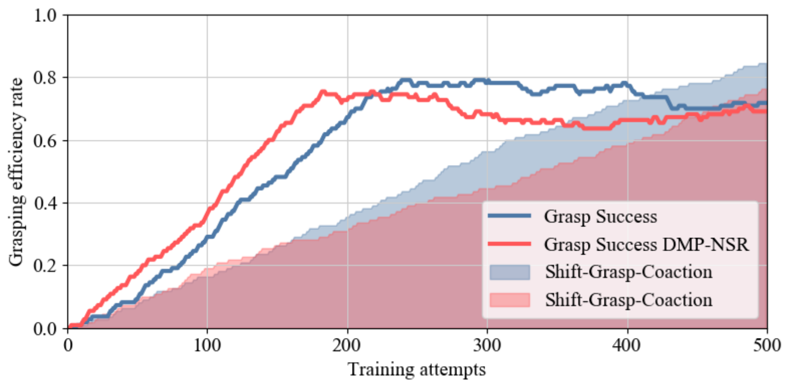
Figure 5.10: Examples of shifting actions that make noticeable changes, (a) two different shifting actions generate noticeable changes, (b) shifting action generates noticeable changes which lead to usable grasping action.

A further examination shall be conducted for the co-action that happens between the two manipulating behaviors (pregrasp and grasp). To do so, we run three learning sessions without pre-grasping rewards ($R_{\psi sh} = 0$), meanwhile, rewards for grasps stay the same. This version of DMP called DMP-No-Shift-Rewards (DMP-NSR). Joint-supervised pre-grasping learning was carried out on the basis of successive grasping that learned in a self-supervised manner. In this case, the agent is unable to recognize the immediate environment changes that happen from pregrasps. But the agent would rather learn pregrasp manipulation from the rewards given to the potential grasps. More alliance and teamwork between the two manipulation actions are, consequently, learned. In addition, the agent comprehends to prioritize actions, if the object is sufficiently graspable then pregrasp becomes unnecessary.

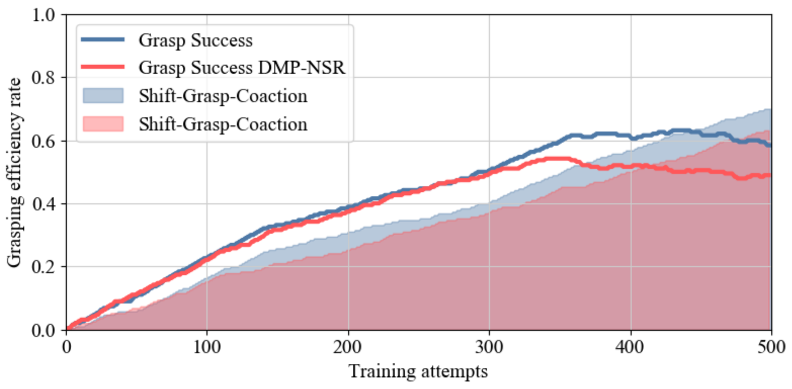
Figure 5.11 demonstrates the findings of two versions of DMP (DMP and DMP-NSR) for multiple different scenarios (sparse and challenging). Grasp success rate of DMP-NSR has shown impressive performance relative to the primarily DMP (which presented in blue trajectories). The two versions of DMP have trajectories which are roughly aligned with each other. DMP-NSR well-performed where the final success rate is slightly less than DMP. It is interesting to note that the agent learns about how the two actions work and achieved $70-80\%$ in just 500 attempts. The shaded areas refer to the success rate of StG. This shows the necessary co-action required to solve picking up task in hard scenarios. Figure 5.11 (a) has achieved the least StG rate, co-action in sparse scenario less needed because of the random arrangments of objects happening in each round. Whilst, Figure 5.11 (b) and (c) require relatively more co-action to approach grasping in the challenging arrangments.



(a) Sparse scenario



(b) Scenario 1

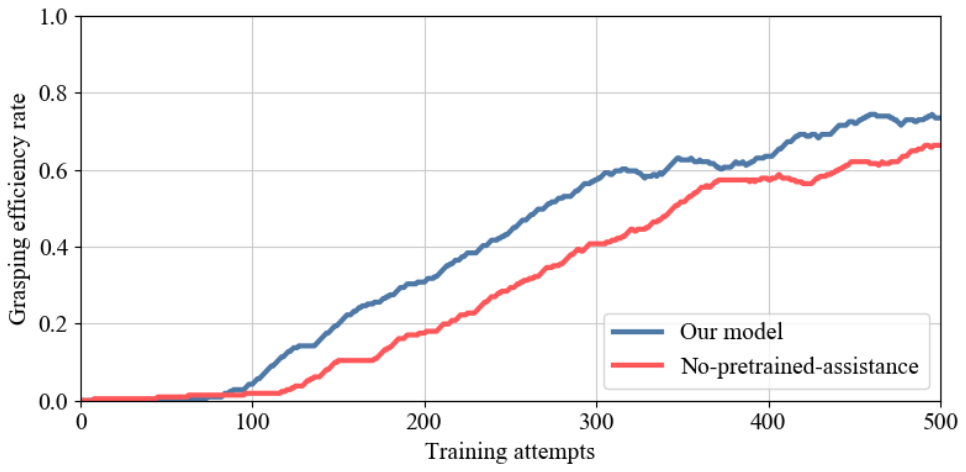


(c) Scenario 3

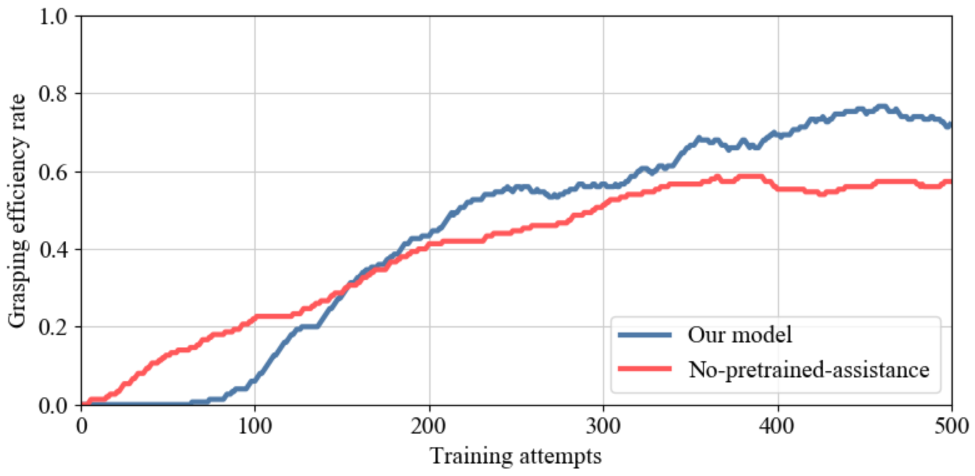
Figure 5.11: Comparing performance between two versions of DMP (DMP and DMP-NSR) trained with and without rewards for pregrasps. ((a) sparse scenario, (b) and (c) challenging scenarios).

5.5.4 Ablation and Short-Term Studies

To demonstrate the benefit of our model, ablation study was carried out to show the performance of our model with and without the pre-trained model (ImageNet) assistance used in the learning scheme on sparse and challenging scenarios. The version named No-pretrained-assistance, the FCN in the perception network starts with random initialization where transform learning step is unavailable for features extractions. Figure 5.12 ((a) and (b)) demonstrate the performance of our model (blue trajectories) and the ablated version



(a) Sparse scenario



(b) Challenging scenario

Figure 5.12: The performance of our model with and without the pre-trained model assistance, (b) sparse scenario, (b) challenging scenario.

