

**Towards the Vision of a Social Robot in every Home:
A Navigation Strategy via Enhanced Subsumption
Architecture**

**by
Kamal M. Othman**

M.A.Sc., Simon Fraser University, 2013

B.Sc., Umm Al-Qura University, 2004

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
School of Mechatronic Systems Engineering
Faculty of Applied Sciences

© Kamal Othman 2020
SIMON FRASER UNIVERSITY
Summer 2020

Approval

Name: Kamal M. Othman

Degree: Doctor of Philosophy

Title: Towards the Vision of a Social Robot in every Home: A Navigation Strategy via Enhanced Subsumption Architecture

Examining Committee:

Chair: Dr. Jason Wang
Assistant Professor

Dr. Ahmad Rad
Senior Supervisor
Professor

Dr. Edward Park
Supervisor
Professor

Dr. Mehrdad Moallem
Supervisor
Professor

Dr. Kamal Gupta
Internal Examiner
Professor
School of Engineering Science

Dr. Joshua Marshall
External Examiner
Associate Professor
Department of Electrical and Computer Engineering
Queen's University

Date Defended/Approved: June 29, 2020

Abstract

In this thesis, we report the studies undertaken in the design and implementation of a behavioristic navigation system for social robots with limited sensors to be deployed in family homes. The project was completed in four phases. Each phase of the project was independently evaluated in virtual or real-time implementation on the NAO humanoid robot.

In the first phase of this research study, we address the problem of indoor room classification via several convolutional neural network (CNN) architectures. The main objective was to recognize different rooms in a family home. We also propose and examine a combination model of CNN and a multi-binary classifier referred to as Error Correcting Output Code (ECOC).

In the second phase, we propose a new dataset referred to as SRIN, which stands for Social Robot Indoor Navigation. This dataset consists of 2D colored images for room classification (termed SRIN-Room) and doorway detection (termed SRIN-Doorway). The main feature of the SRIN dataset is that its images have been purposefully captured for short robots (around 0.5-meter tall). The methodology of collecting SRIN was designed in a way that facilitated generating more samples in the future regardless of where the samples have come from.

In phase three, we propose a novel algorithm to detect a door and its orientation in indoor settings from the view of a social robot equipped with only a monocular camera. The proposed system is designed through the integration of several modules, each of which serves a special purpose.

Finally, we report an end-to-end navigation system for social robots in family homes. The system combines a reactive-based system and a knowledge-based system with learning capabilities in a meaningful manner for social robot applications.

Keywords: Social Robots; Behavioristic Navigation; Subsumption Architecture; Deep Learning; Reinforcement Learning; Indoor Localization

To the soul of my father

Acknowledgements

In the name of Allah, most gracious and most merciful

I express my deepest appreciation to my supervisor, Professor Ahmad Rad, for accepting me to be part of his team and for guiding and supporting me whenever I needed him. I am grateful to have such a caring and understanding supervisor. The academic support during writing my papers and my thesis has been indispensable. Dr. Rad, you helped me to grow academically and personally throughout the years of my MSc and PhD. Thank you for everything! I could not have imagined having a better supervisor and mentor during my Ph.D.

I would like to thank my committee members, Professor Edward Park and Professor Mehrdad Moallem, for their valuable input and guidance. I still remember the first time that they encouraged me to learn and study deep learning and adopting it in my work, which became a passion during my research.

I would like to express my thanks and gratitude to Professor Kamal Gupta, and Dr. Joshua Marshall, for accepting to be my examiners, for reviewing my thesis and providing valuable feedback. Thanks also to Dr. Jason Jiacheng Wang for being the chair of my thesis defence session.

I am grateful to the Ministry of Higher Education in Saudi Arabia for the generous Doctoral funding that has allowed me to pursue my studies in Canada. To my dear friends at the Autonomous and Intelligent System Laboratory (AISL), Mehran, Mehdi, Mohammad, Elfituri, and Armina, many thanks for your help and advice during this journey.

My mom, our daily calls encouraged/motivated me and provided me with endless emotional support. To my dear wife, Abeer, you are amazing. I am so proud of you and jealous too that you were able to complete your Ph.D. degree before me. Dear Abeer, I am fortunate to have you by my side. I would not be able to complete my degree without your support and inspiration.

Table of Contents

Approval	ii
Abstract	iii
Dedication	iv
Acknowledgements.....	v
Table of Contents.....	vi
List of Tables.....	ix
List of Figures.....	x
Chapter 1. Introduction	1
1.1. Background and Motivation	1
1.2. Research Outline	3
1.3. Organization of The Thesis.....	7
1.4. Publications.....	8
Chapter 2. Literature Review	9
2.1. Behavioristic Approach	15
2.1.1. From Biological to Robotic Behaviors	15
2.1.2. Subsumption: An Architecture of Behavioristic Robotics System	17
2.2. Deep Learning	20
2.2.1. CNN for Robotics Applications	25
2.3. Reinforcement Learning	32
2.3.1. RL for Robotic Navigation Applications.....	36
2.4. Conclusion	38
Chapter 3. An Indoor Room Classification System for Social Robots via Integration of CNN and ECOC	39
3.1. Introduction	39
3.2. Related Work	40
3.3. Methodology.....	44
3.3.1. Phase 1: Training and Validation	44
Adopted Scene dataset.....	44
CNN architectures	47
3.3.2. Phase 2: Models Integration.....	48
ECOC Technique	48
3.3.3. Phase 3: Real-Time Implementation.....	50
3.4. Experiments and Results.....	50
3.4.1. Phase1: Validating Room Classification within CNN Models Using Places..	50
3.4.2. Phase 2: Validating Room Classification within the integration of CNN and ECOC	51
Discussion	52
3.4.3. Phase 3: Validating Room Classification on Real-Time Implementation Using Nao robot.....	53
Discussion	57

3.5.	Conclusion	58
Chapter 4. SRIN: A New Dataset for Social Robot Indoor Navigation.....		59
4.1.	Introduction	59
4.2.	Related work	60
4.3.	Collecting Dataset and Methodology.....	62
4.4.	Experiments & Results	65
	Discussion	67
4.5.	Conclusion	67
Chapter 5. Doorway Detection and Direction (3Ds) System for Social Robots via a Monocular Camera		69
5.1.	Introduction	69
5.2.	Related Research	70
5.3.	Proposed System and Methodology	74
	5.3.1. 2D image from Nao monocular camera	75
	5.3.2. CNN-SRIN Doorway Module.....	76
	5.3.3. Depth Module	76
	5.3.4. Pixel-Selection Module.....	77
	5.3.5. Pixel2Angle Module.....	79
5.4.	Experiments and Results.....	80
	5.4.1. Stage 1: CNN-SRIN for Doorway Detection	81
	5.4.2. Stage 2: Angle Extraction from 2D Images Based on Depth Map and Pixel Selection.....	83
	5.4.3. Validating the Overall performance of 3Ds-System in Real-Time Experiments with Nao humanoid robot	85
	Discussion	89
5.5.	Conclusion	91
Chapter 6. Sequential Localizing and Mapping: A navigation strategy via Enhanced Subsumption Architecture.....		92
6.1.	Introduction	92
6.2.	Related Research	93
6.3.	Methodology and Proposed System	98
	6.3.1. Subsumption-Based System.....	100
	Layer 1 - Exploration Task	100
	Layer 2 - Purposive Task	101
	Direction Detection & Go Towards Doorway Modules:	105
	Layer 3 and 4: Achievement and Protective Tasks	106
	6.3.2. Knowledge-Based System	107
	Room Localization & Doorway Detection Modules: (CNN-Based Models).....	107
	Doorway Passing Module: (Canny Edge Detection & Hough Transform)	107
	Topo-Room Module: (A Directional Semantic Topological Map)	108
	6.3.3. Implementation Setup.....	111
6.4.	Experiments and Results.....	112

6.4.1.	Evaluation of Individual Modules.....	112
	Evaluating RL system for obstacle avoidance module	112
	Evaluating doorway edges for “Passing Doorway” module	116
6.4.2.	Evaluation of the Overall System	116
	Scenario 1 – Moving between two rooms with no obstacles.....	116
	Scenario 2 – Moving between two rooms with a different direction	119
	Scenario 3 – Moving between two rooms with obstacles and a final goal	122
	Discussion	127
6.5.	Conclusion	129
Chapter 7. Contributions and Future Work		131
7.1.	Contributions	131
7.2.	Future Work	133
References.....		136
Appendix. Platform & Simulator		159
Nao Humanoid Robot – Hardware		159
	Sonar:	161
	Camera:.....	161
Nao Humanoid Robot – Software.....		162
Webots Simulator.....		164
Nao’s Research Areas		164

List of Tables

Table 2-1:	Summarizing and classifying selected papers of object detection for robotic applications based on CNN architectures since 2016. (S: Simulation & R: Real-Time Experiments).....	31
Table 3-1:	Number of images for each class after cleaning.....	46
Table 3-2:	Comparison of accuracies of fine-tuning different CNN models using all data and clean data (the shaded results are the best for the real-time experiment).....	51
Table 3-3:	Accuracies of 15 binary classifiers.....	52
Table 3-4:	Testing accuracies of all models with top-1 (T1) and top-2 (T2) for all five classes.....	56
Table 3-5:	Confusion matrix of testing images with top-1 & top-2 for five classes..	57
Table 4-1:	Comparison Results: confusion matrix of each class with top1 & top2 predictions for both models (CNN with places & CNN-SRIN) on a Nao robot.....	66
Table 5-1:	CNN-SRIN doorway prediction results on Nao. Shaded results are the false prediction.....	82
Table 5-2:	Confusion Matrix. (TN: True Negative, TP: True Positive, FP: False Positive, FN: False Negative).....	83
Table 5-3:	Real-time experiment results: Depth to angel values for controlling Nao robot.....	85
Table 5-4:	Results of real-time experiments with Nao at AISL in SFU, BC.....	88
Table 5-5:	Qualitative comparison of related algorithms.....	90
Table 6-1:	Qualitative comparison between classic SLAM and SeqLAM.....	129

List of Figures

Figure 1-1:	The design flow of the thesis.....	6
Figure 2-1:	Related research areas to the social robots and the scope of this thesis (shaded).....	14
Figure 2-2:	An illustration of Subsumption architecture with Layers and possible components. (B: Behavior, SF: Sense Function, AF: Act Function, FSM: Finite State Machine, Si: Inhibitor, Ss: Suppressor, R: Releaser, S: Subsumed layer by a higher layer).....	18
Figure 2-3:	Deep learning approaches within machine learning area. CNN is the adopted approach in this thesis.....	21
Figure 2-4:	Fundamental components of CNN architecture.....	22
Figure 2-5:	Mathematical model illustration for each layer in the ConvNet.....	24
Figure 2-6:	The concept of RL.....	33
Figure 2-7:	Temporal Difference algorithms of RL.....	35
Figure 3-1:	Process of simulation and real-time experiments for room classification problem.....	44
Figure 3-2:	Examples of removed images from room dataset of Places. (a) Not belonged to; (b) no furniture; (c) not wide scene; (d) multi-scenes in an image; (e) including texts; (f) fake scene; (g) focusing on people.....	46
Figure 3-3:	Concept of inception architecture.....	47
Figure 3-4:	Error correcting output code (ECOC) process for addressing multi class learning problems. (a) Stage 1: Process of training all binary classifier, and (b) Stage 2: Process of predicting a class of a new image.....	49
Figure 3-5:	Nao humanoid robot during experiments.....	54
Figure 3-6:	Scenes examples taken by Nao humanoid robot.....	55
Figure 4-1:	Nao robot's height with respect to the door handle.....	60
Figure 4-2:	Collecting Process of SRIN Dataset.....	63
Figure 4-3:	Scene samples from SRIN dataset for each category.....	64
Figure 4-4:	The concept of transfer learning of CNN. ConvNet: any pre-trained Convolutional Network & FC: Fully Connected Network.....	65
Figure 5-1:	A block diagram of Sense-Perception-Action control architecture.....	75
Figure 5-2:	Our proposed robotic system for doorway detection and direction (3Ds-system).....	75
Figure 5-3:	Field of View of Nao robot cameras [171].....	76
Figure 5-4:	Depth-Dense Network [207]. The figure is modified for the explanation purpose.....	77
Figure 5-5:	Illustration of correlation and selection of the best pixel.....	78
Figure 5-6:	Pixel2angle module for Nao robot.....	80
Figure 5-7:	Examples of three different scenarios for evaluating 3Ds-system with Nao in real-time experiments.....	86

Figure 5-8:	Pseudocode of implementing the Doorway Detection and Direction system.....	87
Figure 6-1:	Learning-based behavioristic system for homes.....	99
Figure 6-2:	Illustration of robot's direction while exploring.....	101
Figure 6-3:	State's design of RL for an obstacle avoidance module.....	102
Figure 6-4:	Reward's design of RL for an obstacle avoidance module.....	102
Figure 6-5:	Action's design of RL for an obstacle avoidance module.	103
Figure 6-6:	The design of cautious actions.	104
Figure 6-7:	The impact of cautious action with RL system.	105
Figure 6-8:	Doorway's edges detection for "Passing Doorway" module.	108
Figure 6-9:	Main attributes or information within each node in the TDS-Map.....	108
Figure 6-10:	Flowchart of creating and updating the Directional Semantic Topological Map (DST-Map).	110
Figure 6-11:	An illustration of the expected DTS-Map (right) for a simple layout (left).	110
Figure 6-12:	Pseudocode of implementing the overall system sequentially.	111
Figure 6-13:	An apartment virtual model with Nao robot using the Webots simulator.	112
Figure 6-14:	RL training results for both methods: a) Q-learning and b) SARSA.	115
Figure 6-15:	Doorway edges detection for "Passing Doorway" module.....	116
Figure 6-16:	Evaluating the overall system in scenario 1.	119
Figure 6-17:	Evaluating the overall system in scenario 2.	122
Figure 6-18:	Evaluating the overall system in scenario 3.	127

Chapter 1.

Introduction

1.1. Background and Motivation

While not privy to the future and in the absence of a crystal ball, we can only conjecture how our homes will have evolved in twenty-years. The current evidence, though, suggests that the 6G (or higher) technology, Internet of Things (IoT), AI, and Robotics will have had an irreversible and profound impact in all aspects of our life including our homes. We may have a social robot at home that recognizes every member of the family, communicates in natural human language, assists the household with their chores, communicates with appliances, orders goods on-line, monitors our health and security, acts as a companion, and above all, becomes a member of the family with all its implications. Back in 2007, Bill Gates [1] prophesied that there will be an autonomous robot in every home. It is unlikely that this prediction will be realized in this decade; perhaps 2040 is a more realistic date.

The central premise of this thesis, irrespective of the actual time for the realization of social robots in every home, is that mobility and the ability to roam around a house and recognize different rooms and their respective functions must be fundamental in social robots. We refer to this process as a navigation strategy. An informed reader may suggest that this problem is widely studied within the Simultaneous Localization and Mapping (SLAM) and questions the contribution. We agree that the problem is widely studied. Let us go through our motivation to study this problem: First, there are still many open problems in SLAM including data association and automatic landmark assignment. Second, SLAM is formulated within the probabilistic robotics paradigm whereby the pose of the robot and the map coordinates environment are concurrently estimated via one estimation vector. Third, SLAM is counter-intuitive to humans. Consider a scenario that you are in one of the rooms in a house such as a kitchen. You do not update iteratively your position in the kitchen, nor you update the kitchen map. Fourth, in many SLAM algorithms, the robot is manually driven through an environment. Although this latter problem is addressed via autonomous SLAM and Exploration strategies, a small percentage of SLAM papers actually deal with this issue.

Finally, SLAM in dynamic settings has its own challenges and is the subject of further research. To alleviate some of the above shortcomings, the studies in this thesis are based on behavioristic robotics and inspired by humans and animals. As a parallel to SLAM, we coin the term Sequential Localization and Mapping (SeqLAM). Here, we define zones (kitchen, bathroom, etc.); as such, we are not concerned with minute details of updating the robot position in a particular zone. Humans and their pets know a priori that a home has different zones (kitchen, bathroom, bedroom, etc.). The social robot also has this information beforehand. Once, the robot establishes that it is in the kitchen, it remembers its location unless it passes through a door. The spatial relation (map) of two adjacent locations (kitchen and living room, for instance) is determined after the robot recognizes its new location. The idea can be encapsulated in the same manner a human (or a domestic pet) moves from one room (zone) to another. We view SeqLAM as an alternative approach with certain benefits – predominantly in indoor and structured settings.

The research in social robots has substantially increased in the last decade. The social robotics market, including but not limited to domestic, assistance, or entertainment, is projected to grow from USD 3.3 billion in 2019 to USD 9.1 billion by 2024, with a Compound Annual Growth Rate (CAGR) of 22.4% [2]. This estimated growth is understandable as it is expected to happen concurrently with other important statistics in the world. According to the World Health Organization (WHO), the growing number of people aged 65 or older is expected to increase from 524 million in 2010 to nearly 1.5 billion in 2050 [3]. Another fact is also by WHO that mental disorders affect a person in every four people [4]. We argue that social robots will significantly improve health monitoring inside our homes.

Early generations of social robots are already available and are employed in diverse applications, such as eldercare [5], autism [6], supporting children with cancer [7], education [8], [9], or entertainment [10], etc. However, there are still many open questions that need to be resolved before the realization of a social robot in every home becomes practical. Here, we are concerned with two research questions: navigation strategies, and low-cost implementations. The former refers to the capability of a social robot to explore or navigate in an unknown but structured indoor setting, especially in apartments. We will discuss various schemes and state-of-art in robotics navigations in chapters 2 and 6. However, suffice to say in this section that we take on the problem by

augmenting the well-known Brooks' Subsumption architecture [11], which is a classification of behavior-based robotics. Perceiving knowledge and its crude replication by machine have been the on-going motivation and obsession for the behaviour-based robotics researchers. The subsumption architecture is inherently a hierarchical decision-making process whereby an input sensor triggers a corresponding action in a bottom-up manner. The original subsumption was static and deterministic implying that "it did not learn". By adding a learning module, we have transformed the system into a dynamic learning decision-making process. As such, we break the navigation process into a collection of simpler behaviors organized in layers such that higher layers subsume the lower ones. The learned knowledge is used to revisit places and to facilitate path planning in future research such as "go to kitchen and bring a cup of tea". This part, of course, is outside the scope of the thesis.

Social robots must become affordable to find their place in every home. There was a recent survey by Brookings Institution through an online U.S. national poll of 2,021 adult Internet users [12]. It reported that 42% of the users would pay \$250 or less, whereas 10% said that they would pay between \$251 and \$500. Correspondingly, robotics companies started designing small-sized social robots with limited sensors, in which their average cost was around CAD\$1000, such as Buddy and Kuri [13]. Except for very expensive social robots that provide SLAM and extensive proprietary firmware/software and equipped with state-of-art vision and depth sensors, the problem of navigation for low-cost social robots has rarely been reported. In this research, we focus on robots with "limited" sensors and mainly use monocular vision as the sensory input.

1.2. Research Outline

The research project was completed in four interrelated phases as shown in Figure 1-1. In phase 1, we addressed the problem of indoor room classification via several Convolutional Neural Network (CNN) architectures. The main objective was to recognize five indoor classes (bathroom, bedroom, dining room, kitchen, and living room) from the *Places* dataset. Then, we proposed and examined a combination model of CNN and a multi-binary classifier referred to as error correcting output code (ECOC) with the clean data. The proposed models were evaluated in real-time experiments with a Nao humanoid robot.

In phase 2, we proposed our in-house developed dataset referred to as SRIN, which stands for Social Robot Indoor Navigation dataset. This dataset consists of 2D colored images for room classification (termed SRIN-Room) and doorway detection (termed SRIN-Doorway). SRIN-Rooms has 37,288 raw and processed colored images for five main classes: bedrooms, bathrooms, dining rooms, kitchens, and living rooms. The SRIN-Doorway contains 21,947 raw and processed colored images for three main classes: no-door, open-door and closed door. The main feature of the SRIN dataset is that its 2D images have been purposefully captured for short robots (around 0.5-meter tall) such as NAO humanoid robots. In this phase, we focused only on using the SRIN-Room dataset to train a CNN-based model and then tested it on 2D images taken by Nao in new environments.

In phase 3, we focused on addressing doorway detection as a main component of any indoor navigation system. Thus, we proposed a (3Ds) system for Doorway Detection and Direction. The novelty of this system is to detect a door and extract the orientation in indoor settings from the view of a social robot equipped with only a monocular camera. The challenge is to achieve this goal with only a 2D image from a monocular camera. The proposed system is designed through the integration of several modules, each of which serves a special purpose. The detection of the door was addressed by training a CNN model on SRIN-Doorway. Whereas, the direction of the door (from the robot's observation) was achieved by three other modules: Depth module, Pixel-Selection module, and Pixel2Angle module, respectively.

In the final phase, we designed an end-to-end indoor navigation system exclusively designed for social robots with limited sensors for applications in homes. The overall system is a learning-based behavioristic system that combined a reactive system based on subsumption architecture and a knowledge system with learning capabilities. The robot with this system was able to explore a new home-like environment safely while gaining crucial knowledge based on our daily life during navigating. The system integrated several modules from previous phases. While at the same time, there were new modules designed for completing the system, such as avoiding obstacles via reinforcement learning, passing the doorway via Canny edge's detection, building an abstract map called a Directional Semantic Topological Map (DST-Map) within the knowledge system, and other predefined layers within the subsumption architecture. The

individual module and the overall system were evaluated in a virtual environment using Webots simulator [14].

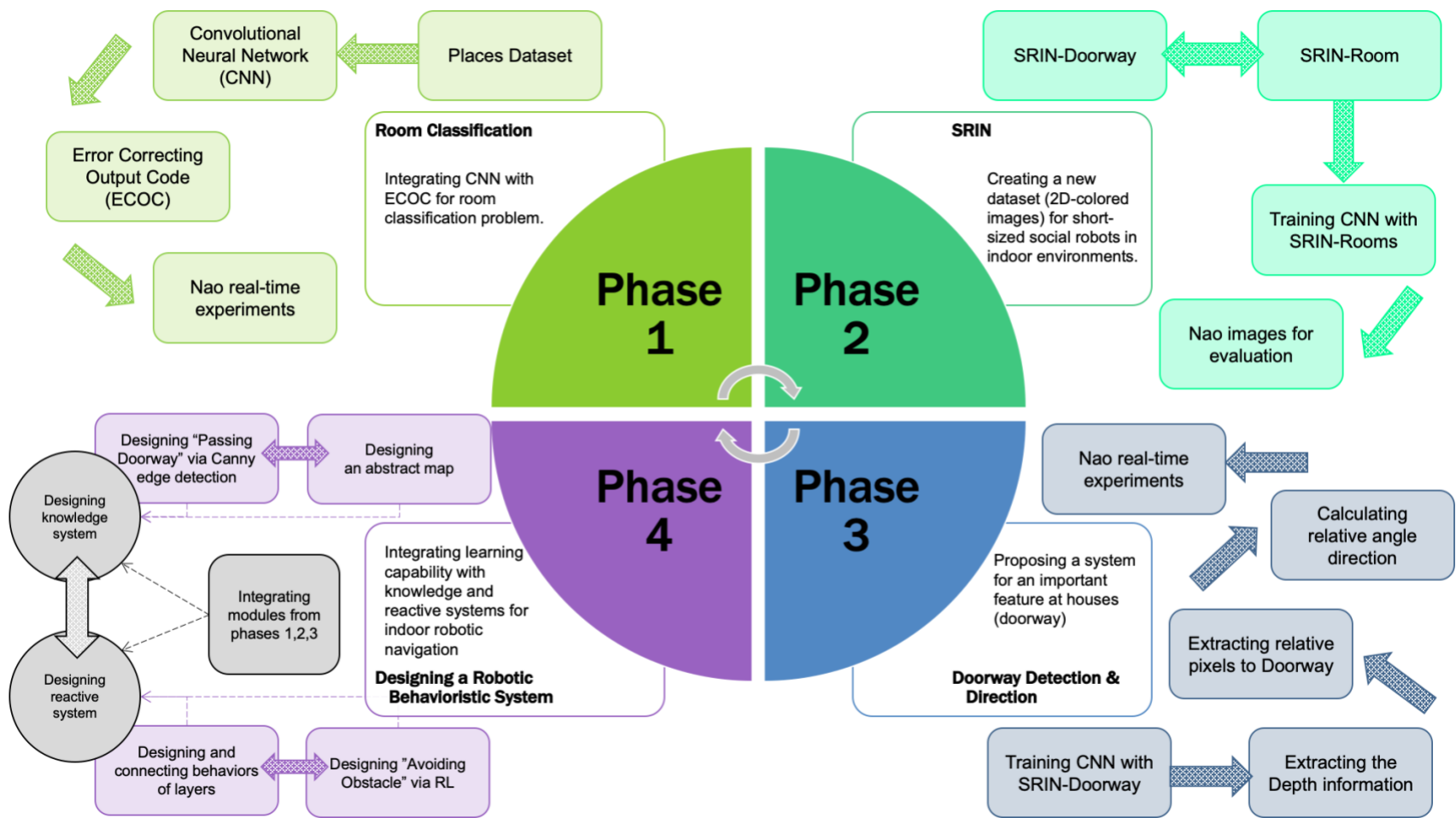


Figure 1-1: The design flow of the thesis.

1.3. Organization of The Thesis

Chapter 2 is aimed to provide a selective yet succinct review of literature pertinent to social robots, navigation, robotic architectures, learning approaches, subsumption architecture, Convolutional Neural Network (CNN) and Reinforcement Learning (RL). The studies related to the aforementioned phases will be explained in detail in chapters 3 to 6, which include the related research of each phase.

In chapter 3, we address the problem of indoor room classification via several convolutional neural network (CNN) architectures, i.e., VGG16, VGG19, & Inception V3. The main objective is to recognize five indoor classes (bathroom, bedroom, dining room, kitchen, and living room) from a Places dataset. We also propose and examine a combination model of CNN and a multi-binary classifier referred to as error correcting output code (ECOC) with a real-time implementation on a NAO humanoid robot.

In chapter 4, we propose a new dataset referred to as SRIN, which stands for **Social Robot Indoor Navigation**. This dataset consists of 2D colored images for room classification (termed SRIN-Room) and doorway detection (termed SRIN-Doorway). The main feature of the SRIN dataset is that its images have been purposefully captured for short robots (around 0.5-meter tall). The evaluation of the dataset will be reported by testing Nao's 2D images and comparing them with the prediction results of the same model with Places dataset [15].

Chapter 5 introduces a novel algorithm to detect a door and its orientation in indoor settings from the view of a social robot equipped with only a monocular camera. The proposed system is designed through the integration of several modules, each of which serves a special purpose. We include simulation results and real-time experiments to demonstrate the performance of the algorithm.

In chapter 6, we propose a reactive system that combines a subsumption-based architecture with a knowledge system for a social robot with limited sensors that is supposed to explore a new apartment (house). Intensive experiments are included to evaluate each module of the system and the overall system via simulation and real-time experiments.

Chapter 7 aims to capture the main ideas and provide a summary of the contributions of the thesis and layout tentative suggestions for the future direction of this research work.

1.4. Publications

At the time of writing this thesis, three journal papers have been published. Also, there is one paper that has been submitted to an international journal. These papers are listed below:

1. Othman, Kamal M., and Ahmad B. Rad. "An indoor room classification system for social robots via integration of CNN and ECOC." *Applied Sciences* 9.3 (2019): 470.
2. Kamal M O, Ahmad B R. SRIN: A New Dataset for Social Robot Indoor Navigation. *Glob J Eng Sci.* 4(5): 2020. GJES.MS.ID.000596.
3. Othman, K.M.; Rad, A.B. A Doorway Detection and Direction (3Ds) System for Social Robots via a Monocular Camera. *Sensors* 2020, 20, 2477.
4. Othman, K.M.; Rad, A.B. Sequential Localizing and Mapping: A navigation strategy via Enhanced Subsumption Architecture. [*In Preparation*].

Chapter 2.

Literature Review

Research in social robotics is multifaceted, interdisciplinary, and indeed technology dependent. Therefore, an exhaustive literature survey is an ambitious undertaking and well beyond the scope of this thesis. Nevertheless, in this chapter, a highly selective yet concise survey of related areas that have a direct impact and significance to studies reported in this thesis are presented. The central objective of this chapter is to set the scene for the reader and provide a context for the chapters ahead.

Social robots are referred to as a special family of autonomous and intelligent robots that are predominantly designed to interact and communicate with humans or other robots (agents) within a collaborative environment. Embodiment is an essential characteristic of this class of robots so that avatars and virtual agents are generally excluded. Social robots are designed for a variety of tasks in a collaborative or service setting and could be deployed in homes (to do household chores, act as a companion to children and seniors, or serve as a butler), hospitals (as a nurse, administrative assistant, etc.), schools (as a teacher), libraries (as a librarian), museums (guides, etc.), to name a few. Other key features of such robots are their ability to recognize people, objects, communicate through voice, and respond to various human emotions. Social robots are designed in all sizes and shapes for a variety of applications but most importantly, they are designed to be acceptable to humans. Depending on the ultimate application, a social robot can be designed as a pet-like, e.g. AIBO [16], or a humanoid, e.g. Nao [17], or a wheeled robot, e.g. Pepper [18], or unmoving robot, e.g. Kasper [19].

Implicit but an indispensable feature of a social robot is its ability to seamlessly move around in the environment for which it is expected to function. Indeed, a social robot cannot realistically perform its dedicated tasks, if it is immobile as it is sensible to suggest that such skills can hardly be isolated from the crucial ability to explore, navigate, or perceive information in the way humans do. Navigation within the above context requires the solution to two distinct but interrelated problems: the ability to know and update the robot location at any time instant, the capacity to move towards a goal in

a safe manner. The former is referred to as the localization problem [20] and requires a relatively accurate map of the environment. When the map is available, the problem is rather straightforward; however, in scenarios where the map is not a priori known, the problem becomes more complex and its solution requires Simultaneous Localization and Mapping (SLAM) [21]. The second problem is referred to as path planning and has its own challenges including but not limited to target identity (perception, searching, recognition, learning), target location (prior map, building a global or local map), and how to reach the goal. The required task within the navigation process is that the robot must be able to arrive to target safely which is referred to as the obstacle avoidance problem [22].

Sensors play a significant role in all autonomous robots' applications including but not limited to localization; environment realizations and representation; object or face detection and recognition; data association; and communication or interaction with people. By the same token, a social robot must be equipped with actuators in order to take appropriate actions. The action decision can then be executed through a motion controller that either operates at a low-level control of motors or a high-level control of behaviors. There are two types of sensors: *proprioceptive* and *exteroceptive* sensors [23]. The proprioceptive (*internal*) sensors, such as gyroscope, accelerators or GPS, are used to measure the internal states of the robot, e.g. its speed or acceleration. On the other hand, the exteroceptive (external) sensors, e.g. sonars, laser, monocular camera, stereo camera or Kinect, provide external knowledge of the environment, for example, distance to an obstacle, or direction to a target. Sensor readings are always noisy which may lead to errors in the measurements; therefore, besides employing appropriate filters, sensor fusion techniques have been widely used to alleviate this problem and improve the performance [24]. Multi-sensor data fusion can be generally applied for creating an environment model or applied within a robotic control system to select behaviors [24]. The latter application is partly within the scope of this thesis as multiple low-cost sensors were used onboard a social robot with limited sensors.

Addressing navigation tasks and interconnecting their functions within a robotic system is referred to as robotic architecture. There are several approaches to robotic architectures for addressing the robotic navigation problem [25]. The traditional approach that emerged in the mid-1970s was the ***deliberative robotic system***, which is also known as SPA (Sense-Plan-Act). The navigation system under this classification is

decomposed into sub-functional modules that are sequentially executed. The core of this system is the plan function, which depends on the availability of a realistic model (map) of the environment; as such, it is referred to as the model-based paradigm. It is goal-oriented and able to solve difficult tasks; nevertheless, it still has some shortcomings. For example, an optimal path planning requires an accurate map, which is a challenging proposition in real-time experiments due to inherent uncertainties, and it is intuitively against human or animal's sub-optimal approach of not requiring a detailed map in similar scenarios. Furthermore, planning in each step leads to delay in accomplishing the required navigation tasks and demands high computation cost. Finally, any error in the system, i.e. perception, model, plan, execution, or motion controller modules, affects the whole system due to the sequential architecture. A behavior-based paradigm coined as a **reactive robotic system** with a promise of addressing some of the above issues emerged in the mid-1980. In this system, plan function is eliminated altogether, and only sense and act functions are coupled. That is why this system is known as SA (Sense-Act) system. This methodology was inspired by animals' behaviors that are mainly governed by stimulus-response coupling. A reactive navigation system is decomposed into sub-behaviors, i.e. multiple instances of sense-act couplings that are integrated in parallel. Therefore, the main key feature of this paradigm is to execute a series of behaviors to enable the robot navigating to a destination with no plan for future nor memory for past knowledge. This system has many advantages: it is suitable for dynamic environments, it has low computation, as it does not require a model nor a plan module, and it is fast. However, it still has shortcomings. First, it is not reliable for complex tasks in complex environments as there is no clear final goal within the system, i.e. no plan. Second, although it provides flexibility due to independence among behaviors, the coordination between behaviors for complex tasks is challenging and often tricky. To overcome these limitations, the **hybrid robotic system** emerged in the early 1990s. As the term suggests, this system integrates the deliberative (model-based) with the reactive (behavior-based) systems to circumvent the weaknesses of the two approaches. The system starts with a model and a plan for a specific mission. Then, a set of behaviors is generated to execute the plan. At regular progression steps, the planner produces a new plan for the new goal or sub-goal, which in turn used to generate a new set of behaviors towards completing the mission. Therefore, the plan function is not executed in each step as in the deliberative paradigm. In this thesis, an architecture that could broadly be classified as a hybrid system is designed for indoor

navigation of social robots. In this research project, we adopted a reactive system and further refined it by augmenting a knowledge system to design a novel robotics navigation system for indoor environments.

Since the early 1980's, a completely different school of thought has emerged and dominated research in robotics navigation. Probabilistic robotics aims to address uncertainties due to sensors, actuators, and the environment [26]. Probabilistic algorithms represent robot and environment information by a probability distribution over a space of mathematical estimations. This approach falls between model-based and behavior-based systems as it incorporates models for sensor measurements and the environment. Therefore, it integrates both model and sensor data in order to control the robot based on statistical mathematics. Yet another approach that has gained popularity in the last two decades is the *Cognitive approaches* that are generally inspired by the human or primates brain functions. The cognition model involves seven mental states: sensing and acquisition, reasoning, attention, recognition, learning, planning, action and coordination and their transitions along with cognitive memory, i.e. long-term memory and short-term memory [27].

Learning feature is an important key in almost all robotic applications. In the last three decades, it has been integrated into navigation systems through machine learning algorithms. Machine learning [28] is a field of artificial intelligence that studies algorithms for learning from experience via extracting patterns from training data. Learning from data can be divided into supervised and unsupervised learning. Supervised learning is the area of designing algorithms that learn from labelled data. Unsupervised learning, on the other hand, is the area of designing algorithms that learn from unlabeled data. There is another approach of machine learning called reinforcement learning (RL) that does not need data for learning. Instead, it depends on the interaction with the environment and learning through a trial-and-error process. Since the main focus of this thesis is on the combination of the reactive system (i.e. a collection of behaviors) and the knowledge system (i.e. perception tasks), we only considered the supervised and the reinforcement learning in this project.

Figure 2-1 shows an overview of the interwoven research areas related to social robotics. The interdisciplinary nature of research in social robotics is evident in this figure and the assimilation of engineering, robotics, artificial intelligence, psychology,

neuroscience, design and more can readily be inferred. The shaded areas depict the specific classifications that fall within the scope of this thesis and are mostly inspired by humans or creatures.

In this chapter, we include an overview of the behavioristic system in section 2.1 with the intention of providing more details about the adopted behavioristic system referred to as the subsumption architecture. Then, section 2.2 introduces the concept of deep learning, specifically Convolutional Neural Network (CNN), as supervised learning with the most common robotics applications. The chapter is concluded by introducing the concept of the map-free approach of Reinforcement Learning (RL) in section 2.3 as well as how the RL approaches were employed in the robotics navigation applications.

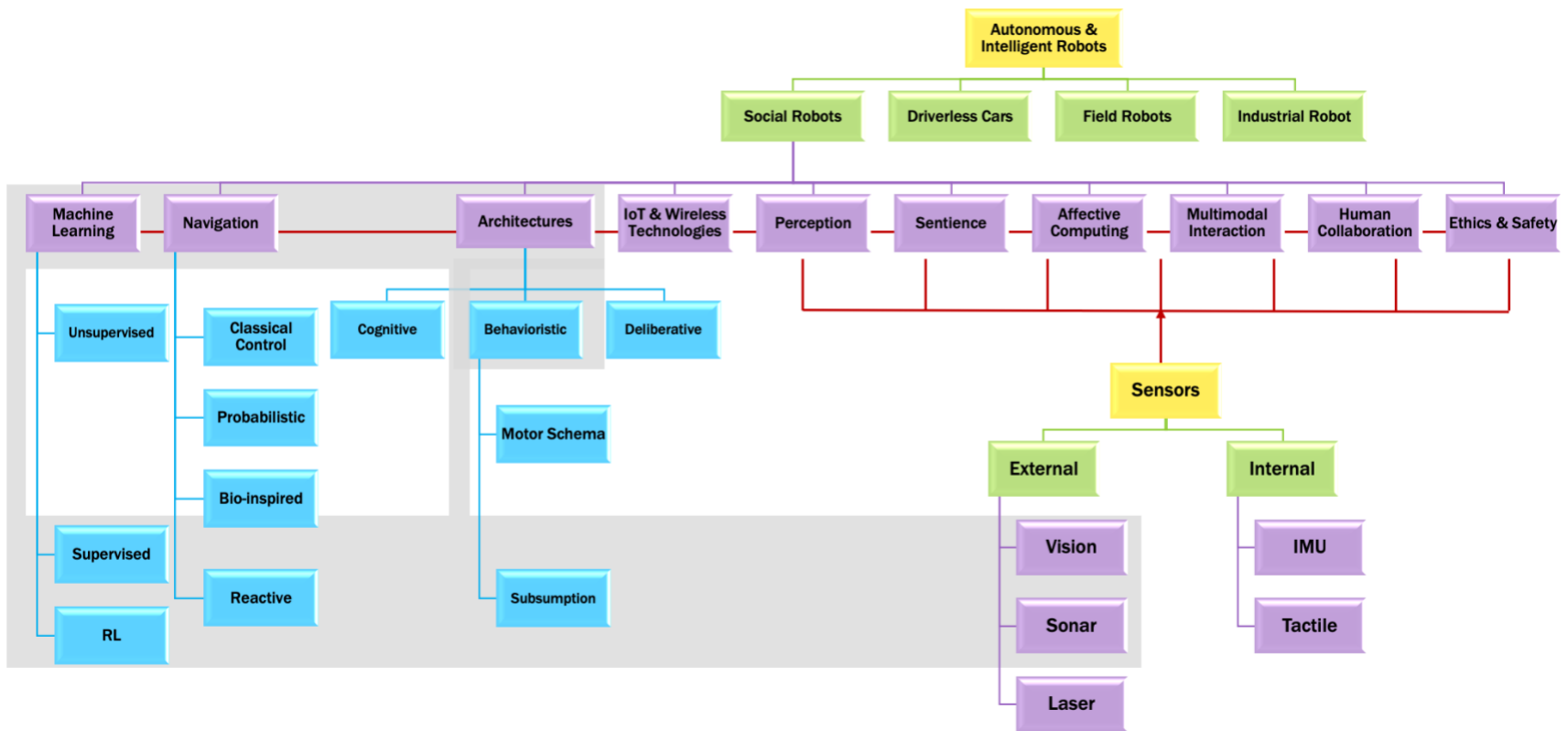


Figure 2-1: Related research areas to the social robots and the scope of this thesis (shaded).

2.1. Behavioristic Approach

2.1.1. From Biological to Robotic Behaviors

Robotic research is interdisciplinary, and researchers are inspired, learned, and engineered findings from other research domains, such as neuroscience (the science of nervous system), psychology (the science of mind), and ethology (the science of animal behaviors). We adopted behavioristic robotics as the backbone research in this thesis. In this section, the main concepts of biological behaviorism are introduced to provide a rationale for behavioristic robotics which will be discussed in the next section.

Behaviorism advocates that learning ensues from interaction with the environment. Therefore, human and animal responses are simply molded by environmental stimuli [29]. A behavior is defined from behaviorism's perspective as a basic capability of animals to make a decision and take an action based on what they sense in order to achieve a specific task [29]. The basic form of behaviors is known as reflexive behavior which has a direct relation between stimulus and response without explicit cognition [30]. Depending on how a stimulus and response is related, different types of reflexive behaviors are formed. For example, lifting the knee immediately when it is tapped by a hammer, it is a form of a *reflex* behavior that its response ends by the end of the stimulus. The other example is the *tax* behavior that the response is a direction towards a stimulus as what happens with a baby turtle that moves towards the brightest light after hatching. The third example is called *fixed action pattern*, in which the response takes a longer time than the stimulus as we can see it in the behavior of fleeing predators [30]. The study of animal behaviors has given more attention as an independent discipline after the study of Konrad Lorenz and Niko Tinbergen who are the fathers of the ethology. They and other researchers have been observing animal activities in their nature from different perspectives. One of the central aspects is observing different types of acquiring behaviors: innate behavior, a sequence of innate behaviors, innate behavior with memory, and learning set of behaviors. The other aspect is observing different ways of controlling behaviors when they are triggered concurrently: dominance, such as hungry versus sleepy, and equilibrium, such as feeding versus fleeing in squirrel [30]. Motivation [31] is an essential aspect that has attracted continuous attention by behaviorists. Studying motivations of behaviors means finding an answer as to why a specific animal acts in a particular manner. Behaviorists have

been looking for cues that are internal, e.g. hunger, or external signals, e.g. the smell of food, that generate specific behaviors without looking for the mental mechanism. Studying the interface between the brain and the behaviors through observing the mechanism of obtaining and processing information is a rather new sub-discipline called neuroethology [32]. In addition, there are different aspects to higher levels such as behavior development and animal cognition. Behavior development studies the change of behaviors based on either evolution theory or learning via interaction with the environment [33]. The latter class of behavior development, i.e. learning via interaction with the environment, will be highlighted in 2.3 as it is considered within the proposed system. Animal cognition is a scientific area that studies the mental functions including learning function, memory, thoughts or environment representation [34].

Robotics researchers are inspired by many biological behaviors and have attempted to replicate them in different applications. The basic form of behavior, i.e. reflexive behavior, can be designed as a sense-act coupling module. A behavior module consists of a sense function that perceives and processes data from the sensors, and an act function that maps input data to action for controlling the robot's actuators. The motivation for behavior is designed as the releaser function that can be a perception function. Other forms of perception functions that modify the behavior module are presented as a suppressor and inhibitor functions, respectively. A suppressor function prevents the sensor's input signal to be sent to the sense function, whereas an inhibitor function prevents the output signal of the act function to be sent to the actuator. All these three perception functions will be highlighted in section 2.1.2 of the adopted behavioristic system in this thesis. An innate and sequence of innate behaviors can be designed as a pre-programmed behavior, and Finite State Machine (FSM), respectively. The collection and combination of a set of behavior modules create a behavior-based robotic system. The two well-known behavior-based robotic control systems are the subsumption [11] and the motor-schema [35]. The subsumption control system mimics the dominance control observation, in which one behavior takes control when many behaviors are triggered concurrently. On the other hand, the motor-schema control system simulates the equilibrium observation that sums the output of concurrent triggered behaviors. Learning behaviors can be addressed through artificial intelligence learning approaches, such as deep learning for perception tasks, and reinforcement learning for action tasks. They will be discussed in section 2.2 and section 2.3, respectively.

2.1.2. Subsumption: An Architecture of Behavioristic Robotics System

Subsumption architecture is the first pure reactive robotic system, which was introduced by Brook in his early work on the behavior-based system [11]. He argued against explicit modelling of the world and notably stated that “*the world is its own best model*” [36]. This system consists of a set of layers as shown in Figure 2-2, in which each layer is responsible for a specific task. These layers are built in parallel where the higher layers have more dominance than the lower ones when they are triggered concurrently, i.e. competition coordination. Each layer is decomposed into basic behaviors or modules. A module is considered as a pre-wired reflexive behavior that connects a sense function with an act function in order to perform a specific behavior. Each behavior can be created, tested, and debugged individually. In the robotics navigation context, the system decomposes the navigation problem into vertical task-achieving behaviors, such as avoid, wander, explore ... etc. Layers are added to the system incrementally, and they are built and activated in parallel, but operated asynchronously. Let us consider that level 0, e.g. “*move around*” behavior, is designed, implemented and debugged. Then, level 1 is added, e.g. “*obstacle avoidance*” behavior, with keeping the function of level 0. Therefore, the robot can move and detect obstacles to be avoided. The main feature of subsumption is that the higher layer can subsume the control of the lower layer, which is the origin of the name, when behaviors are triggered concurrently. Consequently, the higher layer only controls the robot to the overall goal or destination at a specific time. Since interaction between layers might become complicated with complex environments and tasks, the goal in subsumption design is to keep the connection between layers as least as possible, which is more effective for modularity [37]. In short, behaviors in the subsumption system are coordinated and executed such that the robot can interact with the environment and select the best behavior in a sequential manner until the task is completed with no plan for future or any memory of past knowledge.

Each behavior is designed by mapping a stimulus (input sensory data) into a response behavior (actuator output) through connecting its own sense and act functions. In addition, a behavior can be modified by its respective inhibitor, suppressor, or releaser conditional functions as shown in Figure 2-2. An inhibitor function (S_I) is designed to inhibit a behavior from controlling the robot even if the sensor data is available. A

suppressor function (S_s), on the other hand, suppresses the corresponding behavior output, and consequently, no output response is generated. A releaser is like a switch that turns the module on/off based on a particular sensory input. In a case of sequence behaviors, such as finding a refrigerator, opening the door, finding a can, gripping a can, moving gripper out of the refrigerator, and closing the door, this type of task can be accomplished by a Finite State Machine (FSM). This implies that instead of letting one behavior triggering the next behavior, each behavior in this sequence can be activated through the environment. For example, if the state of the refrigerator is opened, then “find a can” behavior will be activated, or if the state of the gripper is closed, then “moving gripper out of refrigerator” will be activated, and so on. Hence, the design and the coordination between behaviors essentially depend on the application or the main task.

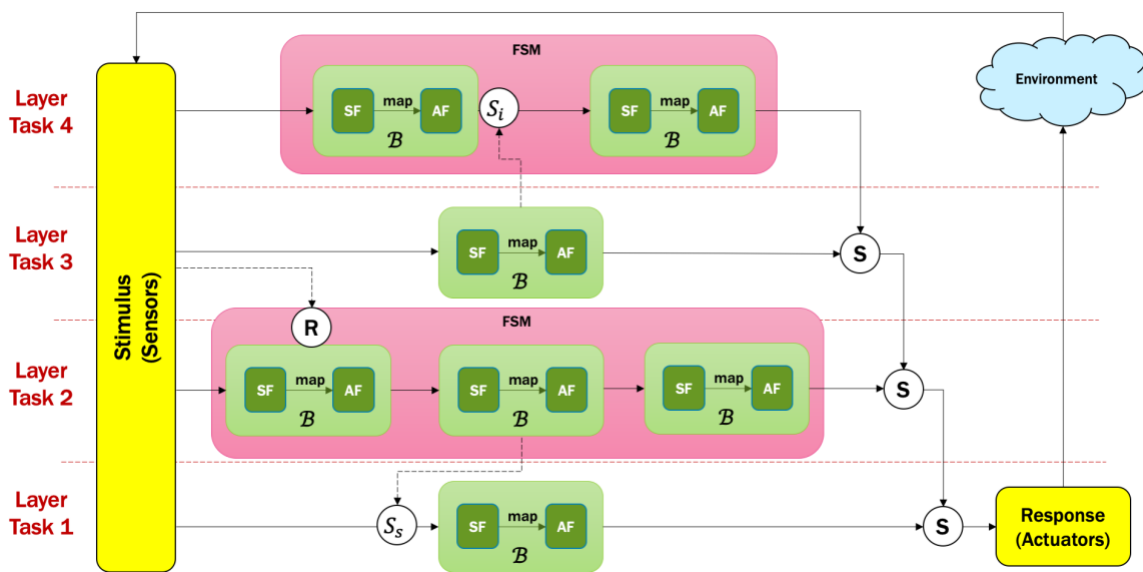


Figure 2-2: An illustration of Subsumption architecture with Layers and possible components. (\mathcal{B} : Behavior, SF: Sense Function, AF: Act Function, FSM: Finite State Machine, S_i : Inhibitor, S_s : Suppressor, R: Releaser, S: Subsumed layer by a higher layer).

Brooks [36] configured a number of MIT’s robots with the subsumption system, *Toto*, *Genphis* and *Seymour*. Subsumption architecture has several advantages including low computation as no need for a model and no plan modules; consequently, it is fast. However, the pure reactive behaviors in subsumption are not straightforward nor reliable for complex tasks in complex environments as there is no clear final goal within

the system, i.e. no plan. In addition, there is no explicit learning, no memory, and no goal-directed motivation [38]. Although some researchers attempted addressing these shortcomings by integrating different algorithms of soft computing with subsumption such as fuzzy logic [39], genetic algorithm [40], or neural network [41], further research on subsumption lost its momentum; particularly due to emergence and success of probabilistic robotics. Integrating learning capability with behaviors can be categorized into two main types: learning coordinating behaviors and learning new behaviors [42]. It is also sporadic regarding employing reinforcement learning (RL) within behaviors, specifically in subsumption architecture. Two early studies were reported by Mahadevan & Connell [43] and Mataric [44] using RL. In [43] RL was combined with statistical clustering and hamming distance in a box-pushing task within subsumption, whereas in [44] RL was applied in a multi-robot domain. Recently, RL has been used with a behavior-based system in order to learn how to perform new tasks, and its performance has been compared with subsumption in [45]. The design consists of two combined phases. First is the imitation phase, which is carried out by using a self-organizing decision tree to emulate the behavior of a skilled operator. Second is the composition phase, which is accomplished by using Q-learning to combine all behaviors and assign learned weights. Then, the outputs of all behaviors from phase one and the learned weights from phase two are fused to perform weighted action. In addition, Q-learning was applied, and its performance was compared within two different behavior-based system: subsumption & motor schema. They were tested on the Lego NXT robot for avoiding obstacles [46].

With the advent of machine learning and particularly deep learning in the last two decades, new opportunities arise to enhance the subsumption architecture and alleviate its shortcomings. In this thesis, we propose a subsumption-based system that is integrated with a knowledge system. We demonstrate its performance to address the open problem of exploration in structured indoor settings in Chapter 6. The new system has learning capabilities for addressing specific perception and action problems. In the next two sections, we review the adopted learning approaches for deep learning and reinforcement learning, respectively.

2.2. Deep Learning

Machine learning (ML) is a research area that predominantly deals with designing a mathematical algorithm to learn from labelled datasets for supervised learning (classification and regression) or from unlabeled datasets for unsupervised learning (clustering and dimensionality reduction). There are different ML methods including but not limited to Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Naïve Bayes (NB), Decision Tree (DT), K-Means (KM), and Artificial Neural Network (ANN) [47]. Deep learning (DL) is the mainstream classification of machine learning and is viewed as the state-of-art in artificial intelligence. It is essentially evolved from artificial neural networks (ANN) dating back to the 1950s [48]. ANN was inspired and a crude attempt to model biological neural networks. It gained popularity in the second half of the twentieth century and led to impressive solutions for complex nonlinear problems that did not fit consistent mathematical models. DL is an ANN architecture with multiple levels of hidden layers that is suited to complex pattern recognition problems. Deep learning is distinctive among machine learning algorithms as it does not require a careful engineering design for extracting features from raw data [49]. The basic form of deep learning architecture is the fully connected (FC) network. It consists of multiple hidden layers, i.e. deep, in which all neurons in one layer are connected to every neuron in the next layer. There are other DL architectures for different preferred applications [50], such as Recurrent Neural Network (RNN for natural language processing), Convolutional Neural Network (CNN for image recognition), Generative Network (GN for predicting or generating output data), or more recently, the so-called Capsule Networks (CapsNet: for image recognition) [51]. These DL architectures have different learning processes and different connection topology among layers to extract features. Since the focus in this thesis is on scene recognition to enhance the navigation process within indoor environments, we adopted the CNN architecture for perceptual tasks. We found that there is no need to adopt the CapsNet for our project as it mainly addresses the lack of rotational invariances in CNN [52], and this is not the case for a scene perceived by a mobile robot. Figure 2-3 depicts the relation between CNN with other ML methods in a network diagram. The mathematical model of CNN will be explained in the next paragraph for understanding its concept and why it is better than the fully connected DL.

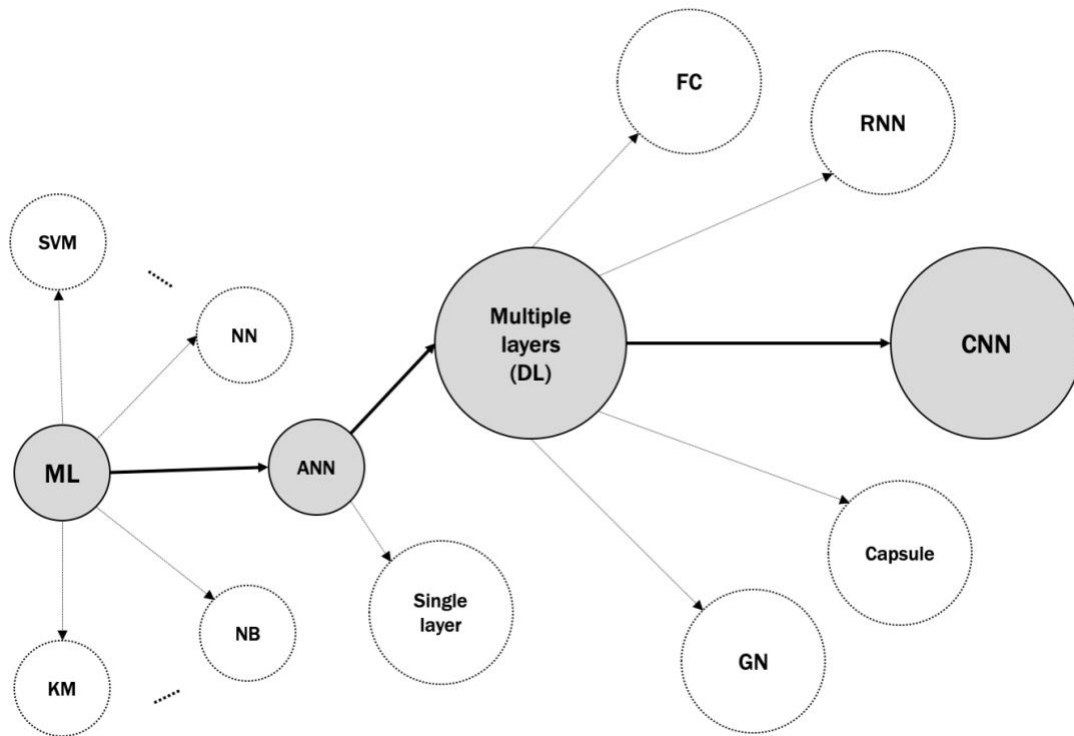


Figure 2-3: Deep learning approaches within machine learning area. CNN is the adopted approach in this thesis.

CNN was proposed by LeCun [53], who introduced the first CNN architecture called LeNet in 1998, after several successful earlier attempts since the 1980s [54], [55]. Due to the huge improvements in data collection and computer hardware between the 1990s and 2012, AlexNet was introduced [56] for addressing the object detection problem using the ubiquitous *ImageNet* to classify 1.2 million images into 1000 different classes. Since 2012, many articulate architectures have been proposed that are essentially built on the early architecture of LeCun, in order to improve the performance on the ImageNet database [57], for example, VGG16 [58], ResNet [59], or Inception [60].

In order to understand the main characteristics of CNN, let us review the CNN structure and its mathematical model. As CNN has been successfully employed for image detection and classification, its main objective is to extract features such as edges, lines, shapes, or colors, without a careful human prior design. The fundamental architectures can be found in LeNet [61] and AlexNet [56]. In general, such networks have three main layers in each stage: convolutional, rectified linear unit (ReLU), and pooling layers. These layers are known together as ConvNet. The architecture is

completed by fully connected (FL) layers for classifying the image from high-level features that have been extracted in multi-stages. In addition, the dropout technique [62], [63] is used to overcome the overfitting problem by ignoring some neurons in the training stage. Figure 2-4 shows the basic architecture of CNN, and the following sub-paragraphs explain the mathematical model of each main layer:

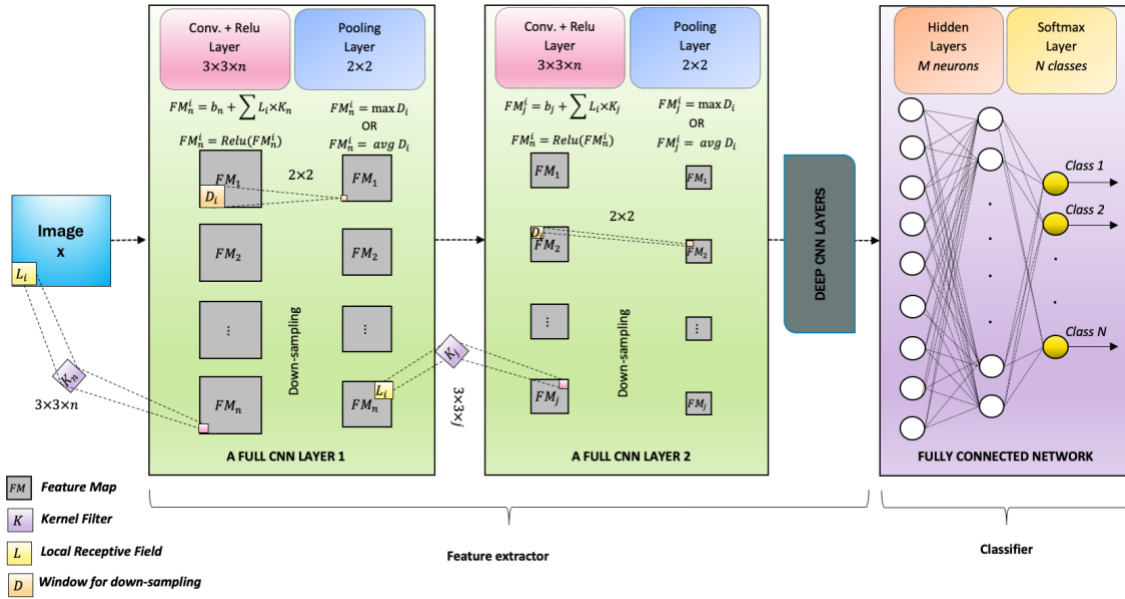


Figure 2-4: Fundamental components of CNN architecture.

• **Convolutional Layers:**

The main purpose of these layers is to extract features from inputs $x = \{L_1, L_2, \dots, L_i\}$, where x is an input image in the first stage while its input features are in the middle stages. Every input can be divided into a set of local receptive fields $\{L_1, L_2, \dots, L_i\}$. There are kernel filters, e.g., 3-by-3 trainable-filters $K = \{k_1, k_2, \dots, k_j\}$ that are used to produce feature maps $FM = \{FM_1, FM_2, \dots, FM_j\}$. The size of each kernel filter and each local receptive field is similar, in which they are used with the bias for calculating each output cell of FM_j as follows:

$$FM_j^i = b_j + \sum k_j \times L_i$$

Let us assume for simplicity that the input is a grayscale, i.e. one channel, of size 5x5 with normalized values of 1 and 0 for white and black, respectively as shown in Figure 2-5 (a). Now, let us further assume that we have a kernel of size 3x3 that slides

on top of the input by one step starting from the top left corner, i.e. it is called stride. Thus, the number of local receptive fields is 9, which is the number of cells of the produced feature map. Each cell of the feature map is using the same kernel multiplied by all local areas in the input of the previous layer. Accordingly, the size of the feature map depends on the size of the input, the size of the kernel and the stride. Optionally, zero cells can be added to the border of the input, it is not shown in Figure 2-5 (a), to have better coverage for the cells in the edge of input, which leads to a bigger size of a feature map if the kernel and stride are same.

- **ReLU:**

As real-time applications are non-linear, the previous layer of linear filters is followed by a non-linear operation as an activation layer. There are various types of activation functions, e.g. Hyperbolic tangent (tanh), sigmoid, or exponential. Recently, the most useful activation function used within CNN architectures is Rectified Linear Unit (*ReLU*) as shown in the below equation and Figure 2-5 (b):

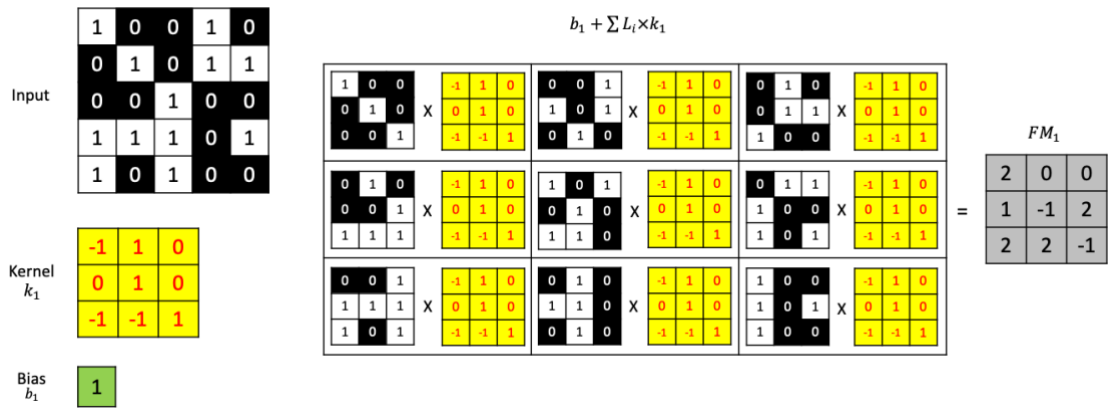
$$f(FM) = \max(0, FM)$$

- **Pooling Layers:**

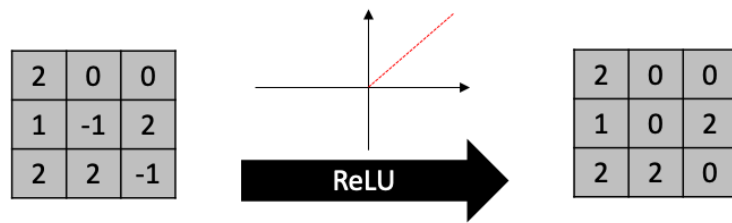
The role of this layer is to reduce the dimension of all features maps while keeping the most important information. It is carried out by taking the average or maximum value of every window, e.g. 2x2 size, of features maps, see Figure 2-5 (c).

- **FL Layers:**

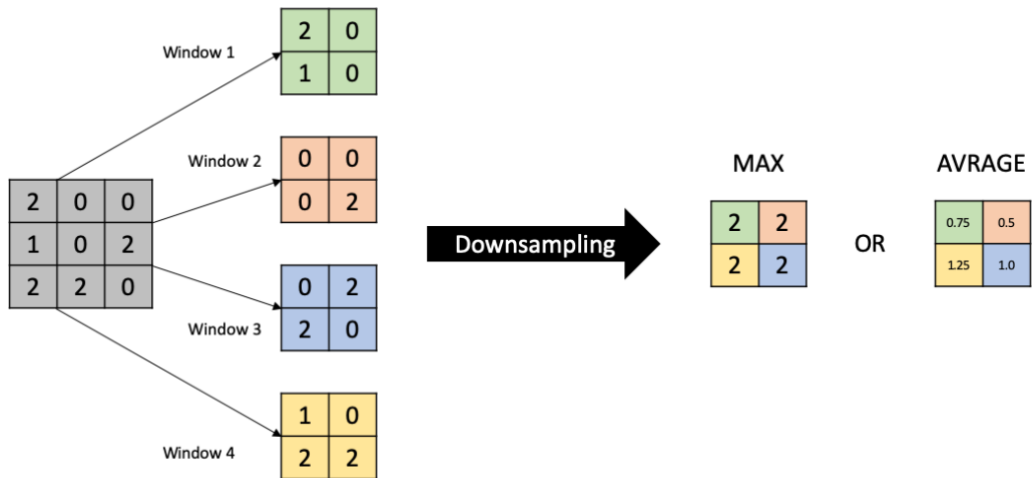
This is a regular neural network that uses all high-level features for classifying an input image to a specific class based on training a set of images.



(a) Convolutional layers



(b) ReLU non-linear function



(c) Pooling layers

Figure 2-5: Mathematical model illustration for each layer in the ConvNet.

The objective of training a CNN architecture is to train and optimize the value of kernels and fully connected weights through backpropagation [54] in order to detect or classify an image with high accuracy. From the CNN architecture in Figure 2-4 and the mathematical illustration of each layer in Figure 2-5, there are four key ideas behind the ConvNet architecture [49]: local connections, shared weights, pooling and the use of many layers. The first concept, i.e. local connection, is due to the high correlation between pixels in the same area. Thus, it is preferable that each cell in the feature map, i.e. it is similar to hidden neurons in FL, is produced from a local area, instead of a specific pixel, of the previous feature map through a kernel that has weights and bias. The second key point of CNN, i.e. shared weights, is that extracting features from input should not depend on its pixel location since this feature can be varied from one image to another. Thus, the shared weights can detect the same patterns from different parts of the input image. Next, the pooling layer or what is known as down-sampling is to decrease the size of the features while maintaining the essential information. Finally, from the shared weights and pooling, deep ConvNet layers lead to a significant reduction of parameters' number comparing to the regular DL.

2.2.1. CNN for Robotics Applications

Since Convolutional Neural Network (CNN) shows an excellent performance in computer vision for learning image features, especially supervised learning applications, the robotics community has applied the CNN model for robotic perception tasks. This section focuses on the recent literature that has adopted CNN for robotic perception applications. Perception with CNN in robotics was studied within two main categories: objects-based and places-based applications [64].

Object-based applications in robotics are considered as the study of addressing object detection problems. Object detection is a computer vision and image processing area that deals with images and extracts features in order to categorize them into a specific class of objects. The task of object detection has been addressed by several CNN architectures in many robotics applications as summarized and classified in Table 2-1. The task of grasping in robotics arms is one of the common robotic applications that depend on the object detection task as well as the location of the object. The work in [65] addressed detecting and grasping the most exposed object from a stack of objects, applied on seven different types of fruits. A simple CNN-based model was designed and

trained on collected RGB-D images by Kinect that separated from the arm robot. In order to achieve the detection and grasping simultaneously, the dataset was labelled by one of the seven fruits that it is mostly exposed and associated with pre-grasp points for grasp detection. In [66], the objective was to optimize the bounding-box rectangle estimation of the object detection in a simulation experiment. This was accomplished through two stages: 1) applying a pre-trained Regional Convolutional Neural Network (R-CNN) on a raw image for getting the bounding box and 2) Applying the CNN-based model on the output of the R-CNN combined with predefined an intent area to optimize the rectangular location within the image. The concept of using tools to improve object detection was discussed in [67]. The authors aimed to assist a wheelchair with an arm called a Kinova Jaco robotic arm to pick an object that was pointed by a user using a laser. The object detection and laser point detection were tackled by applying YOLOv3* system (You Only Look Once system). Then, the grasping pose of the robot arm in the wheelchair was generated based on the object color and the laser depth value. A real-time experiment was reported in [68] on WALK-MAN mobile robot to evaluate the tool grasping task in an indoor environment. It was achieved by designing a CNN model based on encoder-decoder architecture and training it on UMD (RGB-D affordance of tool parts) in order to extract the affordance part of the tool based on RGB-D input image.

Robotic grasping was also tackled for farming applications with agricultural robots. A simulation work for detecting and generating a bounding box around apples in the image within an orchard environment was presented in [69]. This was accomplished via the network of single shot multi-box detector (SSD), which was trained on collected images by an unmanned ground vehicle called Shrimp that was operating in the apple orchards. With the same concept of simulation work of addressing detection problem without a real-time implementation of grasping, a network of Faster R-CNN with VGG19 in [70] was proposed to address the maize seedling detection for a wheeled robot. Twenty thousand images of soil, maize and weeds are collected by the wheeled robot and used for training and testing the proposed model. On the other hand, real-time experiments were conducted on a robotic arm to pick a certain vegetable [71]. Images for seven categories of vegetables were collected by two cameras mounted on the arm robot, in which these images were used for training and analyzing the performance of

* YOLO is a CNN-based model for predicting multiple bounding boxes with their class probabilities [282], [283].

different CNN models: Faster R-CNN, SSD, RFB (Receptive Field Block) Net, YOLOv2 and YOLOv3.

Human-Robotic Interaction (HRI) is another important area in robotic applications that needs to tackle the object detection problem. The simulation work in [72] concentrated on recognizing hand gestures as a classification problem for human-robot collaboration. It was achieved through designing and testing the Fast R-CNN model using collected RGB images from a Kinect camera. Emotion detection is another significant application for HRI as presented in [73], [74]. The first paper provided simulation studies by proposing a CNN model based on Inception architecture that was trained and evaluated on eight different emotion datasets. The latter paper presented a real-time experiment on the Nao humanoid robot by applying a CNN-based model for detecting six different emotions based on facial expression in children using several existed datasets. Another practical robotic system for the Nao robot was proposed for understanding the behaviors of children with autism while playing with toys [75]. The objective was to detect a toy that had received the child's attention as well as to detect the child's hands. The object detector within the system was designed based on the Fast R-CNN model and trained with the ADOSet dataset for the purpose of autism diagnosis. The practical proposed system started by recording videos by Nao's camera and then annotating video frames with object's names using the trained Fast R-CNN. Then, detecting the toy that got the child's attention was achieved by combining hand detection with a decision tree model.

Detecting people is an important problem to be addressed for indoor or outdoor robotics applications. An integrated system of Aggregate Channels Features (ACF) with CNN was proposed [76] in order to detect people for the Human-Aware Robot Navigation problem. The CNN network was fine-tuned using the INRIA dataset that is considered as a benchmark for pedestrian detection. The CNN model takes generated regions from AFC as an input and predicts whether the input region consists of people or not. The proposed system was tested with onboard and off-board cameras. Another practical project [77] suggested a perception system based on depth information to detect people for a robot assistant in hospital environments. The system was designed based on combining Fast R-CNN, Kalman Filter and Hidden Markov Model modules to detect people, classify their situation category (pedestrian, a person in a wheelchair, person in a wheelchair with a person pushing them, person with crutches and person

using a walker), tracking them, and estimate the person position and velocity for providing the appropriate assistance. The authors introduced a new annotated RGB-D dataset for hospital environments collected by Kinect camera on a mobile Festo Robotino robot. In [78], the simulation work targeted moving humans for safe autonomous robotics navigation. A robotic vision system was proposed to detect moving humans and classify their actions based on the CNN model. The system composes of an optical flow descriptor, a CNN human detector, a CNN human action detector and a local search window detector. Detecting and tracking a single person was tackled by showing a real-time implementation on Parrot AR Drone in [79]. The detection task was completed by designing and training an SSD CNN model, whereas the tracking task was achieved by few simple PID controllers.

RoboCup Soccer [80] is a robotic competition that deals with several robotics problems including object detection. The work in [81] proposed a new dataset that created for ball detection in the RoboCup Soccer Standard Platform League by Nao camera. This dataset was employed to train and evaluate a CNN-based model called XNOR-Net in a simulation scenario. Detecting the robot within the soccer field is another object detection problem in the RoboCup that has given attention in some of the robotic studies such as the simulation work of [82], [83]. The work in [82] proposed a two-stages vision system for detecting other robots. The first stage aimed to preprocess the input image in order to extract the region of interest (ROI). Then, the second stage applied and compared different CNN-models for classifying three different types of humanoid robots within the extracted ROI. The other work [83] focused on analyzing the performance of two different CNN-based models, XNOR-Net and SqueezeNet for detecting Nao robot with limited computational resources.

Robotic navigation is a crucial problem that consists of many sub-problems such as localization, mapping, SLAM and path planning. The solution to these problems could include object detection. In [84], A navigation system that integrates people detector module with a SLAM module was proposed using ROS for a Pioneer 3DX robot. The robot equipped with an RGB-D camera that was employed for the people detection task and a Sick Lms200 sensor laser that was important for the SLAM task. The objective of the proposed system was to assist the robot to be able to detect and follow a person. That was done by applying the people detection module to find the person's position and applying the SLAM module to find the robot position within a created map, then

integrating the position of the person and the robot in the path planning module. Two different methods were employed and compared separately for detecting people: Histogram of Oriented Gradients (HOG) and a GoolgeNet CNN-model followed by SVM classifier. The author claimed that the latter approach was more accurate for people's recognition within their proposed navigation system. A practical experiment was presented in [85] in which a mobile robot acquires a 360° image in order to classify the navigational direction based on a CNN-based model. The model was trained by a collected data of 360 images that manually labelled by specific directions using a spherical camera mounted on a mobile robot. The work in [86] aimed to conduct real-time experiments on a Pioneer P3-DX mobile robot with mounted laser and camera in order to track movable objects, e.g. door and people, using R-CNN model within SLAM-based robotic system. The simulation work in [87] presented a vision system for object recognition in an indoor environment. The proposed system combined a CNN trained model with prior knowledge of scene and color to enhance the prediction performance. However, there were no real experiments in robots although it was intended to address the robotic navigation problem. There are several recent works in literature that focus on improving the built map by SLAM with adding semantic information through detecting objects using different CNN-based models, such as a model based on Fast R-CNN [88], YOLO with Kobuki base [89], YOLO with Turtle Bot mobile robot [90], YOLOv3 with simulation work on Robot@Home dataset [91] and multi CNN models for creating 3D bounding box [92]. In addition, there are some studies that take the advantage of the object detection in order to improve the performance of localization, such as integrating SSD model with Kalman filter that applied on HUSKY UGV platform [93], and a trained Fast R-CNN on a dataset that contains images associated with GPS and compass sensors information [94].

There are other general applications that have addressed the object detection problem by CNN models. For example, An Unmanned Aerial Vehicle (UAV) was employed to collect a sequence of images (videos) of avalanche debris for the task of search and rescue victims [95]. The collected videos were used to train and validate a CNN model followed by an SVM classifier that detects the presence or absence of objects. The CNN model was combined with preprocessing and post-processing modules in order to improve the performance of the victim's detection. The work in [96] suggested a deep recurrent neural network called LSTM (Long-Short Term Memory) for

detecting a car generating a descriptive or instructive sentence related to the detected car. The task was achieved by combining the CompCar dataset with collected videos to train the model. For the cleaning purpose, [97] proposed a cascade machine learning system that is able to detect debris in an indoor environment for vacuum robotic application. The system consists of the SSD model for classifying the type of debris (solid vs liquid) during the training stage. Then, it is followed by SVM during the testing stage to classify the hardness of cleaning the detected liquid spills. The experiments were only verified by simulation studies through collecting images using the Aver vision system and Kinect sensor for evaluating the proposed system. Simple models of CNN were also suggested and tested in simulation experiments for detecting clear sky [98], and detecting cracks in concrete bridges [99]. Classifying marine objects into sea cucumber, sea urchin, and scallop has been addressed and tested on Underwater Robot by applying Fast R-CNN [100]. The trained detector was followed by a tracking model called kernelized correlation filter to track detected objects.

Table 2-1: Summarizing and classifying selected papers of object detection for robotic applications based on CNN architectures since 2016. (S: Simulation & R: Real-Time Experiments)

CNN for object detection in robotics	
Robotic Applications	Deep Learning Architectures
Grasping / Farming	CNN-based (<i>Basic ConvNet layers</i>)
S [66],[69],[70]	[65],[74],[76],[78],[85],[87],[92],[95],[98],[99]
R [65],[67],[68],[71]	SSD
HRI	[69],[79],[93],[97]
S [72],[73]	YOLO
R [74],[75]	[67],[89],[90],[91]
People detection	R-CNN
S [78]	[66],[70],[72],[75],[77],[86],[88],[94],[100]
R [76],[77],[79]	Encoder-Decoder
RoboCup	[68]
S [81],[82],[83]	GoogleNet / Inception
R -	[84] / [73]
Navigation	XNOR-Net
S [87],[91],[92]	[81]
R [84],[85],[86],[88],[89],[90],[93],[94]	LSTM
Other	[96]
S [96],[97],[98],[99]	Models Comparison
R [95],[100]	[71],[82],[83]

On the other hand, scene recognition is another important research area that has been applied to numerous robotics applications including navigation and localization. This area is widely studied in literature with different perspectives: features recognition, pose estimation, and semantic classification. The simulation study in [101] adopted CNN to address the viewpoint changes issue in a road-based application. This was achieved

by estimating the depth and generating synthetic views for the current visual scene representing lateral camera shift in order to improve the performance of place recognition. In [102], the indoor place recognition was enhanced by combining the descriptors of two separate CNN models for a given RGB image and a given depth image inputs, respectively. The problem was addressed in [103] by training CNN-models with triplet embedded systems. Other studies also addressed the pose estimation, such as [104]–[106]. The authors in [104] addressed the camera global pose estimation (6 degrees of freedom) via training a CNN-like model, as a regression function, using 2D images for outdoor environments. Whereas, the other two studies focused on solving the relative pose from the camera’s location to the scene. The authors in [105] solved place recognition by improving the efficiency of a topological map. It focused on selecting the best keyframes for every node in the map through a CNN-based model, a sharpness model and a matching model. Then, the localization task was achieved by comparing the current scene with all keyframes in the map and finding the most similar keyframe to estimate the relative pose between the two images. While the authors in [106] proposed a place recognition system based on the CNN model using an omnidirectional (360°) camera. The study was aimed to find out the closest place (exemplar) in the predefined map (saved exemplars) with the smallest metric distance to the current position, in which distance information was used to control the robot to move toward this closest place. Semantic classification is another perspective of place-based robotic applications. In [107], the room classification problem for household service robots was addressed by applying a pre-trained CNN model on a segmented image, i.e. learning through parts. Similarly, the same problem was addressed in [108] by combining CNN with NBNN (Naïve Bayes Nearest Neighbor). In this thesis, semantic classification and doorway detection are considered since they are important tasks for social robots that interact meaningfully with humans and navigate autonomously. Therefore, all related research will be presented and discussed in the related chapters.

2.3. Reinforcement Learning

Reinforcement learning (RL) was proposed by Richard S. Sutton, who is considered as the “father” of the RL, in the early 1980s [109]. RL is a class of machine learning algorithms that was inspired by learning theory in animal behaviors [110], [111]. The main characteristic of RL is that the learning process occurs through the interaction

between the agent and the environment. Therefore, there is no dataset for learning as the other two machine learning classes, i.e. supervised and unsupervised learning. The interaction between the agent and environment is achieved when the agent recognizes the current state within the environment, then it takes an action in order to transit to another state, see Figure 2-6. While the agent gets a positive (*rewards*) or a negative (*punishment*) feedback when it reaches a good state or a bad state, respectively. Therefore, it is a slow and trial-and-error process in which the agent learns from its experience. Accordingly, RL can be considered as a semi-supervised approach as it takes a feature from supervised learning, which is the rewards feedback, while it takes another feature from unsupervised learning, which is no desired output.

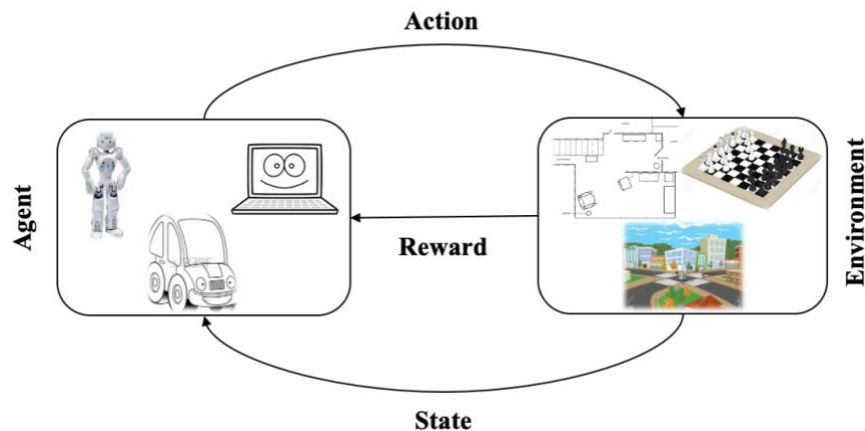


Figure 2-6: The concept of RL.

There are three different RL methods [112]: Dynamic Programming, Monte Carlo and Temporal Differences. Dynamic programming method requires a complete and accurate model of the environment; thus, it is for a model-based system in which the state transition is known. The Monte Carlo is a model-free method, i.e. the state transition is unknown, but is not suited for step by step incremental computation. In other word, it gives the feedback in the end of each experiment or game (*an episode*). The last method is the Temporal Difference (TD) that works with model-free applications and it is a fully incremental computation. In this thesis, the latter method is adopted as it is suitable for our project that deals with a behavioristic system for a model-free application. The main components to design an RL model in any above-mentioned form are:

- *States*: are the representation of perceived data. These representations include the final state, e.g. end of a game, or destination, e.g. final location in robotics. The number of states might be finite, or infinite based on the problem and the way of encoding them.
- *Actions*: are the list of movement behaviors that need to be taken to change the state. The goal is to select the best action in a certain state that leads to a desirable state.
- *Policy*: is the procedure of action selection at a certain state. It can be executed through a simple function or a lookup table, or it can be a stochastic function depending on the problem.
- *Rewards*: are the immediate feedback after every transition. If the transition was good, then positive feedback (rewards) should be given to the agent. However, if the transition was bad, then negative feedback should be given to the agent, which needs to change the policy of selecting the action.
- *State value*: is the collected rewards in the long term. If this value for a pair of (state-action), then it is called Q-value. The main goal of RL is to maximize this value to address the decision-making problem. It should be updated after every step in TD methods.
- *An episode*: is the end of every round of the problem. For example, the end of a game when the agent wins / loses is an episode or in robotics, when the robot achieves a specific task is an episode.