(red trajectories). It is interestingly noted that the pre-trained model has no crucial effects on the performance of our model other than helping to accelerate the learning progress. In Figure 5.12 (b), pre-trained model assists to improve the final performance on challenging scenario (scenario 1).

Shortsightedly planning is required to investigate the agent's ability on short-term strategies. We designed two short-term versions that have smaller discounted factor. The two versions are Short-Sighted-1 and Short-Sighted-2 with $\gamma = 0.125$ and $\gamma = 0.25$ respectively. The agent's performance on short-term strategies is demonstrated in Figure 5.13. It is impressively indicated that the agent could learn at the faster pace and reach around 70% in less training attempts comparing to Figure 5.4. In this case, the agent gives priority to near-future rewards since the long-term rewards are worth less.

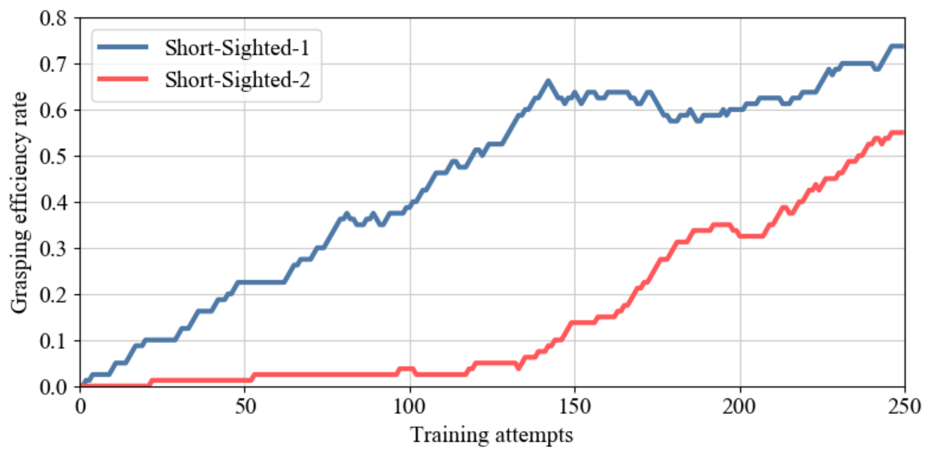


Figure 5.13: Performance of the Short-Sighted manipulation policy on sparse scenario.

5.5.5 Real-World Experiments

The agent has learned manipulating strategies from different arrangements of the limited simulated training sessions. The gained knowledge during simulation-training phase was implemented in real-life scenarios targeting unfamiliar objects arranged in challenging situations. For the real-generalization test, a robot is capable of planning manipulating actions on novel objects which have different shape attributes from the training objects. Figure 5.14 shows the physical experimental setup in the lab. environment, including 6DOF UR5 robot manipulator developed by *UniversalRobot* and 2-finger gripper made by *RobotiQ*. Realsense D435 camera (RGB-D sensor) was fixedly mounted to capture the workspace representation.

Different challenging scenarios of target objects are manually arranged to examine the real robot performance. Figure 5.15 demonstrates examples of challenging unknown situations on which the robot executes StG manipulation approach. Fine-tuning the trained model in the real-world settings is not required. The findings showed that our system was able to generalize to real-world environments and successfully perform StG manipulation approach. Task-specific training data and predefined information of target objects are both not required in the implementations. Video recordings of the experimentations have been provided to show the performance of the proposed system.

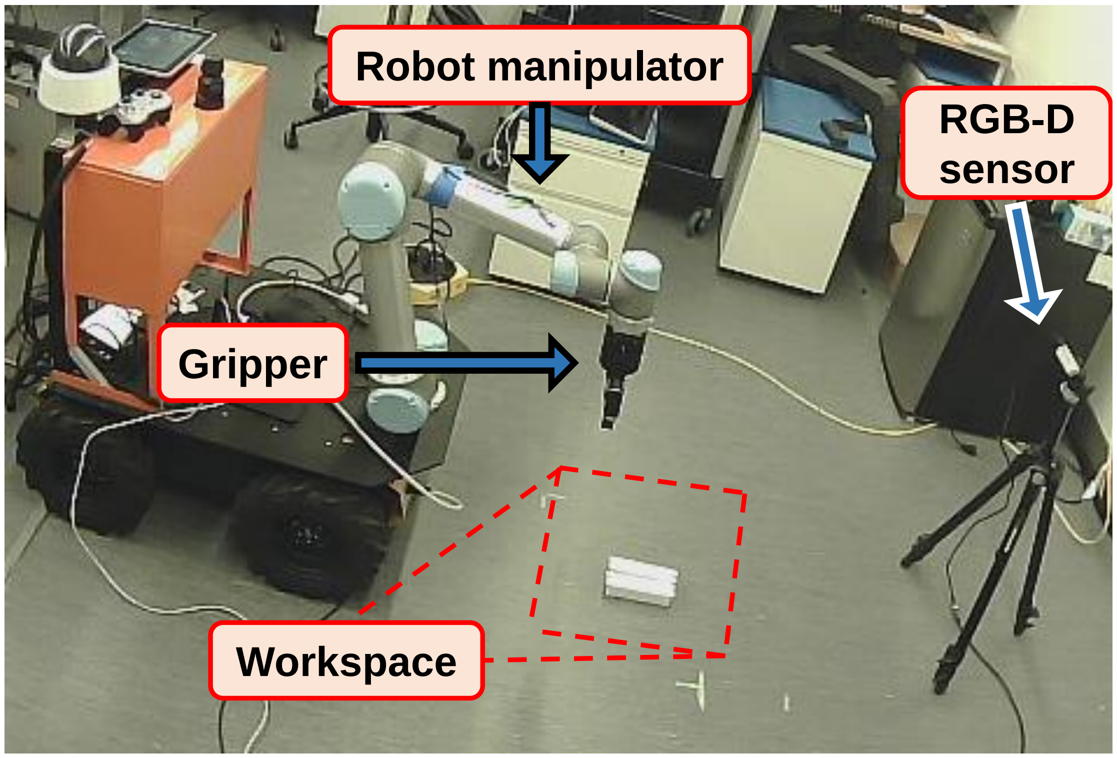


Figure 5.14: Physical experiment setup.

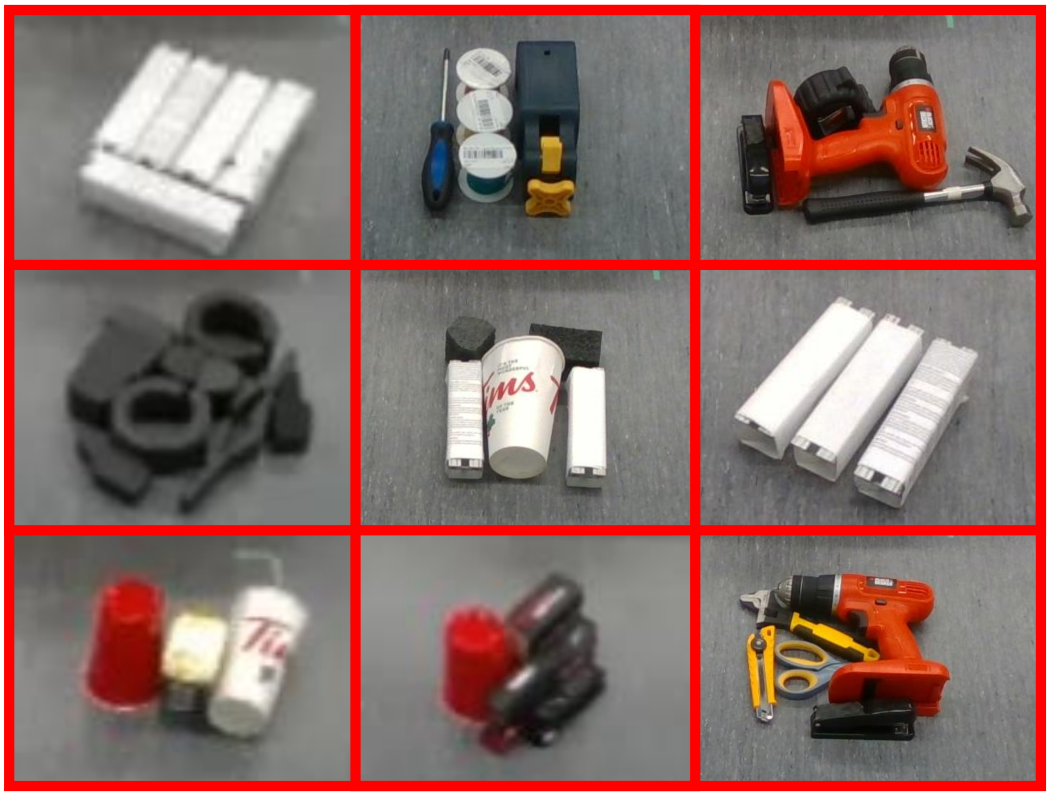


Figure 5.15: Examples of challenging unknown scenarios in real-world experiments.