There are two popular algorithms of the TD method: Q-learning and SARSA (Sense-Act-Reward-Sense-Act). In both algorithms, the objective is to maximize the long-term reward $Q(s, a)$, which represents the long-term reward for the combination of all states with all actions. The policy of selecting an action can be either exploiting or exploring. Exploiting movement is a 100% greed-move that selecting the best-learned action with the highest value of $Q(s, a)$. Exploring movement, on the other hand, is ϵ -greedy-move that selecting a random action. The difference between Q-learning and SARSA is in the way of updating the $Q(s, a)$ in every time step as shown in Figure 2-7. In the Q-learning algorithm, the maximum value of the future $Q(s', a')$ will be used to update the current value of $Q(s, a)$ even if the action a' has not been selected for the next timestep. For that reason, Q-learning is called an off-policy TD control. Whereas, the updating equation of the current value of $Q(s, a)$ in the SARSA algorithm uses $Q(s', a')$ where a' should be the selected action for the next timestep, which is the reason this algorithm is called on-policy TD control. Therefore, they are similar when the policy of

selecting the action is exploiting movement, i.e. 100% greedy move. There are several areas of RL applications including but not limited to games, e.g. backgammon or chess, inventory management, dynamic channels allocation, elevator scheduling, helicopter control, robotics e.g. navigation, grasping, or Robocup soccer [113]. The next section focuses on the related work of RL within mobile robotic navigation.

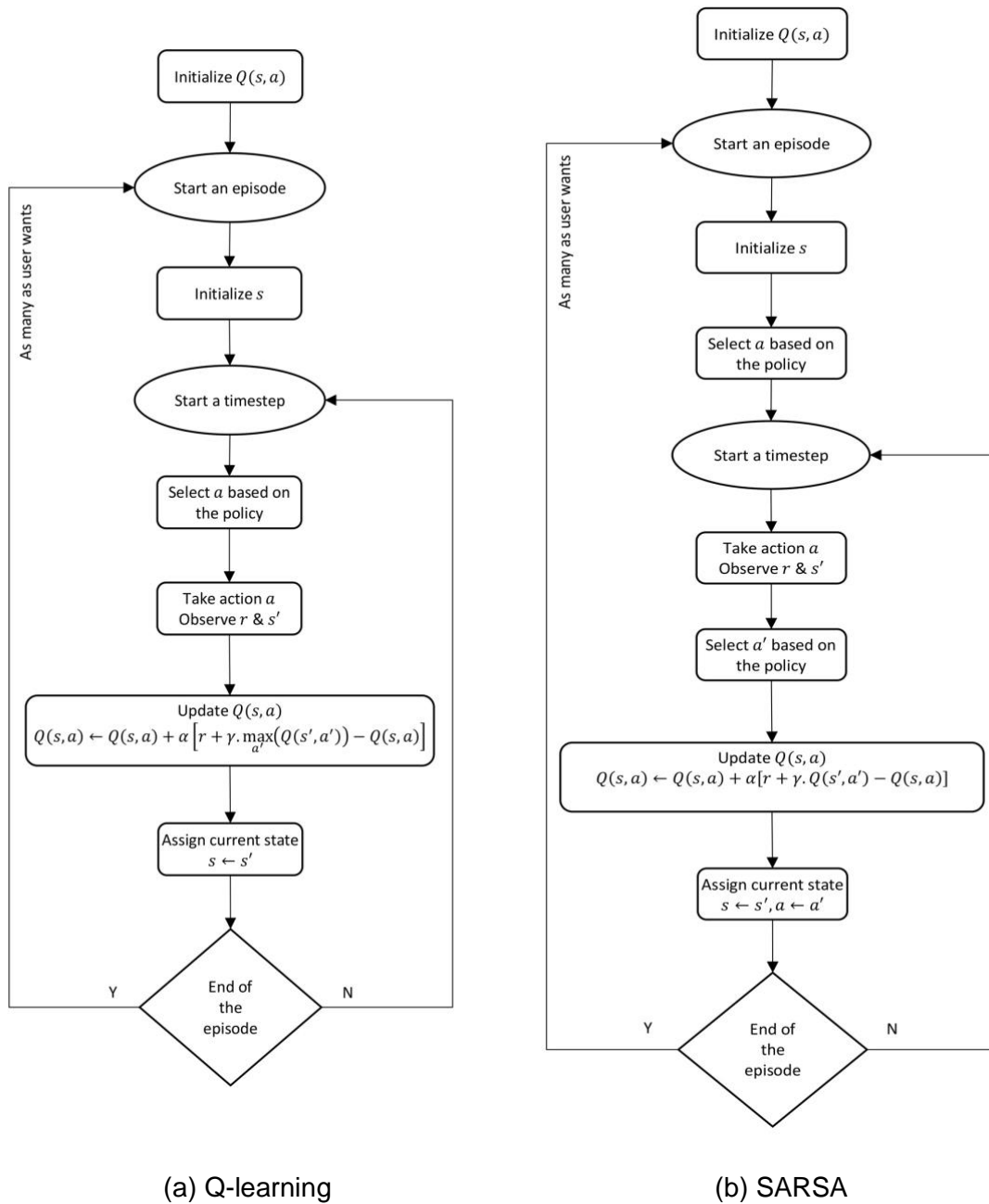


Figure 2-7: Temporal Difference algorithms of RL.

2.3.1. RL for Robotic Navigation Applications

Reinforcement Learning is a promising learning approach for robotics control systems as the robot learns and adapts through the interaction with the dynamic environment. Therefore, RL is considered as a suitable behavioristic approach since it depends on the sense-act process. This section presents examples of robotic applications that have adopted RL with a more focus on the navigation area.

There are several applications within Human-Robot Interaction (HRI) that can be achieved by applying the RL approach in order to assist humans. In [114], A simulation work of Vocal interaction between robot and human was addressed by proposing the RL dialogue system via socially-inspired rewards which is a combination of task-related data and human social signals. An assistive HRI system was proposed [115] to help human to perform a task with less effort. The control system was integrated by the RL model that optimizes the overall system by searching the optimal control parameters.

Learning from human demonstration is another area within HRI that can be addressed with a specific approach called Inverse Reinforcement Learning (IRL) that determines the optimized reward function with given measurements of agent behaviors over time [116]. Therefore, the HRI-IRL can be applied for other purposes than assisting people, such as robotic navigation tasks. The work in [117] used a prior map of the environment and given scored trajectories by a human expert to address navigation problem in a simulation work by estimating the reward function of the IRL system. The simulation work in [118] proposed a solution for faster learning of IRL via failed and successful demonstration in the context of navigation problem with known and deterministic environment as well as in the context of factory domain. The integration of IRL and neural network was applied in a simulation work of [119] to learn the navigation task via human demonstration with large state-space, which is an input of waypoints and trajectory by a user. Real time experiments on Dary1 robot were presented in [120] to address navigation behaviors based on modelling people social behaviors via IRL controller.

There are several other perspectives or tasks of addressing robotic navigation based on RL. One of the important tasks is the obstacle avoidance problem. It was addressed by integrating RL with neural network in a deterministic simulated

environment [121], or with a prior knowledge of a global map for Powerbot wheeled robot [122], or integrating spiking neural network [123]. Integrating RL within a robotic control system is another important area. For example, [45] used RL with a behavior-based system in order to learn how to perform new tasks for soccer application in a simulation work. The design consists of two combined phases: the imitation phase that is carried out by using self-organizing decision tree to emulate the behavior of a skilled operator and the composition phase that is accomplished by using Q- learning to combine all behaviors and assign learned weights. Then, the outputs of all behaviors from the phase one and the learned weights from the phase two are fused to perform weighted action. Another example of integrating RL with behaviouristic system is the work in [124] that focused on coordinating between behaviors with a prior knowledge of the environment and then tested in real experiment with the Pekee robot. The RL was used within a hierarchical navigation system [125] as a local planner for the Probabilistic Roadmaps (PRMs).

There are some works focusing on applying RL with the perspective of prior knowledge of the environment (known robot/object locations) or a deterministic environment (known states-relation, e.g. grid cell map). In [126], a topological map within a deterministic grid-world was built and employed to update the value function of RL system in order to improve the performance. A model-based reinforcement learning system called TEXPLORE was developed in [127] to address navigation problem for quadcopter UAV robot within ROS-Gazebo environment. The system used a parallel architecture of taking action based on current policy and at the same time updating the plan and model to get the new state and reward. Testing Q-learning and SARSA on Robotino robot with pre-defined environment was presented in [128]. The simulation study in [129] aimed to enhance the performance of go-to-goal task with shortest path with providing the location of a target with pre-defined environment. The problem was addressed by integrating spiking neural network with RL. SLAM was applied in [130] to generate a map and then used it in the RL system for path planning within a dynamic environment. In [131], RL was combined with a PID position controller and used for UAV navigation problem within a deterministic environment.

Other robotic navigation applications have also been reported that focus on the RL design. For example, XCS-based RL (Extended Classifier System) [132] was designed to address motion planning for spherical robot [133]. Fuzzy logic was proposed

in [134] to design basic actions that used by the RL to address navigation problem for a Pioneer robot in V-rep simulator. A generalized computation graph was proposed in [135] for RL instead of a full model based or model free based approach in order to learn from few samples in real world, which was applied in a real-world RC car. The design of RL can be based on the robot's design as presented in [136] using 6 sonars mounted on the left, front and right of AmigoBot within a spiral maze.

In this thesis, we focus on the appropriate design of RL within behavioristic robotic system. The design of RL is for obstacle avoidance behavior based on only two sonars and cautious actions as will be discussed in Chapter 6.

2.4. Conclusion

In this chapter, we presented an overview of the most related research areas for social robotics. In section 2.1, we discussed the behavioristic approach in robotics that is inspired by the behaviour of the living creatures. The section provided an overview of the subsumption architecture which is adopted in this research. Then, the convolutional neural network (CNN) and its mathematical model were presented in section 2.2 that also included CNN's main applications in robotics. In Section 2.3, we provided another learning approach called reinforcement learning (RL). The section began with explaining the concept of RL, then it was followed by its applications in robotic navigation. The literature review in this chapter was the theoretical foundation of the research reported in this thesis.

Chapter 3.

An Indoor Room Classification System for Social Robots via Integration of CNN and ECOC*

3.1. Introduction

The prospect of a social robot in every home may be realized within the next two decades. There are already many researchers in academe and tech industries that are actively studying and designing prototypes of such robots. The open research objectives are diverse and include but not limited to emotion recognition, perception, pattern recognition (face, object, scene, and voice), and navigation. These robots are expected to be employed as companions to seniors and children, housekeeping, surveillance, etc. [137], [138]. In order to accomplish such tasks, it is essential that the robot seamlessly recognizes its own location inside the home – similar to humans who are effortlessly aware of their whereabouts at any instant, e.g. kitchen or living room. This knowledge is a pretext for many navigation indoor scenarios and facilitates the robot's movement in the house. The study in this chapter is not about designing social robots per say; it addresses one of many problems that collectively contribute towards efficient operation of such robots; namely knowing its location in the house at any given instant. Classification is a core computer vision problem whereby data streams are categorized into specific classes in accordance to learning their specific features. The problem has been addressed by different supervised machine learning algorithms [28]. The Convolutional Neural Network (CNN) [53], [54] is generally regarded as the state-of-the-art algorithm in deep learning for visual purposes, e.g. face recognition and object detection, especially after the pioneering work reported in [56]. This algorithm surpasses conventional machine learning algorithms by integrating the feature extraction and classification problems without the requirement of careful human design [49].

The main objective of this chapter is to identify different household rooms for social robotic applications in houses. It is part of designing social robots to be employed at such environments. The problem of indoor navigation can be addressed by different

* This chapter is mainly reproduced from paper 1 on section 1.4 (page 8).

approaches; here, we propose a CNN solution for real time implementation for such social robots. We examine several CNN architectures within a home setting. The latest scene dataset called Places [15] is adopted in the study. We downloaded the most common five indoor classes for houses (bedrooms, dining rooms, kitchens, living room, and bathrooms) from the dataset. However, we noted that each class included a sizeable number of irrelevant scenes; we, therefore, reduced the number of samples by removing unrelated images from each class. We will then propose a combination solution of CNN with multi-binary classifiers, referred to as ECOC [139]. These models are evaluated experimentally on a NAO humanoid robot [140].

The rest of the chapter is organized as follows: Section 3.2 provides a literature review of the room classification problem in robotics applications. In section 3.3, we introduce the methodology of this study, which is divided into three phases. The objective and the process of each phase are explained. In addition, this section gives a brief review of the adopted dataset, the adopted CNN architectures and the ECOC algorithm. Then, in section 3.4, we present simulation studies and real-time experiments with their results for each phase including some discussions. The chapter is concluded in section 3.5 by summarizing the study and results.

3.2. Related Work

Recognizing different rooms in a home environment based on their specific function is an important problem for social robots, and its solution not only facilitates seamless movement from one place to another, it is the basis of all other tasks, including assistance to humans inside the house or performing various functions in the context of the robot's overall tasks. An interaction with a human might be in the form of "*Please go to the kitchen and bring a cup of water*". This problem has attracted the attention of robotics researchers in the last decade, and several conventional machine learning methods have been employed to address room classification in indoor settings. One of the early studies reported by Burgard's group in [141] was to address semantic place classification of indoor environments by extracting features from a laser range data using AdaBoost algorithm. The experiments were conducted in a real office environment using sequential binary classifiers for differentiating between room, corridor, doorway, and hallway. It was suggested that the sequential binary AdaBoost classifiers were much more accurate than multi-class AdaBoost. The study was further extended in [142] and

[143] by extracting features from laser and camera for classifying six different places: doorways, a laboratory, a kitchen, a seminar room, and a corridor, as well as examining the effect of the Hidden Markov Model on the final classification. The same algorithm, i.e., AdaBoost, was trained in [144] using SIFT features of online images for seven different rooms. It examined the performance of different number of classes and different possible pairs of classes, where the success of the average of binary classifiers was 77%.

Robotics researchers also employed the well-known SVM algorithm for the room classification problem using different sensors. In [145], laser data was used to build a hierarchical model, in which the hierarchy is employed for training and testing SVMs to classify 25 living rooms, 6 corridors, 35 bathrooms, and 28 bedrooms. Although this study reported an accuracy of 84.38%, laser data generally do not provide rich information, and require substantial processing to extract useful features. In contrast, vision features are used in other studies in order to train SVMs. In [146], a voting technique was used to combine 3D features to GIST 2D features, and these were used for training SVMs to classify six indoor places: bathrooms, bedrooms, eating places, kitchens, living rooms, and offices. Furthermore, SVM and Random Forests (RF) classifiers were used and compared in [147] to classify five places: corridors, laboratories, offices, kitchens, and study rooms using RGB-D images from a Kinect sensor. Room detection has also been addressed as an unsupervised learning problem using unlabeled images. In [148], SIFT features and 3D representation were used to extract convex spaces for clustering images based on similarities. In addition, stereo imagery was used in [149] for room detection and modeling by fusing 2D features with geometry data acquired from pixel-wise stereo for representing 3D scenes. The study was completed by modeling walls, rooms, and doorways using many techniques of extracting features, depth diffusion, depth segmentation, and clustering in order to detect room functionalities. The problem has also been addressed from different perspectives, such as the study in [150], in which the authors addressed the context-awareness problem for service robots by developing a system that identified 3D objects using online information. As we can note from previous research, the main drawback is the huge effort required to extract features. This weakness can be overcome by adopting a convolutional neural network (CNN) algorithm.

Convolutional Neural Network (CNN) is a successful approach of deep learning for computer vision application. It has been got an attention after the huge achievements of LeNet [53] and AlexNet [56] in the image classification problem. There are two main advantages of this algorithm over other machine learning algorithms and the conventional fully connected feedforward neural networks. First, CNN extracts and learns features from raw images without requiring a careful engineering design for extracting features in advance [49]. Second, CNN considers the spatial structure of the image by translating inputs to outputs through shared filters [61]. Since the huge improvements in data collection and computer hardware between 1990s and 2012, AlexNet was introduced [56] for addressing the object detection problem using the ubiquitous *ImageNet* to classify 1.2 million images into 1000 different classes. Since 2012, many articulate architectures have been proposed that are essentially built on the early architecture of LeNet, in order to improve the performance on ImageNet database [57]. However, the effective progress that was demonstrated on ImageNet for object classification by these pre-trained models has not shown the same success for the scene classification problem. Consequently, the first significant dataset for scene-centric images, referred to as Places, was proposed in [15]. In general, indoor scene classification is challenging due to features' similarity in different categories. This problem has been studied with different learning methods as well as CNN, which so far has been employed in few studies. In [151], a solution was proposed by designing a model that combined local and global information. The same problem was addressed by applying a probabilistic hierarchical model, which associates low-level features to objects via an object classifier, and objects to scenes via contextual relations [152]. There are also some research studies that have employed CNN for learning robots in indoor environments. Ursic et al. [107] addressed the room classification problem for household service robots. The performance of a pre-trained hybrid-CNN model was examined in [15] on segmented images, i.e., learning through parts, of eight classes from the *Indoor67* dataset. The result generated 85.16% accuracy using a part-based model, which is close to the accuracy of the original hybrid-CNN, 86.45%. However, learning through parts gave much better accuracies on deformed images than the original model. The authors in [153] took advantage of CNN for scene recognition in laboratory environments, with 89.9% accuracy, to enhance the indoor localization performance of a multi-sensor fusion system using smartphones. Furthermore, the objective of [154] was to find the best retraining approach for a dynamically learning robot in indoor office

environments. The paper examined and compared different approaches when adding new features from new images into a learned CNN model, considering the accuracy and training time. The new added images to the features database were the failed ones that were selected and corrected by the user. The authors simulated one of the categories to be the new environment. This paper reported that a pre-trained CNN with a KNN classifier was the most appropriate approach for real robots, as it gave a reasonable accuracy with the shortest training time. All their experiments were executed on the VidRILO dataset [155] using only its RGB frames, i.e., excluding the D frame. The methodology of this presentation, however, is different from previous studies, as we examined several CNN architectures with five categories of indoor scene rooms, i.e., bathrooms, bedrooms, dining rooms, kitchens, and living rooms downloaded from the Places dataset. In addition, these models were examined after cleaning and reducing the number of samples. Furthermore, a combination of CNN and multi-binary classifiers method called ECOC was proposed and evaluated in order to improve the real-time performance on a Nao humanoid robot.

Error correcting output code (ECOC) is a decomposition technique that was proposed by Dietterich and Bakiri for addressing multiclass learning problems [139]. There are a few studies in reported literature that have employed ECOC within CNN architecture, but from a different perspective than the employed in the work of this chapter. Deng et al. [156] used ECOC in order to address the target code issue by replacing the one-hot encoding with Hamming code in the last layer of CNN, which helped reduce the number of neurons in that layer. Then, the CNN model and CNN-ECOC, i.e., CNN with the new target codes, were trained and evaluated separately and the results were compared. Additionally, the same problem was solved in [157] using a different code algorithm referred to as Hadamard code. ECOC within CNN has also been employed in medical applications [158], [159], in which a pre-trained CNN was employed only for extracting features, then multi-binary SVM classifiers trained and combined with ECOC, referred to as ECOC-SVM. Up to our knowledge, this is the only reported work combining and a fine-tuning CNN with ECOC for robotics applications that design multi-binary classifiers of CNN and compare the performance with regular CNN for multiple classes.

3.3. Methodology

The process of this work can be divided into three phases, as shown in Figure 3-1. Phase 1 was aimed at fine-tuning three different CNN models, i.e., VGG16, VGG19, and Inception V3, through transfer learning process on five categories of rooms from the Places205 dataset, in order to select the best model for real experiments on the Nao robot. In addition, all models were examined in this phase after cleaning the dataset by removing all unrelated images to the scenes, and the results were compared before and after cleaning the dataset. In phase 2, the goal was to design multi-binary classifiers of the selected CNN from phase 1 and combine their results through the ECOC algorithm and ECOC-REG. Finally, testing the selected CNN, CNN-ECOC, and CNN-ECOC REG through real experiments on a Nao robot, and comparing the results were the goal of phase 3. The importance of this phase is to show how these models performed on real time experiments with robot's images, e.g., Nao, in which those images are quite different in the level of view from the existed dataset.

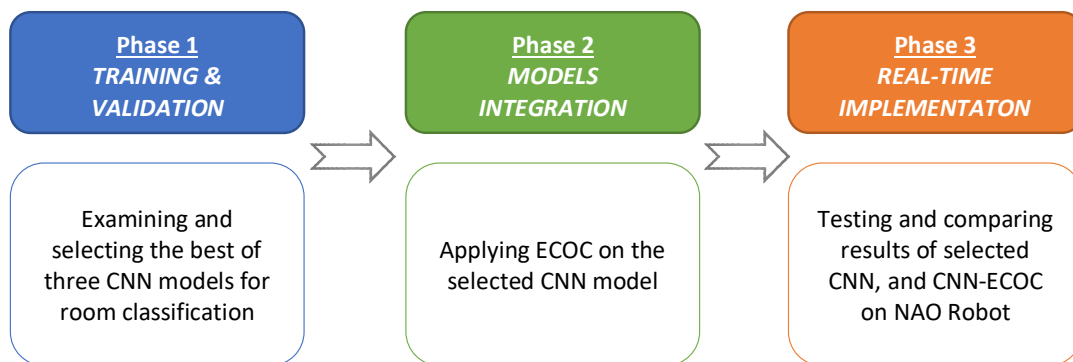


Figure 3-1: Process of simulation and real-time experiments for room classification problem.

3.3.1. Phase 1: Training and Validation

Adopted Scene dataset

There are several scene datasets proposed in the literature for addressing object/scene detection or classification problems. Some of them are small-scale datasets such as the 15-scene dataset, UIUC Sports, and CMU 300, and some are large-scale datasets such as 80 Million Tiny Image Dataset, PASCAL, ImageNet,

LabelMe, SUN, and Places [160]. The dataset can be 3D scene, such as SUNCG [161], or it can be images for a particular environment with geo-referenced pose information for each image, such as TUM and NavVis [162]. These datasets can be classified into two view types: object-centric datasets, e.g., ImageNet, and scene-centric datasets, e.g., Places [15]. Places is the latest and largest scene-centric dataset, which is provided by MIT Computer Science and Artificial Intelligence Laboratory for the purpose of CNN training. It has a repository of around 2.5 million images classified into 205 categories, and for this reason it is called the Places205 dataset. This dataset is updated and extended with more images classified into 365 categories in [163], which is called Places365.

Since this project is within the scope of household robotics applications, five categories of images were selected to be downloaded from Places205 for addressing room–scene classification problems for social robots using CNN models. The five categories are: bedroom, dining-room, kitchen, living-room, and bathroom, which most, if not all, houses have. It should be noted that the corridor category is not available in Places205 and Places365 at the time of this work. This category is important in this research and will be incorporated in the design once it is available. 11,600 images/category were used to train the CNN model, where 20% of images were used for validation, i.e., 2320 images/category.

Cleaning dataset

The Places dataset is regarded to be very important in the field of computer vision and deep learning. However, there are some issues with the downloaded images for real time robotic applications that affect the learning process. Therefore, we manually excluded some images from all five categories, based on criteria that are shown in the few examples in Figure 3-2.

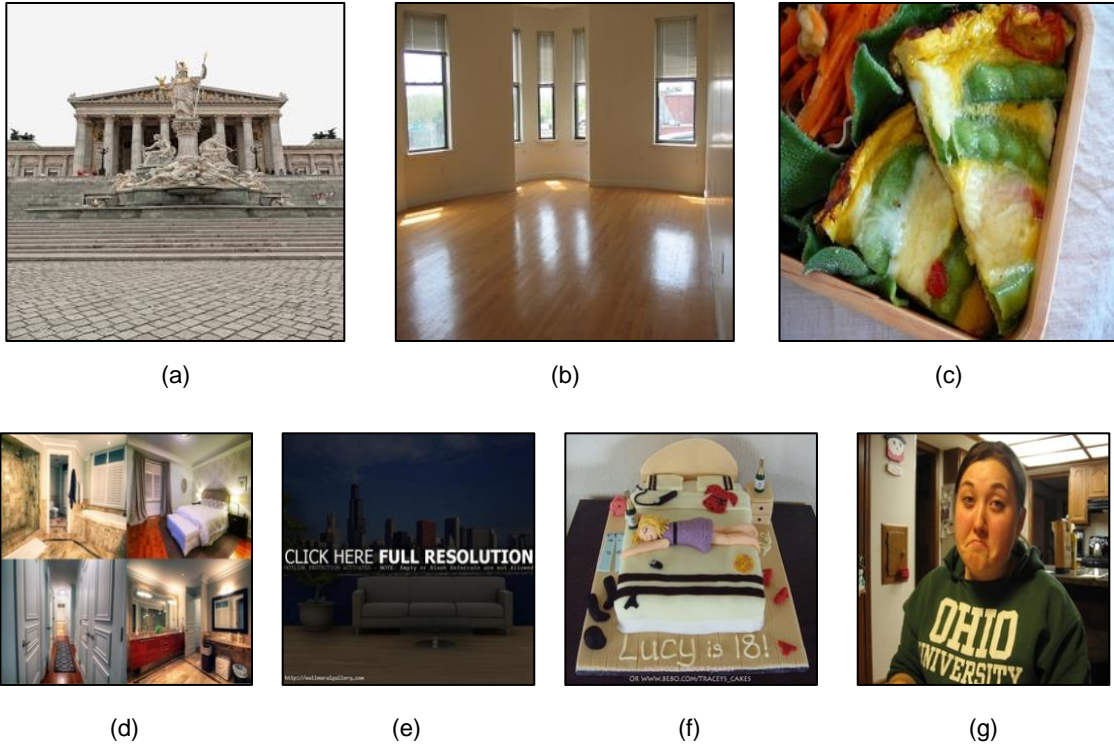


Figure 3-2: Examples of removed images from room dataset of Places. (a) Not belonged to; (b) no furniture; (c) not wide scene; (d) multi-scenes in an image; (e) including texts; (f) fake scene; (g) focusing on people.

Table 3-1 shows the percentage of the data that were deemed irrelevant form each category. After cleaning the data, we noted that the remaining images for bedrooms were the highest and for kitchens were the lowest (Table 3-1). The reader might note the high percentage of irrelevant images in each category, which justifies the need for cleaning data.

Table 3-1: Number of images for each class after cleaning.

Class	# images out of 11,600	% of removed images
Bedrooms	9323	19.6 %
Dining Rooms	7919	31.7 %
Kitchens	6611	43.0 %
Living Rooms	8571	26.1 %
Bathrooms	8959	22.8 %

CNN architectures

Since our main concern was to recognize only five room classes, we did not require huge amounts of data. Additionally, the learned features from pre-trained models in literature are relevant to the room classification problem, therefore transfer learning of pre-trained models was the best strategy for this work, instead of training a CNN model from scratch. Transfer learning can be achieved through two main steps. The first step is to proceed room images to non-trainable ConvNet in order to extract features, then use these features to train our new classifier, i.e., SoftMax layer. The second step is to retrain the whole network, i.e., ConvNet and classifier, with a smaller learning rate, while freezing a few layers of the ConvNet. Several CNN models were fine-tuned for this project, i.e., VGG16, VGG19 [58], and Inception V3 [60], with different freezing layers to be trained through transfer learning process. All these CNN models were followed by a similar fully connected (FC) layer. VGG16 and VGG19 architectures are similar to what was shown in Figure 2-4. Whereas Inception architecture is shown in Figure 3-3. We considered these architectures in this project based on their popularity and performance with the limited samples and classes in this project. The main objective of the VGG network is to improve the performance by increasing the depth of layers to 16 or 19. Whereas, the Inception network focuses on reducing the high computational cost in the VGG network by merging 3×3 and 5×5 filters that are preceded by 1×1 filters.

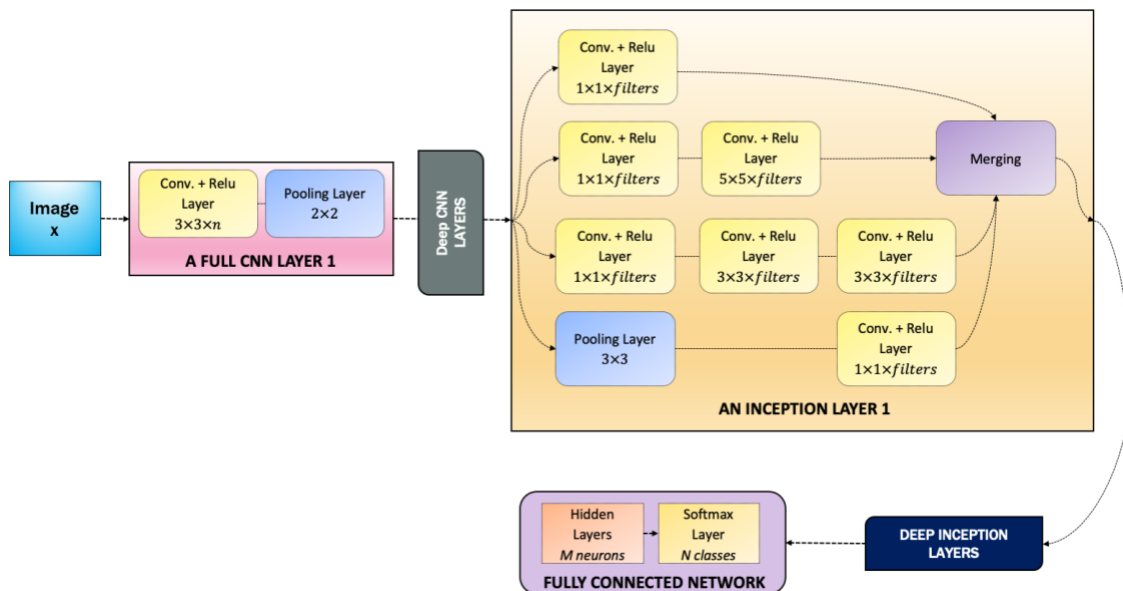


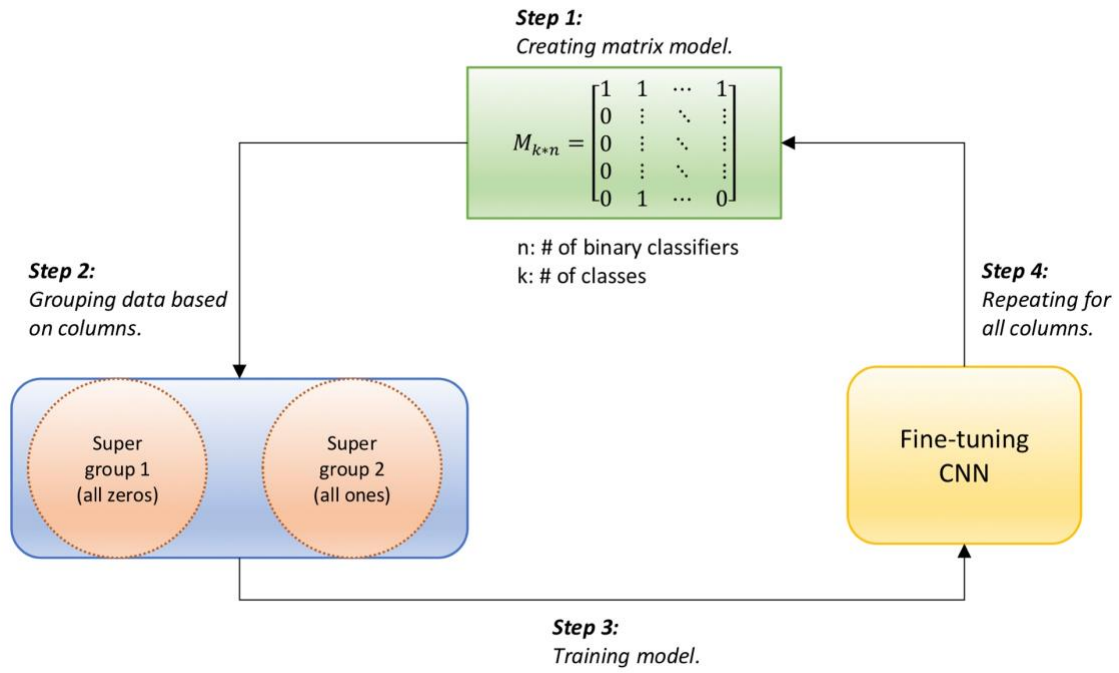
Figure 3-3: Concept of inception architecture.

3.3.2. Phase 2: Models Integration

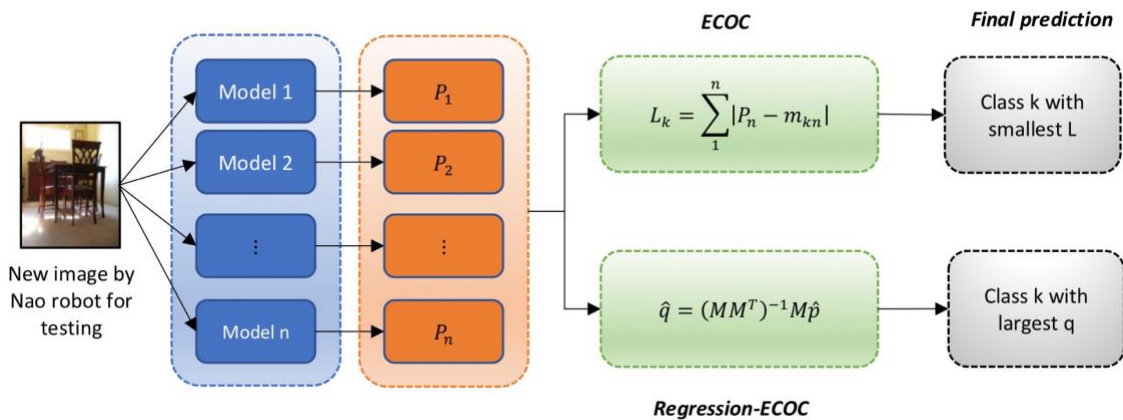
ECOC Technique

In practice, a regular CNN classifier via transfer learning for multiple room classes may not be sufficient to assist the robot in making the right decision, due to high similarities among the rooms and the different views from robots compared to the exist dataset. Thus, it is very important to find a way to improve the robot's best decision, even when applying a CNN classifier with high accuracy. We suggest adopting a decoding approach to address this problem [164]. The purpose of this technique is to improve the practical performance by designing a new classifier that combines multi-binary classifiers through an algorithm. This approach is also known in literature as decomposition [165] or plug-in classification technique (PICT) [166]. There were two main reasons for adopting this technique for this project. The first reason was to take advantage of higher accuracy with binary classes. The second reason was that designing multi-binary classifiers was feasible in the room classification problem, as the number of room classes in houses is limited.

One of the most popular decomposition techniques is error correcting output code (ECOC). It was proposed by Dietterich and Bakiri to address multiclass learning problems [139]. The main concept of this algorithm is to create a binary matrix code that represents multi-binary classifiers of two super groups. Each group consists of many classes, in order to alleviate the overall error in order to obtain the right classification. As shown in Figure 3-4, the algorithm consists of two stages. The first stage is to create and train multi-binary classifiers. It starts with creating a binary matrix code, in which the number of rows represents the number of classes and the number of columns represents the number of binary models. The data is then classified into two super groups based on zeros and ones in respective columns, where all classes with zeros are assembled in the first group, and the rest of classes are collected in the second group. The last step in this stage is to train all binary classifiers, i.e., the best CNN architecture for this problem, based on columns of the matrix using their super groups. The second stage is to predict a new image using all trained models. Each model gives a probability of predicting one of the super groups. After getting all probabilities of all models, we calculate the distance between all predictions p and each row in the matrix code. The smallest distance will be considered the correct class of that input image.



(a)



(b)

Figure 3-4: Error correcting output code (ECOC) process for addressing multi class learning problems. (a) Stage 1: Process of training all binary classifier, and (b) Stage 2: Process of predicting a class of a new image.

The alternative way to get the overall classification is *Regression-ECOC*, which is using the least squares instead of Euclidean distance. Thus, the correct class will be the maximum value of the following equation:

$$\hat{q} = (MM^T)^{-1}M\hat{p}$$

3.3.3. Phase 3: Real-Time Implementation

This phase is the main purpose of this project, which is to test out the performance of different models on Nao robot using its monocular camera. The goal is that the Nao robot should be able to predict the room class using several trained models from phase 1 and 2. Nao will be transferred to several houses in Vancouver, BC, Canada for real-time experiments. It is important to mention here that Nao's height is only 573 mm, whereas its top camera is located at the level of 514.29 mm. This short height is expected to influence the overall results, which will direct our future work. Through the main software of Nao, i.e., *Naoqi*, several methods can be used from *AIPhotoCapture* module for real experiments as explained in the Appendix.

3.4. Experiments and Results

All codes of this study were programmed by Python 2, in which the CNN architectures were built via Keras API as a high-level building block for deep learning and TensorFlow backend for the low-level operations. Training and validating CNN models were executed offline through Graham cluster provided by Compute Canada Database for concurrent running of several tasks or programs. However, the real time experiments were implemented via connecting Nao robot to a laptop with 2GB graphics memory through the Wi-Fi. Accordingly, the implementation python code for real-time experiments was encoded in Python 2 with Naoqi API, see the Appendix, while applying the pretrained model for classifying different scenes captured by Nao's camera.

3.4.1. Phase1: Validating Room Classification within CNN Models Using Places

Several CNN models were fine-tuned through transfer learning process for this project, i.e., VGG16, VGG19, and Inception V3, with different freezing layers to be

trained. All these CNN models were followed by a similar fully connected (FC) layer. FC begins with an average pooling layer, then a layer of 1024 neurons with the *ReLU* activation function, and ends with a logistic layer to predict one of the five classes. Keras provides a compile method with different optimizers for learning process [167]. In the first stage of the fine-tuning, the *Adam* optimizer was used with a 0.001 learning rate, whereas we applied the SGD (stochastic gradient descent) optimizer in the second stage with learning rate of 0.0001 and momentum of 0.9. All models were trained for 10 epochs in each stage with both the original data as well as the cleaned data, and with different non-trainable layers. It was noticed that training with more epochs did not provide that much improvement in the final accuracy, but it took a very long time in the training process. Table 3-2 shows the superior results of all models trained with clean data compared to all data. The best result shown from these experiments is VGG19 and VGG16 with 0 freezing layers using clean data, which gives an accuracy of 93.61% and 93.29%, respectively.

Table 3-2: Comparison of accuracies of fine-tuning different CNN models using all data and clean data (the shaded results are the best for the real-time experiment).

CNN Models	Non-trainable layers	All Data		Clean Data	
		Time	Accuracy %	Time	Accuracy %
VGG16	15	11:50:40	86.03	8:16:16	89.69
	11	11:50:41	88.09	8:15:38	91.49
	7	11:53:42	88.9	8:18:49	93.22
	0	12:13:55	87.78	8:34:45	93.29
VGG19	20	13:11:07	78.69	9:16:00	82.50
	17	13:14:56	86.22	9:18:38	89.65
	0	13:43:40	90.30	9:40:52	93.61
Inception V3	299	10:17:29	75.12	7:8:33	78.83
	249	10:17:46	79.11	7:07:50	84.05

3.4.2. Phase 2: Validating Room Classification within the integration of CNN and ECOC

The best two models in phase 1 were VGG19 and VGG16 with all layers fine-tuned. Although this work was carried out through one of Compute-Canada Servers [168], i.e. Graham, there are many works in robotic applications that can be processed using local machines. Therefore, considering the time of training is an important factor

for this phase, which has multiple binary classifiers for training. For this reason, the selected model to be trained in this phase was the VGG16 with 0 freezing layers, which has an accuracy of 93.29% that is quite similar to the best one. The binary classifiers can be designed through grouping classes based on an exhausted matrix code, as explained in [139]. The following 5×15 matrix is the best for this experiment, as it does not have any repeated and complimented columns.

$$M_{5 \times 15} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} \textit{bathroom} \\ \textit{bedroom} \\ \textit{dining room} \\ \textit{kitchen} \\ \textit{living room} \end{bmatrix}$$

Let us take an example of the classifier in column 3 of matrix M , which has [1 0 0 1 0] values. The first group of this classifier would be the classes with zero value, i.e., bedrooms, dining rooms, and living rooms. The second group is the classes with ones, i.e., bathrooms & kitchens. Table 3-3 shows the validation accuracies of all 15-binary fine-tuned VGG16 classifiers. The main advantage of the binary classifier is high accuracy depending on classification, as it reached 98.5% for this project, and the average of all 15 classifiers was 95.37%, which is still higher than the multi-classification approach.

Table 3-3: Accuracies of 15 binary classifiers.