Table 5.2 reports the findings from the physical experiments on 11 different challenging scenarios. Numbers of target objects are varied between 3 to 10 covering various shapes of objects. The robot could robustly accomplish the given manipulation task with an average of 64% grasp-success rate and 68% of action ratio.

Figure 5.16 demonstrates an example of StG approach during the real-world experiments. The red dotted arrows represent the direction of shifting action and grasping point. By executing shifting action, the robot first makes the targets more graspable and then performs grasping. Left side of Figure 5.16 shows the shifting action indicated by the heightmap shift representation. Grasp action representation from the heightmap is shown on right side of the figure.

Table 5.2: Action efficiency, grasp-success rate, and action ratio (mean%) from real-world performance on challenging situations. Number of objects varies between (3-10) in different scenarios.

No. of scenario	Action efficiency	Grasp success rate	Action ratio
Scenario 1	37.5	50	75
Scenario 2	33.33	46	73.33
Scenario 3	38.46	56	70
Scenario 4	44.45	57	77.77
Scenario 5	26.31	46	58
Scenario 6	43.8	60	71.43
Scenario 7	27.77	56	50
Scenario 8	35.71	59	61
Scenario 9	40.2	67	60
Scenario 10	75	100	75
Scenario 11	75	100	75

DMP system was implemented in real robot settings to carry out further generalization testing operating on challenging novel scenarios. Figure 5.17 demonstrates the manipulating performance of the robot utilizing StG approach.

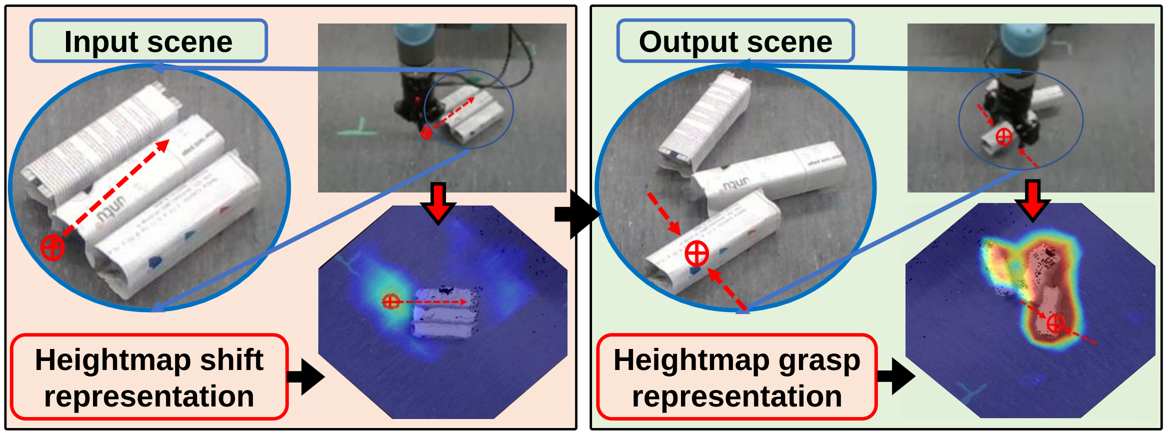

Input scene
Heightmap shift representation
Output scene
Heightmap grasp representation

Figure 5.16: (Shift-to-Grasp) example in the real-world experiments.

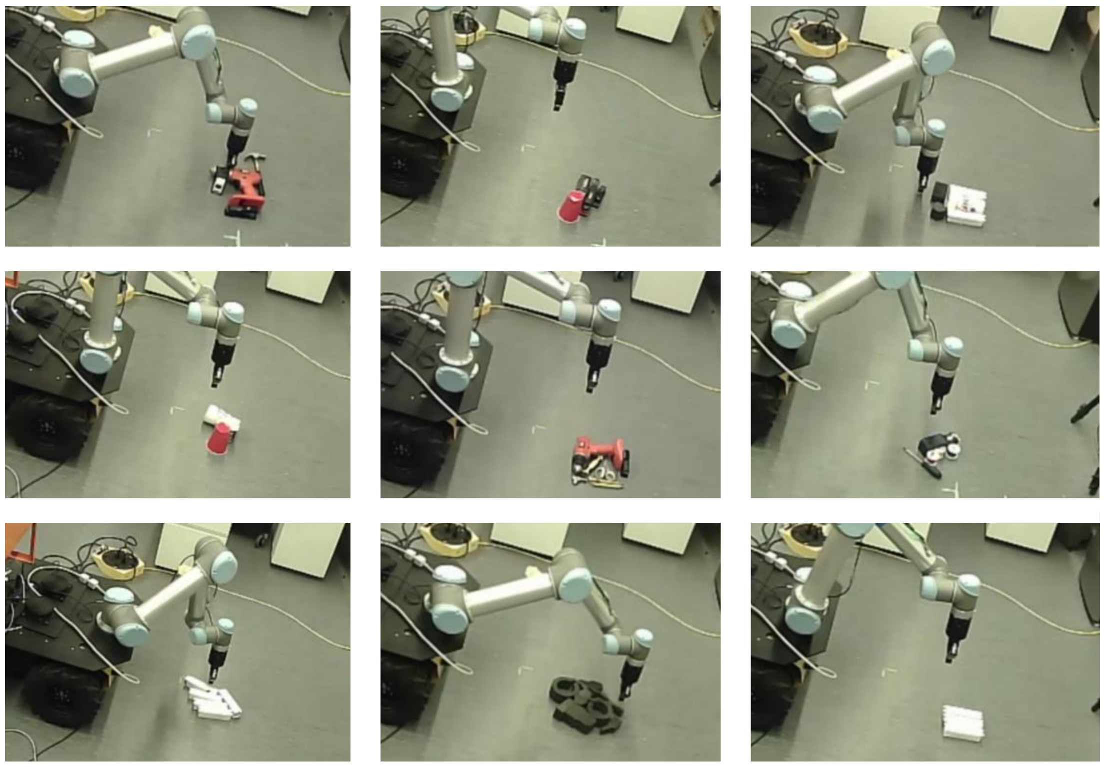


Figure 5.17: Generalization performance of the proposed DMP system on different challenging scenarios in real-world settings.

5.6 Summary

In this Chapter, we developed an end-to-end self-supervised manipulation framework that is able, in addition to grasping, to learn pregrasp in a strategic way that aids to increase the rate of grasp success. Our framework proposes optimal control strategies that approach grasping in challenging scenarios where objects necessarily need pregrasps assistance. Model-free DRL was used to build an autonomous agent which learns from scratch, from visual observations to sequential decision-making. The agent was trained on simple scenarios for a few hours. The acquired knowledge was, then, transferred to generalize on novel challenging scenarios without requiring predefined object information nor task-specific re-training data. The experiment's findings showed the significant promise of joint-learning for sequential policies to approach Shift-to-Grasp task based on minimal raw experience and simple objects. In which pregrasp manipulations assist to declutter objects and generate intended grasps. Besides this, grasping benefits efficiently by removing objects. In both simulation and real-world settings, the proposed system was successfully experimented and verified. The robot is efficiently able to handle different novel scenarios in challenging situations, without requiring fine-tuning in the implementation. A demo-video was provided to show the training scenario as well as successful performance of pregrasp manipulations and grasping actions. The model is agnostic to object identity where it operates without object segmentation or classification. As future works, it would be important to understand 6D manipulation policy.

Chapter 6. Conclusion and Future Work

6.1 Conclusions

In many industries, the use of autonomous mobile manipulators has increased, allowing these systems to handle, organise and proceed various assets autonomously. The two main industries that use this technology are production facilities and delivery services that manage a large inventory and benefit from a powerful and autonomous robot. Applied robots use sensing network technology and control algorithms to improve their efficiency in autonomy and usability. However, the sensory network technique in the mobile manipulation applications leads to complicated and expensive system. In Chapter 3, we addressed the challenge of employing only vision feedback to control a mobile manipulation system. We proposed a complete autonomous mobile manipulator system for flexible industrial production utilizing uncostly single camera. The proposed model has built-in vision system to achieve industrial requirements. For instance, pick and place, assembly, machine tending, packaging, inventory tracking, sorting, and transportation. The proposed system comprises of deep net model and 3D visual servoing process, incorporated in a sophisticated mobile manipulator. The complete framework is developed in two main steps: firstly, the perception network for recognising and estimating 3D target objects by using an effective model architecture of deep-CNN and pose estimation algorithms. Second, the information of the pose estimate was then used to control the movement of the AMM system in the 3D visual servoing scheme. The perception network was entirely trained on computer-generated datasets that could be reasonably produced to cover the object variants which are necessary for robotic manipulation applications. The datasets include sufficient possibilities of objects poses in different environments and illumination conditions. The synthetic-trained model of the perception network was, then, incorporated in real-life experiments without the need for post-refinements nor extra re-training. The findings were generalized successfully targeting different real-world objects in various backgrounds, occlusions events, and lighting conditions. Autonomous implementation, therefore, was performed with the visual servoing control law in 3D real-world space. The proposed system was physically tested (on 6 DOF manipulator arm mounted on a differential robot-base) in addition to simulation tests to extract the performance characteristics of the robot model. Video recordings were provided to show the experimentations of AMM system.

The ability of grasping unfamiliar objects (unknown during training) is essentially required to provide robust grasping in the applications of autonomous robotic manipulation. Recent solutions typically require either pre-defined information of target objects (e.g., pose estimation or 3D CAD models) or task-specific training data. In addition, millions of grasping attempts and massive training datasets might be required for the implementations. However, this makes the developed systems difficult to scale up for generalization on novel objects. Thus, Chapter 4 introduced a robotic grasping strategy based on the model-free deep reinforcement learning, named Deep Reinforcement Grasp Policy (DRGP). The proposed system uses simple geometric objects in training and generalizes efficiently on novel objects. It does not require pre-defined object information nor task-specific training data in implementations. The agent learns grasping strategies from random arrangements of limited simulated objects. The gained knowledge during simulation-training sessions was, then, implemented in real-life scenarios targeting unfamiliar objects. Learning process in Chapter 4 happens from scratch (starting from input pixels to joint velocities), to explore grasping strategies through trial-and-error manner. Experimentations (simulation and real-life) in Chapter 4 showed that the agent was well-versed in handling different scenarios of novel objects without fine-tuning.

Manipulating unknown objects is another challenge that accompanies the applications of robotic manipulation, which is also considered in this thesis. Inspired by humans' intuitive strategies that push apart the clutter of objects and achieve grasping. Conducting this task by robots could be one of the biggest bottlenecks in the field of autonomous manipulation. In practical advanced applications of household and service prospective robots, this challenge has become much more relevant and demanding. In challenging situations, robots are necessarily required to execute non-prehensile actions (e.g., pushing, toppling, squeezing, or rolling) in order to facilitate grasping. Classic solutions typically require agnostic pre-grasping actions prior to grasping. However, these solutions play a loose role and causing delays. The non-prehensile action should have intended utility and effect on the consecutive grasping action, because it is a sequential decision-making problem. Chapter 5 proposed a self-learning robotic system for robot-unfamiliar-object manipulation which can be applied for the purposes of independent household and service robots. For instance, home chores robots for the elderly and disabled. In Chapter 5, data-driven manipulation of shifting and grasping as sequences of acts was proposed and implemented, which synergizes to provide better grasp outcomes in such challenging scenarios. We showed in Chapter 5 the possibility of joint-learning for sequential robotic manipulation. The agent proved that pregrasp policies can be jointly supervised by future grasping policies (which are self-supervised), both of which are simultaneously trained. Practical findings showed the capabilities of the proposed system that learned directly from only visual observations to approach non-trivial sequential manipulation problem. Without the need for predefined object information nor task-specific training data. The agent showed the performance of real-world generalizations on adversarial scenarios based on minimal simulated training knowledge. In both simulation and real-world settings, the proposed system was successfully experimented and verified. The robot is efficiently able to handle different novel scenarios in challenging situations, without requiring fine-tuning in the implementation. A demo-video was provided to show the training scenario as well as successful performance of pregrasp manipulations and grasping actions. The model is agnostic to object identity where it operates without object segmentation or classification.

6.2 Recommendations and Future Work

Most of the existing mobile manipulator approaches utilize vision sensor with predefined calibration parameters. Our proposed system in Chapter 3 is particularly no different, hence, it would be interesting to set the system without the need to camera calibration parameters. While our experimentations have covered a wide variety of household target objects, further experiments aimed at transparent objects will also be interesting. This will open further possibilities to handle different set of target objects. AMM system developed in Chapter 3 is targeted to perform robot-object-interaction tasks for the industrial applications. However, the goal of applying actual object grasping was set as future work. For example, autonomous mobile manipulator integrated with pick-and-place task.

We use visual affordances (in Chapter 4 and Chapter 5) that are interpreted as pixel-wise Q-values to formulate shifting and grasping actions for manipulation tasks. This formulation could be extended to explore multi-task learning challenge which includes more complicated primitive actions of potential manipulations. For instance, twisting a doorknob, pushing to open a door, or dragging a drawer. This type of challenges requires global understanding of the physical properties of target objects and surroundings. Primitive actions for such defined tasks require successive reasoning and understanding because it is multi-stage sequential decision-making problem.

Our proposed visual perception network (for manipulating objects) was modelled to detect objects for 3D grasping task, included height information and single rotational degree. However, carrying out 6D grasping (included 3D translation and 3D orientation) would open further possibilities.

The proposed grasping and manipulating systems (Chapter 4 and Chapter 5) addressed the top-down object manipulation challenge in the experimentations. However, different scenarios like objects on the table or on the rack would require special modifications. The current system is unable to handle lateral object manipulation. However, since our system is scalable, approaching objects from the side is potentially possible. The depth-sensor should be place at the end-effector of the manipulator in order to capture the workspace from a pre-defined arm position. The arm should move first to the pre-defined position to observe the workspace and then decide the action.

Although, we focus on rich and diverse task which is grasping, our model-free framework potentially allows to extend the applicability by considering an object placement task. Since our system is scalable, it is possible to consider a place-learning network for object placement in addition to grasp-learning network. In this case, the agent is able to learn jointly grasp-to-place task by using the model-free reinforcement learning. The place-network could be modelled similarly to the grasp-network, but it has its own rewarding policy which enables the robot learning the task progressively. Perception network could have two convolutional neural networks, one for grasping and another one for placement. This results in two visual affordances pixel-wise Q-maps (one for grasping and one for placement). The two networks learn jointly because it's a sequential decision-making challenge.