Binary classifiers	Bath vs. All	2	3	4	5	6	7	Bed vs. All	9	10	11	Dining vs. All	13	Kitchen vs. All	Living vs. All	Average
Accuracy %	98.50	93.62	97.38	92.01	94.84	93.95	96.06	95.73	95.59	96.10	94.94	95.89	92.76	97.95	95.31	95.37

Discussion

One of the most important features that has been studied by researchers is deepness, which is significant in most of the known CNN architectures. It is reported that the deeper and wider the architecture is designed, the better features will be learned [60]. However, this requires a huge dataset, i.e., millions of samples, in order to avoid the overfitting issue. Unfortunately, this is not always the case in robotics applications, wherein in most applications, the number of classes for a specific robotic problem such as room classification is very limited, which means the number of samples might be only thousands, i.e., a small dataset. Therefore, the very deep CNN architecture will most likely lead to overfitting problem, as happened with ResNet [59] in this work, which is

why their results are excluded from these experiments. Although the Inception V3 results were not over-fitted, they were less accurate than the architectures with less deepness, i.e., VGG16 and VGG19. The reason might be related to the scene-centric type of dataset, in which learning hierarchical representations can be difficult with more deepness. For this work, we had two ways to address this problem for robotic application: either designing a new CNN architecture for a small dataset or adopting an existing CNN with less deepness and improving robot's decision by integrating another method. The latter solution was preferable, so we adopted VGG16, i.e., the least deepness of all three architectures, and integrated it with ECOC in a way it was practicable for real time robotic implementation. One could ask why we adopted CNN from the beginning—as discussed before, CNN extracts and learns features from raw images without requiring a careful engineering design that shown superiority over conventional approaches in computer vision.

Binary classifier results for ECOC explain the challenge of feature similarities in a house's rooms. Let us discuss the obvious results of 'class vs. all' in classifiers number 1, 8, 12, 14, and 15. The most distinguishable rooms are the bathroom and kitchen, as shown in classifiers 1 and 14 respectively. Meanwhile, the other three classes are very similar to each other. There are many reasons related to the dataset or the architecture of VGG16 that the model is less accurate with these similar rooms than the distinguishable ones. The first reason is having some sharable objects in different rooms, such as tables or TVs, or the wide variety styles of those rooms such as open/closed spaces, or even culture-based styles, e.g., no beds for sleeping. The second reason is that the architecture with less deepness will not be able to differentiate between objects similar in shape, e.g., rectangular shapes in dining tables, coffee tables, and beds from different rooms. Therefore, it is a tradeoff between learning deep features and having a small dataset. The results of the third phase will determine the direction of the future solution.

3.4.3. Phase 3: Validating Room Classification on Real-Time Implementation Using Nao robot

The proposed methods were tested and compared practically in five different houses using the Nao humanoid robot as shown in Figure 3-5. The goal is that the robot should be able to predict the room class using the three different models, i.e., Regular

CNN, CNN-ECOC & CNN-ECOC-REG, in which all models return the probability for each class. Since rooms in the houses had different sizes, layouts, furniture, etc., Nao was positioned in different spots in the rooms, i.e., center, corners, beside the wall or the door, during different time of the day under different light conditions. Accordingly, 56 images were taken by its top camera as follows: 12 bathrooms, 13 bedrooms, 6 dining rooms, 13 kitchens, and 12 living rooms, while the robot's head faced the x direction and the top camera covered 60.97° horizontally and 47.64° vertically. Figure 3-6 shows some examples of scenes taken by the Nao humanoid robot. An important point to be noted is that Nao is very short compared to an average human's height, and its camera is mounted about 514 mm from the floor level. This is quite different from the field of view of the images from the adopted dataset. Consequently, the highest (top-1) probabilities were negatively affected, especially in the kitchen class. However, the second highest (top-2) probabilities show the superiority of CNN-ECOC and CNN-ECOC-REG over the regular CNN for prediction of most of the five classes.

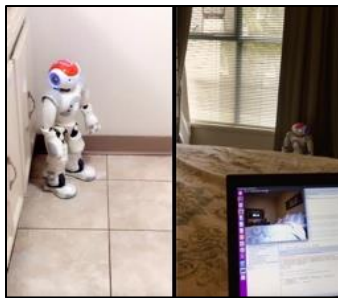


Figure 3-5: Nao humanoid robot during experiments.

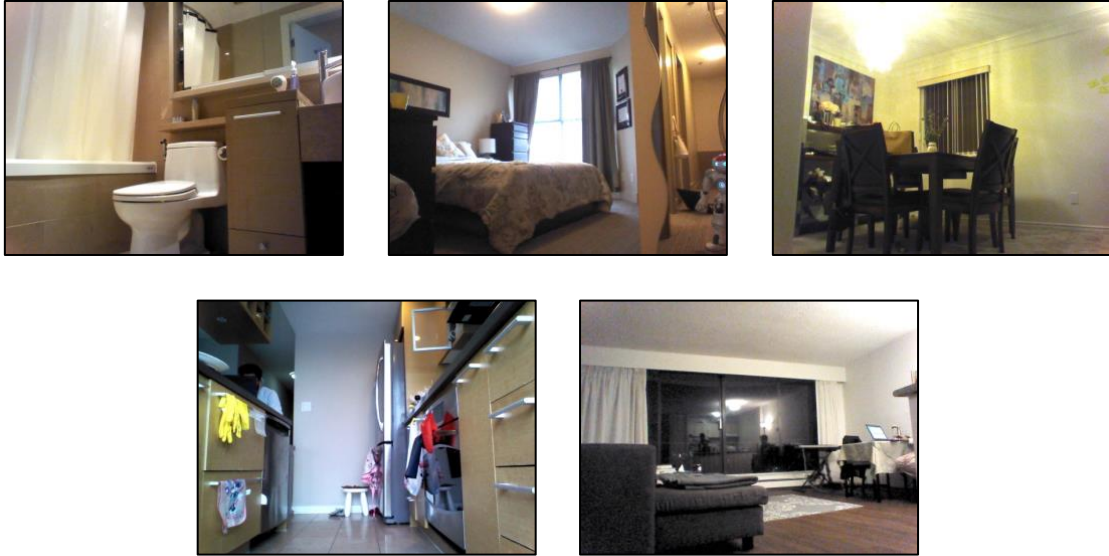


Figure 3-6: Scenes examples taken by Nao humanoid robot*.

Table 3-4 shows the testing accuracy of the five classes, whereas Table 3-5 gives confusion matrix for both top-1 and top-2 predictions. Notice that the confusion matrix of top-1 tables has all 56 images, however the confusion matrix of top-2 tables includes only the number of false predicted images in the top-1 table. The best room prediction was the bathroom, where accuracy was 100% with all three models. Therefore, the bathroom was considered to be the most distinctive room. All three models were similar at predicting bedrooms, in which their top-1 accuracy was 69.2%, increasing to 100% in top-2. The distinction in performance between these models is shown in the prediction of dining rooms, kitchens, and living rooms. The top-1 predictions of dining rooms for three models were similar, however, the top-2 prediction of CNN-ECOC and CNN-ECOC-REG increased to 100%, which was better than CNN. In the living room cases, the CNN and CNN-ECOC-REG gave the best rate of 50% in the top-1, however the CNN-ECOC and CNN-ECOC-REG were able to increase up to 91.7% and 100% in their top-2 predictions, respectively. Although the kitchen top-1 prediction was the worst in all models, it surprisingly showed a huge improvement in top-2 prediction for CNN-ECOC and CNN-ECOC-REG compared to CNN. Overall, the three models performed similarly in the top-1 prediction, however the multi-binary classifier

* All Captured images by Nao were saved and added to the same project in this link: <https://github.com/KamalOthman/SRIN-Dataset.git>

solutions, i.e., CNN-ECOC and CNN-ECOC-REG, gave much better performance in the top-2 results in contrast to the CNN for multi-classification.

Table 3-4: Testing accuracies of all models with top-1 (T1) and top-2 (T2) for all five classes.

Model	Bath		Bed		Dining		Kitchen		Living	
	T1	T2	T1	T2	T1	T2	T1	T2	T1	T2
CNN	100	-	69.2	100	66.7	83.3	15.4	61.5	50	75
CNN-ECOC	100	-	69.2	100	66.7	100	15.4	100	41.7	91.7
CNN-ECOC-REG	100	-	69.2	100	66.7	100	30.8	100	50	100

Table 3-5: Confusion matrix of testing images with top-1 & top-2 for five classes.

		# Images	# False Images in Top 1	bath	bed	din	kit	liv
Top-1 with CNN	bath	12		12				
	bed	13		4	9			
	din	6		2		4		
	kit	13		10			2	1
	liv	12		4	2			6
Top-2 with CNN	bath		0					
	bed		4		4			
	din		2		1	1		
	kit		11	1	4		6	
	liv		6		2	1		3
Top-1 with CNN-ECOC	bath	12		12				
	bed	13		4	9			
	din	6		2		4		
	kit	13		11			2	
	liv	12		6	1			5
Top-2 with CNN-ECOC	bath		0					
	bed		4		4			
	din		2		1	1		
	kit		11				11	
	liv		7			1		6
Top-1 with CNN-ECOC-REG	bath	12		12				
	bed	13		4	9			
	din	6		2		4		
	kit	13		9			4	
	liv	12		4	1	1		6
Top-2 with CNN-ECOC-REG	bath		0					
	bed		4		4			
	din		2			2		
	kit		9				9	
	liv		6					6

Discussion

After obtaining results of the multi-binary classifiers in Table 3-3, we expected to get the best results with the most distinguishable rooms, i.e., bathrooms & kitchens, on phase 3. The results were as expected with bathrooms, but not with kitchens. The

reason is the short height of Nao, which captured scenes that are totally different from what was learnt from the dataset. In the kitchen case, the robot captured the cabinets and drawers rather than capturing the top view of stoves or other appliances. Thus, the robot's first prediction was mostly bathrooms instead of kitchens. The ideal solution is to have our own dataset for social robots in the height range of 0.5–1.0 m, which will be proposed and discussed in the next chapter.

3.5. Conclusion

This chapter focused on addressing the room classification problem for social robots. The CNN deep learning approach was adopted for this purpose because of its superiority in pattern recognition applications [169]. Several CNN architectures were examined, fine-tuned and trained on five room classifications via Places dataset. We concluded that VGG16 was the best model, with 93.29% of validation accuracy after cleaning the dataset by excluding all mislabeled images. In addition, we proposed and examined a combination of CNN with ECOC, a multi-binary classifier approach, in order to address the error in practical prediction. The validation accuracy reached 98.5% in one of the binary classifiers and 95.37% in the average of all binary classifiers. The CNN and the combination model of CNN and ECOC in both forms, i.e., CNN-ECOC and CNN-ECOC-REG, were evaluated practically on a NAO humanoid robot. The results show the superiority of the combination model over the regular CNN.

During the course of this part of the research, we discovered that the existing datasets are not quite suitable for medium-sized (short) social robots with limited sensors. Hence, we embarked on the possibility of designing a new dataset for such robots. The process will be explained in the next chapter.

Chapter 4.

SRIN: A New Dataset for Social Robot Indoor Navigation*

4.1. Introduction

Providing a seamless and reliable solution to indoor navigation is a central research problem in robotics as resolving this challenge is a precursor for success of many activities of a social robot. Indeed, achieving the ultimate objective of having a social robot in every home depends on a reliable solution to this problem. Social robots will be part of the family as pets are. They interact and assist in chores and will keep company for minors and seniors. Within this context, they must be able to flawlessly roam around the home and be able to identify different locations and their functionalities in a house. In the previous chapter, we presented a CNN-based model (*Convolutional Neural Network*) that demonstrated promising results with respect to room classification in an indoor setting. As training a CNN-model requires a significant number of samples, there are many models trained on popular computer visions datasets, such as ImageNet [170] and Places [163]. However, adopting pre-trained CNN models that learned features from computer vision datasets to be tested in real-time experiments on social robots, e.g. Nao humanoid robot [171] was not overwhelmingly successful [172]. We suggest that a dedicated dataset as opposed to general datasets such as ImageNet or Places could drastically improve the performance in real-time experiments.

Thus, the objective of this chapter is to report a new dataset called SRIN, which stands for Social Robot Indoor Navigation, for room classification and doorway detection applications in indoor environments – particularly in homes. This dataset has its unique feature for social robots with medium size and height, such as Nao humanoid robot, shown in Figure 4-1 which indicates the height of the robot with respect to the door handle. SRIN is a 2D RGB images dataset that consists over 75000 raw and augmented images for several rooms and doorways, i.e. SRIN-Rooms and SRIN-Doorway datasets, respectively. The distinct feature of the dataset is that all collected images were captured

* This chapter is mainly reproduced from paper 2 in section 1.4 (page 8).

at the height of 0.5 m, which will be practically a useful dataset for social robots with medium size.

The rest of the chapter is organized as follows: Section 4.2 introduces related studies and presents different examples of robotics datasets as well as different examples of social robots with medium size. Next, we introduce the process of collecting SRIN dataset in section 4.3. We include some experiments and results in section 4.4 that demonstrates the superiority of SRIN to train CNN-model for robotics application and compare them with the results in the previous chapter. We conclude the chapter with additional remarks in section 4.5.

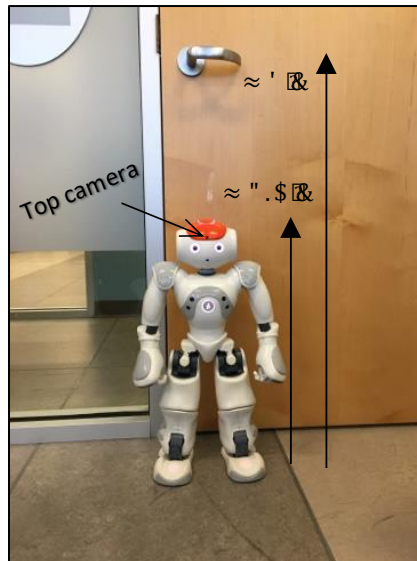


Figure 4-1: Nao robot's height with respect to the door handle.

4.2. Related work

By and large, the performance of indoor robotic navigation tasks that employ general computer vision datasets has been rather poor [173], [174]. To circumvent this shortcoming, robotics researchers have started developing their own dataset based on the robot's platform and its onboard sensors. We suggest that creating a robot-specific dataset improves the performance of navigation tasks. In [175], a dataset was acquired from a specific environment (only two locations of labs) using Pioneer and Virtual Me robots in which the camera is mounted in the height of 88 cm and 117 cm, respectively. Although it is reported that 100-500 images per class for 17 classes were collected, the variety of the images was limited for the two main locations. The dataset of

Robot@Home [173] is a collection of 83 time-stamped observations obtained from a mobile robot equipped with 4 RGB-D cameras and a 2D laser for robotics mapping applications in indoor domestic environments. Autonomous Robot Indoor Dataset, ARID [174], is an object-centric dataset that was collected by RGB-D camera on a customized Pioneer mobile robot, in which the camera was mounted at a height of over 1m. The main objective of that study was to test and compare previous CNN performances to recognize 51 objects using more than 6000 RGB-D image scenes. Another dataset referred to as AdobelIndoorNav [176] was collected for Deep Reinforcement Learning (DRL) robotics applications from 24 scenes. Each scene consisted of 3D reconstruction point cloud, 360 panoramic view from grid locations, and 4 different images in 4 different directions using sensors mounted on a Turtlebot mobile robot.

An affordable social robot in every home is one of the robotics researchers dreams in the near future. Such a robot most likely has a limited number of sensors, such as a monocular camera instead of a stereo camera or expensive 3D sensors. Such domestic social robots are probably medium sized and are the same height as domestic pets. There are already several types of mobile robots with a reasonable size in the market, i.e. less than 1 m. Among such robots are those with the average height of ~ 0.6 m, such as Nao by Aldebaran [171], QRio by Sony [177], [178], Zeno by Hanson Robotics [179], [180], Manava by Diwakar Vaish in the labs of A-SET Training & Research Institutes [181], and DARwIn-OP by Dinnes Hong in Robotics and Mechanisms Laboratory [182], [183]. There are also other class of robots which are a bit shorter, i.e. the average of 0.4 m, such as HOVIS [184] and Surena-Mini [185], or a bit taller, i.e. the average of 0.8 m, such as Poppy [186]. However, there is no suitable dataset for this kind of robots that can be used to address houses' environments navigation tasks such as recognizing room classes. Furthermore, testing a pure learned CNN-based model that trained on an existing computer vision dataset, such as Places dataset, with a medium-sized robot like Nao, gives an undesirable performance as shown in [172]. Therefore, we propose a new *scene-centric* dataset called SRIN, which stands for Social Robot Indoor Navigation dataset. This dataset has been collected to be employed for indoor navigation tasks on Nao humanoid robot or any other robot with a similar height.

4.3. Collecting Dataset and Methodology

The main feature of the RGB images of SRIN dataset is that all raw images were taken at a height of 0.5 m above ground from several houses in Vancouver, BC. In order to generalize our work for any social robot similar to Nao as well as to simplify the process of increasing dataset in future without depending on taking Nao to different houses, it was preferable to start taking images by personal cameras restricted to the condition of the 0.5 m height. These images were used to train and validate a CNN-based model, then the trained model was tested on Nao robot in new environments. SRIN dataset contains +75000 of the raw and augmented images for several rooms and doorways, i.e. SRIN-Rooms and SRIN-Doorway datasets. The procedure of collecting and increasing this dataset has been done through three main steps as shown in Figure 4-2:

1. Over 500 raw images of five rooms classes (bedroom, bathroom, dining room, kitchen and living room) and three doorway classes (open-door, closed door or no-door) were captured from seven different houses.
 - a. From each room, 5-8 images were collected with different scenes from different angles.
 - b. In front of each door, 5-9 images were collected with different scenes from different distances and angles.
 - c. If the room had a window, then more images were taken during the day, based on the sun light, and the night, based on the house's lights.
 - d. All raw images were resized to a resolution of 256×256 .
2. Each image was flipped and changed with Gaussian noise separately using OpenCV libraries. Therefore, the number of images was tripled.
3. An aggressive augmentation process using Keras API [167] was applied on all images, creating the new dataset. This augmentation includes a random combination of rotation, width shift, height shift, shear range, zoom range and channel shift range.

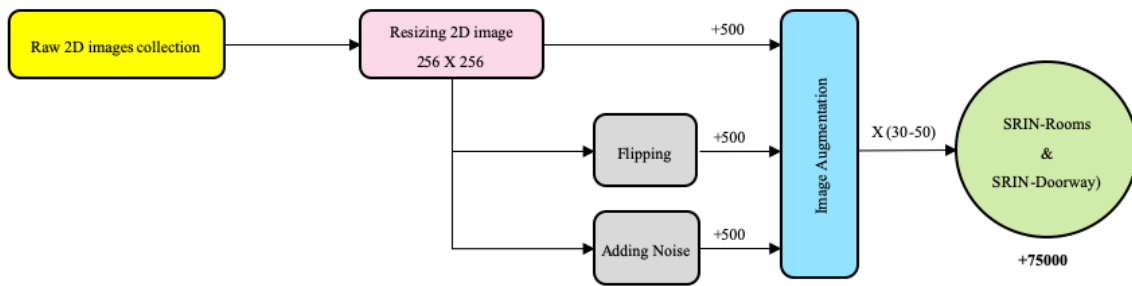


Figure 4-2: Collecting Process of SRIN Dataset.

SRIN dataset has several unique characteristics among which are: images are captured from the view of short robots as opposed to humans' view; and the dataset allows other researchers add images from indoor homes of different countries, cultures, etc. The criteria are also basic as it required only RGB images from different layouts form different rooms in a home. Figure 4-3 provides clarification of how the SRIN's content has been further processed by showing a raw image and its associated processed and augmented images per category.






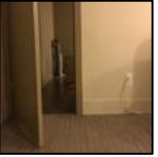









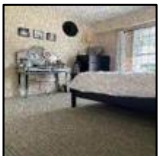



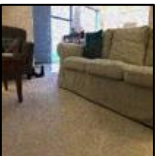







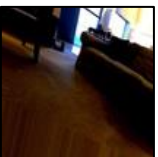



	SRIN-Rooms Samples					SRIN-Doorway Samples		
	<i>Bed</i>	<i>Bath</i>	<i>Din</i>	<i>Kit</i>	<i>Liv</i>	<i>No-door</i>	<i>Open door</i>	<i>Closed door</i>
Raw data samples (resized)								
Processed samples	<i>flipped</i> 							
	<i>Added noise</i> 							
Augmented samples								

Figure 4-3: Scene samples from SRIN dataset for each category*.

* All samples and python codes can be downloaded from the author's GitHub page via this link: <https://github.com/KamalOthman/SRIN-Dataset.git>

4.4. Experiments & Results

In this section, we present the superiority of training the CNN-based model using SRIN dataset, i.e. let's call it CNN-SRIN, over Places dataset for indoor robotic applications. We present the validation accuracy, then test the trained model on real-time experiments with Nao robot. For appropriate validation and comparison, we applied the same CNN training process, via the concept of transfer learning as shown in Figure 4-4 and explained in detail in [172], with only SRIN-Rooms dataset for training the model. Then both CNN-SRIN results and CNN with places results in [172] have been compared with each other. The number of room samples in SRIN-Room dataset was 37288 images divided as follow: [bathrooms: 7538, bedrooms: 7634, dining rooms: 7561, kitchen: 7185, living rooms: 7370]. 20% of images of each class have been used for validation. Thus, the total images for training were 29832 samples, while 7456 samples were employed for validation.

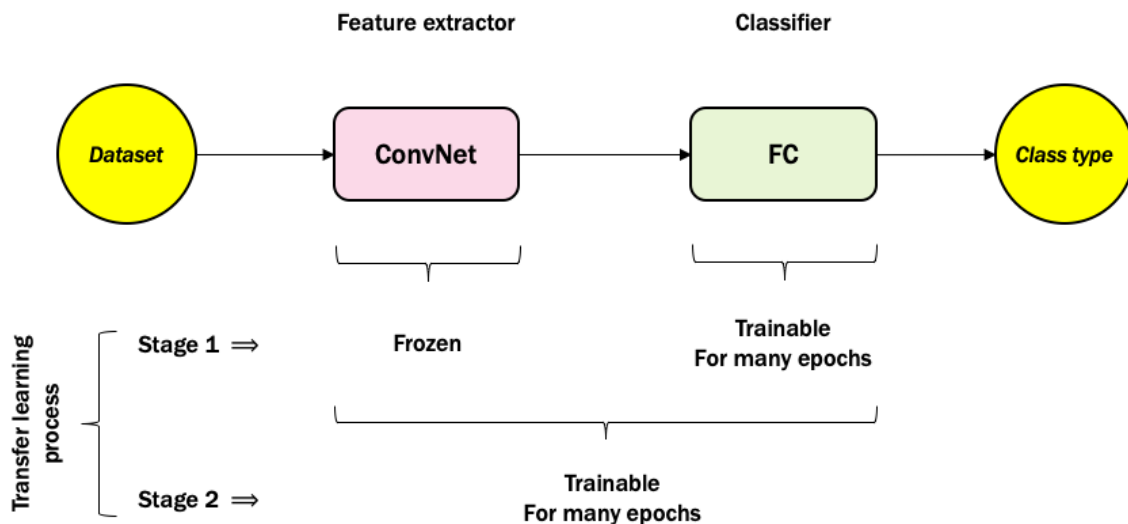


Figure 4-4: The concept of transfer learning of CNN. ConvNet: any pre-trained Convolutional Network & FC: Fully Connected Network.

As mentioned in [172], training the CNN model using places dataset for room classification problem was accomplished through two stages, see Figure 4-4. First stage was the training of the classifier part, i.e. fully connected network, while the feature extractor part, i.e. VGG16, was frozen, which implies that all their weights were non-trainable. The average of validation accuracy of this stage could not increase beyond

87%. In order to improve the performance, we added the second stage whereby the whole model was trained, i.e. both the feature extractor and the classifier parts. The validation accuracy reached to 93.29% after around 8:35 hours of training process on Graham cluster provided by Compute Canada Database [168]. Interestingly, training CNN-SRIN reached a validation accuracy of 97.3% after 1:32 hours of applying only the first stage of CNN transfer learning that is shown in Figure 4-4. Therefore, there was no need to go through the second stage.

As real-time experiments with Nao was our main concern, we tested the trained CNN-SRIN model on the same Nao images from [172]. Figure 3-6 from Section 3.4.3 shows some examples of Nao images that were used for real-time experiments. Table 4-1 shows the comparison results of rooms predictions on Nao images for both trained models, i.e. CNN with places dataset and CNN-SRIN. All results are shown as a confusion matrix for each class. Each model gives the top-1 and top-2 predictions, in which the top-2 is the results of the false prediction from top-1 results. As it is obvious in the results of all confusion matrices of Table 4-1, the overall correct prediction from top-1 and top-2 of CNN-SRIN outperformed the CNN model with Places. Results of four room classes in CNN-SRIN attained 100% correct prediction in top-2 results. We believe that with increasing the SRIN dataset in future, the prediction of top-1 for real-time experiments on Nao shall be improved as well.

Table 4-1: Comparison Results: confusion matrix of each class with top1 & top2 predictions for both models (CNN with places & CNN-SRIN) on a Nao robot.

CNN-Room Classification with Places dataset (from Table 3-5)																		
		# images	Bath	Bed	Din	Kit	Liv	%			# false	Bath	Bed	Din	Kit	Liv	%	Overall %
TOP1	Bath	12	12					100	TOP2	0							-	100
	Bed	13	4	9				69.2		4		4					100	100
	Din	6	2		4			66.7		2		1	1				50.0	83.3
	Kit	13	10			2	1	15.4		11	1	4		6			54.5	61.5
	Liv	12	4	2			6	50.0		6		2	1		3	50.0	75	
CNN-Room Classification with SRIN dataset																		
		# images	Bath	Bed	Din	Kit	Liv	%			# false	Bath	Bed	Din	Kit	Liv	%	Overall %
TOP1	Bath	12	12					100	TOP2	0							-	100
	Bed	13		12			1	92.3		1		1					100	100
	Din	6			2		4	33.3		4			4				100	100
	Kit	13	6	1	1	3	2	23.1		10		1	1	5	3	50	61.5	
	Liv	12	1	6			5	41.7		7					7	100	100	

Discussion

As this work was concerned about the practical performance on a medium-sized mobile robot, i.e. Nao, Table 4-1 provided the promising practical results of SRIN dataset. Let us discuss the CNN-SRIN results based on the type of rooms. The “*bathroom*” is the most class with its unique features in its scenes in which the CNN-Room models (with Places and SRIN) predict 100% correctly in top1. Similarly, the “*bedroom*” is considered as a unique class that both CNN-models were able to get the 100 % correct predictions in top2. However, CNN-SRIN was much better in top1 results as it predicts 92.3% comparing to the 69.2% of CNN with Places. Next, despite the low percentage of CNN-SRIN prediction for the “*dining room*” and “*living room*” in top1, both classes were reached up to 100% in top2 results which is considered a huge improvement compared to their results of CNN with Places. The most arguable class is the “*kitchen*”. As discussed in section 3.4.3 of the previous chapter, the robot captured the cabinets and drawers rather than capturing the top view of stoves or other appliances. Thus, first predictions of CNN with Places were mostly bathrooms instead of kitchens. However, we consider this case was improved with CNN-SRIN from two perspectives. The first perspective is that the top1 percentage raised from 15.4% to 23.1% although they are quite low. The second perspective is that the number of the first predictions as “*bathroom*” were less than CNN with places, which means the CNN-SRIN model learned different features. While, both models were reached the same percentage in the top2 results, the questions now are how can practically top1 results be improved and how can the robot make a decision based on top1 results? First, we believe that with increasing the SRIN dataset in future, the prediction of top-1 for real-time experiments on Nao shall be improved as well. Second, as what we noticed with most of the “*dining room*” predictions that their probabilities of CNN-SRIN as “*living room*” very close to their probabilities as “*dining room*”. In this case, a threshold between top1 and top2 probabilities’ values can be assigned to make the right decision as well as capturing different views while navigating to maximize the correct prediction.

4.5. Conclusion

This chapter presented a new Social Robot Indoor Navigation dataset called SRIN. It consisted of 2D colored images for both rooms and doorways, i.e. SRIN-Rooms & SRIN-Doorways, respectively. SRIN is a useful dataset for medium-sized social robots

in indoor environments, specifically houses. SRIN was validated through training a CNN-based model using SRIN-Rooms dataset, and then tested on the Nao's images for real-time experiments validation. The novelty of this work was illustrated when the validation accuracy of CNN-SRIN for room classification reached to 97.3% in a relatively short time. This was a huge improvement compared to training the same architecture with Places dataset, shown in the previous chapter, that reached 93.29% of validation accuracy after a long time. In addition, the significance of this work was also shown through the comparison of the performance of two models on real-time experiments for Nao. It showed a big improvement in predicting bedrooms and slightly different performance of other classes in Top1. However, it reached 100% in the Top2 of the correct predictions for four classes out of five. We believe that with increasing the SRIN dataset in the future, the prediction of top-1 for real-time experiments on Nao shall be improved as well.

In this and the previous chapter, we proposed two different solutions for social robots to determine their own whereabouts inside a house based on an ability to classify each room. In the next chapter, we will focus on addressing another crucial component for indoor navigation. We will propose a visual-based system for addressing the doorway detection and direction via only a monocular camera.

Chapter 5.

Doorway Detection and Direction (3Ds) System for Social Robots via a Monocular Camera*

5.1. Introduction

Navigating in indoor environments inevitably requires detection and crossing doors that are regarded as integral parts of any indoor setting, particularly in human habitats (homes). Whereas this task does not require much effort for humans and their pets, it is a challenge for social and other autonomous robots. As such, it is desired that social robots have the same skill and are able to move around a house seamlessly and know their own whereabouts based on an ability to classify each room and its functionality, as reported in Chapter 3 and Chapter 4, respectively. Indoor navigation is inherently multifaceted and includes several tasks including but not limited to localization, mapping, SLAM, path planning, object, and scene recognition. However, the capacity to detect doors and their orientation are critical in any navigation system and are the main subject of this chapter; though the related problem of passing through a door is not within the scope of this study in this chapter. This research question has attracted attention by many researchers on robotics and as we shall discuss in section 5.2, the detection and navigation through a doorway are mostly addressed via sensor fusion techniques, deployment of rather expensive built-in sensor(s) on-board the robot, or augmenting the environment by appropriate and dedicated sensors or Quick Response (QR) Codes.

The motivation for this study in this chapter is the following question: can this problem also be solved practically via a monocular camera? Therefore, the objective of the study in this chapter is to design a system just for detecting and directing a social robot towards a doorway using only a monocular camera that captures only a 2D image. The proposed system is one of the components of an end-to-end navigation strategy inspired by Behavioristic Robotics, in particular the ubiquitous Brook's Subsumption architecture [11] for social robots with limited sensors. This methodology is based on the

* This chapter is mainly reproduced from paper 3 in section 1.4 (page 8).

Sense-Perception-Act theme that is essentially a discrete decision-making process as opposed to methods generally categorized under iterative processes. The two methodologies are fundamentally different and can be viewed as alternative approaches. Depending on specific applications, one or the other is preferable. We argue that for indoor navigation, which is generally regarded as partially known structured environments, the former approach has certain operational advantages including comparable computational cost and robustness. In addition, it is argued that the behavioristic approaches present a balance between the accuracy versus functionality. The complete navigation system will be discussed in the next chapter, whereas the focus here is to address the subtask of detecting a doorway direction within the context of indoor navigation.

The rest of the chapter is organized as follows: in section 5.2, we present related studies for door detection and navigation through it. Then, the proposed system with some details of each module will be discussed in section 5.3. We will then include and discuss some experiments in section 5.4. We conclude the chapter with tentative conclusions and outline the contributions.

5.2. Related Research

A door is a significant obstacle hindering smooth indoor robotics navigation. Consequently, a social robot can move around rooms only if it is capable of detecting and passing through the door safely. There are several approaches that have addressed the problem of doorway detection. Solutions based on probabilistic methods were reported in [187], [188]. In [187], the authors focused on the mapping problem by employing the Expectation-Maximization (EM) algorithm to segment typical corridor environments into doors and walls using camera and laser sensors mounted on a pioneer robot. They assumed that all doors in the corridor had the same shape and color, which were the main extracted features from the vision system. The main task of the laser was to detect dynamic objects in the corridor (doors being open or closed). The authors in [188] extracted features from camera and sonars and applied a graphical Bayesian network to differentiate doors from walls. Both papers focused on typical corridor environments. The problem was also addressed in [189] by designing an image-based geometric model. The model detected doors by connecting corners and edges, then differentiating them from shelves or other similar shape objects by extracting the

concave and convex information. Although it was not explicitly mentioned that this study considered hallway environments, but one can infer that the door was a concave object with respect to the wall in the hallway, or outside the rooms. Alternative methods based on 3D point cloud data to detect and differentiate doors from walls using RANSAC (Random Sample Consensus) estimator were reported in [190], [191]. Sensor fusion is another approach to address doorway detection as reported in [192]. The paper suggested a sequential process that fused laser data with images to detect doors in corridor environments. It started with applying the X-histogram method on the laser scan data to detect walls. Then, it combined the wall detection laser data with an image to identify the region of interest (ROI). Subsequently, the ROI was combined with the integral image to calculate the vertical lines in the walls using Haar-like features to detect the doors in the corridor environments. Machine learning (ML) has also been applied to solve this problem. A conventional machine learning method known as Adaboost was employed in [193]. The authors implemented that algorithm on a Pioneer2DX robot to extract weak features from the camera, i.e. color, knob, frame, gap and texture, and from the laser, i.e. door width and concavity in order to use them in a strong classifier. The key objective of this method was to make sure that the extraction of features was accurate. Another promising method in machine learning is Convolutional Neural Networks (CNN) for images and Region CNN (R-CNN) for object detection that was proposed in [194], [195], respectively. The first paper [194] used 20 door images with the same features. By applying different image processing, the images were increased up to 20500 images, where the positive samples were 2500 and the negative samples were 18000. Note that there is a big difference between the two samples. They applied a simple CNN with 3 stages structure to learn door detection in a typical environment. The validation accuracy reached up to 73.1% for the 856 positive samples. The latter paper [195] addressed a different problem of cabin doors detection. It was completed via applying R-CNN on 11 videos. The algorithm started with the prediction of an area of the door, then it applied a mathematical morphology approach to detect if that area was a door or not by extracting a handle and footplate.

Furthermore, several studies took advantage of detecting the door as an important feature for the navigation process. In [196], the authors proposed a system to address the exploration problem in an indoor environment for a Pioneer3 robot using its stereo camera. First, it detected the door using the image-geometric approach. Then,

both the Dynamic Window Approach (DWA) and A* algorithms were applied to address the obstacle avoidance and path planning problems, respectively. The probabilistic method is among the common approaches for addressing navigation problems by considering the advantage of door detection [197], [198]. The authors in [197] mainly focused on controlling the manipulator of the PR2 mobile robot to open the door as well as to plug itself into a standard socket. However, their related work was in detecting doors using conventional vision methods and moving the base of the PR2 robot by designing a deliberative robotic control system that combined a probabilistic localization, 3D obstacle detection, and path planning with a given 2D occupancy grid map. Whereas the authors in [198] applied probabilistic methods on laser data for door detection to improve the localization and mapping performance in corridor environments. In contrast, the ubiquitous statistical machine learning algorithm of the Gaussian Mixture Model (GMM) was applied to a semi-autonomous wheelchair in the Gazebo simulator [199]. A nonlinear adaptive controller was proposed in [200] to help a big four-wheeled robot to cross the door after applying a sensor-based approach to detect it using a Kinect camera. Similarly, passing the door in a corridor environment for a wheelchair with 3 cameras was the objective of the algorithm reported in [201]. The problem was addressed by applying an image geometric-based method for detecting doors and designing a Lyapunov-based controller based on visual features for following the corridor and passing through the door.

It is important that we also point out to some other studies in computer vision that have broadly addressed the depth estimation problem via a monocular camera, although not particularly employed for doorway detection. The study in [202] described an algebraic representation based on the image geometry and using the vanishing point and line to extract 3D measurements from 2D images. The extracted measurements were the distance between parallel planes from a reference plane (e.g. the ground plane), the area and length ratio of a plane parallel to the reference plane, and the Cartesian location of the camera (x, y, z). Alternatively, the Structure-from-Motion technique (SfM) is a well-known approach to address 3D reconstruction from multiple 2D images as discussed in several studies, such as [203]–[205]. SfM was adopted in these studies was to address the feature detection and matching among the input sequence of images; thus, the camera parameters were recovered. Then, the incremental SfM with the integration of the Multi-View Stereo technique was applied to reconstruct the 3D

information. Other studies such as [206], [207] adopted supervised learning approaches based on datasets of 2D images with corresponding depth maps. The first study [206] used collected images with corresponding laser dataset to train a probabilistic supervised model that depends on the appropriate extraction of local and global features. Also, the authors studied the performance of using the monocular cues for the stereovision system. On the other hand, the latter study [207] used two different RGB-D datasets for training a proposed encoder-decoder architecture. The authors presented the success of their network as compared with other studies in the field of depth estimation from 2D images. From a different perspective, a framework was proposed in [208] that integrated the Adaboost method of machine learning and dynamic optimization to estimate 3D structure from 2D images of an outdoor environment. There are alternative solutions based on image processing techniques for the depth recovery challenge such as using a sharpening filter [209], using defocus cues [210], or computing salient regions and image compressing based on blur cues (focus/defocus) [211].

In contrast to the aforementioned studies, this project focuses on three main objectives. The first goal is to address the doorway detection for indoor environments based on a CNN-like model which provides a better performance and higher accuracy than [194], which adopted the same CNN approach. The main motivation to adopt the CNN approach over other machine learning methods was that it does not require a careful a priori human design. The second goal of this work is to calculate the relative angle direction of the robot with respect to the doorway from a 2D image. The angle direction is an important information for controlling the robot towards the target. Therefore, a global or explicit Cartesian position, as well as distance information towards the doorway are excluded; although they might provide crucial cues for other robotic applications. Also, our study focuses on the discrete decision of the sense-perception-act theme, which is unlike other visual servo techniques, such as [201], that address the navigation problem continuously with the integration of a conventional controller. The third goal is to compute the angle direction from only a still 2D image via a monocular camera, which can be inferred through estimating the depth information. Therefore, we have adopted the model from [207], for estimating the depth values from a 2D still image with no need for additional image preprocessing, over other computer vision methods, such as machine learning methods that depend on careful engineered designs, and SfM

that needs a sequence of 2D images. Besides, the work in [207] is considered as one of the state-of-the-art studies in the field of estimating depth information from 2D images as the author presented the success of their network compared with other studies in the research area. Accordingly, we propose a Doorway Detection and Direction 3Ds-system for social robots with limited sensors (monocular camera). This system can detect an open door and then can direct the robot toward the doorway based only on a 2D image that is captured by a monocular camera. The system combines several modules with different approaches: learning-based, pixel-based and triangular-based methods.

5.3. Proposed System and Methodology

The key concept of the proposed system is based on the Sense-Perception-Action architecture, see Figure 5-1. Accordingly, the proposed 3Ds-system for detecting a doorway and directing a social robot towards it is shown in Figure 5-2. It consists of several modules to enable a social robot equipped with only a monocular camera, i.e. Nao robot, to provide an appropriate angle toward the doorway from its current location. The algorithm is initiated by acquiring a 2D image using the top camera of Nao. This image is then passed to the CNN-SRIN Doorway module to classify the image as either an open door or a no-door scene. SRIN is a dataset for indoor settings specifically designed for short robots such as Nao. If the image is classified as an open-door scene, the depth module is triggered to construct a depth map using the captured 2D image. Next, the Pixel-Selection module is applied to the depth map to determine the best pixel that represents the doorway location. Finally, the selected pixel is passed to the Pixel2Angle module that converts the depth value of that pixel into an appropriate angle which will be used to guide the robot towards the door. The Pixel2Angle module is triggered only if there is no obstacle between the robot and the doorway, which can be detected via a vertical correlation in the Pixel-Selection module.

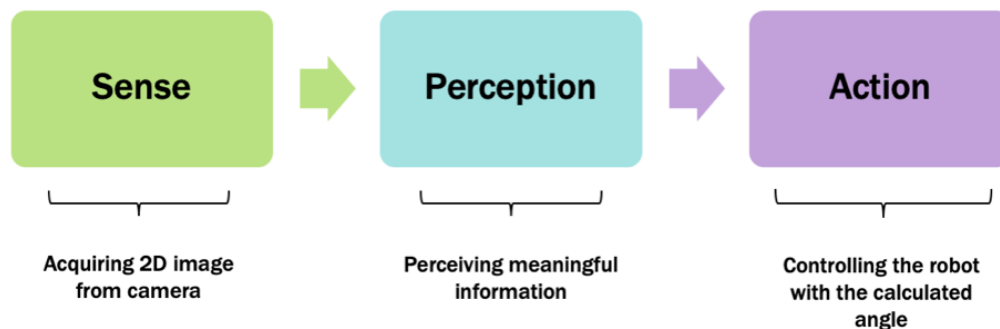


Figure 5-1: A block diagram of Sense-Perception-Action control architecture.

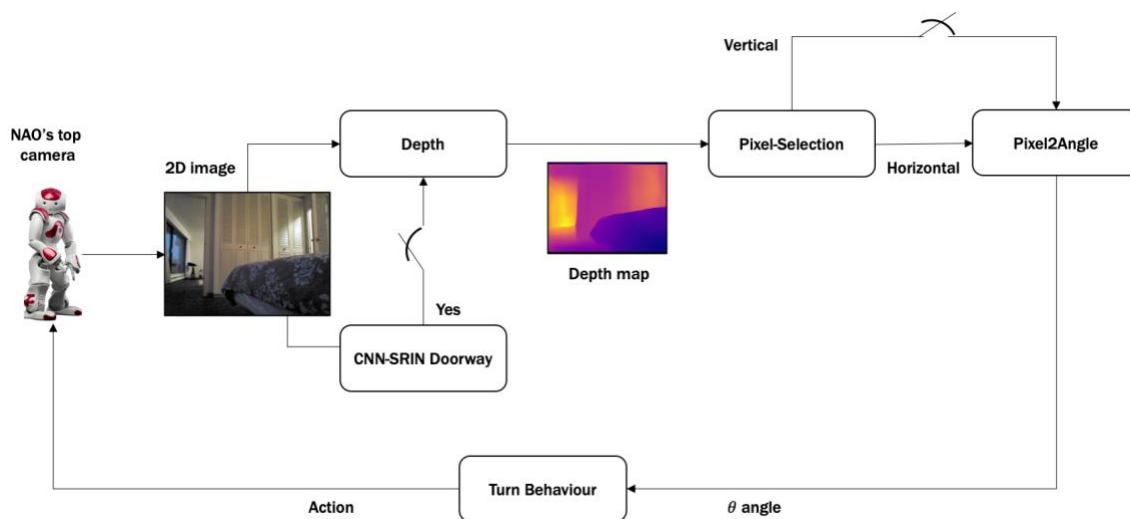


Figure 5-2: Our proposed robotic system for doorway detection and direction (3Ds-system).

The following sub-sections will explain the function of each module in more detail. As the algorithm is meant for social robots, we present these modules for the Nao humanoid robot. The same algorithm can be readily applied to any (social) robot equipped only with a monocular camera.

5.3.1. 2D image from Nao monocular camera

The Nao humanoid robot has two monocular cameras that are mounted vertically on its face. Since there is no overlap between them, the system is not considered as a stereo camera set, i.e. there is no direct depth information or direct way to extract the depth values. For this project, we employ the top camera to extract a 2D image, which is

set up with a size of 640×480 . The specifications of the camera are crucial for achieving the purpose of this project successfully, specifically the horizontal Field of View, $FoV_w = 60.9^\circ$, and the vertical Field of View, $FoV_h = 47.6^\circ$ as shown in Figure 5-3. As our goal is to control the direction of the Nao robot, then the FoV_w will be used in the calculation of the Pixel2Angle module.

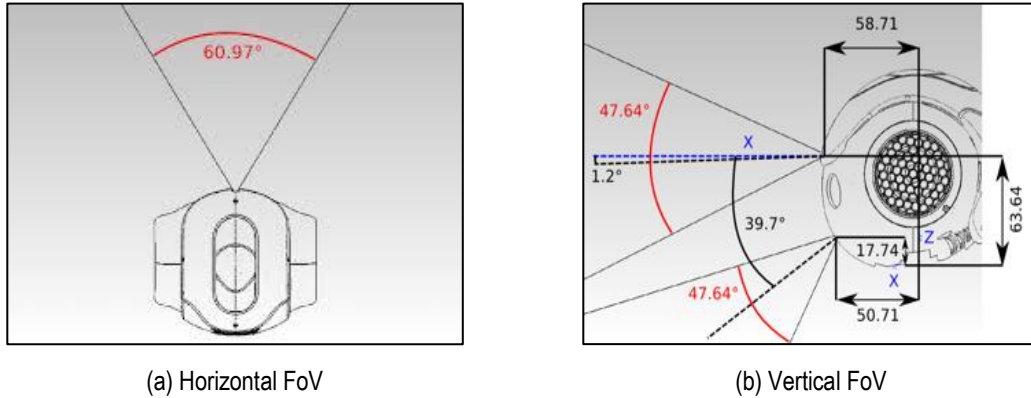


Figure 5-3: Field of View of Nao robot cameras [171].

5.3.2. CNN-SRIN Doorway Module

The aim of this module is to detect whether or not the scene in front of the robot is a door. We achieve this goal by training a CNN model via the transfer learning process as shown in Figure 4-4 from last chapter using our collected SRIN dataset. Thus, we call this model CNN-SRIN throughout the chapter. There are two classes of SRIN dataset for doorway used for training CNN model: no-door and open-door, in which this module will be useful for any indoor robotic visual navigation system. Within the proposed 3Ds system, the following module will be triggered if the robot detects an open-door with CNN-SRIN.

5.3.3. Depth Module

The objective of this module is to estimate a depth map from a 2D image extracted from Nao's monocular camera. Estimating depth information from 2D-colored images is among open research problems in computer vision. We have adopted the trained Depth Dense network from [207], which is considered as state-of-the-art in this area. The Depth Dense network is designed based on the encoder-decoder architecture

as shown in Figure 5-4. The encoder part is a pre-trained CNN architecture, specifically DenseNet 169, which has layers for extracting features through the down-sampling process. The decoder part has layers for constructing the estimated depth information through the up-sampling process. Every layer in the decoder was fed by the output of a specific layer in the encoder, i.e. this concept is referred to as skip connection. The network was trained while keeping the encoder part frozen, i.e. transfer learning process, using two different RGB-D datasets: NYU Depth-v2 [212] and KITTI [213]. Both datasets provide RGB images as inputs, whereas the depth map is the respective output. The authors [207] presented the success of their network compared with other work in the field of estimating depth information from 2D images. For that reason, this trained model has been adopted to test and estimate 2D images from Nao within a robotic application. The Nao 2D image is fed to the Depth Dense Network in size of 640×480 , where the network will estimate the depth information of size 320×240 . All depth map pixels carry a value from 0 to 1, in which the value 1 is the deepest distance.

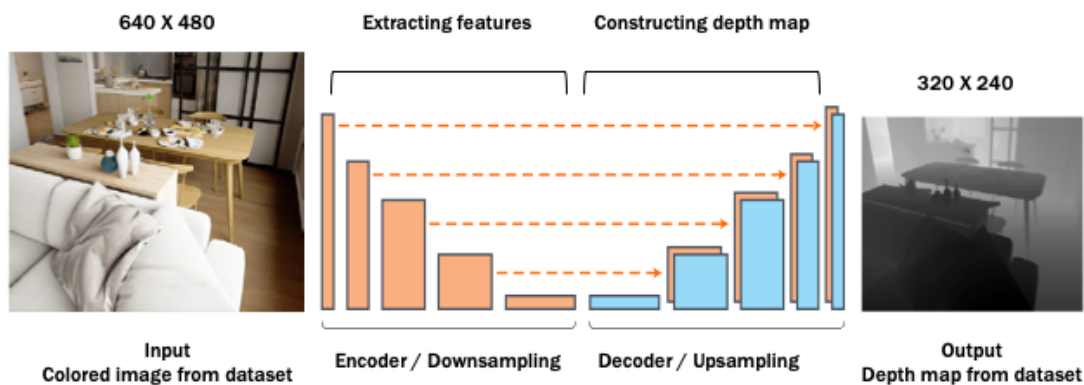


Figure 5-4: Depth-Dense Network [207]. The figure is modified for the explanation purpose.

5.3.4. Pixel-Selection Module

This module is designed with a premise that the pixel with the deepest value is associated with the doorway. Therefore, the simple way to select a pixel related to the doorway is the maximum depth value from the depth map; let us call them Max-Pixel & Max-Depth. However, the Max-Pixel is not the best one for the robot direction as it might be very close to the edge of the door, or it might be close to the top corners of the room as will be shown later in images 1 and 4 of Table 5-3, respectively. For this reason, we

need to find the Best-Pixel for the robot direction based on the horizontal correlation in the lower half of the image. This can be obtained by comparing every two adjacent pixels starting from the Max-Pixel in both directions, i.e. right by incrementing the width and left by decrementing the width. If the difference of the depth values is less than a $threshold = 0.01 \text{ unit}$, then we move to the next pixel that is next to the current pixel. The proposed algorithm keeps comparing every two adjacent pixels from the right and left until the difference of depth is greater than a threshold value from both directions. This implies that the most likely pixel is related to the edges of the door. Then, the Best-Pixel is the mid pixel between the last right and left correlated pixels. This Best-Pixel will be passed on to the Pixel2Angle module.

In many cases, the robot can detect a door while there is an obstacle between the robot and the door. Therefore, we need to find a Trigger-Pixel to make sure that there is no obstacle in the way to the door, and then to trigger the next module. This can be performed by applying the idea of pixel correlation vertically to the depth map through the bottom direction only, i.e. incrementing the height value from the Max-Pixel, with a $threshold = 0.045 \text{ unit}$. If the height of the last correlated bottom pixel is over 200, then this will trigger the next module to find the proper angle. Otherwise, it is implied that there is an obstacle in the way towards the door. In that scenario, there is no need to calculate the angle in the Pixel2Angle module. Figure 5-5 illustrates the concept of pixel correlation and selection from the 2D depth map.

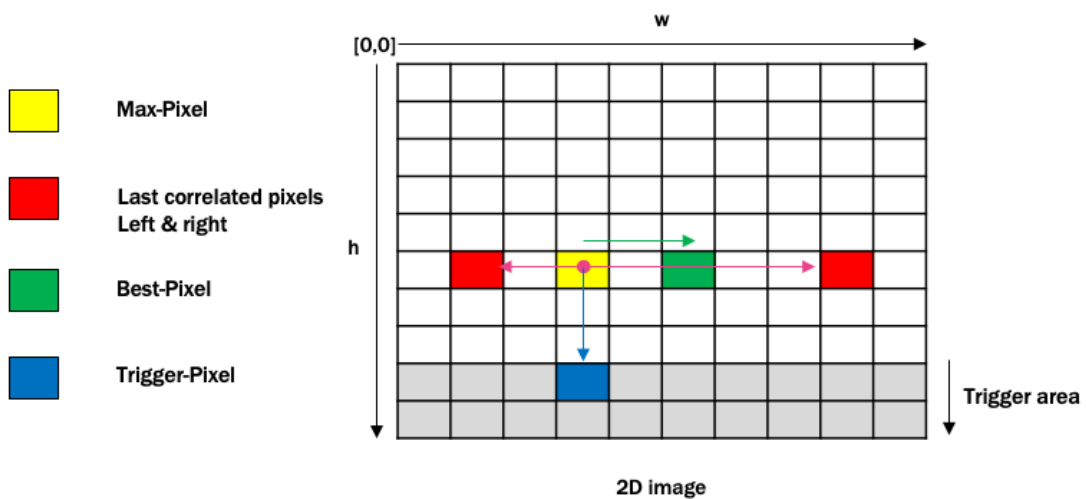
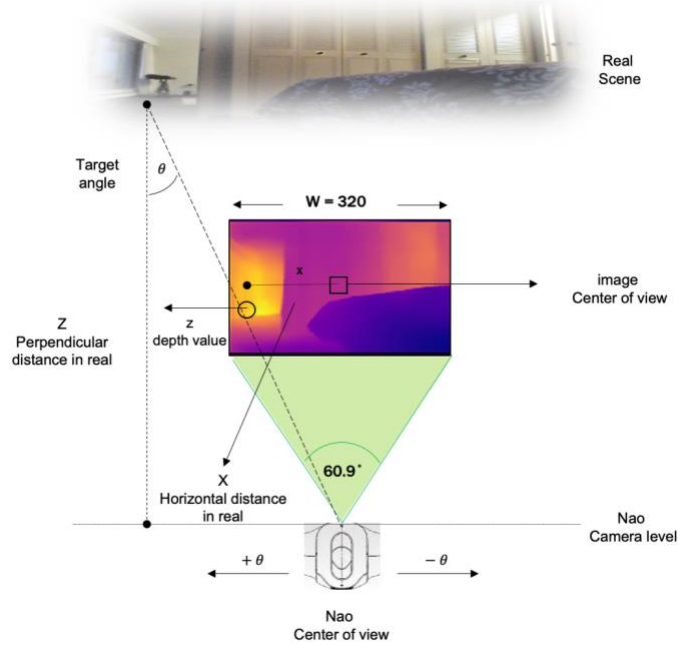


Figure 5-5: Illustration of correlation and selection of the best pixel.

5.3.5. Pixel2Angle Module

After selecting the Best-Pixel toward the doorway and making sure that the door's pixel is located on the trigger area of the depth map, i.e. no obstacle in the way to the door, then we apply the Pixel2Angle module for calculating the proper and approximate angle direction toward the door. It is a simple, but effective triangular algorithm applied to the selected pixel. As our goal is that Nao turns left ($+\theta$) or right ($-\theta$), then the calculation will be focusing on the horizontal pixel values, although the vertical calculation can be processed similarly for other applications. Figure 5-6 presents an idea of how this module works and how the target's angle is calculated. As illustrated in Figure 5-6a, the robot center view is represented as the center pixel in the depth image, the depth value of the selected pixel is the perpendicular distance between the target and the robot location. Therefore, the real horizontal distance X between the robot and the target is represented as the number of pixels from the selected pixel $Pixel_{best}$ and the center pixel $Pixel_{center}$ in the depth map. First, we need to find the horizontal length x between the selected pixel $Pixel_{best}$ and the center pixel $Pixel_{center}$ from the depth map. Then, we need to calculate the angular size of each pixel α_{pixel} in the depth map by dividing the Field of View (FoV) by the size of the depth image. The horizontal Field of View of Nao is $FoV_w = 60.9^\circ$, whereas the width of the depth map from the Depth module is 320 pixels. Thus, each pixel in the depth image has 0.19° angular size. After that, it is easy to calculate the desired angle θ_w between Nao and the target direction toward the door by multiplying the angular size α_{pixel} by the horizontal length x . This angle will be passed to Nao as a negative value if the $Pixel_{best}$ is in the right half of the depth map; otherwise, it is positive. For other applications that deal with distances, if the unit of the depth map is known, e.g. depth in meter, then it worth it to calculate the distance to the target, i.e. the door in our application. First, we calculate the real horizontal distance X by multiplying the Tangent of the desired angle θ_h by the depth value Z . Then, we can find the distance using the Pythagorean equation. Figure 5-6b gives the mathematical algorithm of this module.



(a) Pixel2Angle module illustration

Pixel-Angle Algorithm:

- 1) Find image horizontal length:

$$x = |P_{best} - P_{center}|$$

- 2) Calculate the angular size of each pixel:

$$\alpha_{pixel} = \frac{FoV_w}{w_{image}} = \frac{60.9^\circ}{320} = 0.19^\circ$$

- 3) Find the target angle:

$$\theta_w = x * \alpha_{pixel}$$

$$\theta_w = \begin{cases} -\theta_w, & P_{best} \text{ in the right half} \\ \theta_w, & P_{best} \text{ in the left half} \end{cases}$$

- 4) This step is useful to find the distance to the target if the depth unit is known.
(not used in this work):

$$X = Z * \tan(\theta_w)$$

$$target_{distance} = \sqrt{X^2 + Z^2}$$

(b) Pixel2Angle module calculation

Figure 5-6: Pixel2angle module for Nao robot.

5.4. Experiments and Results

All modules in the proposed system were programmed in Python 2 except the Depth module which was programmed in Python 3 by the author. Therefore, a python module called subprocess was used for managing and combining different modules from

different python virtual environments. The overall system was evaluated using a laptop with a 2GB memory size of the GPU for simulation experiments with pre-captured images by Nao as well as for real-time experiments via connecting real Nao robot with the laptop through Wi-Fi.

The results of all modules of this project are presented into two stages: the doorway detection stage and the angle extraction based on depth and pixel selection. In the first stage, the system detects the doorway via CNN-SRIN model. The second stage presents the results of other modules on some selected images from the first stage. Afterward, we present real experiments with a Nao robot in a new environment in order to validate the overall performance of the 3Ds-system.

5.4.1. Stage 1: CNN-SRIN for Doorway Detection

The design of CNN-SRIN architecture consists of a features extractor via VGG16 and an image classifier via Fully Connected (FC) network using Keras API [167]. In this project, the first stage of transfer learning concept shown in Figure 4-4 was only applied to the CNN-SRIN architecture, for which VGG16 was frozen while FC was trainable. FC began with an average pooling layer, then a layer of 1024 neurons with the Rectified Linear Unit (*ReLU*) activation function. The model was terminated with a logistic layer to predict one of the two classes: no-door vs open-door. In this stage, the learning rate was 0.001 and Adam optimizer ran for 10 epochs. The no-door class consisted of 7062 images, whereas the open-door class included 7432 images. We trained the CNN-SRIN model for doorway detection on the Graham cluster provided by Compute Canada Database [168] for several epochs, 10, 20 and 30, respectively. The validation accuracy reached 95.79% after 36 minutes for the 10 epochs. Whereas it increased up to 97.96% after 1:10 hour of training for 20 epochs, and 97.51% after 1:46 hour of training for 30 epochs. Accordingly, the trained model with 20 epochs was adopted to be tested on new images collected by Nao humanoid robot since the model has the highest validation accuracy within a reasonable period of time on the Graham cluster. We randomly selected twelve related images, i.e. six images for each class, from the Nao's point of view. Table 5-1 shows all images with their predictions. The model successfully predicted five images out of six with the correct class for each category, i.e. a total of 10 correct predictions as shown in Table 5-2. These results validated that this module within the 3Ds-system will be a good trigger for the next module.

Table 5-1: CNN-SRIN doorway prediction results on Nao. Shaded results are the false prediction.



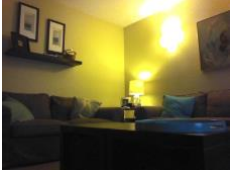

No-door		Open-door	
Nao image	CNN-SRIN Prediction	Nao images	CNN-SRIN Prediction
	No-door		Open-door
	Open-door		No-door
	No-door		Open-door
	No-door		Open-door
	No-door		Open-door
	No-door		Open-door

Table 5-2: Confusion Matrix. (TN: True Negative, TP: True Positive, FP: False Positive, FN: False Negative).

12 images		Prediction	
		No-door	Open-door
Actual	No-door = 6	TN=5	FP=1
	Open-door = 6	FN=1	TP=5
Percentage %		83.3 / 16.7	83.3 / 16.7

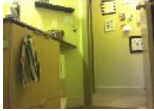




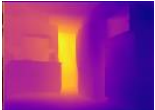




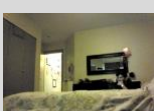
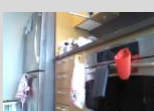

5.4.2. Stage 2: Angle Extraction from 2D Images Based on Depth Map and Pixel Selection

The next modules of the 3Ds-system were tested on several real-time images from the previous module in order to get a practical proof of the successful performance. The expected outputs of the angles are in the range of $[-30^\circ, 30^\circ]$ as the Nao's horizontal $FoV_w = 60.9^\circ$. We selected the six open-door images of Nao robot as well as the image of no-door with the false-positive prediction. All results of these modules are presented in Table 5-3. This table presents every Nao's 2D image with its CNN-SRIN trigger status. If the status is "Yes", then the rest of the other modules' results are presented. The depth module provides a depth colored image, in which the yellowish pixels are considered as far distances to a specific target, whereas the dark pixels represent very close objects. Then, the pixel selection module results are provided as follows: maximum depth value with its pixel, best-selected pixel with its depth value, and the vertical trigger status with its pixel. All depth values are rounded to two decimal points in this table for simplification. The last column of this table shows the calculated angle value from the last module if it is triggered by the previous module, otherwise, it gives a not applicable "n/a" value which means the robot does not receive any signal. The positive angle value means the robot turns left, whereas the negative value is for turning right.

The overall results show the success of the proposed system for detecting and navigating the robot toward a door in the indoor environments. It can be seen in image 1 & 3-5, that the 3Ds-system successfully detects the doorway and estimates proper angle to direct Nao. The interesting results are shown in the shaded results, which need to be discussed further. The system was able to detect an obstacle between the robot and the

doorway when the pixel-based module applied, as shown in image 2. Therefore, it did not send any angular value to Nao. Although the 3Ds-system could not detect a door in image 6, this does not affect the overall performance as there is an obstacle in front of the robot, which will be detected by pixel module and no angular value will be expected to be sent to Nao. The last tested image, i.e. image 7, showed the false positive prediction of the CNN-SRIN trigger module. Since it predicted that there was a door, the other module was triggered and obtained its results. The angular value of image 7 leads the Nao robot to the free space direction, which is considered relatively as a good action within a navigation system that would lead to the doorway. This is certainly not conclusive evidence as it is possible to fail in other cases.

Table 5-3: Real-time experiment results: Depth to angel values for controlling Nao robot.

Nao 2D image	CNN-SRIN Trigger	Depth Map 240X320	Max pixel	Max depth	Best Pixel	Best Depth	Vertical trigger	Angle in degree
	Yes		[185, 194]	0.24	[185, 255]	0.23	True [238,255]	- 18.1
	Yes		[145, 157]	0.46	[145, 201]	0.40	False [185,201]	n/a
	Yes		[120, 130]	0.50	[120, 135]	0.50	True [238,135]	4.8
	Yes		[183, 73]	0.37	[183, 42]	0.32	True [238,42]	22.5
	Yes		[166, 41]	0.52	[166, 39]	0.52	True [238,39]	23.0
	No (false)	-	-	-	-	-	-	-
	Yes (false)		[188, 0]	0.27	[188,19]	0.23	True [238,19]	26.8

5.4.3. Validating the Overall performance of 3Ds-System in Real-Time Experiments with Nao humanoid robot

As this work focuses on the door detection and direction, we evaluated the process by testing the 3Ds-system in real-time experiments with Nao in a new indoor environment. These experiments were carried out at Autonomous Intelligent System Laboratory (AISL) at Simon Fraser University (SFU). For practical purposes, it is important to mention that the Depth module is implemented on python 3 version,

whereas Naoqi API works with python 2 version. Therefore, different modules in the 3Ds-system should be managed and combined via a python module called *subprocess* that includes *Popen* constructor for executing a child program with its suitable python virtual environment in a new process. The goal of these experiments is to show that Nao is able to detect the doorway and direct itself towards the doorway properly with a correct angle value. Simultaneously, it is able to detect an obstacle in the way to the door and prohibit applying the angle direction. We considered three different scenarios for this evaluated experiment: Nao is in front of the doorway from different distances and angles, Nao is not in front of the door, and Nao is in front of the door while an obstacle is in the way to the door, see some examples in Figure 5-7. The process of implementing all modules of the proposed system altogether is shown as a pseudocode in Figure 5-8.



a) Doorway



b) No door



c) Doorway with an obstacle

Figure 5-7: Examples of three different scenarios for evaluating 3Ds-system with Nao in real-time experiments.

Pseudocode

```
Capture a new scene by Nao

# Predicting the doorway
Apply "CNN-SRIN Doorway" Module


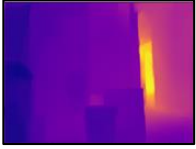


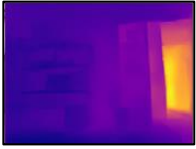




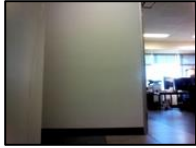
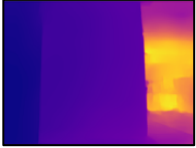

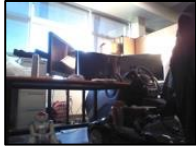
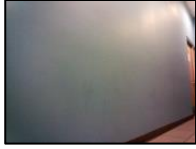

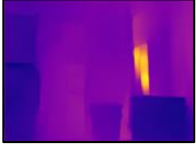
if prediction == "open door":
    # Getting the depth image from 2D image
    Apply the "Depth" Module

    # Extracting the pixel that related to the doorway from depth image
    Apply the "Pixel-Selection" Module
    if the vertical-trigger == False:
        print ("There is no obstacle")
        # Calculating the angel direction towards the doorway from the selected pixel
        Apply the "Pixel2Angle" Module
    else:
        print ("There is an obstacle")
        The "Pixel2Angle" module is prohibited
else:
    print ("Nao cannot see the doorway in this direction")
```

Figure 5-8: Pseudocode of implementing the Doorway Detection and Direction system.

Table 5-4 provides the results of several real-time experiments with the Nao humanoid robot. It shows the validation results of the three aforementioned scenarios. As we can see, there are Nao's perceptions before and after the "Turn" behavior. Nao decides to turn based on the acquired output from each module in the 3Ds-system. These results show the success of our proposed system in practice.

Table 5-4: Results of real-time experiments with Nao at AISL in SFU, BC.

Scenario	Experiment	Input	Modules outputs			Turning action
		Nao perception	Nao decision	Depth perception	Important values	Nao perception after turning
Doorway	1		Open door		Best pixel = [143, 246] Z = 0.54 Vertical trigger: True $\theta = -16.37^\circ$ Turn Right	
	2		Open door		Best pixel = [152, 283] Z = 0.78 Vertical trigger: True $\theta = -23.41^\circ$ Turn Right	
	3		Open door		Best pixel = [182, 36] Z = 0.58 Vertical trigger: True $\theta = 23.60^\circ$ Turn Left	
	4		Open door		Best pixel = [177, 291] Z = 0.64 Vertical trigger: True $\theta = -24.93^\circ$ Turn Right	
No door	5		No door	Prohibiting other modules		
	6		No door	Prohibiting other modules		
Door with an obstacle	7		Open door		Best pixel = [131, 237] Z = 0.41 Vertical trigger: False $\theta = \text{None}$ No Turn	Prohibiting to turn

Discussion

The number of experiments in section 5.4.3 may appear to be inconclusive. We included only 6 experiments for different scenarios as the other attempts within the accessed area were almost similar to what is reported. Besides, we can consider the experiments on Nao's images from section 5.4.2 as an extra validation since the angle outputs are the values that are supposed to be passed to the robot to turn, which is similar to what presented in section 5.4.3. However, implementing further experiments in different environments, such as schools or community centers, in the future are potentially useful validation steps but not within the scope of this chapter. In addition, more images will be useful to be collected in future by Nao, or any similar social robot. We encourage the Nao robot's community to assist and provide more indoor environment's images to improve the validation performance of the doorway detection in section 5.4.1. Table 5-5 presents a comparative evaluation of the proposed algorithm versus related methods outlined in section 5.2. We acknowledge that this comparison is subjective and inferred from the source papers. Nevertheless, the main features of each algorithm including respective computational resources as well as their relative robustness are listed.

Table 5-5: Qualitative comparison of related algorithms.

Objectives	Papers	Main methods	Hardware / Data type	Required information	Computational cost / Robustness	Environments	Output information
Extracting depth from a 2D image	[202]	Image geometry	Simulation work / 2D still image	Vanishing point & line, reference plane	High / High	Static	Dimensions
	[203]–[205]	SfM	Simulation work / 2D sequenced & overlapped images	N/A	High / High	Static	Feature detection & matching for 3D reconstruction
	[206]–[208]	CNN-based Supervised learning	Simulation work / 2D images with associated depth	Dataset	High / High	Dynamic	predicting depth values
Only door detection	[187]	(EM) probabilistic	Camera & Laser / Images & laser polar readings	Pre-map	Medium / Medium	Static / Corridor	Segmentation with assumption of only dynamic door
	[188]	Graphical Bayesian network	Camera & Sonars / Images & sonar polar readings	N/A	Medium / Medium	Static / Corridor	Differentiating doors from walls to build GVG-map
	[189]	Image geometry	Camera / 2D still image	N/A	Medium / Medium	Static/ Corridor	Extracting the concave and convex information
	[191]	RANSAC & ACF detector	Project Tango Tablet / 3D points cloud data	Dataset	High / High	Static	Differentiating doors from walls
	[192]	Sensor fusion	Camera & Laser / sequenced images & Laser polar readings	N/A	High / High	Static	Detecting the wall and then extracting door edges
	[193]	Adaboost supervised learning	Camera & Laser / Images & Laser polar readings	Extracted features & dataset	High / Medium	Static	Accuracy of extracting features of doors
	[194], [195]	CNN-based supervised learning / image processing	Camera / [Images], [Videos]	Dataset	High / Medium	Static closed door	Discrete door direction / extracting certain features
	[196]	Image geometry + DWA & A*	Stereo Camera / Overlapped images	Pre-map	High / Medium	Static	Obstacle avoidance & path planning
Door detection & navigation	[197]	Image processing + Probabilistic method	Stereo Camera & Laser / 3D data points	Pre-map	Medium / Medium	Static	Demonstration of opening doors by manipulator
	[198]	Probabilistic method	Laser / Continuous laser readings	Pre-map	Medium / Medium	Static with assumption of moving doors	Enhancing the map with an explicit Door Representation
	[200], [201]	Sensor-based + conventional controller	Kinect / 3D overlapped images	Extracted features	High / High	Static	Passing through door
	This study	CNN-based + reactive approach	Camera / 2D still images	Dataset & FoV	High / Medium	Dynamic / Any indoor environment	Extracting angle direction toward the door

5.5. Conclusion

In this chapter, we proposed a doorway detection algorithm that will ultimately be used in indoor navigation for social robots with limited sensors. We designed a robotic system referred to as the 3Ds-system, which stands for Doorway Detection and Direction system that was applied and tested on a Nao humanoid robot. The goal of the proposed system was to control the Nao direction towards the doorway based on a 2D image from a monocular camera. The system takes a 2D colored image and provides an angular value in degrees via a combination of several modules. CNN-SRIN doorway module for detecting a doorway was applied on Nao images after getting a validating accuracy of 97.96%. Then, the Depth module, Pixel-Selection module and Pixel2Angle module were applied on the input of 2D images for directing Nao towards the doorway. The practical results were promising, and they demonstrated the success of the proposed system for Nao. The proposed system can be applied to any other similar social robot, by acquiring the proper angle direction toward the door. The overall system has been validated by implementing the 3Ds-system on Nao within a new environment, specifically in AISL at SFU Canada. We suggest that the proposed system is very useful in robotic navigation applications for medium-sized robots with limited sensors, such as a monocular camera, in structured indoor environments.

At this stage, we were able to design several significant modules for the indoor navigation task in houses based on a monocular camera and implemented on a Nao robot. The first two modules were for classifying the room location and doorway detection. The third module is for extracting the angle direction towards the doorway from the robot's perspective. We will adopt these modules for the indoor navigation system. In Chapter 6, we will present a design and a test of a learning-based behavioristic system that consists of several modules for addressing localization and mapping problems for social robots in houses.

Chapter 6.

Sequential Localizing and Mapping: A navigation strategy via Enhanced Subsumption Architecture*

6.1. Introduction

Designing a robotic navigation system with limited sensors that is capable of exploring an indoor environment while the robot concurrently updates its pose and generates/updates its map is still an open research area. As outlined in Chapter 2, this problem has been studied within the context of probabilistic and behavioristic paradigms, respectively. Whereas, the former was studied extensively with varying degrees of success; the latter has been gaining attention in the last decade through advancements of deep learning and availability of algorithms inspired by the natural world. We have adopted the general behavioristic architecture (Sense-Act system) in this research. It could be argued that the problem is more challenging than a purely deliberative design that depends on prior planning and an accurate map. Also, the conventional behavioristic design often acts blindly and does not explicitly include a learning module that is essential for the robot to function in a purposeful manner. To circumvent this shortcoming, we propose an end-to-end behavioristic robotic system that not only guides a social robot to explore an indoor environment without any prior knowledge but also builds a local map and registers its location within that map. We coin the term Sequential Localization and Mapping (SeqLAM) not to be confused with the widely popular probabilistic Simultaneous Localization and Mapping (SLAM) algorithm [21]. The proposed system consists of several layers based on Brook's subsumption architecture [11], in which each layer is responsible for a specific task. The goal of the system is to enable a social robot to navigate an indoor environment safely and purposively, to avoid obstacles, to go to a specific location within the environment, and to build a map for future visits. The purposive capability of the system is manifested through an integrated knowledge-based system that provides the robot's location and builds an abstract map sequentially. In contrast to SLAM that employs an explicit incremental coordinate-based location (pose), a zone-based location is identified in SeqLAM. As for the abstract map,

* This chapter is mainly reproduced from paper 4 in section 1.4 (page 8).

we employ a Directional Semantic Topological Map (DST-Map) that takes advantages of different zones and the spatial relationships between the zones.

We will go through the detailed components of the proposed system in this chapter. However, we need to set the scene and provide context for the design. Here, the term indoor environment is specific and is reserved for human habitats, i.e. an apartment. As it will be shown later, the zones are referred to five classes (bathroom, bedroom, dining room, kitchen, and living room) that are generally present in an apartment. The proposed design can be readily extended to other indoor settings such as a hospital, a library, an office, etc. To demonstrate its performance, the system is implemented and tested virtually on the Nao humanoid robot within a house environment.

The rest of the chapter is organized as follows: in section 6.2, we present related studies of performing exploration and SLAM with a focus on reactive systems integrated with knowledge systems. Then, the proposed system and the design of each module will be explained in section 6.3. We will then include and discuss experiments for individual modules and the overall system in section 6.4. The chapter concludes with a summary and further remarks on the overall system in section 6.5.

6.2. Related Research

Consider a scenario whereby a social robot is required to safely explore and learn in a new environment; in the context of this study, a new apartment. In robotics literature, this task is broadly referred to as robotics exploration which includes wandering in an unknown environment with the purpose of gaining information of that environment (building a map) using mainly exteroceptive sensors. This problem has been addressed by different approaches, including but not limited to geometric or frontier-based methods (*maintaining boundaries*) [214], [215], information-theoretic or probabilistic methods (*minimizing uncertainty*) [216]–[218], and data-driven or learning-based methods (*predicting a map via trained network on a dataset of partial maps*) [219], [220]. Simultaneous Localization and Mapping (SLAM) is considered as the main process to accomplish this task. SLAM is defined as a robotics navigation process through which a robot builds a map of an unknown environment while simultaneously estimates its pose within the created map. SLAM is essentially a probability-based

technique as it deals with an inherently uncertain and noisy environment. Several probabilistic techniques could be employed in a SLAM algorithm such as Kalman Filter (KF), Particle Filter (PF) or Expectation-Maximization (EM) [221], [222].

In parallel, bio-inspired SLAM approaches have also been studied. Such algorithms are distinguished by designs that are motivated by nature to address the robotic navigation problems via developing and validating a biological design, e.g. RatSLAM and BatSLAM. The RatSLAM [223], [224] is a visual-based structure that was inspired by the connection of different types of cells in the rat's hippocampus. The structure is a fixed-weights network for pose cells. The BatSLAM [225], on the other hand, has basically the same structure as the RatSLAM, however, it is based on biomimetic sonars instead of a monocular camera.

Designing a high-level control architecture as an end-to-end system is an important solution for robotic navigation tasks, such as exploration and SLAM. There are several types of robotic architectures that were discussed in Chapter 2. Since a map does not generally exist a priori, behavior-based architectures, i.e. reactive method, can be considered as an alternative solution. One of the key behavior-based architectures is the ubiquitous *subsumption* architecture that was proposed by Brook [11]. The main characteristic of this methodology is to eliminate the plan function from the navigation system while the system decouples the sense and the act functions in the form of distinct behaviors. The subsumption architecture falls under the behavioristic psychology that generally claims a behavior is triggered by the environment as opposed to cognitive psychology that argues mental representations play a causal role in behavior [226]. As such, the original architecture did not explicitly learn from experience. We argue that while adhering to the overall architecture, we can enhance it by embedding learning and incorporating learned knowledge in decision making while navigating. Accordingly, in this chapter, we focus on studying these issues by developing an integrated indoor navigation system via an enhanced subsumption architecture. This perspective has been studied by other researchers, albeit quite different from the design herein. One of the early studies was by Arkin [227] who presented a hybrid architecture that combined two independent levels of planning and action, which were based on the potential field

* It is different from the low-level control systems that make sure the motors for moving are stable or not oscillated [284].

method [228]. He also discussed the definition and the importance of maintaining knowledge within a robotic system in his celebrated textbook “*Behavior-Based Robotics*” [29]. Similarly, the same concept was adopted in several studies with different methods. In [229], a hybrid deliberative-reactive architecture was proposed in which behaviors of the reactive system were designed based on fuzzy logic, while the path planning was addressed based on a prior given map of a static environment. In [230], the authors presented a hybrid navigation system (deliberative and reactive) with incomplete knowledge, i.e. known positions of some static obstacles. The deliberative part was designed based on a binary grid map with the A* algorithm to generate a global path. While the reactive part was designed based on the DH-bug algorithm. It was tested via simulation studies on a Pioneer robot with laser and sonar sensors. Furthermore, the studies in [231], [232] suggested a cognitive method for planning level and a learner method for the reactive level. The authors used a minefield simulator to evaluate the performance of the BDI-FALCON hybrid system for an autonomous vehicle with five sonar sensors. The FALCON is the low-level reinforcement learner, while the BDI (Belief-Desire-Intention) is the high-level planner using prior data. The FALCON system took the action when there was no available plan. Once the plan was created, then the FALCON system was suppressed, and the action was executed by the BDI system. These aforementioned studies demonstrated designs based on a deliberative system that generally required an accurate map which they could use it independently with the behavioristic domain.

Alternatively, Mataric [233] designed an architecture that integrated a map representation with a reactive system, which was tested on a mobile robot equipped with a ring of sonars and a compass within an office environment. The main three levels of the architecture were: subsumption for navigation, wall detection (left, right corridor), and map building (topological) that consisted of nodes with four attributes, including metric information. In [234], Case-Based Reasoning (CBR) was suggested to address the knowledge for a reactive system within a static environment. The concept of CBR is to solve a current problem by retrieving past experiences. The created cases, i.e. pairs of sense-act, in the study consisted of sonars readings, robot direction, and goal direction, while the output was the heading direction. Conversely, the study in [235] focused only on the learning ability within a reactive system. It suggested a combination of two independent systems: a reactive system based on potential field method, and a learning

system based on Reinforcement Learning (RL). The RL component was integrated to coordinate layers in the reactive system that enhanced the robot's movement toward the goal within an unknown non-convex environment. Also, the reactive navigation was addressed in simulation trials [236] by designing two simple behaviors (avoid obstacle and go to goal) within actor-critic architecture for a wheeled robot in a static environment. The reactive system was combined with a trajectory generator and a tracking control system in a hierarchical theme. In addition, planning a trajectory for reactive navigation was solved in [237] based on the law of electromagnetism that leads the arm robot to a desired predefined position while avoiding unknown obstacles. From these studies, the static environments are the main assumption of the reactive systems by combining different types of knowledge/learning or focusing on the trajectory problem based on a partially known environment.

Within the probabilistic approaches, studies that integrated SLAM with various robotic control systems have been also reported. Visual SLAM was used for a deliberative control system in [238]. The author presented a theoretical control architecture for outdoor navigation using only a single camera. The proposed system starts with two visual modules: structure from motion and visual SLAM. The first module took a sequence of 2D images of the same scene from different viewpoints to get depth information, i.e. reconstruct a 3D map. The depth information was passed to "Avoid Obstacle" behavior. The other module performed SLAM via images followed by a path planner that provided waypoints to be used for "Go to Goal" behavior. The outputs of these two behaviors were fused as a final command control. In contrast, the author [239] combined the probabilistic SLAM module with behavior-based motion module within a control system to address the exploration problem for an aerial robot. The SLAM module provided the estimated robot position, whereas the motion module provided the desired position. Both values were passed to a space-state model (low-level) controller to minimize the error. In addition, the authors in [240] aimed to improve the performance of a reactive system by integrating probabilistic SLAM to address a biohazard search mission in an unknown environment. The SLAM algorithm was a probabilistic approach called GMapping that built a metric grid map based on particle filter and localized the robot position within the map. Whereas, the reactive system consisted of a collection of behaviors that were connected based on the form of a finite state machine (FSM) or a finite state automaton (FSA). Behaviors in the reactive system were triggered by the

spatial memory of SLAM instead of the stimulus inputs. All their experiments were executed on a Pioneer 3 wheeled robot with laser for performing SLAM and camera for detecting the target within a limited and static environment. As we noticed from these studies, probabilistic SLAM for a static environment, which is mainly for uncertainty issue, was added to the system. However, the uncertainty in position and mapping, and the static environment are not viable assumptions in real settings.

In contrast to the aforementioned studies, we focus on designing a robotic exploration system that integrates learning and knowledge capability to a reactive system for social robots with limited sensors. Thus, the assumption of static environments is relaxed. Besides, the knowledge system will be designed for addressing localization and mapping in an abstract manner. Accordingly, we design a behavioristic system based on subsumption architecture that controls a social robot. Throughout the thesis, we have focused on robots with limited sensors, particularly those equipped with a monocular camera. We design a knowledge system that builds a Directional Semantic Topological Map (DST-Map) incrementally, which is accessed by all layers in the subsumption system. Therefore, behaviors in layers can be triggered based on the direct stimulus from sensors as well as the gained information in the DST-Map. Thus, the proposed system will build a map and sequentially localize the robot within the abstracted created map during the exploration process. We refer to it as a sequential localization and mapping (SeqLAM) that does not localize incrementally but identifies itself in a predefined zone. In addition, an appropriate reinforcement learning module is designed for adaptive behavior while exploring using only two sonars. All designed modules and the overall system are implemented and tested on the Nao humanoid robot within a house environment using Webots simulator [14].