In Chapter 5, we utilized only visual data in the self-learning scheme. Visual feedback is an important option which comes from the bigger family of perceptual modalities, such as tactile-sensor, force-and-torque, lidar, and motion sensors. While examination of an additional sensing channel might result in increased complexity, it would also be interesting to study the object-interaction manipulation.

Eye-to-hand camera installation was considered for object manipulation in Chapter 5, where a depth-camera was fixedly placed on a tripod observing the workspace including the robot. This requires precise camera calibration process in order to obtain the sensor parameters. If the camera position was slightly moved, manipulating with target objects would have affected. Two recommendations could be applied to avoid such an issue and meet specific requirements for manipulation application. First, we could have placed the camera at the bottom of the mobile robot and achieved one calibration process. This configuration could also benefit to carry out different robotic uses, for instance navigation purposes combined with objects manipulation. Second option could have worked by placing the camera at the end-effector of the arm. The manipulator arm needs first to move to certain pose to capture the workspace then decide the execution. In addition, further manipulation studies without the need to the calibration process would also be interesting.

Our proposed end-to-end grasping system learns in entirely self-supervised manner. This is different than current systems which separate the learning scheme and data collection step. It would be interesting in the future to train our system based on the precise human-annotated-data to consider grasp-oriented task to the target object rather than grasp-agnostic to the object. Another suggestion could be utilizing deep-learning net to learn and generalize from minimal raw experience collected by humans.

References

[1] A. Borji, M. M. Cheng, Q. Hou, H. Jiang, and J. Li, "Salient object detection: A survey," *Computational visual media,* vol. 5, no. 2, pp. 117-150, 2019.

[2] B. Mandal, "Appearance Based Robot and Human Activity Recognition System," *arXiv preprint arXiv:1602.01608,* 2016.

[3] P. Loncomilla, J. Ruiz-del-Solar, and L. Martínez, "Object recognition using local invariant features for robotic applications: A survey," *Pattern Recognition,* vol. 60, pp. 499-514, 2016.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision,* vol. 60, no. 2, pp. 91-110, 2004.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer vision and image understanding,* vol. 110, no. 3, pp. 346-359, 2008.

[6] A. Andreopoulos and J. K. Tsotsos, "50 years of object recognition: Directions forward," *Computer vision and image understanding,* vol. 117, no. 8, pp. 827-891, 2013.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition,* 2016, pp. 770-778.

[8] Z. Lei, D. Yi, and S. Z. Li, "Learning stacked image descriptor for face recognition," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 26, no. 9, pp. 1685-1696, 2015.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE transactions on pattern analysis and machine intelligence,* vol. 38, no. 1, pp. 142-158, 2015.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems,* vol. 25, pp. 1097-1105, 2012.

[11] P. Fischer, A. Dosovitskiy, and T. Brox, "Descriptor matching with convolutional neural networks: a comparison to sift," *arXiv preprint arXiv:1405.5769,* 2014.

[12] N. Kruger *et al.*, "Deep hierarchies in the primate visual cortex: What can we learn for computer vision?," *IEEE transactions on pattern analysis and machine intelligence,* vol. 35, no. 8, pp. 1847-1871, 2012.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition,* 2014, pp. 580-587.

[14] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision,* 2015, pp. 1440-1448.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence,* vol. 39, no. 6, pp. 1137-1149, 2016.

[16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition,* 2016, pp. 779-788.

[17] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263-7271.

[18] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961-2969.

[19] W. Liu *et al.*, "Ssd: Single shot multibox detector," in *European conference on computer vision*, 2016, pp. 21-37: Springer.

[20] J. Li, X. Liang, S. Shen, T. Xu, J. Feng, and S. Yan, "Scale-aware fast R-CNN for pedestrian detection," *IEEE transactions on Multimedia*, vol. 20, no. 4, pp. 985-996, 2017.

[21] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, "Object detection networks on convolutional feature maps," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1476-1481, 2016.

[22] L. Xie, T. Ahmad, L. Jin, Y. Liu, and S. Zhang, "A new CNN-based method for multi-directional car license plate detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 507-517, 2018.

[23] E. Simo-Serra, A. Ramisa, G. Alenyà, C. Torras, and F. Moreno-Noguer, "Single image 3D human pose estimation from noisy observations," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2673-2680: IEEE.

[24] D. Tome, C. Russell, and L. Agapito, "Lifting from the deep: Convolutional 3d pose estimation from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2500-2509.

[25] H. Yasin, U. Iqbal, B. Kruger, A. Weber, and J. Gall, "A dual-source approach for 3d pose estimation from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4948-4956.

[26] B. Ahn, D.-G. Choi, J. Park, and I. S. Kweon, "Real-time head pose estimation using multi-task deep neural network," *Robotics and Autonomous Systems*, vol. 103, pp. 1-12, 2018.

[27] J. Sock, S. Hamidreza Kasaei, L. Seabra Lopes, and T.-K. Kim, "Multi-view 6D object pose estimation and camera motion planning using RGBD images," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 2228-2235.

[28] M. Hayat, M. Bennamoun, and A. A. El-Sallam, "An RGB–D based image set classification for robust face recognition from Kinect data," *Neurocomputing*, vol. 171, pp. 889-900, 2016.

[29] C. Rodriguez-Garavito, G. Camacho-Munoz, D. Álvarez-Martínez, K. V. Cardenas, D. M. Rojas, and A. Grimaldos, "3D object pose estimation for robotic packing applications," in *Workshop on Engineering Applications*, 2018, pp. 453-463: Springer.

[30] M. Schwarz, H. Schulz, and S. Behnke, "RGB-D object recognition and pose estimation based on pre-trained convolutional neural network features," in *2015 IEEE international conference on robotics and automation (ICRA)*, 2015, pp. 1329-1335: IEEE.

[31] M. Tian, L. Pan, M. H. Ang Jr, and G. H. Lee, "Robust 6D Object Pose Estimation by Learning RGB-D Features," *arXiv preprint arXiv:2003.00188*, 2020.

[32] A. Al-Shanoon, H. Lang, and Y. Wang, "Vision-Based Hand Gesture Recognition With Deep Machine Learning for Visual Servoing," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2018, vol. 51814, p. V05BT07A050: American Society of Mechanical Engineers.

[33] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154-6162.

[34] S. Gidaris and N. Komodakis, "Object detection via a multi-region and semantic segmentation-aware cnn model," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1134-1142.

[35] J. Hung and A. Carpenter, "Applying faster R-CNN for object detection on malaria images," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 56-61.

[36] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 292-301.

[37] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun, "Cascaded pyramid network for multi-person pose estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7103-7112.

[38] T. T. Do, M. Cai, T. Pham, and I. Reid, "Deep-6dpose: Recovering 6d object pose from a single rgb image," *arXiv preprint arXiv:1802.10367*, 2018.

[39] J. Wu *et al.*, "Real-time object pose estimation with pose interpreter networks," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 6798-6805: IEEE.

[40] S. Yuan *et al.*, "Depth-based 3d hand pose estimation: From current achievements to future goals," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2636-2645.

[41] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv preprint arXiv:1711.00199*, 2017.

[42] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake, "Label fusion: A pipeline for generating ground truth labels for real rgbd data of cluttered scenes," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1-8: IEEE.

[43] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. 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)*, 2017, pp. 23-30: IEEE.

[44] J. Tremblay, T. To, and S. Birchfield, "Falling things: A synthetic dataset for 3d object detection and pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 2038-2041.