With the proposed behavioristic system, associated issues to the exploration and SLAM with classical techniques will be addressed or improved. For example, the semantic information within the DST-Map helps SLAM via recognizing the scene and classifying the room's type that gives a meaningful localization and mapping. Consequently, other robotic applications will be improved because of the meaningful task such as the interaction between humans and robots or switching the planning function from a path issue to a task issue, e.g. robot needs to go to the bedroom. Furthermore, The DST-Map implicitly addresses the associated issues to SLAM: high dimensionality and data association [221], [222]. The former issue is addressed within

our proposed system as the nodes in the DST-Map represents the high-level locations instead of objects or free space locations. Whereas, the latter issue is addressed through matching the connection between nodes within the created DST-Map. If there are multiple connected rooms, e.g. two different bedrooms are connected to two different bathrooms, then it is flexible to add a new module to the system for image matching. In contrast to the most SLAM studies that focused on addressing localization and mapping while controlling the robot was executed manually, this work addressing the control part via a collection of behaviors that is useful during any interaction between humans and robots.

6.3. Methodology and Proposed System

The proposed system is a hierarchical design inspired by the behavioristic paradigm of subsumption architecture augmented with a knowledge system. The behavioristic system is essentially multi-layered whereby each layer is dedicated and responsible for a specific task, see Figure 6-1. In principle, the first layer (exploration task) is continuously active during the entire navigation task. The second layer (purposive task) takes the control and subsumes the exploration function. Similarly, whenever a higher layer is triggered then the lower layers will be subsumed as explained in Chapter 2, section 2.1.2. However, the decision of the control layer does not only rely upon on the stimulus functions but also on the information from the knowledge system. The learning-based knowledge system is responsible for building a Directional Semantic Topological Map (DST-Map) that depends on the zone-based location and the related direction via a monocular camera. Most layers in the behavioristic system have bidirectional access to the knowledge system.

The detailed design with all layers, perception and action interconnected modules is shown in Figure 6-1. The first layer is always activated for exploring the environment and finding the purpose of the navigation task. It has only action modules with no perception modules: “Turn” and “Move Straight” behaviors. This implies that these action modules are not functions of any sensor’s data, therefore, this layer is always activated unless the higher layers subsume its control. The perception and action modules in the second layer are designed to control the robot to move towards the sub-goal, for instance the doorway, in a safe manner (purposive task). Therefore, this layer has two perception modules: “*Obstacle Detection*” module that is stimulated by sonar

sensor, and “*Direction Detection*” module that is triggered by the knowledge system. Whereas, the action modules of this layer that control the robot are either “*Go Toward Doorway*”, “*Avoid Obstacle*” or the weighted summation of both behaviors based on the perception modules outputs. Next, the achievement task in the third layer is responsible to make sure that the robot reaches the final goal assigned by a companion. Thus, the “*Command Detection*” module can be designed using the speaker to detect the companion’s command, such as “*Come to the living room*”, as well as comparing the commands with the current information in the knowledge system. If the robot reached the goal, then the navigation’s process will be ended by the action “*Sitting down*”. However, if the robot’s battery level drops to a certain level in any time during the navigation process, then the protective task in the fourth layer will be triggered and subsumes all other layers. Thus, the “*Charging*” action will take the control, which is basically in this work considered as a robot’s request to be charged. It can be more complicated module by designing an autonomous “*Charging*” module with more global information of the socket’s location or by local information of the socket’s detection in every room, which are out of the scope.

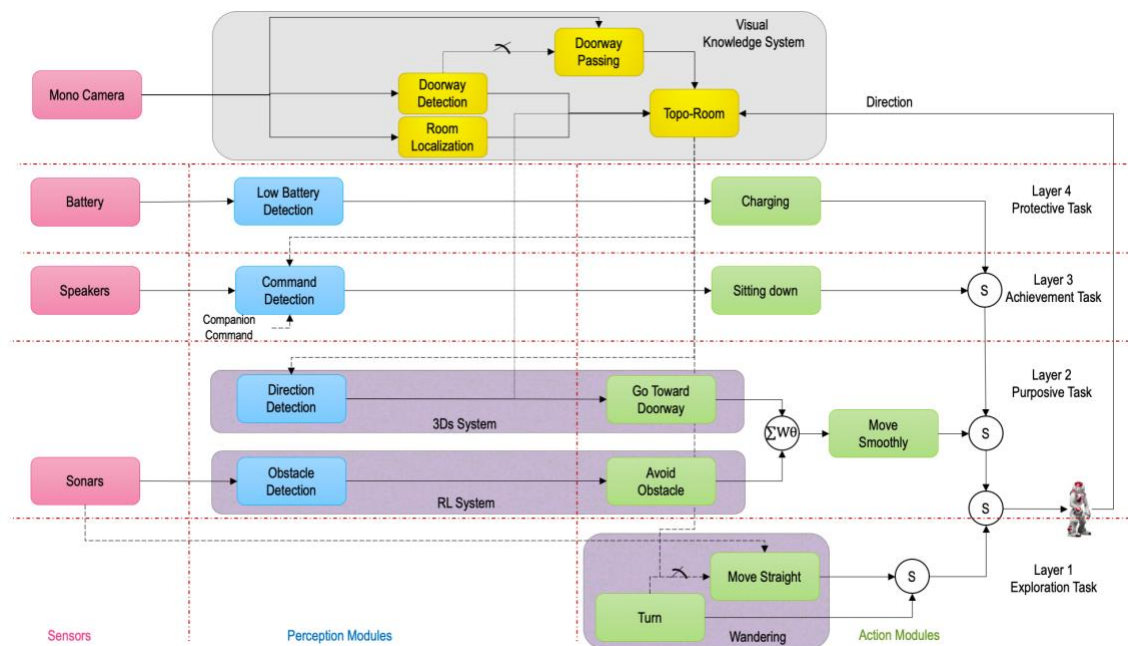


Figure 6-1: Learning-based behavioristic system for homes.

Let us go through the list of assumptions made in developing this system. First, we have not included corridors within the five type of zones (rooms) in the CNN model with labelled images (see Chapter 3). Second, we assume the practical predictions of

classifying the room (see Chapter 4) and detecting the doorway (see Chapter 5) are always true positive within the navigation task. Third, we have assumed that the shape of all rooms is rectangular, and the connection between rooms is through one of the four directions (*east*: 0° , *west*: 180° , *north*: 90° , *south*: -90°). Fourth, as the “command detection” cannot be tested with the simulator, the goal will be assigned by the user. We believe that future work in this area is needed to design a system that relaxes these assumptions. In the following sub-sections, we explain the design of each module in more details.

6.3.1. Subsumption-Based System

The objective of this part is to design a collection of behaviors and interconnect them properly within four main layers as follow:

Layer 1 - Exploration Task

Turn Module:

This module consists three predefined turning angles $\{90^\circ, 180^\circ, -90^\circ\}$ in an order that are related to all other possible directions in a room from any current direction. We assume all rooms are broadly rectangular shape. The robot is supposed to detect the doorway in one of these directions, and the associated direction with the doorway will be passed to the knowledge system. The reason of this particular order is to minimize the number of turning. Let us assume that the robot enters in a new room, which implies a doorway is behind the robot, as illustrated in Figure 6-2. The first direction that the robot tries to detect another doorway is the front, which is considered as 0° angle. If there is no doorway, then the robot selects the option of 90° (turn left). If there is still no doorway, then the robot turns to the other direction by applying 180° from the last direction. By now, the robot detects all three directions of this new room with only two moves. The last angle -90° (turning right) will be applied when there is no other doorway in the room and the robot is supposed to turn back to the previous doorway. The global direction will be passed to the knowledge system for the direction between nodes in the DST-Map.

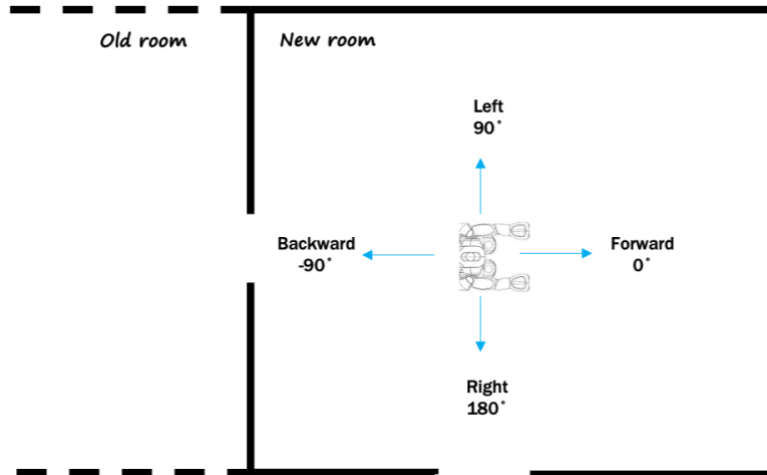


Figure 6-2: Illustration of robot's direction while exploring.

Move-Straight Module:

This module aims to change the scene in front of the robot by changing the position. It is triggered by either the “Turn” module in the same layer or the “Topo-Room” module in the knowledge system. “Turn” module triggers this module when the robot examines all directions but cannot detect a doorway. This is most likely due to the doorway being out of the robot’s view from its current position. Then, this module controls the robot to move straight through the best free space of the four directions based on the sonar values in order to change the view and improve the chance of detecting the doorway from a new spot. Therefore, the sonar values will be always passed to this module. The “Topo-Room” module triggers this module when the knowledge system gives a positive sign for passing the doorway, then it moves straight to make sure it is completely outside the previous room.

Layer 2 - Purposive Task

Obstacle Detection & Avoid Obstacles Modules: (RL System Based on Sonars and Cautious Actions)

The key feature of this module is to design an adaptive behavior with a learning capability. The ultimate goal is to design an appropriate RL model (*states, actions and rewards*) based on limited sensors on Nao, only two sonars. Each sonar provides a distance to an obstacle. We classify any distance into three classes {*very close distance, close distance, far distance*}. Therefore, we can get nine different states as shown in Figure 6-3. As we can see that the best state is [2,2] where the robot has a good

distance from obstacles from the front, whereas the worst is [0,0] where the robot is very close to obstacles from both sonars.

$$S_l \& S_r = \begin{cases} 2, & 1.5\text{m} < \text{far distance} \leq 2\text{m} \\ 1, & 0.8\text{m} < \text{close distance} \leq 1.5\text{m} \\ 0, & \text{very close distance} \leq 0.8\text{m} \end{cases}$$

States	S_l	S_r
1	0	0
2	0	1
3	0	2
4	1	0
5	1	1
6	1	2
7	2	0
8	2	1
9	2	2

(a) Three classes of sonar distances

(b) RL-OAB states

Figure 6-3: State's design of RL for an obstacle avoidance module.

The reward function is designed based on the above defined states, see Figure 6-4. We divide the reward system into state's rewards R_s and transition rewards R_t . R_s provide a positive or a negative reward based on the new state after taking a specific action. Whereas, R_t is a measure of how good or bad the transition is; thus, it is calculated based on the difference of the old and new states. For example, if the robot was in state [0,1] and it moves to state [0,2], then the $R_s = -1$ in both states. However, since the transition is good, the robot gains positive reward for that transition $R_t = \text{sum}([0,2]) - \text{sum}([0,1]) = +1$. Then, the total reward will be the summation of the R_s and R_t . Every episode is terminated either when the robot is able to move more than 500 seconds, or when it falls down which is associated with a final negative reward, i.e. $r = -5$. The time of 500 seconds was selected based on several experiments, and it was found that it is a suitable time for an episode.

$$R_s = \begin{cases} -2, & \text{if state} = [0,0] \\ -1, & \text{if state has } \{0\} \\ 0, & \text{if state} = [1,1] \\ 1, & \text{if state} = [2,1] \text{ or } [1,2] \\ 2, & \text{if state} = [2,2] \end{cases}$$

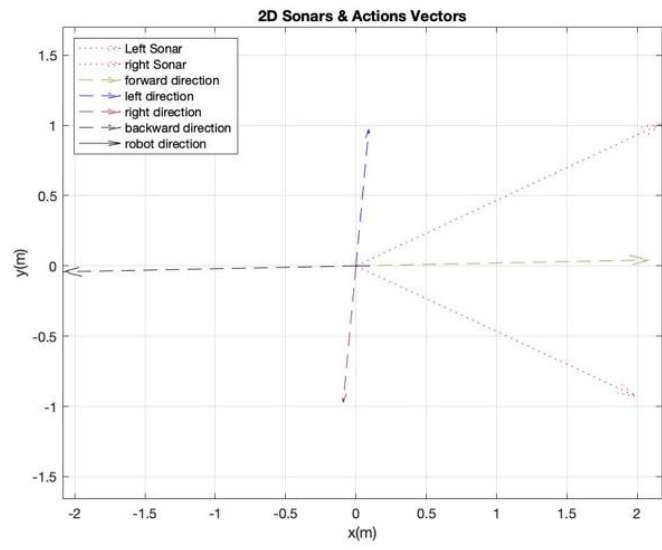
(a) States rewards for RL-OAB.

$$R_t = \text{sum}(\text{state}_{\text{new}}) - \text{sum}(\text{state}_{\text{old}})$$

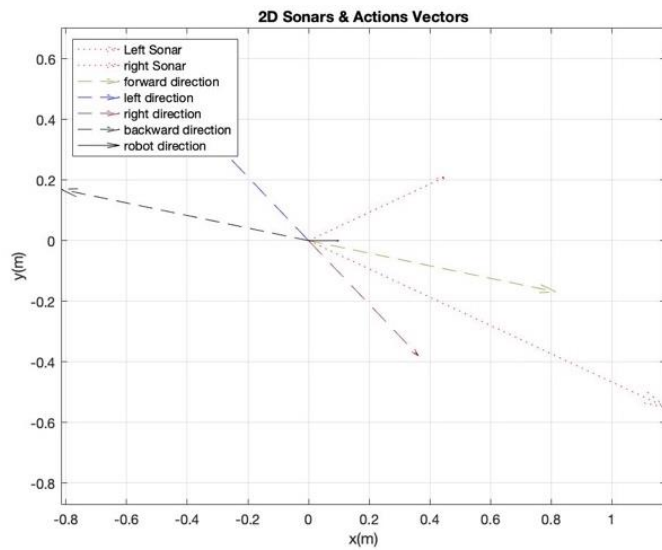
(b) Transition rewards for RL-OAB.

Figure 6-4: Reward's design of RL for an obstacle avoidance module.

The last important part of the RL system is the action function. The main four actions in the obstacle avoidance behavior are {*Go Forward=GF*, *Turn Left=TL*, *Turn Right=TR*, *Turn Back=TB*}. In order to avoid assigning predefined angles and distances for each action, we design an action function that takes in to account the two sonars values to determine the distance and angle of each possible action. In other word, the exact values of distance and angle vary in every step for each action. This is achieved by calculating four different direction vectors, in which each one belongs to a specific action. Figure 6-5 shows different examples.



(a) Far-Far State



(b) Very Close-Close State

Figure 6-5: Action's design of RL for an obstacle avoidance module.

In order to avoid any dangerous action during the RL process, we suggest *cautious actions* by weighing each direction vector based on its current state. Let us assume that the robot is in the [2,2] state, which means the robot is far from obstacles from both sonars as shown in the example of Figure 6-5 (a). If the robot selected the TB action, which has a large magnitude of distance, then it is better this value has a low weight with this state as the robot does not know what obstacles are in the back. Therefore, each direction vector can be weighted by one of the three values $W = \{w_1 = 0.8, w_2 = 0.5, w_3 = 0.2\}$. The cautious action process and the associated weights are shown in Figure 6-6, while the effect of weights on the action vectors is shown in Figure 6-7.

Direction Weights	GF	TL	TR	TB
S1=[0 0]	0.2	0.2	0.2	0.8
S2=[0 1]	0.2	0.2	0.5	0.2
S3=[0 2]	0.2	0.2	0.8	0.2
S4=[1 0]	0.2	0.5	0.2	0.2
S5=[1 1]	0.5	0.5	0.5	0.2
S6=[1 2]	0.5	0.5	0.8	0.2
S7=[2 0]	0.2	0.8	0.2	0.2
S8=[2 1]	0.5	0.8	0.5	0.2
S9=[2 2]	0.8	0.5	0.5	0.2

(a) Associated weights with states

Cautious Action Process:

- 1) Get the left and right sonars reading in a polar coordinate:

$$P_l = [d_l; 25^\circ]$$

$$P_r = [d_r; -25^\circ]$$

- 2) Transform Points to a cartesian coordinate:

$$P_l = \begin{bmatrix} x_l \\ y_l \end{bmatrix} = \begin{bmatrix} d_l \cos 25^\circ \\ d_l \sin 25^\circ \end{bmatrix}$$

$$P_r = \begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} d_r \cos -25^\circ \\ d_r \sin -25^\circ \end{bmatrix}$$

- 3) Find the four direction vectors (Go Forward u_f , Go Backward: u_b , Turn Left: u_l , Turn Right: u_r)

$$u_f = \frac{P_l + P_r}{2}; u_b = -u_f$$

$$u_l = \frac{P_l - P_r}{2}; u_r = -u_l$$

- 4) Multiply each vector to the appropriate weight based on their current state.

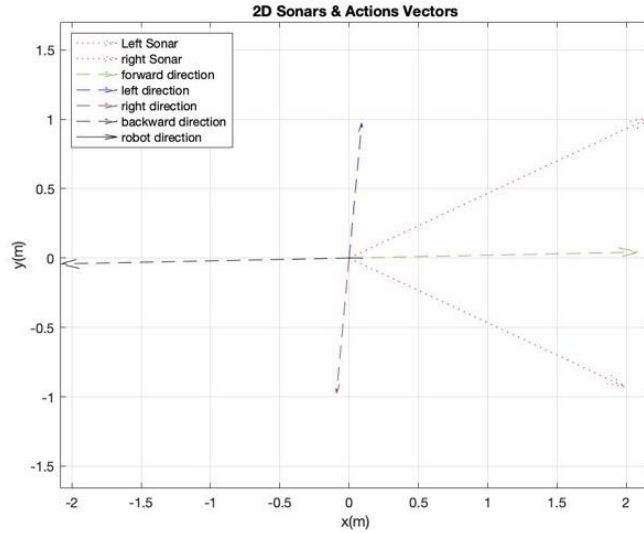
- 5) Transform Points to a polar coordinate:

$$u_{cartesian} = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow u_{polar} = \begin{bmatrix} \rho \\ \theta \end{bmatrix} = \begin{bmatrix} \sqrt{x^2 + y^2} \\ \text{atan}\left(\frac{y}{x}\right) \end{bmatrix}$$

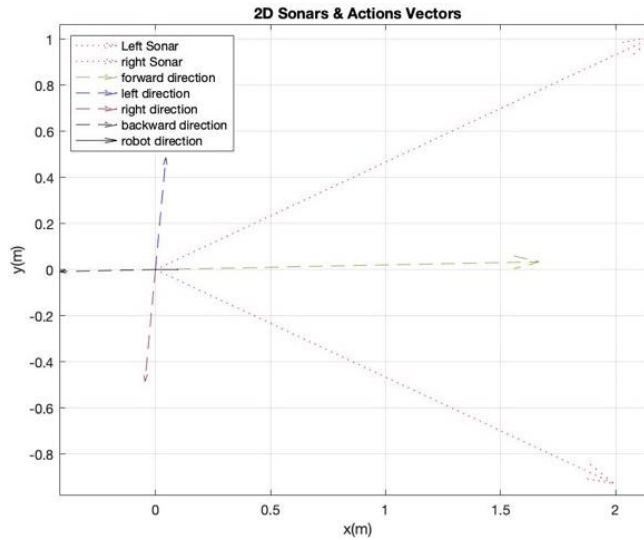
- 6) All action vectors will be passed to the RL system.
-

(b) The process of calculating a cautious action

Figure 6-6: The design of cautious actions.



(a) Four directions before weighing.



(b) Four directions after weighing.

Figure 6-7: The impact of cautious action with RL system.

Direction Detection & Go Towards Doorway Modules:

The “*Direction Detection*” perception module is triggered by the provided information from the knowledge system. If the doorway detection attribute in “*Topo-Room*” map is positive, then this module will be activated for running the system as explained in Chapter 5. As this system is able to detect if the deepest information is not related to the doorway, then the doorway detecting within the knowledge system will be updated. However, if the deepest information is related to the doorway, the calculated

angle will be passed to the “Go Toward Doorway” action module to turn and translate the robot toward the doorway.

Move Smoothly Module:

In cases where the robot should move to the detected direction of the doorway where there is an obstacle in the way, designing a smooth move is important. This action module is the weighted summation of the output from the preceding action modules, i.e. “Avoid Obstacle” and “Go Toward Doorway”. Let us call the action from “Avoid Obstacle”, “Go Toward Doorway” and “Move Smooth” as a_o , a_g and a_s , respectively, in which each action consists of angle and translation values. When the robot is very close to the obstacle, then the “Avoid Obstacle” action takes the full or the higher control. Whereas, if there is no obstacle or they are far enough, then the “Go Toward Doorway” action takes the higher weight. This is executed by applying the following weighted summation function:

$$a_s = w * a_o + (1 - w) * a_g$$

Layer 3 and 4: Achievement and Protective Tasks

For an end-to-end system, modules in these two layers are designed using the pre-defined functions from the Naoqi API, see the Appendix. The achievement task is to end the navigation process via following a voice command made by a companion using Nao’s speaker. The “Command Detection” perception module can be designed by detecting one of the five room classes from room classification component as presented in Chapter 3. By comparing the command and the information in the knowledge system, the “Sitting Down” action module will be run as an indication of completing the navigation process. Similarly, the protective task is important to keep the robot protected from falling if the battery is out of charge. The battery level is checked all the time during the process of navigation. If the level is low, e.g. less than 20%, then the system applies the “Charging” action module. As this function is not mainly a part of navigation in this project, so we keep this module simple by applying a request function that the robot asks to be charged.

6.3.2. Knowledge-Based System

The behavior-based approach for the navigation system was inspired by the concept of behaviorism, which speculates that behaviors are triggered by the environment. In contrast, having mental representations that play a causal role in behaviors was the assumption stated by cognitive psychology [226]. Accordingly, we modify the subsumption-based system by integrating a knowledge-based system that is a crucial part in the learning phase. We adopt the topological-based mapping approach for achieving the knowledge part of the system. Topological map [241] is a graphical representation of the environment that consists of nodes and edges. Nodes represent different places while edges are the connection between relative positions of these places.

Room Localization & Doorway Detection Modules: (CNN-Based Models)

The knowledge-based system aims to create a Directional Semantic Topological Map via designing a module called “*Topo-Room*”, which will be explained below. Therefore, the system has to begin with “*Room Localization*” and “*Doorway Detection*” perception modules that successfully designed and tested using CNN model as explained in detail in Chapter 3 & Chapter 4, and Chapter 5, respectively. “*Room Localization*” provides the semantic feature, whereas “*Doorway Detection*” provides the directional feature to the topological map.

Doorway Passing Module: (Canny Edge Detection & Hough Transform)

When the robot identifies the room’s type and detects the doorway, then there is no need to keep applying them again while the robot wanders in the same room. Thus, this is the key role of integrating a knowledge system with the behavior-based navigation system. When the robot moves to a new room, then it needs to capture a new image for adding new knowledge to the “*Topo-Room*”. Therefore, we design a “*Passing Doorway*” perception module based on the Canny edge detection [242] and the Hough transform [243], as an indicator that the robot left the current room and arrived at a new room. The key idea is to extract the two edges of the doorway from a 2D image towards the doorway. The distance in pixels between the two edges will be increased while the robot gets closer to the door. When the two edges are out of the robot’s view, then it is most likely that there are new edges will be detected with a smaller distance than the previous

one, as will be shown in section 6.4.1. Figure 6-8 shows the practical process of getting doorway edges.

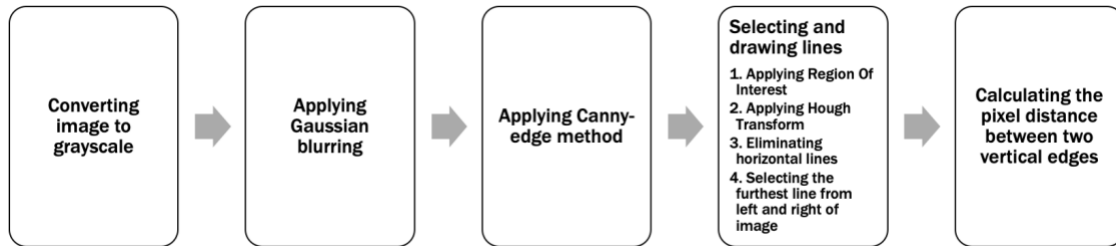


Figure 6-8: Doorway’s edges detection for “Passing Doorway” module.

Topo-Room Module: (A Directional Semantic Topological Map)

The “Topo-Room” module is a Directional Semantic Topological Map (DST-Map). It is semantic as each node associated with a specific class of room that is provided by “Room Localization” perception module. In addition, “Topo-Room” is directional because the relative position is based on the four directions {East, West, North, South} that can be extracted by “Doorway Detection” perception module and the predefined angles in the first layer (exploration task). The objective of this module is to build an abstract map for the house environment. The abstract map is a high-level representation that saves the connection between rooms. Thus, the created map is a collection of nodes that represent rooms, and these nodes are connected by edges that represent the direction towards the doorway. There are eight attributes associated with each node, as shown in Figure 6-9. They are as follow:

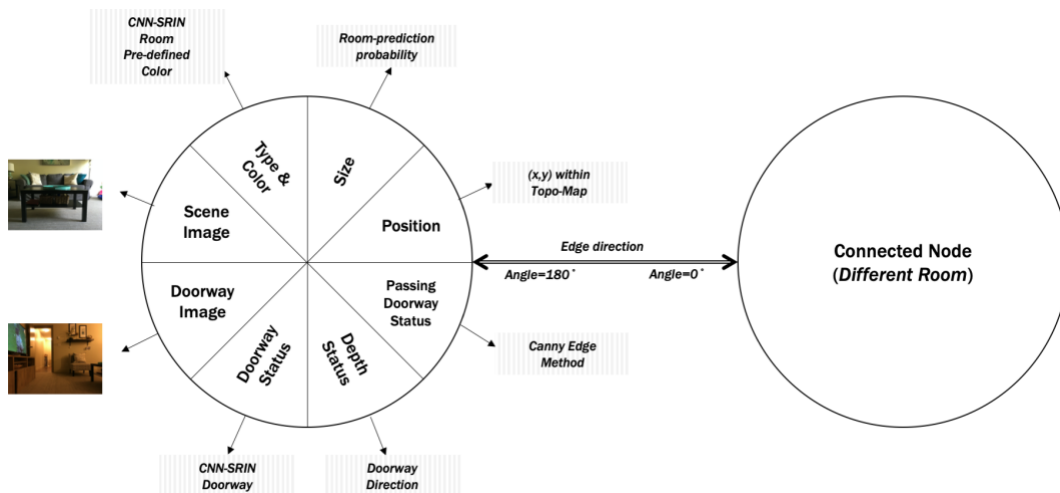


Figure 6-9: Main attributes or information within each node in the TDS-Map.

- The first two attributes are the type with a predefined color and the size. They are extracted from the “Room Localization” module, in which the type is the prediction of the room class and the size is the prediction’s probability.
- The node’s position, i.e. the third attribute, is assigned based on one of the four directions (*east*: 0°, *west*: 180°, *north*: 90°, *south*: –90°) as the key is to find the relation direction between nodes. Thus, we assume that all nodes’ positions depend on the position of the first node. In other word, we assume that the first node for the first classified room is positioned in the (0,0) of the map, and the direction of its doorway is always in the *east*: 0°. Then, the position of the next room depends on the position of the previous room and the global direction of the doorway by applying these two equations:

$$next_{node_x} = pre_{node_x} + \cos(pre_{door_angle})$$

$$next_{node_y} = pre_{node_y} + \sin(pre_{door_angle})$$

- Each node is associated with a saved scene image as the fourth attribute for future needs, such as appearance association or matching.
- The other attributes are related to the doorway within the room, which are extracted from “Door Detection” perception module. Doorway status gives information about doorway detection as positive or negative. If it is positive, then the image will be saved as the sixth attribute, i.e. doorway image. The seventh attribute is for depth status that give information about the execution of “Doorway Direction”. If the module is executed by calculating the depth and direction toward the doorway, then its status is positive. Finally, the attribute of passing doorway status gives positive or negative information based on the “Passing Doorway” module, while it saves the distance between the two edges every time that is needed. If its status is positive, then the process of gaining new information and creating a new node will start again.

The important attribute related to the edges is a bidirectional angle from the four directions between two classified rooms. The flowchart in Figure 6-10 shows the map building process of the “Topo-Room” module within the knowledge-based system. The expected DST-Map is an abstract map that contains nodes of rooms with their positions

in the map space, and the edges between nodes that show the angular relation between two relative nodes, as shown in the illustrated example of Figure 6-11.

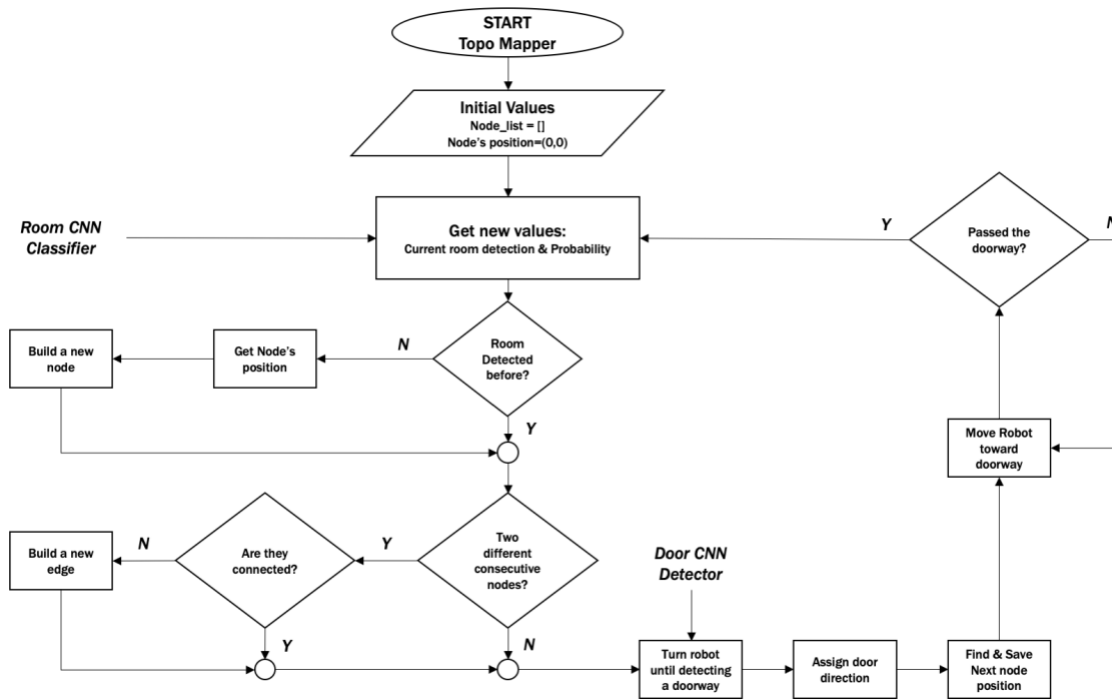


Figure 6-10: Flowchart of creating and updating the Directional Semantic Topological Map (DST-Map).

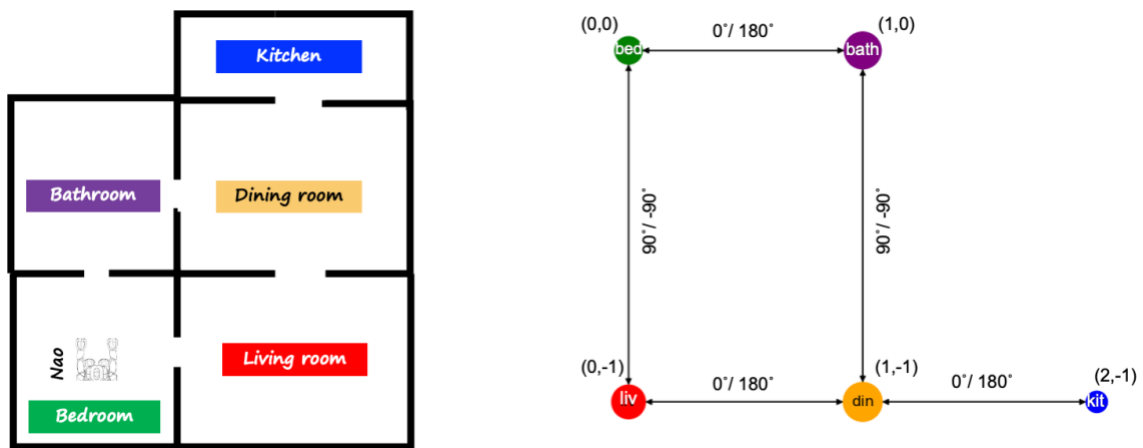


Figure 6-11: An illustration of the expected DTS-Map (right) for a simple layout (left).

6.3.3. Implementation Setup

All modules were programmed by python language using a Laptop with a 64-bit Linux operating system with 8GB RAM. It has a Graphic Process Unit of Quadro K620M with 2GB total memory. As the system was implemented on a small memory size of GPU, all modules were executed in a sequential manner. Therefore, Figure 6-12 shows the pseudocode of implementing all modules together within the system.

Pseudocode

```
new room = True
For each step:
  # Go to the knowledge system
  If new room = True: #(Exploring new room)
    Capture a picture by robot
    Run CNN-Room & CNN-Doorway
  If doorway status = True & depth status = True:
    Run Passing Doorway module, get edge distance
    & compare with previous distance
    if distance > previous distance
      new room = True
    else:
      new room = False
  else:
    Capture another picture
    Run CNN-Doorway
  Update DST-Map

  ## Go to Subsumption system
  direction = 0°
  Acquire data from battery, speaker and sonars
  Access DST-Map information
  if battery < threshold:
    Activate the protective layer & break
  if CNN-Room prediction = command detection by user :
    Activate the achievement layer & break
  if doorway status = True or sonar values < very close range:
    Activate the purposive layer
    If depth status = False:
      Run Direction Detection, get depth value
      update depth status = True in DST-Map
    Check the sonars values
    Run the weighted smooth move
  else:
    Activate the exploration layer, update direction angle
  Update direction angle in DST-Map
```

Figure 6-12: Pseudocode of implementing the overall system sequentially.

6.4. Experiments and Results

We have created an apartment model* with Nao humanoid robot using Webots simulator [14] for executing all virtual-time experiments, see Figure 6-13.



Figure 6-13: An apartment virtual model with Nao robot using the Webots simulator.

6.4.1. Evaluation of Individual Modules

The previous chapters presented the individual evaluation and successful results of the “Room Localization” and “Doorway Detection” within the knowledge system, and the “Direction Detection” within the layer of the purposive task. The evaluation of the other important modules is shown in this section.

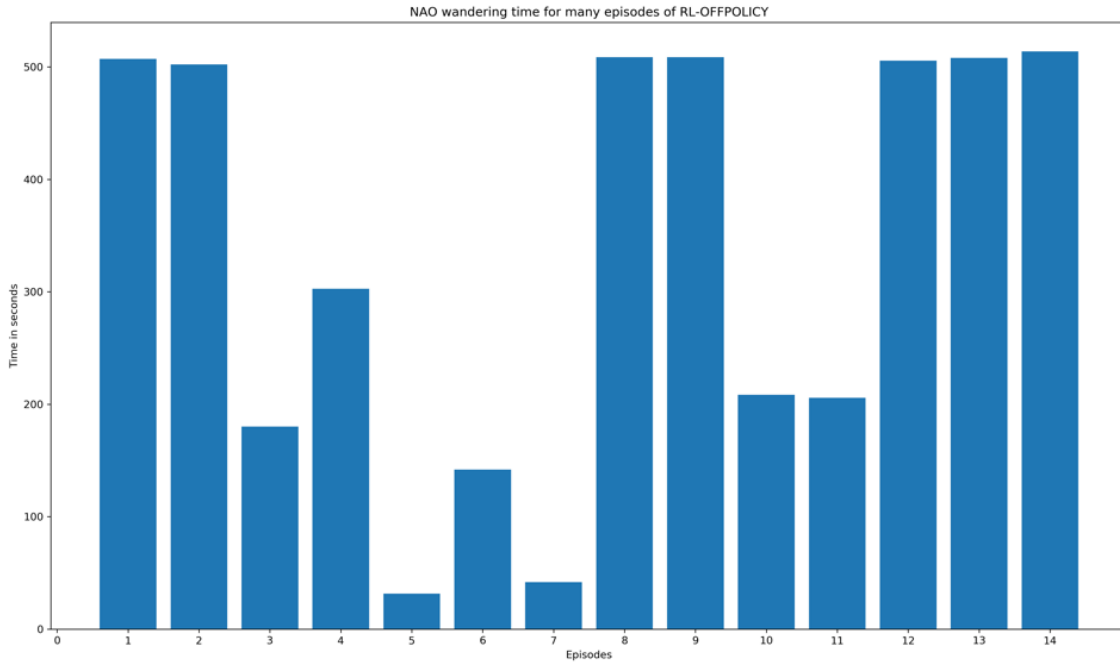
Evaluating RL system for obstacle avoidance module

All experiments have been executed within the virtual environment with a learning rate $\alpha = 0.2$, and a discount factor $\gamma = 0.8$. In order to ensure having a good learning process, an ϵ -greedy added to the RL process thus the robot will not be stuck in certain areas, and it can face all states situation as well as learn the best associated

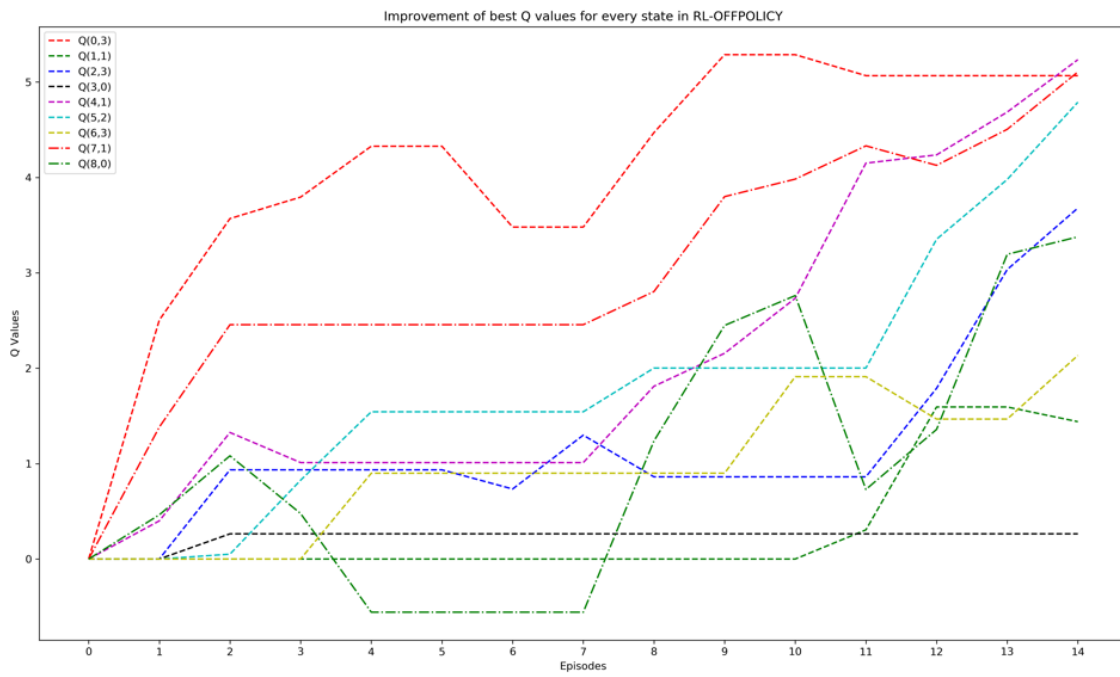
* The apartment model can be found in the official webpage of Webots in this link: https://www.cyberbotics.com/doc/guide/samples-environments#complete_apartment-wbt

behavior. We suggest a gradual ascent ϵ -greedy for every 10-time steps. Every step is considered as an action that is taken by the robot during the process. A gradual ascent ϵ -greedy combines the exploring and exploiting movements in every episode. So, we started with 60% of ϵ -greedy in the first 10-time steps in which the robot starts with exploring movements in the first 4 steps. Then, the ϵ -greedy is increased to 70% in which the robot explores in the first 3 steps of the second 10-time steps. Therefore, the robot increases the exploiting movement and decreases the exploring movement every 10-time steps until reaching 50-time steps when the robot moves with fully exploiting based on what have been learned during the RL process. Since the objective of this experiment is that the robot keeps avoiding obstacles as long as it can, then time of wandering is the key factor of evaluating the performance of this experiment. Each episode is ended either when the time of wandering is reached 500 seconds or when Nao falls down. The process of RL is terminated when Nao is able to wander without falling for three consecutive episodes. Thus, Nao is able to move about 1500 seconds, i.e. over 25 minutes.