[45] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *2015 international conference on advanced robotics (ICAR)*, 2015, pp. 510-517: IEEE.

[46] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," *arXiv preprint arXiv:1809.10790*, 2018.

[47] R. Figueiredo, A. Dehban, P. Moreno, A. Bernardino, J. Santos-Victor, and H. Araújo, "A robust and efficient framework for fast cylinder detection," *Robotics and Autonomous Systems,* vol. 117, pp. 17-28, 2019.

[48] J. Tremblay *et al.*, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 969-977.

[49] F. Chaumette and S. Hutchinson, "Visual servo control. I. Basic approaches," *IEEE Robotics & Automation Magazine,* vol. 13, no. 4, pp. 82-90, 2006.

[50] F. Chaumette and S. Hutchinson, "Visual servo control. II. Advanced approaches [Tutorial]," *IEEE Robotics & Automation Magazine,* vol. 14, no. 1, pp. 109-118, 2007.

[51] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*. 2006.

[52] P. Corke, *Robotics, vision and control: fundamental algorithms in MATLAB® second, completely revised*. Springer, 2017.

[53] A. Ansar and K. Daniilidis, "Linear pose estimation from points or lines," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, no. 5, pp. 578-589, 2003.

[54] E. Marchand, H. Uchiyama, and F. Spindler, "Pose estimation for augmented reality: a hands-on survey," *IEEE transactions on visualization and computer graphics,* vol. 22, no. 12, pp. 2633-2651, 2015.

[55] D. Xu, J. Lu, P. Wang, Z. Zhang, and Z. Liang, "Partially decoupled image-based visual servoing using different sensitive features," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 47, no. 8, pp. 2233-2243, 2017.

[56] A. Cherubini, F. Spindler, and F. Chaumette, "Autonomous visual navigation and laser-based moving obstacle avoidance," *IEEE Transactions on Intelligent Transportation Systems,* vol. 15, no. 5, pp. 2101-2110, 2014.

[57] C. Teulière and E. Marchand, "A dense and direct approach to visual servoing using depth maps," *IEEE Transactions on Robotics,* vol. 30, no. 5, pp. 1242-1249, 2014.

[58] Q. Bateux, E. Marchand, J. Leitner, F. Chaumette, and P. Corke, "Training deep neural networks for visual servoing," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1-8: IEEE.

[59] E. Marchand, "Subspace-based direct visual servoing," *IEEE Robotics and Automation Letters,* vol. 4, no. 3, pp. 2699-2706, 2019.

[60] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1521-1529.

[61] M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3828-3836.

[62] R. A. Wilson, "Control system for automatic guided vehicles," ed: Google Patents, 1989.

[63] A. Al-Shanoon, A. H. Tan, H. Lang, and Y. Wang, "Mobile Robot Regulation with Position Based Visual Servoing," in *2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, 2018, pp. 1-6: IEEE.

[64] V. Lippiello *et al.*, "Hybrid visual servoing with hierarchical task composition for aerial manipulation," *IEEE Robotics and Automation Letters,* vol. 1, no. 1, pp. 259-266, 2016.

[65] C. Y. Tsai, C.-C. Wong, C.-J. Yu, C.-C. Liu, and T.-Y. Liu, "A hybrid switched reactive-based visual servo control of 5-DOF robot manipulators for pick-and-place tasks," *IEEE Systems Journal,* vol. 9, no. 1, pp. 119-130, 2015.

[66] G. Allibert, E. Courtial, and F. Chaumette, "Predictive control for constrained image-based visual servoing," *IEEE Transactions on Robotics,* vol. 26, no. 5, pp. 933-939, 2010.

[67] H. Shi, G. Sun, Y. Wang, and K.-S. Hwang, "Adaptive image-based visual servoing with temporary loss of the visual signal," *IEEE Transactions on Industrial Informatics,* vol. 15, no. 4, pp. 1956-1965, 2018.

[68] A. Hao Tan, A. Al-Shanoon, H. Lang, and M. El-Gindy, "Mobile Robot Regulation With Image Based Visual Servoing," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference,* 2018, vol. 51807, p. V05AT07A078: American Society of Mechanical Engineers.

[69] F. Janabi-Sharifi, L. Deng, and W. J. Wilson, "Comparison of basic visual servoing methods," *IEEE/ASME Transactions on Mechatronics,* vol. 16, no. 5, pp. 967-983, 2010.

[70] C. López-Franco, J. Hernández-Barragán, A. Y. Alanis, N. Arana-Daniel, and M. López-Franco, "Inverse kinematics of mobile manipulators based on differential evolution," *International Journal of Advanced Robotic Systems,* vol. 15, no. 1, p. 1729881417752738, 2018.

[71] S. Avilés *et al.*, "Simulation of a Mobile Manipulator on Webots," *International Journal of Online Engineering,* vol. 14, no. 2, 2018.

[72] A. T. Azar, H. H. Ammar, and H. Mliki, "Fuzzy Logic Controller ith Color Vision System Tracking for Mobile Manipulator Robot," in *International Conference on Advanced Machine Learning Technologies and Applications*, 2018, pp. 138-146: Springer.

[73] T. Kornuta and C. Zieliński, "Robot control system design exemplified by multi-camera visual servoing," *Journal of Intelligent & Robotic Systems,* vol. 77, no. 3-4, pp. 499-523, 2015.

[74] A. Herzog *et al.*, "Learning of grasp selection based on shape-templates," *Autonomous Robots,* vol. 36, no. 1-2, pp. 51-65, 2014.

[75] C. Goldfeder, M. Ciocarlie, H. Dang, and P. K. Allen, "The columbia grasp database," in *2009 IEEE international conference on robotics and automation*, 2009, pp. 1710-1716: IEEE.

[76] J. Mahler *et al.*, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *arXiv preprint arXiv:1703.09312,* 2017.

[77] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research,* vol. 36, no. 13-14, pp. 1455-1473, 2017.

[78] R. Detry, J. Papon, and L. Matthies, "Task-oriented grasping with semantic and geometric scene understanding," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 3266-3273: IEEE.

[79] K. Fang *et al.*, "Learning task-oriented grasping for tool manipulation from simulated self-supervision," *The International Journal of Robotics Research,* vol. 39, no. 2-3, pp. 202-216, 2020.

[80] M. Kokic, D. Kragic, and J. Bohg, "Learning Task-Oriented Grasping From Human Activity Datasets," *IEEE Robotics and Automation Letters,* vol. 5, no. 2, pp. 3352-3359, 2020.

[81] H. Han, G. Paul, and T. Matsubara, "Model-based reinforcement learning approach for deformable linear object manipulation," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, 2017, pp. 750-755: IEEE.

[82] M. Van der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, "Learning Continuous 3D Reconstructions for Geometrically Aware Grasping," *arXiv preprint arXiv:1910.00983*, 2019.

[83] L. Zaidi, J. A. Corrales, B. C. Bouzgarrou, Y. Mezouar, and L. Sabourin, "Model-based strategy for grasping 3D deformable objects using a multi-fingered robotic hand," *Robotics and Autonomous Systems*, vol. 95, pp. 196-206, 2017.

[84] J. Mahler *et al.*, "Learning ambidextrous robot grasping policies," *Science Robotics,* vol. 4, no. 26, 2019.

[85] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2901-2910.

[86] C. Wang *et al.*, "Densefusion: 6d object pose estimation by iterative dense fusion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3343-3352.

[87] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox, "Self-supervised 6d object pose estimation for robot manipulation," *arXiv preprint arXiv:1909.10159,* 2019.

[88] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research,* vol. 34, no. 4-5, pp. 705-724, 2015.

[89] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1316-1322: IEEE.

[90] S. Iqbal *et al.*, "Toward sim-to-real directional semantic grasping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 7247-7253: IEEE.

[91] K. Kleeberger, R. Bormann, W. Kraus, and M. F. Huber, "A survey on learning-based robotic grasping," *Current Robotics Reports*, pp. 1-11, 2020.

[92] S. Kumra and C. Kanan, "Robotic grasp detection using deep convolutional neural networks," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 2017, pp. 769-776: IEEE.

[93] D. Morrison, P. Corke, and J. Leitner, "Learning robust, real-time, reactive robotic grasping," *The International Journal of Robotics Research,* vol. 39, no. 2-3, pp. 183-201, 2020.

[94] J. Varley, J. Weisz, J. Weiss, and P. Allen, "Generating multi-fingered robotic grasps via deep learning," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 2015, pp. 4415-4420: IEEE.

[95] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on Robotics,* vol. 30, no. 2, pp. 289-309, 2013.

[96] J. Watson, J. Hughes, and F. Iida, "Real-world, real-time robotic grasping with convolutional neural networks," in *Annual Conference Towards Autonomous Robotic Systems*, 2017, pp. 617-626: Springer.

[97] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "Graspnet: A large-scale clustered and densely annotated datase for object grasping," *arXiv preprint arXiv:1912.13470,* 2019.

[98] P. Hebert *et al.*, "Combined shape, appearance and silhouette for simultaneous manipulator and object tracking," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 2405-2412: IEEE.

[99] L. Berscheid, P. Meißner, and T. Kröger, "Self-supervised learning for precise pick-and-place without object model," *IEEE Robotics and Automation Letters,* vol. 5, no. 3, pp. 4828-4835, 2020.

[100] R. Calandra *et al.*, "More than a feeling: Learning to grasp and regrasp using vision and touch," *IEEE Robotics and Automation Letters,* vol. 3, no. 4, pp. 3300-3307, 2018.

[101] E. Jang, S. Vijayanarasimhan, P. Pastor, J. Ibarz, and S. Levine, "End-to-end learning of semantic grasping," *arXiv preprint arXiv:1707.01932,* 2017.

[102] T. Mar, V. Tikhanoff, G. Metta, and L. Natale, "Self-supervised learning of grasp dependent tool affordances on the iCub Humanoid robot," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 3200-3206: IEEE.

[103] A. Murali, Y. Li, D. Gandhi, and A. Gupta, "Learning to grasp without seeing," in *International Symposium on Experimental Robotics*, 2018, pp. 375-386: Springer.

[104] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE international conference on robotics and automation (ICRA)*, 2016, pp. 3406-3413: IEEE.

[105] Q. Shao *et al.*, "Suction Grasp Region Prediction using Self-supervised Learning for Object Picking in Dense Clutter," in *2019 IEEE 5th International Conference on Mechatronics System and Robots (ICMSR)*, 2019, pp. 7-12: IEEE.

[106] S. Song, A. Zeng, J. Lee, and T. Funkhouser, "Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations," *IEEE Robotics and Automation Letters,* vol. 5, no. 3, pp. 4978-4985, 2020.

[107] X. Yan *et al.*, "Data-efficient learning for sim-to-real robotic grasping using deep point cloud prediction networks," *arXiv preprint arXiv:1906.08989,* 2019.

[108] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research,* vol. 37, no. 4-5, pp. 421-436, 2018.

[109] E. Matsumoto, M. Saito, A. Kume, and J. Tan, "End-to-end learning of object grasp poses in the Amazon Robotics Challenge," in *Advances on Robotic Item Picking*: Springer, 2020, pp. 63-72.

[110] M. Breyer, F. Furrer, T. Novkovic, R. Siegwart, and J. Nieto, "Flexible Robotic Grasping with Sim-to-Real Transfer based Reinforcement Learning," *ArXiv,* vol. abs/1803.04996, 2018.

[111] H. Liang, X. Lou, and C. Choi, "Knowledge Induced Deep Q-Network for a Slide-to-Wall Object Grasping," *arXiv preprint arXiv:1910.03781,* 2019.

[112] M. Gualtieri, A. Ten Pas, K. Saenko, and R. Platt, "High precision grasp pose detection in dense clutter," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 598-605: IEEE.

[113] M. Gualtieri, A. ten Pas, and R. Platt, "Pick and place without geometric object models," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7433-7440: IEEE.

[114] G. Xiang and J. Su, "Task-Oriented Deep Reinforcement Learning for Robotic Skill Acquisition and Control," *IEEE Transactions on Cybernetics,* 2019.

[115] F. Sadeghi and S. Levine, "Cad2rl: Real single-image flight without a single real image," *arXiv preprint arXiv:1611.04201,* 2016.

[116] B. Luo, Y. Yang, and D. Liu, "Adaptive Q-Learning for Data-Based Optimal Output Regulation With Experience Replay," *IEEE transactions on cybernetics,* vol. 48, no. 12, pp. 3337-3348, 2018.

[117] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*, 2017, pp. 3389-3396: IEEE.

[118] A. Ghadirzadeh, A. Maki, D. Kragic, and M. Björkman, "Deep predictive policy training using reinforcement learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2351-2358: IEEE.

[119] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, 2016, vol. 30, no. 1.

[120] Z. Chen, M. Lin, Z. Jia, and S. Jian, "Towards Generalization and Data Efficient Learning of Deep Robotic Grasping," *arXiv preprint arXiv:2007.00982,* 2020.

[121] H. Jo and J.-B. Song, "Object-Independent Grasping in Heavy Clutter," *Applied Sciences,* vol. 10, no. 3, p. 804, 2020.

[122] B. Sauvet, F. Lévesque, S. Park, P. Cardou, and C. Gosselin, "Model-based grasping of unknown objects from a random pile," *Robotics,* vol. 8, no. 3, p. 79, 2019.

[123] S. Song, A. Zeng, J. Lee, and T. Funkhouser, "Grasping in the Wild: Learning 6DoF Closed-Loop Grasping from Low-Cost Demonstrations," *arXiv preprint arXiv:1912.04344,* 2019.

[124] A. Sahbani, S. El-Khoury, and P. Bidaud, "An overview of 3D object grasp synthesis algorithms," *Robotics and Autonomous Systems,* vol. 60, no. 3, pp. 326-336, 2012.

[125] J. Weisz and P. K. Allen, "Pose error robust grasping from contact wrench space metrics," in *2012 IEEE international conference on robotics and automation*, 2012, pp. 557-562: IEEE.

[126] D. Quillen, E. Jang, O. Nachum, C. Finn, J. Ibarz, and S. Levine, "Deep reinforcement learning for vision-based robotic grasping: A simulated comparative evaluation of off-policy methods," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6284-6291: IEEE.

[127] F. Qin, D. Xu, D. Zhang, and Y. Li, "Robotic skill learning for precision assembly with microscopic vision and force feedback," *IEEE/ASME Transactions on Mechatronics,* vol. 24, no. 3, pp. 1117-1128, 2019.

[128] S. Joshi, S. Kumra, and F. Sahin, "Robotic grasping using deep reinforcement learning," in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, 2020, pp. 1461-1466: IEEE.

[129] D. Morrison, P. Corke, and J. Leitner, "EGAD! an Evolved Grasping Analysis Dataset for diversity and reproducibility in robotic manipulation," *IEEE Robotics and Automation Letters,* 2020.

[130] O. Taheri, N. Ghorbani, M. J. Black, and D. Tzionas, "GRAB: A dataset of whole-body human grasping of objects," in *European Conference on Computer Vision*, 2020, pp. 581-600: Springer.

[131] M. Van der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, "Learning continuous 3d reconstructions for geometrically aware grasping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 11516-11522: IEEE.