Figure 6-14 shows the RL results by presenting the wandering time and learning improvement of Q-values in every episode for both Q-learning, Figure 6-14 (a), and SARSA, Figure 6-14 (b), methods with gradual ϵ -greedy. Nao was able to wander around the living room for three consecutive episodes without colliding obstacles after 14 episodes using Q-learning method as shown in Figure 6-14 (a.1), while it learned much faster using SARSA method as shown in Figure 6-14 (b.1). Furthermore, the improvements of the best Q-value for each state in the RL system shows that SARSA model was more stable than the Q-learning model as shown in Figure 6-14 (a.2 and b.2). Therefore, the trained model of SARSA was adopted for the overall system's evaluation.

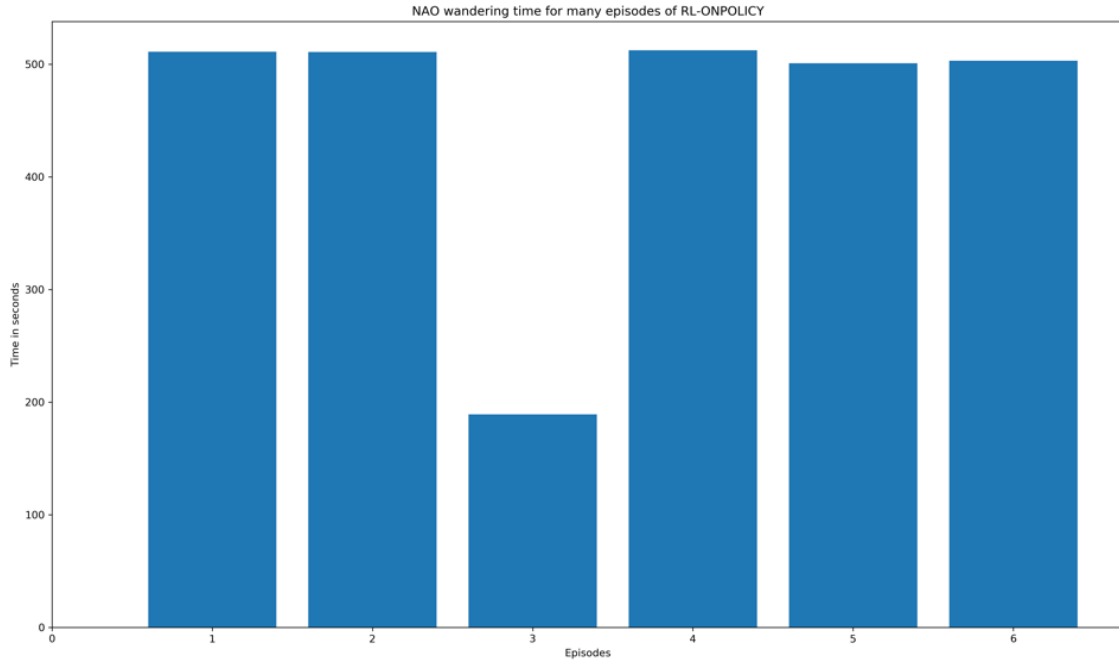


a.1) Wandering time in all episodes.

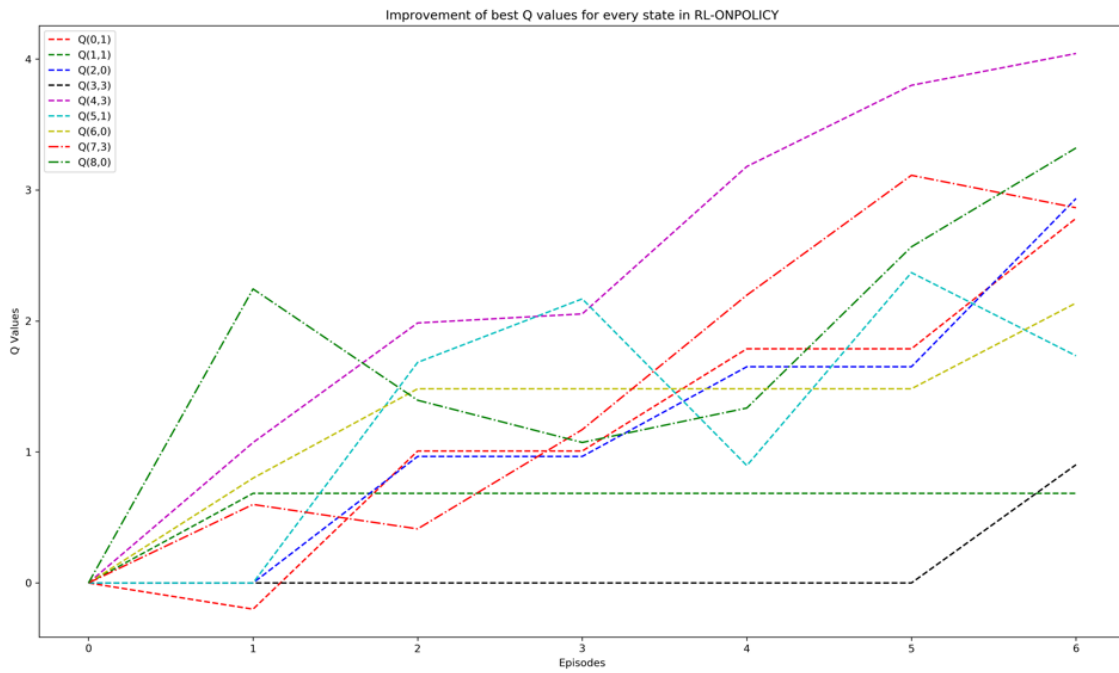


a.2) Best Q-values improvement

(a) Q-learning with 60% gradual ϵ -greedy



b.1) Wandering time in all episodes.



b.2) Best Q-values improvement

(b) SARSA with 60% gradual ϵ -greedy

Figure 6-14: RL training results for both methods: a) Q-learning and b) SARSA.

Evaluating doorway edges for "Passing Doorway" module

This module was tested with a simple real-time experiment with Nao robot to evaluate the concept of passing the doorway based on detecting edges. Figure 6-15 shows a sequence of images while Nao moves toward the doorway in the AISL lab at SFU. As we can see that the distance between the two detected edges of the doorway is increasing when the robot gets closer to the door, see Figure 6-15 (a)-(c). Whereas the distance becomes smaller when the robot passed the doorway as shown Figure 6-15 (d).

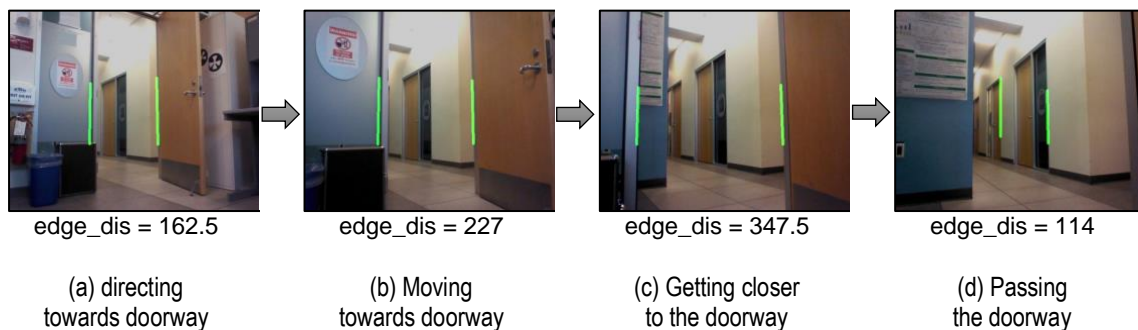


Figure 6-15: Doorway edges detection for "Passing Doorway" module.













6.4.2. Evaluation of the Overall System

Several experiments with different scenarios are presented below for a virtual evaluation of the overall system using a Webots model [14]. The objective of these scenarios is to show that the robot is able to move between two different rooms, specifically the kitchen and the living room, safely. Thus, it maintains information of two connected rooms within the knowledge system, i.e. two connected nodes with an edge. This can be extended for exploring other rooms. Therefore, the results will not show the mapping part. All results for all scenarios are presented in two parts: a) The actions' decision making based on the behavior and knowledge systems in every step, and b) the doorway perception's output for both depth and edge detecting during the exploration process.

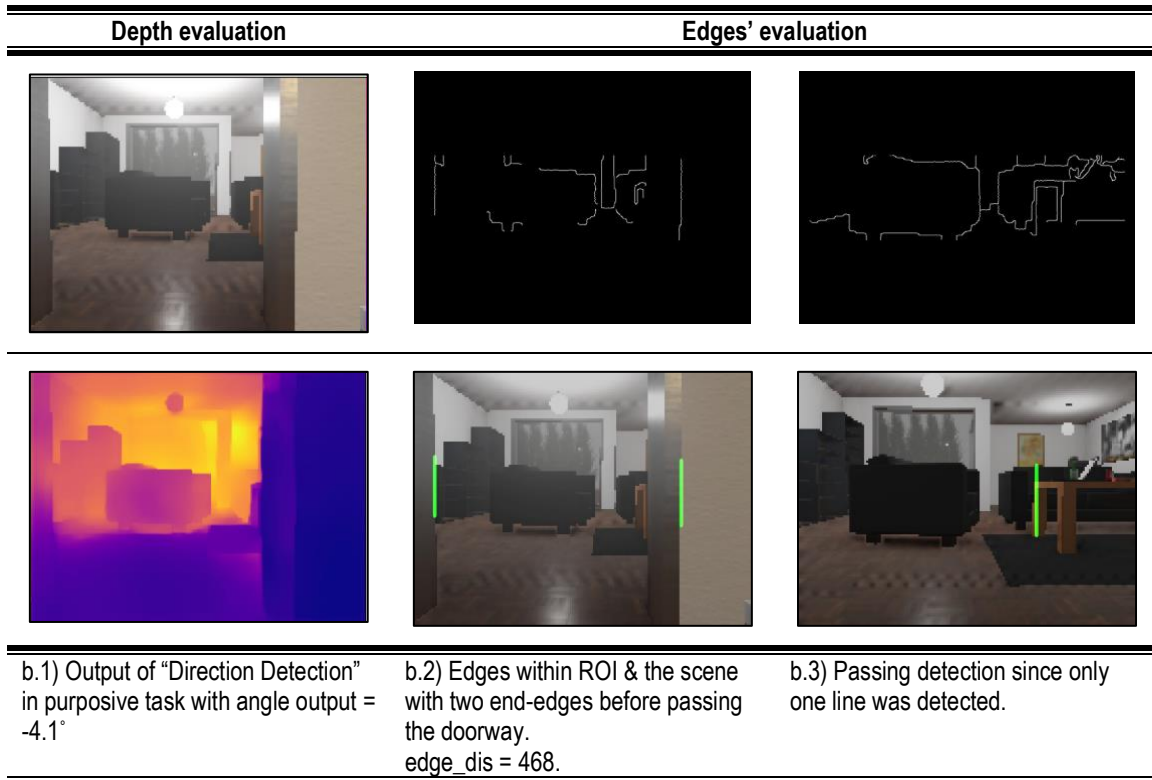
Scenario 1 – Moving between two rooms with no obstacles

The demonstration of this scenario is to test the robot ability to move from the kitchen to the living room while no obstacles in the way. We try to test the robot ability to

recognize the current room and find the right direction of the doorway to connect the two subsequent rooms within the map. Also, the robot is tested to decide when it passed the room to start a new room recognition and updated the map. The detailed perceptions and actions of the system for this scenario are shown in Figure 6-16. The robot started from a position close to the doorway, which is in the left side of robot, in the kitchen while there was know any other obstacles. The robot was able to predict the kitchen with a highest probability of 79.4% compared to other classes, while the status of doorway detection was “no-door”. Therefore, the first layer (exploration) was activated, and the robot turned by 90° to the left. Now, the robot was able to detect the doorway and change its status in the DST-Map, thus the “Direction Detection” was activated and calculated a small angle between the robot and the doorway. During the experiments, we considered any calculated angles from depth information within the range of $[-10^{\circ}, 10^{\circ}]$ as small values, so the robot did not need to turn that small values. Hence, the robot almost directed to the doorway, the “Passing-Doorway” module detected the doorway edges and calculated the distance. After that, the second layer (purposive) was activated, in which the “Go Toward Doorway” took the all weight of the smooth movement. Then, the system compared the new edge distance, which was zero as only one edge was detected, with the previous saved distance in the knowledge system. Since it was smaller, the status of the “Passing Doorway” attribute within the DST-Map was changed, and the “Move Straight” action in the first layer (exploration) was activated to make sure a full pass to the new room. Subsequently, gaining new knowledge started again by classifying the new room as a living room with 93.7%.

Steps	1	2	3	4
Webots screenshots				
Zoom in to Nao robot				
Scenes from top camera				
Description	a.1) The start position within kitchen.	a.2) Activating the "Turn" action from exploration task.	a.3) Activating the "Go Toward Doorway" action from the purposive task with a full weight.	a.4) Activation of "Move Straight" action from exploration task after passing the doorway.

(a) Subsumption decision making based on local data and maintained knowledge.








(b) Some Perception results within exploration process.

Figure 6-16: Evaluating the overall system in scenario 1.

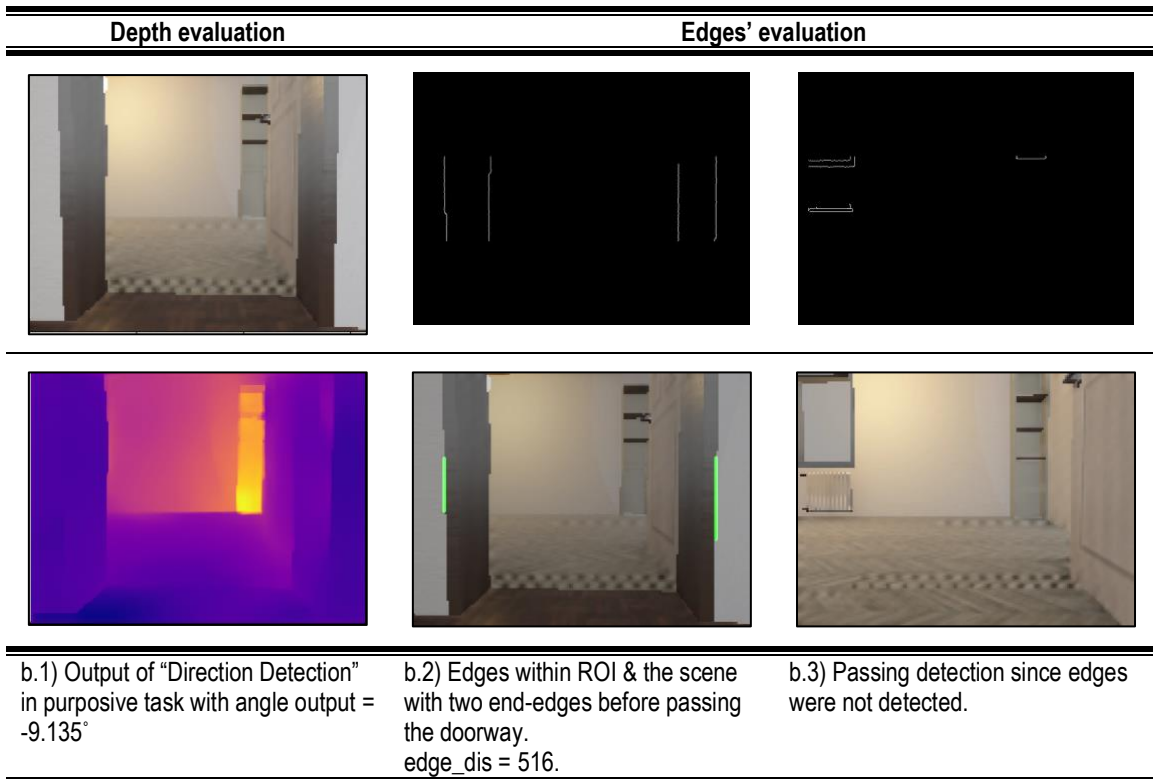
Scenario 2 – Moving between two rooms with a different direction

The difference in this scenario is that the direction of the doorway is in the right side of the robot. The advantage of angle directions order in the exploration task can be showed in this scenario. Similar to the first scenario, the robot is tested to recognize current room and find the right direction of the doorway as well as it is tested to pass the room and start a new room recognition with updating the map information. We tried in this scenario to start the experiment from the opposite room, i.e. living room, as shown in the detailed results of Figure 6-17. The first obtained information by the knowledge system was classifying the room correctly with a 96.4% prediction's probability. The exploration task was activated twice with turning the robot by 90° and 180° consecutively until detecting the doorway. Once the doorway was detected and its corresponding attribute was updated in the knowledge system, the depth information was calculated with a small value. Consequently, the doorway edges were found with a large distance between edges, which it indicates that the robot is very close to the door and ready to pass it. As there were no obstacles, the "Go Toward Doorway" action in the purposive

task was activated and took the all weight of “Smooth Move”. Once the “Passing Doorway” status updated in the knowledge system, the robot started gaining a new knowledge and building a new node of kitchen room with 83.3% prediction’s probability.

Steps	1	2	3	4
Webots screenshots		2.1 		
		2.2 		
Zoom in to Nao robot		2.1 		
		2.2 		
Scenes from top camera		2.1 		
		2.2 		
Description	a.1) The start position within Living room.	a.2) Sequence of activating the "Turn" action from exploration task.	a.3) Activating the "Go Toward Doorway" action from the purposive task with a full weight.	a.4) Activating the "Move Straight" action from exploration task for full passing.

(a) Subsumption decision making based on local data and maintained knowledge.



(b) Some Perception results within exploration process.

Figure 6-17: Evaluating the overall system in scenario 2.

Scenario 3 – Moving between two rooms with obstacles and a final goal

Including the objectives from last scenarios, the objective of this scenario is also to test obstacles avoidance and moving smoothly towards the doorway as shown in Figure 6-18 by adding extra objects around the starting position. Also, the final target is assigned from the beginning to test the performance of the achievement task. The robot started from the kitchen and its goal to go to the living room as a final targeted room. We designed the system to activate the weighted "Avoiding Obstacle" action if one of the sonars less than 1m within the "Move Smoothly" action, while a full weighted "Avoiding Obstacle" if one of the sonars is less than 0.4m. So, the robot started by gaining information about 96.3% prediction of the kitchen. Then, the exploration task was activated, and the robot turned by 90° to the left direction. Now, many perceptual modules were activated sequentially to acquire doorway-related information, i.e. doorway detection, doorway direction, and doorway edges as explained in the previous

scenarios. We noticed that during the robot's actions from the purposive task, the sonar readings were not less than 1m. Therefore, the robot kept moving toward the doorway. The doorway edges were detected, and their distances were calculated three time consecutively. Once the edges' distance became smaller than the previous save distance, the status of "Passing Doorway" was updated to positive. Now instead of activating "Move Straight" action form the exploration task for a full pass through the doorway, the system activated the "Avoid Obstacle" action within the purposive task via RL since the sonars values were (*left=0.38m and right=2.5m*). When the robot completely moved away from the obstacle, which was the door edges in this experiment, gaining a new information started by classifying the new room as a living room with 97.3%. Since the classified room matched the target room, the "Sitting Down" action was activated as an indication of ending the process.

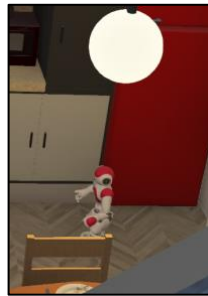
Steps	1	2	3	4	5
-------	---	---	---	---	---









Webots screenshots



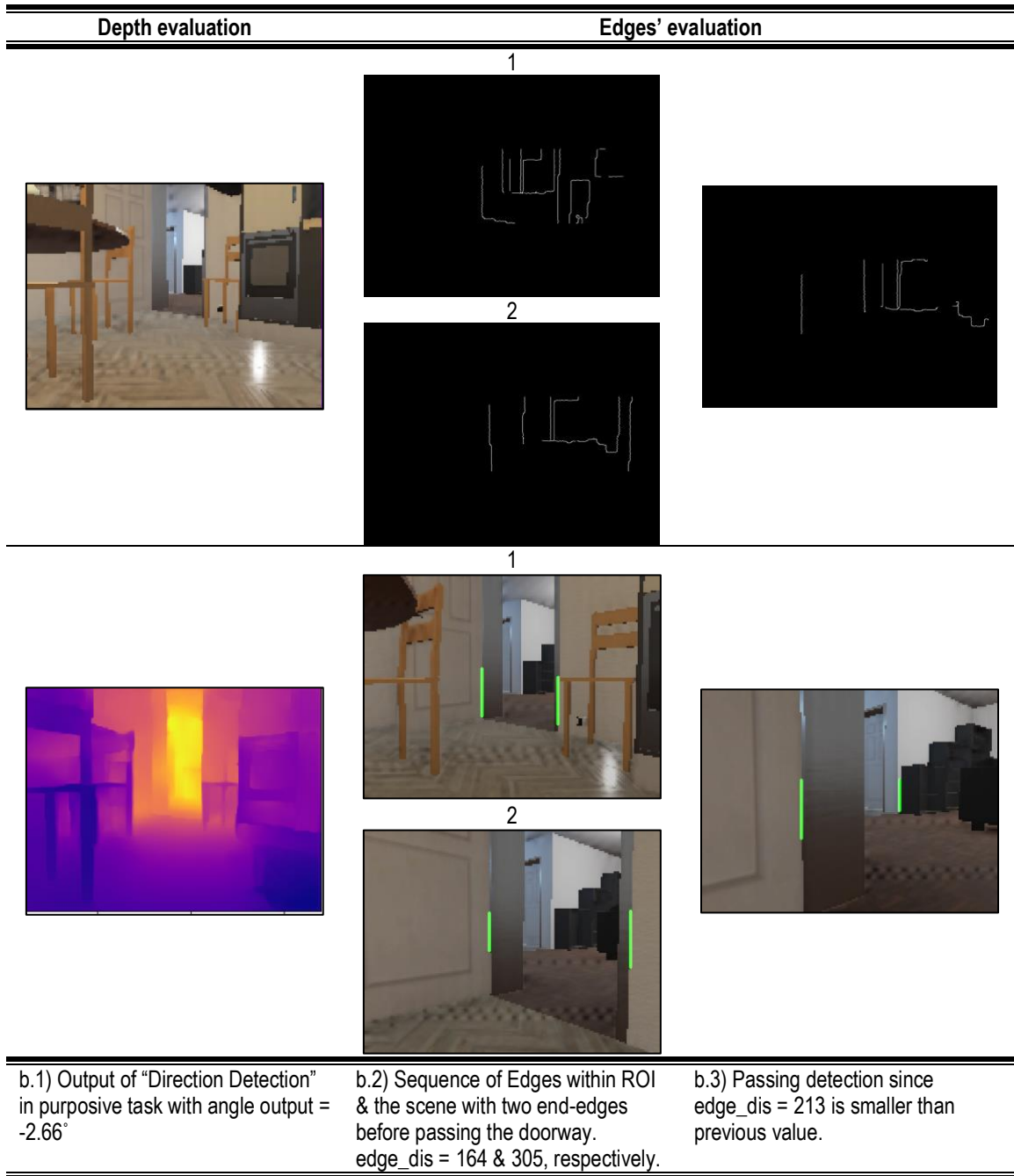
Steps	1	2	3	4	5
-------	---	---	---	---	---

Zoom in to Nao robot



Steps	1	2	3	4	5
Scenes from top camera			3.1 	4.1 	
			3.2 		
				4.2 	
			3.3 		
Description	a.1) The start position within Kitchen with adding table's obstacle.	a.2) Activating the "Turn" action from exploration task.	a.3) Sequence of activating the "Go Toward Doorway" action from the purposive task with a full weight.	a.4) Activation of "Move Straight" action from exploration task after passing the doorway. Then a full weight of avoiding obstacles was activated before gaining new knowledge.	a.5) Activation of the "Sitting Down" action as an end of the exploration process.

(a) Subsumption decision making based on local data and maintained knowledge.



(b) Some Perception results within exploration process.

Figure 6-18: Evaluating the overall system in scenario 3.

Discussion

The proposed system is flexible as each module can be designed, tested and modified individually and within the overall system. The virtual experiment results are promising and demonstrate that the system can be adopted and modified for any social

robot with limited sensors for domestic applications. In this section, we discuss some of the limitation or areas for improving results. The RL for “Avoiding obstacle” worked very well during the training stage within the area of the living room. However, as this room was spacious; in order to get a better validation, the RL could be applied and tested in a congested room to improve its Q-values. Also, the depth images with their calculated angles from scenario 1 & 2 were not the best expected angles to keep the robot’s direction exactly in the middle of the doorway. This might be due to the inaccuracies in simulator’s view as the result from the real-time experiments from the previous chapters were much better and accurate. Thus, we ignored the small values in the virtual experiments and let the robot moved straight towards the doorway. In addition, adding a class of corridors for the room classification project is important for future work in order to extend the exploration process between more rooms. Thus, building the topological map will be more meaningful. Finally, this system can be tested and modified in real-time experiments in several houses.

Although extensive simulation runs with Webots were conducted; the real-time experiment with NAO could not be completed as this part of the project was completed in May 2020 when there was lockdown due to COVID-19 and access to the laboratory was not possible. Having said that, we suggest that the real-time experiments with NAO will produce similar results as Webots since this software uses the same NAOqi API as the actual NAO humanoid robot.

Therefore, the proposed system is an alternative solution for addressing localization and mapping sequentially for indoor environment, specifically homes. We coined the term Sequential Localization and Mapping (SeqLAM), and provided a qualitative comparison to the widely popular probabilistic Simultaneous Localization and Mapping (SLAM) algorithm as shown in Table 6-1.

Table 6-1: Qualitative comparison between classic SLAM and SeqLAM.

Features	SLAM	SeqLAM
Philosophy	Probabilistic	Behavioristic
A priori knowledge	Assigning Landmarks	Zone based (in homes)
Update information	Incremental robot's pose and map	Identifying zones, then generating a map sequentially
Map	Accurate	Sketch / spatial relationship
Pose	Accurate	Not applicable
Dynamic settings	Numerous challenges	Moderate
Computation load	High	Moderate
Comparison to human reasoning	Counter-intuitive	Intuitive
Generality	Outdoor / Indoor	Indoor
Data Association	Significant challenge	Can be incorporated via image matching
Human intervention	Robot is mostly driven manually (except for autonomous exploration)	Autonomous
Application	Universal (but needs to be tailor made)	Only homes (can be tailor made for other indoor settings, e.g. hospitals)

6.5. Conclusion

This chapter presented an end-to-end navigation system for social robots with limited sensors designed exclusively for homes. The design combined a subsumption-based system and knowledge-based system. The subsumption system consisted of a collection of behaviors arranged in layers, in which each layer was responsible for a specific task and got activated based on the sensor data. Whereas, the knowledge system consisted of several visual-learning modules to gain information about the environment to build a high-level meaningful map as well as to access all layers in subsumption and trigger the appropriate action. Some modules were evaluated individually in virtual or real-time implementation with the Nao robot. For example, an RL model was designed properly with two different approaches, Q-learning and SARSA, for obstacle avoidance as an adaptive behavior using only two sonars, and the model was

evaluated virtually by observing the time of exploration and number of epochs. The model with the SARSA approach learned faster than the Q-learning. "Passing Doorway" was the other module that was evaluated individually in this chapter. It was tested in a simple real-time experiment to evaluate the concept and to be adopted with the overall system. On the other hand, the overall system was tested virtually using Webots simulator with different scenarios. Although there were some restrictions and assumptions, the performance of all scenarios is acceptable.

Chapter 7.

Contributions and Future Work

7.1. Contributions

The aim of this section is to capture the main findings and contributions reported in the previous chapters. We would like to modestly claim the following contributions:

- Addressing the room classification problem for social robots is the first contribution. The CNN deep learning approach was adopted for this purpose because of its superiority in the areas of pattern recognition. Several CNN architectures were examined by fine-tuning them on five rooms classes of the Places dataset, in order to find out the best model for real-life experiments. It was found that VGG16 is the best model to be adopted, with 93.29% of validation accuracy after cleaning the dataset by excluding all mislabeled images. In addition, we proposed and examined a combination of CNN with ECOC, a multi-binary classifier approach, in order to address the error in practical prediction. The validation accuracy reached 98.5% in one of the binary classifiers and 95.37% in the average of all binary classifiers. The CNN model and the combination models of CNN and ECOC in both forms, i.e., CNN-ECOC and CNN-ECOC-REG, were evaluated practically on the Nao humanoid robot. The results show the superiority of the combination model over the regular CNN.
- The second contribution is the proposal of a new Social Robot Indoor Navigation dataset called SRIN. It consists of 2D colored images for both rooms and doorways, i.e. SRIN-Rooms & SRIN-Doorways, respectively. SRIN is a useful dataset for medium-sized social robots in indoor environments, specifically houses. SRIN has been validated through training a CNN-based model using SRIN-Rooms dataset, and then tested on Nao's images for real-time experiments validation. The novelty of this work was illustrated when the validation accuracy of CNN-SRIN for room classification reached to 97.3% in a relatively short time. This was a huge improvement compared to training the same architecture with the Places dataset that reached 93.29% of validation

accuracy after a long time. In addition, the significance of this work was also shown through the comparison of the performance of two models on real-time experiments for Nao. It shows a big improvement in predicting bedrooms and slightly different performance of other classes in Top1. However, it reached 100% in the Top2 of the correct predictions for four classes out of five. We believe that with increasing the SRIN dataset in the future, the prediction of top-1 for real-time experiments on Nao shall be improved as well.

- Addressing doorway detection is the third contribution as an important feature for any indoor navigation system. We proposed a robotic system called the 3Ds-system, which stands for Doorway Detection and Direction system that was applied and tested on a Nao humanoid robot. The goal of the proposed system was to control the Nao direction towards the doorway based on a 2D image from a monocular camera. The system takes a 2D colored image and provides an angular value in degrees via a combination of several modules. CNN-SRIN doorway module for detecting a doorway was applied on Nao images after getting a validating accuracy of 97.96%. Then, the Depth module, Pixel-Selection module and Pixel2Angle module were applied on the input of 2D images for directing Nao towards the doorway. The practical results are promising and demonstrate the success of the proposed system for Nao. The proposed system can be applied to any other similar social robot, by acquiring the proper angle direction toward the door. The overall system has been validated by implementing the 3Ds-system on Nao within a new environment, specifically in AISL at SFU Canada. We suggest that the proposed system is very useful in robotic navigation applications for medium-sized robots with limited sensors, such as a monocular camera, in structured indoor environments.
- The last contribution is the design of an end-to-end navigation system for social robots with limited sensors within apartments' environments. The design combined a subsumption-based system and knowledge-based system. The subsumption system consisted of a collection of behaviors arranged in layers, in which each layer was responsible for a specific task and got activated based on the sensor data. Whereas, the knowledge system consisted of several visual-learning modules to gain information about the environment to

build a high-level meaningful map as well as to access all layers in subsumption and trigger the appropriate action. Some modules were evaluated individually in virtual or real-time implementation with the Nao robot. For example, an RL model was designed properly with two different approaches, Q-learning and SARSA, for obstacle avoidance as an adaptive behavior using only two sonars, and the model was evaluated virtually by observing the time of exploration and number of epochs. The model with the SARSA approach learned faster than the Q-learning. "Passing Doorway" was the other module that was evaluated individually. It was tested in a simple real-time experiment to evaluate the concept and to be adopted with the overall system. On the other hand, the overall system was tested virtually using Webots simulator with different scenarios. Although there were some restrictions and assumptions, the performance of all scenarios demonstrated promising practical results. Like any other research, the next chapter for me is to continue research in this fascinating area.

7.2. Future Work

Our main objective in this research program was to study a behavior-based navigation system for social robots with limited sensors at homes. Tackling the navigation problem is considered a crucial task for social robotics and other autonomous robotics applications. The following discussion presents some thoughts and recommendations for future research in this area:

- Although CNN-like models show promising results for classification and detection applications, they still require a huge dataset. However, there are many robotic applications with a small dataset. In this work, for example, we had to increase the SRIN dataset samples by applying image augmentation to train the model and avoiding the overfitting issue. We think that it is useful for future work to design a better algorithm that learns from a small dataset. This will be practical to train the model online using the robot's processors with a low computational cost. Studying this problem will create intuitive and general intelligence as people who do not need a huge data for classification or detection tasks, e.g. room classification.

- We did not include corridors that link different rooms. It is important to add corridor as a specific class to be learned.
- We focused on social robots with limited sensors, e.g. a monocular camera. However, the progress in the field of visual sensors improves rapidly while drastically reducing the cost. Thus, it is recommended for future work to employ visual depth sensors within the proposed system for a more robust navigation process including incorporating object detection (relevant to each room).
- As a social robot at home could serve as a companion to humans, it is important to address the task planning issue. It deals with the reasoning actions to reach a goal, in which it is opposed to the motion planning that depends on the configuration space [244]. The task planning can be designed and integrated to our proposed system, in which it takes the advantage of the saved information in the directional semantic topological map (DST-Map) as well as the collection of behaviors in the subsumption layers.
- In the age of the Internet of Things (IoT), merging IoT with robotic technologies will be an interesting area for research. This is known as the Internet of Robotic Things (IoRT) [245]. Accordingly, the navigation process can be studied by designing a robotic system that communicates and transmits data with other machines in the environment, i.e. machine to machine (M2M) communication, for more robust performance.

2020 will be registered in history as the year that the social and economic systems received a huge shock due to Coronavirus Pandemic (COVID-19). At the time of writing this chapter, like millions of people around the world, I have been working from home due to menace caused by this naked-eye invisible beast. As of today, the number of fatalities is 343,097 and over 5,383,900 people around the world are infected. only within almost four months [246]. Governments, around the world, have activated an extraordinary set of measures including travel restriction and strict quarantine directives for weeks that might be extended for months. Although outside the scope of this thesis,

* These numbers of cases were taken on May 23, 2020. However, the number has been increasing unexpectedly.

we suggest that specially designed social robots will contribute as robotics nurses and physicians at homes and at hospitals. I am keen to continue my research in the health sector at the next stage of my career.

References

- [1] B. Gates, "A robot in every home.," *Sci. Am.*, vol. 296, no. 1, pp. 58–65, 2007.
- [2] "Household Robots Market," *Marketsandmarkets*. [Online]. Available: <https://www.marketsandmarkets.com/Market-Reports/household-robot-market-253781130.html>. [Accessed: 09-Apr-2020].
- [3] W. H. Organisation, "Global health and ageing." US National Institute on Ageing Bethesda, 2011.
- [4] W. H. Organization and others, "Mental disorders affect one in four people," *World Heal. Rep.*, 2001.
- [5] D. Portugal, P. Alvito, E. Christodoulou, G. Samaras, and J. Dias, "A Study on the Deployment of a Service Robot in an Elderly Care Center," *Int. J. Soc. Robot.*, 2019.
- [6] F. S. Melo *et al.*, "Project INSIDE: towards autonomous semi-unstructured human–robot social interaction in autism therapy," *Artif. Intell. Med.*, 2019.
- [7] A. Meghdari *et al.*, "Arash: A social robot buddy to support children with cancer in a hospital environment," *Proc. Inst. Mech. Eng. Part H J. Eng. Med.*, 2018.
- [8] R. Guyonneau and F. Mercier, "IstiABot, an open source mobile robot for education and research," in *12th International Workshop on Robot Motion and Control, RoMoCo 2019 - Workshop Proceedings*, 2019.
- [9] F. Arvin, J. Espinosa, B. Bird, A. West, S. Watson, and B. Lennox, "Mona: an Affordable Open-Source Mobile Robot for Education and Research," *J. Intell. Robot. Syst. Theory Appl.*, 2019.
- [10] I. Aaltonen, A. Arvola, P. Heikkilä, and H. Lammi, "Hello pepper, may i tickle you?: Children's and adults' responses to an entertainment robot at a shopping mall," in *ACM/IEEE International Conference on Human-Robot Interaction*, 2017.
- [11] R. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robot. Autom.*, vol. 2, no. 1, 1986.
- [12] D. M. West, "Brookings survey finds 52 percent believe robots will perform most human activities in 30 years." [Online]. Available: <https://www.brookings.edu/blog/techtank/2018/06/21/brookings-survey-finds-52-percent-believe-robots-will-perform-most-human-activities-in-30-years/>. [Accessed: 09-Apr-2020].
- [13] R. Bogue, "Domestic robots: Has their time finally come?," *Ind. Rob.*, 2017.

- [14] "Webots." [Online]. Available: <https://cyberbotics.com>. [Accessed: 10-Nov-2019].
- [15] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning Deep Features for Scene Recognition using Places Database," *Adv. Neural Inf. Process. Syst.* 27, pp. 487–495, 2014.
- [16] Sony, "Aibo Unleash Wonder." [Online]. Available: <https://us.aibo.com>. [Accessed: 15-Apr-2020].
- [17] SoftBank Robotics, "Nao Humanoid Robot." [Online]. Available: <https://www.softbankrobotics.com/emea/en/nao>. [Accessed: 15-Apr-2020].
- [18] SoftBank Robotics, "Pepper the humanoid and programmable robot." [Online]. Available: <https://www.softbankrobotics.com/emea/en/pepper>. [Accessed: 15-Apr-2020].
- [19] University of Hertfordshire, "Kaspar the social robot." [Online]. Available: <https://www.herts.ac.uk/kaspar/the-social-robot>. [Accessed: 15-Apr-2020].
- [20] J. J. Leonard and H. F. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *IEEE Trans. Robot. Autom.*, vol. 7, no. 3, pp. 376–382, 1991.
- [21] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): part I The Essential Algorithms," *Robot. Autom. Mag.*, vol. 2, pp. 99–110, 2006.
- [22] V. Kunchev, L. Jain, V. Ivancevic, and A. Finn, "Path planning and obstacle avoidance for autonomous mobile robots: A review," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2006.
- [23] H. I. Christensen and G. D. Hager, "Sensing and Estimation," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Cham: Springer International Publishing, 2016, pp. 91–112.
- [24] H. Durrant-Whyte and T. C. Henderson, "Multisensor data fusion," in *Springer Handbook of Robotics*, 2016.
- [25] D. Nakhaeinia, S. H. Tang, S. B. M. Noor, and O. Motlagh, "A review of control architectures for autonomous navigation of mobile robots," *Int. J. Phys. Sci.*, vol. 6, no. 2, pp. 169–174, 2011.
- [26] S. Thrun, B. Wolfram, and F. Dieter, *Probabilistic robotics*. 2005.
- [27] S. Patnaik, *Robot Cognition and Navigation: An Experiment with Mobile Robots*. Springer Science & Business Media, 2007.

- [28] P. Louridas and C. Ebert, "Machine Learning," *IEEE Softw.*, vol. 33, no. 5, pp. 110–115, 2016.
- [29] R. C. Arkin, *Behavior-based robotics*. MIT press, 1998.
- [30] R. R. Murphy, *Introduction to AI robotics*, vol. 108. 2000.
- [31] P. Jensen, "Motivation and Organization of Behaviour," in *The ethology of domestic animals: an introductory text*, Cabi, 2017, pp. 38–61.
- [32] J. J. Bolhuis and L.-A. E. Giraldeau, "The Study of Animal Behavior," in *The behavior of animals: Mechanisms, function, and evolution.*, Blackwell Publishing, 2005, pp. 1–10.
- [33] D. F. Sherry, "Brain and Behaviour," in *The behavior of animals: Mechanisms, function, and evolution.*, Blackwell Publishing, 2005, pp. 97–118.
- [34] N. J. Emery and N. S. Clayton, "Animal Cognition," in *The behavior of animals: Mechanisms, function, and evolution.*, Blackwell Publishing, 2005, pp. 170–196.
- [35] R. C. Arkin, "Motor Schema — Based Mobile Robot Navigation," *Int. J. Rob. Res.*, 1989.
- [36] R. A. Brooks, "Elephants don't play chess," *Robotics and Autonomous Systems*, vol. 6, no. 1–2. pp. 3–15, 1990.
- [37] Maja J. Mataric, "The robotics primer," *Choice Rev. Online*, vol. 45, no. 06, pp. 45-3222-45–3222, 2008.
- [38] H. Q. Chong, A. H. Tan, and G. W. Ng, "Integrated cognitive architectures: A survey," *Artif. Intell. Rev.*, vol. 28, no. 2, pp. 103–130, 2007.
- [39] F. Zal, T. S. Chen, S. W. Chi, and C. H. Kuo, "Fuzzy controller based subsumption behavior architecture for autonomous robotic wheelchair," *2013 Int. Conf. Adv. Robot. Intell. Syst. ARIS 2013 - Conf. Proc.*, pp. 158–163, 2013.
- [40] J. Mwaura and E. Keedwell, "Evolving robot sub-behaviour modules using Gene Expression Programming," *Genet. Program. Evolvable Mach.*, vol. 16, no. 2, pp. 95–131, 2014.
- [41] T. Thompson, F. Milne, A. Andrew, and J. Levine, "Improving control through subsumption in the EvoTanks domain," in *CIG2009 - 2009 IEEE Symposium on Computational Intelligence and Games*, 2009, pp. 363–370.
- [42] R. A. Brooks and M. J. Mataric, "Real robots, real learning problems," in *Robot learning*, Boston: Kluwer Academic Publishers, 1993, pp. 193–234.