[132] L. Yang *et al.*, "Rigid-soft interactive learning for robust grasping," *IEEE Robotics and Automation Letters,* vol. 5, no. 2, pp. 1720-1727, 2020.

[133] R. Li and H. Qiao, "A Survey of Methods and Strategies for High-Precision Robotic Grasping and Assembly Tasks—Some New Trends," *IEEE/ASME Transactions on Mechatronics,* vol. 24, no. 6, pp. 2718-2732, 2019.

[134] L. Berscheid, P. Meißner, and T. Kröger, "Robot learning of shifting objects for grasping in cluttered environments," *arXiv preprint arXiv:1907.11035,* 2019.

[135] L. Pinto and A. Gupta, "Learning to push by grasping: Using multiple tasks for effective learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2161-2168: IEEE.

[136] S. Goyal, A. Ruina, and J. Papadopoulos, "Planar sliding with dry friction part 1. limit surface and moment function," *Wear,* vol. 143, no. 2, pp. 307-330, 1991.

[137] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *The International Journal of Robotics Research,* vol. 5, no. 3, pp. 53-71, 1986.

[138] Y. Li, Y. Chen, Y. Yang, and Y. Li, "Soft robotic grippers based on particle transmission," *IEEE/ASME Transactions on Mechatronics,* vol. 24, no. 3, pp. 969-978, 2019.

[139] M. R. Dogar and S. S. Srinivasa, "A planning framework for non-prehensile manipulation under clutter and uncertainty," *Autonomous Robots,* vol. 33, no. 3, pp. 217-236, 2012.

[140] I. Clavera, D. Held, and P. Abbeel, "Policy transfer via modularity and reward guiding," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1537-1544: IEEE.

[141] D. Omrčen, C. Böge, T. Asfour, A. Ude, and R. Dillmann, "Autonomous acquisition of pushing actions to support object grasping with a humanoid robot," in *2009 9th IEEE-RAS International Conference on Humanoid Robots*, 2009, pp. 277-283: IEEE.

[142] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4238-4245: IEEE.

[143] A. Boularias, J. Bagnell, and A. Stentz, "Learning to manipulate unknown objects in clutter by reinforcement," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015, vol. 29, no. 1.

[144] S. Manschitz, "Learning Sequential Skills for Robot Manipulation Tasks," Technische Universität, 2018.

[145] M. Danielczuk, J. Mahler, C. Correa, and K. Goldberg, "Linear push policies to increase grasp access for robot bin picking," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 2018, pp. 1249-1256: IEEE.

[146] A. Eitel, N. Hauff, and W. Burgard, "Learning to singulate objects using a push proposal network," in *Robotics Research*: Springer, 2020, pp. 405-419.

[147] P. Ni, W. Zhang, H. Zhang, and Q. Cao, "Learning efficient push and grasp policy in a totebox from simulation," *Advanced Robotics*, pp. 1-15, 2020.

[148] J. Stüber, C. Zito, and R. Stolkin, "Let's Push Things Forward: A Survey on Robot Pushing," *Frontiers in Robotics and AI*, vol. 7, p. 8, 2020.

[149] V. Annem, P. Rajendran, S. Thakar, and S. K. Gupta, "Towards remote teleoperation of a semi-autonomous mobile manipulator system in machine tending tasks," in *International Manufacturing Science and Engineering Conference*, 2019, vol. 58745, p. V001T02A027: American Society of Mechanical Engineers.

[150] A. Dömel, S. Kriegel, M. Kaßecker, M. Brucker, T. Bodenmüller, and M. Suppa, "Toward fully autonomous mobile manipulation for industrial environments," *International Journal of Advanced Robotic Systems*, vol. 14, no. 4, p. 1729881417718588, 2017.

[151] F. Chen, M. Selvaggio, and D. G. Caldwell, "Dexterous grasping by manipulability selection for mobile manipulator with visual guidance," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 1202-1210, 2018.

[152] G. Z. Gandler, C. H. Ek, M. Björkman, R. Stolkin, and Y. Bekiroglu, "Object shape estimation and modeling, based on sparse Gaussian process implicit surfaces, combining visual data and tactile exploration," *Robotics and Autonomous Systems*, vol. 126, p. 103433, 2020.

[153] C. Wang *et al.*, "Learning Mobile Manipulation through Deep Reinforcement Learning," *Sensors*, vol. 20, no. 3, p. 939, 2020.

[154] M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter, "Whole-body mpc for a dynamically stable mobile manipulator," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3687-3694, 2019.

[155] V. Pilania and K. Gupta, "Mobile manipulator planning under uncertainty in unknown environments," *The International Journal of Robotics Research*, vol. 37, no. 2-3, pp. 316-339, 2018.

[156] S. E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016, pp. 4724-4732.

[157] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision,* vol. 81, no. 2, p. 155, 2009.

[158] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.

[159] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp. 248-255: Ieee.

[160] A. Paszke *et al.*, "Automatic differentiation in pytorch," 2017.

[161] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980,* 2014.

[162] C. T. Kelley, *Iterative methods for optimization*. SIAM, 1999.

[163] Y. Wang, H. Lang, and C. De Silva, "A hybrid visual servoing controller for robust manipulation using mobile robots," *IEEE/ASME Transactions on Mechatronics,* vol. 15, no. 5, pp. 757-769, 2010.

[164] M. Arbabmir and M. Ebrahimi, "Visual–inertial state estimation with camera and camera–IMU calibration," *Robotics and Autonomous Systems,* vol. 120, p. 103249, 2019.

[165] R. Jonschkowski, C. Eppner, S. Höfer, R. Martín-Martín, and O. Brock, "Probabilistic multi-class segmentation for the amazon picking challenge," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 1-7: IEEE.

[166] A. Saxena, J. Driemeyer, J. Kearns, and A. Y. Ng, "Robotic grasping of novel objects," in *Advances in neural information processing systems*, 2007, pp. 1209-1216.

[167] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE transactions on cybernetics,* 2020.

[168] D. Kalashnikov *et al.*, "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," *arXiv preprint arXiv:1806.10293,* 2018.

[169] W. Sheng, A. Thobbi, and Y. Gu, "An integrated framework for human–robot collaborative manipulation," *IEEE transactions on cybernetics,* vol. 45, no. 10, pp. 2030-2041, 2014.

[170] K. Shao, Z. Tang, Y. Zhu, N. Li, and D. Zhao, "A survey of deep reinforcement learning in video games," *arXiv preprint arXiv:1912.10944,* 2019.

[171] W. Woof and K. Chen, "Learning to play general video-games via an object embedding network," in *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, 2018, pp. 1-8: IEEE.

[172] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602,* 2013.

[173] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[174] E. Rohmer, S. P. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1321-1326: IEEE.

[175] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *nature,* vol. 518, no. 7540, pp. 529-533, 2015.

[176] A. Zeng *et al.*, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *2018 IEEE international conference on robotics and automation (ICRA)*, 2018, pp. 1-8: IEEE.

[177] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431-3440.

[178] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700-4708.

[179] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952,* 2015.

[180] I. Popov *et al.*, "Data-efficient deep reinforcement learning for dexterous manipulation," *arXiv preprint arXiv:1704.03073,* 2017.

[181] M. Bauza and A. Rodriguez, "A probabilistic data-driven model for planar pushing," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3008-3015: IEEE.

[182] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2786-2793: IEEE.

[183] F. R. Hogan and A. Rodriguez, "Feedback control of the pusher-slider system: A story of hybrid and underactuated contact dynamics," in *Algorithmic Foundations of Robotics XII*: Springer, 2020, pp. 800-815.

[184] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[185] Y. Yang, H. Liang, and C. Choi, "A Deep Learning Approach to Grasping the Invisible," *IEEE Robotics and Automation Letters,* vol. 5, no. 2, pp. 2232-2239, 2020.