- [43] S. Mahadevan and J. Connell, "Automatic programming of behavior-based robots using reinforcement learning," *Artificial Intelligence*, vol. 55, no. 2–3, pp. 311–365, 1992.
- [44] M. J. Mataric, "Reinforcement Learning in the Multi-Robot Domain," *Auton. Robots*, vol. 4, pp. 73–83, 1997.
- [45] Kao-Shing Hwang, Yu-Jen Chen, and Chun-Ju Wu, "Fusion of Multiple Behaviors Using Layered Reinforcement Learning," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 42, no. 4, pp. 999–1004, 2012.
- [46] H. Wicaksono, H. Khoswanto, and S. Kuswadi, "Behaviors Coordination and Learning on Autonomous Navigation of Physical Robot," *TELKOMNIKA Telecommunication, Computing, Electronics and Control*, vol. 9, no. 3, pp. 473–482, 2013.
- [47] S. Badillo *et al.*, "An Introduction to Machine Learning," *Clin. Pharmacol. Ther.*, 2020.
- [48] I. A. Basheer and M. Hajmeer, "Artificial neural networks: Fundamentals, computing, design, and application," *J. Microbiol. Methods*, 2000.
- [49] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 1, pp. 436–444, 2015.
- [50] S. Pouyanfar *et al.*, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Computing Surveys*. 2018.
- [51] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in Neural Information Processing Systems*, 2017.
- [52] E. Xi, S. Bing, and Y. Jin, "Capsule network performance on complex data," *arXiv Prepr. arXiv1712.03480*, 2017.
- [53] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.
- [54] Y. LeCun *et al.*, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.
- [55] L. D. Le Cun Jackel, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, B. Le Cun, J. Denker, and D. Henderson, "Handwritten Digit Recognition with a Back-Propagation Network," *Adv. Neural Inf. Process. Syst.*, pp. 396–404, 1990.
- [56] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Adv. Neural Inf. Process. Syst.*, pp. 1–9, 2012.

- [57] A. Canziani, A. Paszke, and E. Culurciello, "An Analysis of Deep Neural Network Models for Practical Applications," *Arxiv*, p. 7, 2016.
- [58] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *Int. Conf. Learn. Represent.*, pp. 1–14, 2015.
- [59] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
- [60] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June, pp. 1–9.
- [61] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems*, 2010, pp. 253–256.
- [62] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv Prepr. arXiv1207.0580*, 2012.
- [63] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2013.
- [64] L. Tai and M. Liu, "Deep-learning in mobile robotics-from perception to control systems: A survey on why and why not," *arXiv Prepr. arXiv1612.07139*, 2016.
- [65] D. Guo, T. Kong, F. Sun, and H. Liu, "Object discovery and grasp detection with a shared convolutional neural network," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2016.
- [66] F. H. Zunjani, S. Sen, H. Shekhar, A. Powale, D. Godnaik, and G. C. Nandi, "Intent-based Object Grasping by a Robot using Deep Learning," in *2018 IEEE 8th International Advance Computing Conference (IACC)*, 2018, pp. 246–251.
- [67] M. Zhong *et al.*, "Assistive grasping based on laser-point detection with application to wheelchair-mounted robotic arms," *Sensors (Switzerland)*, 2019.
- [68] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Detecting object affordances with convolutional neural networks," in *IEEE International Conference on Intelligent Robots and Systems*, 2016.
- [69] Q. Liang, J. Long, W. Zhu, Y. Wang, and W. Sun, "Apple recognition based on Convolutional Neural Network Framework," in *2018 13th World Congress on Intelligent Control and Automation (WCICA)*, 2018, pp. 1751–1756.

- [70] L. Quan *et al.*, “Maize seedling detection under different growth stages and complex field environments based on an improved Faster R–CNN,” *Biosyst. Eng.*, 2019.
- [71] Y. Y. Zheng, J. L. Kong, X. B. Jin, T. L. Su, M. J. Nie, and Y. T. Bai, “Real-Time Vegetables Recognition System based on Deep Learning Network for Agricultural Robots,” in *Proceedings 2018 Chinese Automation Congress, CAC 2018*, 2019.
- [72] C. Nuzzi, S. Pasinetti, M. Lancini, F. Docchio, and G. Sansoni, “Deep Learning Based Machine Vision: First Steps Towards a Hand Gesture Recognition Set Up for Collaborative Robots,” in *2018 Workshop on Metrology for Industry 4.0 and IoT, MetroInd 4.0 and IoT 2018 - Proceedings*, 2018.
- [73] S. Jaiswal and G. C. Nandi, “Robust real-time emotion detection system using CNN architecture,” *Neural Comput. Appl.*, pp. 1–10, 2019.
- [74] A. Lopez-Rincon, “Emotion Recognition using Facial Expressions in Children using the NAO Robot,” in *2019 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, pp. 146–153.
- [75] M. Presecan, F. Petric, and Z. Kovacic, “Object Classification for Child Behavior Observation in the Context of Autism Diagnostics Using a Deep Learning-based Approach,” in *2018 26th International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2018*, 2018.
- [76] D. Ribeiro, A. Mateus, P. Miraldo, and J. C. Nascimento, “A real-time Deep Learning pedestrian detector for robot navigation,” in *2017 IEEE International Conference on Autonomous Robot Systems and Competitions, ICARSC 2017*, 2017.
- [77] A. Vasquez, M. Kollmitz, A. Eitel, and W. Burgard, “Deep Detection of People and their Mobility Aids for a Hospital Robot,” in *2017 European Conference on Mobile Robots, ECMR 2017*, 2017.
- [78] S. Hoshino and K. Niimura, “Optical Flow for Real-Time Human Detection and Action Recognition Based on CNN Classifiers,” *J. Adv. Comput. Intell. Intell. Informatics*, vol. 23, no. 4, pp. 735–742, 2019.
- [79] A. Rohan, M. Rabah, and S. H. Kim, “Convolutional Neural Network-Based Real-Time Object Detection and Tracking for Parrot AR Drone 2,” *IEEE Access*, 2019.
- [80] “RoboCup.” [Online]. Available: <https://www.robocup.org>. [Accessed: 06-Dec-2019].
- [81] S. O’Keefe and R. Villing, “A benchmark data set and evaluation of deep learning architectures for ball detection in the robocup SPL,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.

- [82] M. Javadi, S. M. Azar, S. Azami, S. S. Ghidary, S. Sadeghnejad, and J. Baltes, "Humanoid robot detection using deep learning: A speed-accuracy tradeoff," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.
- [83] N. Cruz, K. Lobos-Tsunekawa, and J. Ruiz-Del-Solar, "Using convolutional neural networks in robots with limited computational resources: Detecting NAO robots while playing soccer," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.
- [84] A. T. Angonese and P. F. F. Rosa, "Integration of People Detection and Simultaneous Localization and Mapping Systems for an Autonomous Robotic Platform," in *Proceedings - 13th Latin American Robotics Symposium and 4th Brazilian Symposium on Robotics, LARS/SBR 2016*, 2016.
- [85] L. Ran, Y. Zhang, Q. Zhang, and T. Yang, "Convolutional neural network-based robot navigation using uncalibrated spherical images," *Sensors (Switzerland)*, 2017.
- [86] M. S. Bahraini, A. B. Rad, and M. Bozorg, "SLAM in dynamic environments: A deep learning approach for moving object tracking using ML-RANSAC algorithm," *Sensors (Switzerland)*, 2019.
- [87] X. Ding *et al.*, "Prior knowledge-based deep learning method for indoor object recognition and application," *Syst. Sci. Control Eng.*, 2018.
- [88] H. Sun, Z. Meng, and M. H. Ang, "Semantic mapping and semantics-boosted navigation with path creation on a mobile robot," in *2017 IEEE International Conference on Cybernetics and Intelligent Systems, CIS 2017 and IEEE Conference on Robotics, Automation and Mechatronics, RAM 2017 - Proceedings*, 2018.
- [89] D. Bersan, R. Martins, M. Campos, and E. R. Nascimento, "Semantic map augmentation for robot navigation: A learning approach based on visual and depth data," in *Proceedings - 15th Latin American Robotics Symposium, 6th Brazilian Robotics Symposium and 9th Workshop on Robotics in Education, LARS/SBR/WRE 2018*, 2018.
- [90] P. Maolanon, K. Sukvichai, N. Chayopitak, and A. Takahashi, "Indoor Room Identify and Mapping with Virtual based SLAM using Furnitures and Household Objects Relationship based on CNNs," in *10th International Conference on Information and Communication Technology for Embedded Systems, IC-ICTES 2019 - Proceedings*, 2019.
- [91] D. Chaves, J. R. Ruiz-Sarmiento, N. Petkov, and J. Gonzalez-Jimenez, "Integration of CNN into a Robotic Architecture to Build Semantic Maps of Indoor Environments," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019.

- [92] L. Wang *et al.*, "Multi-channel convolutional neural network based 3D object detection for indoor robot environmental perception," *Sensors (Switzerland)*, 2019.
- [93] W. Lin, X. Ren, J. Hu, Y. He, Z. Li, and M. Tong, "Fast, robust and accurate posture detection algorithm based on Kalman filter and SSD for AGV," *Neurocomputing*, 2018.
- [94] S. Nilwong, D. Hossain, S. I. Kaneko, and G. Capi, "Deep learning-based landmark detection for mobile robot outdoor localization," *Machines*, 2019.
- [95] M. B. Bejiga, A. Zeggada, A. Nouffidj, and F. Melgani, "A convolutional neural network approach for assisting avalanche search and rescue operations with UAV imagery," *Remote Sens.*, 2017.
- [96] L. Chen, Y. He, and L. Fan, "Let the robot tell: Describe car image with natural language via LSTM," *Pattern Recognit. Lett.*, 2017.
- [97] B. Ramalingam, A. K. Lakshmanan, M. Ilyas, A. V. Le, and M. R. Elara, "Cascaded machine-learning technique for debris classification in floor-cleaning robot application," *Appl. Sci.*, 2018.
- [98] C. Kertesz, "Clear Sky Detection with Deep Learning and an Inexpensive Infrared Camera for Robot Telescopes," in *2018 15th International Conference on Control, Automation, Robotics and Vision, ICARCV 2018*, 2018.
- [99] Y. Li, H. Li, and H. Wang, "Pixel-wise crack detection using deep local pattern predictor for robot application," *Sensors (Switzerland)*, 2018.
- [100] F. Xu *et al.*, "Real-time detecting method of marine small object with underwater robot vision," in *2018 OCEANS - MTS/IEEE Kobe Techno-Oceans, OCEANS - Kobe 2018*, 2018.
- [101] M. Milford *et al.*, "Sequence searching with deep-learnt depth for condition- and viewpoint-invariant route-based place recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2015.
- [102] E. Sizikova, I. K. Singh, B. Georgescu, M. Halber, K. Ma, and T. Chen, "Enhancing place recognition using joint intensity - Depth analysis and synthetic data," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016.
- [103] C. Park, J. Jang, L. Zhang, and J. Il Jung, "Light-weight visual place recognition using convolutional neural network for mobile robots," in *2018 IEEE International Conference on Consumer Electronics, ICCE 2018*, 2018.

- [104] T. Naseer and W. Burgard, "Deep regression for monocular camera-based 6-DoF global localization in outdoor environments," in *IEEE International Conference on Intelligent Robots and Systems*, 2017.
- [105] J. Ma and J. Zhao, "Robust Topological Navigation via Convolutional Neural Network Feature and Sharpness Measure," *IEEE Access*, 2017.
- [106] T. H. Wang, H. J. Huang, J. T. Lin, C. W. Hu, K. H. Zeng, and M. Sun, "Omnidirectional CNN for Visual Place Recognition and Navigation," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2018.
- [107] P. Ursic, R. Mandeljc, A. Leonardis, and M. Kristan, "Part-based room categorization for household service robots," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2016, vol. 2016-June, pp. 2287–2294.
- [108] M. Mancini, S. R. Bulò, E. Ricci, and B. Caputo, "Learning Deep NBNN Representations for Robust Place Categorization," *IEEE Robot. Autom. Lett.*, 2017.
- [109] R. S. Sutton and a G. Barto, "Temporal credit assignment in reinforcement learning," *Comput. Sci.*, 1984.
- [110] J. M. Pearce, *Animal Learning and Cognition: An Introduction (3rd ed.)*. 2008.
- [111] W. S. Terry, *Learning and Memory - Basic Principles, Processes, and Procedures*. 2016.
- [112] R. S. Sutton and A. G. Barto, "Reinforcement learning: an introduction.," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, p. 1054, 1998.
- [113] C. Szepesvári, "Algorithms for reinforcement learning," in *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2010.
- [114] E. Ferreira and F. Lefèvre, "Reinforcement-learning based dialogue system for human-robot interactions with socially-inspired rewards," *Comput. Speech Lang.*, 2015.
- [115] H. Modares, I. Ranatunga, F. L. Lewis, and D. O. Popa, "Optimized Assistive Human-Robot Interaction Using Reinforcement Learning," *IEEE Trans. Cybern.*, 2016.
- [116] A. Ng and S. Russell, "Algorithms for inverse reinforcement learning," *Proc. Seventeenth Int. Conf. Mach. Learn.*, 2000.
- [117] O. Saha and P. Dasgupta, "Improved reward estimation for efficient robot navigation using inverse reinforcement learning," in *2017 NASA/ESA Conference on Adaptive Hardware and Systems, AHS 2017*, 2017.

- [118] K. Shiarlis, J. Messias, and S. Whiteson, "Inverse reinforcement learning from failure," in *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 2016, pp. 1060–1068.
- [119] C. Xia and A. El Kamel, "Neural inverse reinforcement learning in autonomous navigation," *Rob. Auton. Syst.*, 2016.
- [120] B. Okal and K. O. Arras, "Learning socially normative robot navigation behaviors with Bayesian inverse reinforcement learning," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2016.
- [121] X. I. A. Chen and A. El Kamel, "A reinforcement learning method of obstacle avoidance for industrial mobile vehicles in unknown environments using neural network," in *Proceedings of the 21st International Conference on Industrial Engineering and Engineering Management 2014*, 2015, pp. 671–675.
- [122] M. Duguleana and G. Mogan, "Neural networks based reinforcement learning for mobile robots obstacle avoidance," *Expert Syst. Appl.*, 2016.
- [123] M. S. Shim and P. Li, "Biologically inspired reinforcement learning for mobile robot collision avoidance," in *Proceedings of the International Joint Conference on Neural Networks*, 2017.
- [124] L. Khriji, F. Touati, K. Benhmed, and A. Al-Yahmedi, "Mobile robot navigation based on Q-learning technique," *Int. J. Adv. Robot. Syst.*, 2011.
- [125] A. Faust *et al.*, "PRM-RL: Long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2018.
- [126] M. B. Hafez and C. K. Loo, "Topological Q-learning with internally guided exploration for mobile robot navigation," *Neural Comput. Appl.*, 2015.
- [127] N. Imanberdiyev, C. Fu, E. Kayacan, and I. M. Chen, "Autonomous navigation of UAV by using real-time model-based reinforcement learning," in *2016 14th International Conference on Control, Automation, Robotics and Vision, ICARCV 2016*, 2017.
- [128] N. Altuntas, E. Imal, N. Emanet, and C. N. Öztürk, "Reinforcement learning-based mobile robot navigation," *Turkish J. Electr. Eng. Comput. Sci.*, 2016.
- [129] A. Mahadevuni and P. Li, "Navigating mobile robots to target in near shortest time using reinforcement learning with spiking neural networks," in *Proceedings of the International Joint Conference on Neural Networks*, 2017.
- [130] C. C. E. Chewu and V. Manoj Kumar, "Autonomous navigation of a mobile robot in dynamic indoor environments using SLAM and reinforcement learning," in *IOP Conference Series: Materials Science and Engineering*, 2018.

- [131] H. X. Pham, H. M. La, D. Feil-Seifer, and L. V Nguyen, "Autonomous uav navigation using reinforcement learning," *arXiv Prepr. arXiv1801.05086*, 2018.
- [132] M. V Butz and S. W. Wilson, "An algorithmic description of XCS," in *International Workshop on Learning Classifier Systems*, 2000, pp. 253–272.
- [133] M. Roozegar, M. J. Mahjoob, M. J. Esfandyari, and M. S. Panahi, "XCS-based reinforcement learning algorithm for motion planning of a spherical mobile robot," *Appl. Intell.*, vol. 45, no. 3, pp. 736–746, 2016.
- [134] A. Adib and B. Masoumi, "Mobile robots navigation in unknown environments by using fuzzy logic and learning automata," in *2017 Artificial Intelligence and Robotics (IRANOPEN)*, 2017, pp. 58–63.
- [135] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine, "Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1–8.
- [136] B. Zuo, S. Member, J. Chen, L. Wang, and Y. Wang, "A Reinforcement Learning Based Robotic Navigation System," *Syst. Man Cybern. (SMC), 2014 IEEE Int. Conf. on. IEEE, 2014.*, pp. 3452–3457, 2014.
- [137] R. Campa, "The Rise of Social Robots : A Review of the Recent Literature," *J. Evol. Technol.*, vol. 26, no. 1–2016, pp. 106–113, 2016.
- [138] C. Mejia, "Bibliometric Analysis of Social Robotics Research: Identifying Research Trends and Knowledgebase," *Appl. Sci.*, vol. 7, no. 12, p. 1316, 2017.
- [139] T. G. Dietterich and G. Bakiri, "Solving Multiclass Learning Problems via Error-Correcting Output Codes," *Jouranal Artifical Intell. Res.*, vol. 2, pp. 263–286, 1995.
- [140] "SoftBank Robotics." [Online]. Available: <https://www.softbankrobotics.com/emea/en/company>. [Accessed: 15-Oct-2019].
- [141] O. M. Mozos, C. Stachniss, and W. Burgard, "Supervised learning of places from range data using AdaBoost," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2005, vol. 2005, pp. 1730–1735.
- [142] A. Rottmann, Ó. M. Mozos, C. Stachniss, and W. Burgard, "Semantic Place Classification of Indoor Environments with Mobile Robots using Boosting," *Proc. 20th Natl. Conf. Artif. Intell. - Vol. 3*, pp. 1306–1311, 2005.
- [143] Ó. Martínez Mozos, R. Triebel, P. Jensfelt, A. Rottmann, and W. Burgard, "Supervised semantic labeling of places using information extracted from sensor data," *Rob. Auton. Syst.*, vol. 55, no. 5, pp. 391–402, 2007.

- [144] B. Ayers and M. Boutell, "Home interior classification using SIFT keypoint histograms," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.
- [145] P. Ursic, M. Kristan, D. Skocaj, and A. Leonardis, "Room classification using a hierarchical representation of space," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 1371–1378.
- [146] A. Swadzba and S. Wachsmuth, "Indoor scene classification using combined 3d and gist features," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 6493 LNCS, no. PART 2, pp. 201–215.
- [147] O. M. Mozos, H. Mizutani, R. Kurazume, and T. Hasegawa, "Categorization of indoor places using the Kinect sensor," *Sensors (Switzerland)*, vol. 12, no. 5, pp. 6695–6711, 2012.
- [148] Z. Zivkovic, O. Booij, and B. Kröse, "From images to rooms," *Rob. Auton. Syst.*, vol. 55, no. 5, pp. 411–418, 2007.
- [149] K. M. Varadarajan and M. Vincze, "Functional Room Detection and Modeling using Stereo Imagery in Domestic Environments," in *Workshop on Semantic Perception, Mapping and Exploration at IEEE International Conference on Robotics and Automation (ICRA 2011)*, 2011.
- [150] T. Varvadoukas, E. Giannakidou, J. V. Gómez, and N. Mavridis, "Indoor furniture and room recognition for a robot using internet-derived models and object context," in *Proceedings - 10th International Conference on Frontiers of Information Technology, FIT 2012*, 2012, pp. 122–128.
- [151] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, 2009, pp. 413–420.
- [152] P. Espinace, T. Kollar, A. Soto, and N. Roy, "Indoor scene recognition through object detection," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010, pp. 1406–1413.
- [153] M. Liu *et al.*, "Scene Recognition for Indoor Localization Using a Multi-Sensor Fusion Approach," *Sensors*, vol. 17, no. 12, p. 2847, 2017.
- [154] E. Cruz, J. C. Rangel, F. Gomez-Donoso, Z. Bauer, M. Cazorla, and J. Garcia-Rodriguez, "Finding the Place: How to Train and Use Convolutional Neural Networks for a Dynamically Learning Robot," in *Proceedings of the International Joint Conference on Neural Networks*, 2018.
- [155] J. Martínez-Gómez, I. García-Varea, M. Cazorla, and V. Morell, "ViDRILo: The visual and depth robot indoor localization with objects information dataset," *Int. J. Rob. Res.*, 2015.

- [156] H. Deng, G. Stathopoulos, and C. Y. Suen, "Error-correcting output coding for the convolutional neural network for optical character recognition," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2009.
- [157] S. Yang, P. Luo, C. C. Loy, K. Shum, and X. Tang, "Deep Representation Learning with Target Coding," *Twenty-Ninth AAAI Conf. Artif. Intell.*, 2015.
- [158] M. K. Abd-Eliah, A. I. Awad, A. A. M. Khalaf, and H. F. A. Hamed, "Two-phase multi-model automatic brain tumour diagnosis system from magnetic resonance images using convolutional neural networks," *Eurasip J. Image Video Process.*, 2018.
- [159] U. O. Dorj, K. K. Lee, J. Y. Choi, and M. Lee, "The skin cancer classification using deep convolutional neural network," *Multimed. Tools Appl.*, vol. 77, no. 8, pp. 9909–9924, 2018.
- [160] K. C. . Chen C., Ren Y., "Big Visual Data Analysis Scene Classification and Geometric Labeling," Singapore: SpringerBriefs in Electrical and Computer Engineering. Springer, 2016.
- [161] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-Janua, pp. 190–198.
- [162] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-Based Localization Using LSTMs for Structured Feature Correlation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, vol. 2017-October, pp. 627–637.
- [163] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million Image Database for Scene Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.
- [164] A. Rocha and S. K. Goldenstein, "Multiclass from binary: Expanding One-versus-all, one-versus-one and ECOC-based approaches," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 25, no. 2, pp. 289–302, 2014.
- [165] M. Aly, "Survey on multiclass classification methods," *Neural Netw.*, no. November, pp. 1–9, 2005.
- [166] G. James and T. Hastie, "The error coding method and PICTs?," *J. Comput. Graph. Stat.*, vol. 7, no. 3, pp. 377–387, 1998.
- [167] F. Chollet, "Keras Documentation," *Keras.io*, 2015. [Online]. Available: <https://keras.io>.

- [168] "Compute Canada." [Online]. Available: <https://www.computecanada.ca>.
- [169] S. Hijazi, R. Kumar, and C. Rowen, "Using convolutional neural networks for image recognition," *Cadence Des. Syst. Inc. San Jose, CA, USA*, pp. 1–12, 2015.
- [170] J. D. J. Deng, W. D. W. Dong, R. Socher, L.-J. L. L.-J. Li, K. L. K. Li, and L. F.-F. L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 2–9, 2009.
- [171] "Nao Documentation." [Online]. Available: http://doc.aldebaran.com/2-1/home_nao.html. [Accessed: 01-Dec-2019].
- [172] K. M. Othman and A. B. Rad, "An Indoor Room Classification System for Social Robots via Integration of CNN and ECOC," *Appl. Sci.*, vol. 9, no. 3, p. 470, Jan. 2019.
- [173] J. R. Ruiz-Sarmiento, C. Galindo, and J. Gonzalez-Jimenez, "Robot@Home, a robotic dataset for semantic mapping of home environments," *Int. J. Rob. Res.*, 2017.
- [174] M. Reza Loghmani, B. Caputo, and M. Vincze, "Recognizing objects in-the-wild: Where do we stand?," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2018.
- [175] R. Sahdev and J. K. Tsotsos, "Indoor place recognition system for localization of mobile robots," in *Proceedings - 2016 13th Conference on Computer and Robot Vision, CRV 2016*, 2016.
- [176] K. Mo, H. Li, Z. Lin, and J.-Y. Lee, "The AdobelIndoorNav Dataset: Towards Deep Reinforcement Learning based Real-world Indoor Robot Visual Navigation," Feb. 2018.
- [177] "Qrio, the robot that could," *IEEE Spectr.*, vol. 41, no. 5, pp. 34–37.
- [178] "Qrio." [Online]. Available: <https://robots.ieee.org/robots/qrio/>.
- [179] D. Hanson *et al.*, "Zeno: A cognitive character," in *AAAI Workshop - Technical Report*, 2008.
- [180] "Zeno research robot." [Online]. Available: <https://www.hansonrobotics.com/zeno/>.
- [181] "Meet Manav, India's first 3D-printed humanoid robot." [Online]. Available: <https://www.livemint.com/Industry/rc86lu7h3rb44087oDts1H/Meet-Manav-Indias-first-3Dprinted-humanoid-robot.html>.

- [182] I. Ha, Y. Tamura, H. Asama, J. Han, and D. W. Hong, "Development of open humanoid platform DARwIn-OP," in *Proceedings of the SICE Annual Conference*, 2011.
- [183] "DARwIn OP: Open Platform Humanoid Robot for Research and Education." [Online]. Available: <http://www.romela.org/darwin-op-open-platform-humanoid-robot-for-research-and-education/>.
- [184] "HOVIS Guide." [Online]. Available: http://hovis.co.kr/guide/hovis_eco_eng.html.
- [185] A. Nikkhah, A. Yousefi-Koma, R. Mirjalili, and H. M. Farimani, "Design and Implementation of Small-sized 3D Printed Surena-Mini Humanoid Platform," in *5th RSI International Conference on Robotics and Mechatronics, IcRoM 2017*, 2018.
- [186] M. Lapeyre *et al.*, "Poppy Project: Open-Source Fabrication of 3D Printed Humanoid Robot for Science, Education and Art," *Digital Intelligence 2014*. 2014.
- [187] D. Anguelov, D. Koller, E. Parker, and S. Thrun, "Detecting and modeling doors with mobile robots," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2004.
- [188] J.-S. Lee, N. L. Doh, W. K. Chung, B.-J. You, and Y. Il Youm, "Door Detection Algorithm of Mobile Robot in Hallway Using PC-Camera," in *Proceedings of the 21st International Symposium on Automation and Robotics in Construction*, 2017.
- [189] Y. Tian, X. Yang, and A. Arditi, "Computer vision-based door detection for accessibility of unfamiliar environments to blind persons," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2010.
- [190] M. Derry and B. Argall, "Automated doorway detection for assistive shared-control wheelchairs," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2013.
- [191] B. Kakillioglu, K. Ozcan, and S. Velipasalar, "Doorway detection for autonomous indoor navigation of unmanned vehicles," in *Proceedings - International Conference on Image Processing, ICIP*, 2016.
- [192] C. Fernández-Caramés, V. Moreno, B. Curto, J. F. Rodríguez-Aragón, and F. J. Serrano, "A Real-time Door Detection System for Domestic Robotic Navigation," *J. Intell. Robot. Syst. Theory Appl.*, pp. 119–136, 2014.
- [193] J. Hensler, M. Blaich, and O. Bittel, "Real-time door detection based on AdaBoost learning algorithm," in *Communications in Computer and Information Science*, 2010.

- [194] W. Chen, T. Qu, Y. Zhou, K. Weng, G. Wang, and G. Fu, "Door recognition and deep learning algorithm for visual based robot navigation," in *2014 IEEE International Conference on Robotics and Biomimetics, IEEE ROBIO 2014*, 2014.
- [195] R. Jin, B. Andonovski, Z. Tu, J. Wang, J. Yuan, and D. M. Tham, "A framework based on deep learning and mathematical morphology for cabin door detection in an automated aerobridge docking system," in *2017 Asian Control Conference, ASCC 2017*, 2018.
- [196] H. Zhang, L. Dou, H. Fang, and J. Chen, "Autonomous indoor exploration of mobile robots based on door-guidance and improved dynamic window approach," in *2009 IEEE International Conference on Robotics and Biomimetics, ROBIO 2009*, 2009.
- [197] W. Meeussen *et al.*, "Autonomous door opening and plugging in with a personal robot," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010.
- [198] M. Nieuwenhuisen, J. Stückler, and S. Behnke, "Improving indoor navigation of autonomous robots by an explicit representation of doors," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010.
- [199] A. Goil, M. Derry, and B. D. Argall, "Using machine learning to blend human and robot controls for assisted wheelchair navigation," in *IEEE International Conference on Rehabilitation Robotics*, 2013.
- [200] D. Dai *et al.*, "Detecting, locating and crossing a door for a wide indoor surveillance robot," in *2013 IEEE International Conference on Robotics and Biomimetics, ROBIO 2013*, 2013.
- [201] F. Pasteau, V. K. Narayanan, M. Babel, and F. Chaumette, "A visual servoing approach for autonomous corridor following and doorway passing in a wheelchair," *Rob. Auton. Syst.*, 2016.
- [202] A. Criminisi, I. Reid, and A. Zisserman, "Single view metrology," *Int. J. Comput. Vis.*, 2000.
- [203] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Reconstructing building interiors from images," 2010.
- [204] S. Fuhrmann, F. Langguth, and M. Goesele, "MVE – A Multi-View Reconstruction Environment," in *Eurographics Workshop on Graphics and Cultural Heritage*, 2014.
- [205] S. Fuhrmann, F. Langguth, N. Moehrle, M. Waechter, and M. Goesele, "MVE - An image-based reconstruction environment," *Comput. Graph.*, 2015.

- [206] A. Saxena, S. H. Chung, and A. Y. Ng, "3-D depth reconstruction from a single still image," *Int. J. Comput. Vis.*, 2008.
- [207] I. Alhashim and P. Wonka, "High Quality Monocular Depth Estimation via Transfer Learning," Dec. 2018.
- [208] T. Yu, J.-H. Zou, and Q.-B. Song, "3D Reconstruction from a Single Still Image Based on Monocular Vision of an Uncalibrated Camera," in *ITM Web of Conferences*, 2017, vol. 12, p. 1018.
- [209] V. Aslantas, "A depth estimation algorithm with a single image," *Opt. Express*, vol. 15, no. 8, pp. 5024–5029, 2007.
- [210] C. Tang, C. Hou, and Z. Song, "Depth recovery and refinement from a single image using defocus cues," *J. Mod. Opt.*, 2015.
- [211] M. T. Khanna, K. Rai, S. Chaudhury, and B. Lall, "Perceptual depth preserving saliency based image compression," in *ACM International Conference Proceeding Series*, 2015.
- [212] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor Segmentation and Support Inference from RGBD Images," in *Computer Vision -- ECCV 2012*, 2012, pp. 746–760.
- [213] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Rob. Res.*, 2013.
- [214] B. Yamauchi, "Frontier-based approach for autonomous exploration," in *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA*, 1997.
- [215] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *IEEE International Conference on Intelligent Robots and Systems*, 2017.
- [216] P. Whaite and F. P. Ferrie, "Autonomous exploration: Driven by uncertainty," *IEEE Trans. Pattern Anal. Mach. Intell.*, 1997.
- [217] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using rao-blackwellized particle filters," in *Robotics: Science and Systems*, 2005.
- [218] M. G. Jadidi, J. V. Miro, and G. Dissanayake, "Mutual information-based exploration on continuous occupancy maps," in *IEEE International Conference on Intelligent Robots and Systems*, 2015.
- [219] S. Bai, F. Chen, and B. Englot, "Toward autonomous mapping and exploration for mobile robots through deep supervised learning," in *IEEE International Conference on Intelligent Robots and Systems*, 2017.

- [220] R. Shrestha, F. P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned map prediction for enhanced mobile robot exploration," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2019.
- [221] J. Aulinas, Y. Petillot, J. Salvi, and X. Lladó, "The SLAM problem: A survey," in *Frontiers in Artificial Intelligence and Applications*, 2008.
- [222] T. S. Ho, Y. C. Fai, and E. S. L. Ming, "Simultaneous localization and mapping survey based on filtering techniques," in *2015 10th Asian Control Conference: Emerging Control Techniques for a Sustainable World, ASCC 2015*, 2015.
- [223] M. J. Milford, G. F. Wyeth, and D. Prasser, "RatSLAM: A hippocampal model for simultaneous localization and mapping," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2004.
- [224] M. Milford and G. Wyeth, "Persistent navigation and mapping using a biologically inspired slam system," *Int. J. Rob. Res.*, 2010.
- [225] J. Steckel and H. Peremans, "BatSLAM: Simultaneous Localization and Mapping Using Biomimetic Sonar," *PLoS One*, 2013.
- [226] S. J. Shettleworth, "Animal cognition and animal behaviour," *Animal Behaviour*. 2001.
- [227] R. C. Arkin, "Integrating behavioral, perceptual, and world knowledge in reactive navigation," *Rob. Auton. Syst.*, 1990.
- [228] O. Khatib, "REAL-TIME OBSTACLE AVOIDANCE FOR MANIPULATORS AND MOBILE ROBOTS.," *Int. J. Rob. Res.*, 1986.
- [229] E. Aguirre and A. González, "Fuzzy behaviors for mobile robot navigation: Design, coordination and fusion," *Int. J. Approx. Reason.*, 2000.
- [230] Y. Zhu, T. Zhang, J. Song, and X. Li, "A new hybrid navigation algorithm for mobile robots in environments with incomplete knowledge," *Knowledge-Based Syst.*, 2012.
- [231] S. Karim, L. Sonenberg, and A. H. Tan, "A hybrid architecture combining reactive plan execution and reactive learning," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2006.
- [232] A. H. Tan, Y. S. Ong, and A. Tapanuj, "A hybrid agent architecture integrating desire, intention and reinforcement learning," *Expert Syst. Appl.*, 2011.
- [233] M. J. Mataric, "Integration of representation into goal-driven behavior-based robots," *IEEE Trans. Robot. Autom.*, vol. 8, no. 3, pp. 304–312, 1992.

- [234] C. Urdiales, E. J. Perez, J. Vázquez-Salceda, M. Sánchez-Marrè, and F. Sandoval, "A purely reactive navigation scheme for dynamic environments using Case-Based Reasoning," *Auton. Robots*, 2006.
- [235] V. F. Da Silva, A. H. Selvatici, and A. H. R. Costa, "Navigation towards a goal position: From reactive to generalised learned control," in *Journal of Physics: Conference Series*, 2011.
- [236] Z. Hendzel and M. Szuster, "Neural dynamic programming in reactive navigation of wheeled mobile robot," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012.
- [237] A. Ataka, H. K. Lam, and K. Althoefer, "Reactive Magnetic-Field-Inspired navigation method for robots in unknown convex 3-D environments," *IEEE Robot. Autom. Lett.*, 2018.
- [238] G. De Cubber, S. A. Berrabah, D. Doroftei, Y. Baudoin, and H. Sahli, "Combining dense structure from motion and visual SLAM in a behavior-based robot control architecture," *Int. J. Adv. Robot. Syst.*, 2010.
- [239] D. Gómez-Anaya, R. Mungu\`ia, E. Guerra, and A. Grau, "Full autonomous navigation for an aerial robot using behavior-based control motion and SLAM," in *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, 2014, pp. 1–6.
- [240] S. Jiang and R. C. Arkin, "SLAM-Based Spatial Memory for Behavior-Based Robots," *IFAC-PapersOnLine*, 2015.
- [241] J. Boal, Á. Sánchez-Miralles, and Á. Arranz, "Topological simultaneous localization and mapping: A survey," *Robotica*, 2014.
- [242] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, 1986.
- [243] J. Illingworth and J. Kittler, "A survey of the hough transform," *Comput. Vision, Graph. Image Process.*, 1988.
- [244] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2014.
- [245] P. Simoens, M. Dragone, and A. Saffiotti, "The Internet of Robotic Things: A review of concept, added value and applications," *Int. J. Adv. Robot. Syst.*, 2018.
- [246] Worldometer, "COVID-19 CORONAVIRUS PANDEMIC." [Online]. Available: <https://www.worldometers.info/coronavirus/>. [Accessed: 23-May-2020].

- [247] R. Gelin, "NAO," in *Humanoid Robotics: A Reference*, A. Goswami and P. Vadakkepat, Eds. Dordrecht: Springer Netherlands, 2019, pp. 147–168.
- [248] "ALDEBARAN Robotics equips 200 schools with NAO humanoid robot." [Online]. Available: <https://www.expo21xx.com/news/aldebaran-robotics-humanio-robots/>. [Accessed: 09-Nov-2019].
- [249] L. Kleeman and R. Kuc, "Sonar Sensing," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Cham: Springer International Publishing, 2016, pp. 753–782.
- [250] S. Sivaraman and M. M. Trivedi, "Combining monocular and stereo-vision for real-time vehicle ranging and tracking on multilane highways," in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2011.
- [251] K. M. Othman, "Implementation of EKF-SLAM on NAO Humanoid Robot," Simon Fraser University, 2013.
- [252] "NAOqi enabled controller for simulated NAO robots in Webots." [Online]. Available: <https://github.com/cyberbotics/naoqisim>. [Accessed: 10-Nov-2019].
- [253] Š. Fojtů, M. Havlena, and T. Pajdla, "Nao robot localization and navigation using fusion of odometry and visual sensor data," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012.
- [254] C. Wei, J. Xu, C. Wang, P. Wiggers, and K. Hindriks, "An approach to navigation for the humanoid robot Nao in domestic environments," *Toward Auton. Robot. Syst.*, vol. 8069 LNAI, no. Springer Berlin Heidelberg, pp. 298–310, 2014.
- [255] L. George and A. Mazel, "Humanoid robot indoor navigation based on 2D bar codes: Application to the NAO robot," in *IEEE-RAS International Conference on Humanoid Robots*, 2015.
- [256] J. D. Chacón, "Visual-Topological Mapping: An approach for indoor robot navigation," Master Thesis. Univ. of Groningen, 2011.
- [257] J. Xu, R. Wang, S. Yue, and L. C. Kiong, "A Cognitive Model for Humanoid Robot Navigation and Mapping using Alderbaran NAO," *arXiv Prepr. arXiv1405.3095*, 2014.
- [258] J. Delgado-Galvan, A. Navarro-Ramirez, J. Nunez-Varela, C. Puente-Montejano, and F. Martinez-Perez, "Vision-based humanoid robot navigation in a featureless environment," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015.

- [259] S. Wen, K. M. Othman, A. B. Rad, Y. Zhang, and Y. Zhao, "Indoor SLAM using laser and camera with closed-loop controller for NAO humanoid robot," *Abstr. Appl. Anal.*, vol. 2014, 2014.
- [260] S. Müller, C. Weber, and S. Wermter, "RatSLAM on humanoids - A bio-inspired SLAM model adapted to a humanoid robot," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014.
- [261] A. K. Rath, D. R. Parhi, H. C. Das, M. K. Muni, and P. B. Kumar, "Analysis and use of fuzzy intelligent technique for navigation of humanoid robot in obstacle prone zone," *Def. Technol.*, 2018.
- [262] O. Melinte, L. Vladareanu, and I.-A. Gal, "NAO robot fuzzy obstacle avoidance in virtual environment," *Period. Eng. Nat. Sci.*, vol. 7, no. 1, pp. 318–323, 2019.
- [263] Z. Wang, X. Wen, Y. Song, X. Mao, W. Li, and G. Chen, "Navigation of a humanoid robot via head gestures based on global and local live videos on Google Glass," in *I2MTC 2017 - 2017 IEEE International Instrumentation and Measurement Technology Conference, Proceedings*, 2017.
- [264] I. M. Ariffin, A. I. H. M. Rasidi, H. Yussof, M. A. Miskam, and A. R. Omar, "Vision tracking application for mobile navigation using Humanoid robot Nao," in *2015 International Symposium on Micro-NanoMechatronics and Human Science, MHS 2015*, 2016.
- [265] A. Gardecki, M. Podpora, and A. Kawala-Janik, "Implementation of an External Laser Scanner into Control System of the NAO Robot," *IFAC-PapersOnLine*, vol. 51, no. 6, pp. 231–237, 2018.
- [266] J. Figat and W. Kasprzak, "NAO-mark vs QR-code recognition by NAO robot vision," *Adv. Intell. Syst. Comput.*, 2015.
- [267] M. Alam, L. Vidyaratne, T. Wash, and K. M. Iftekharruddin, "Deep SRN for robust object recognition: A case study with NAO humanoid robot," in *Conference Proceedings - IEEE SOUTHEASTCON*, 2016.
- [268] H. Zhu, H. Yi, R. Chellali, and L. Feng, "Object Recognition and Localization Algorithm base on NAO Robot," in *RO-MAN 2018 - 27th IEEE International Symposium on Robot and Human Interactive Communication*, 2018.
- [269] L. Xie and C. Deng, "Object Detection of NAO Robot Based on a Spectrum Model," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.
- [270] N. Kofinas, E. Orfanoudakis, and M. G. Lagoudakis, "Complete analytical inverse kinematics for NAO," in *Proceedings of the 2013 13th International Conference on Autonomous Robot Systems, ROBOTICA 2013*, 2013.

- [271] Y. Jiang, F. Chang, and F. Liang, "Kinematics modeling and trajectory planning for NAO robot object grasping based on image analysis," in *Proceedings - 2017 Chinese Automation Congress, CAC 2017*, 2017.
- [272] O. Tutsoy, D. Erol Barkana, and S. Colak, "Learning to balance an NAO robot using reinforcement learning with symbolic inverse kinematic," *Trans. Inst. Meas. Control*, 2017.
- [273] D. Albani, A. Youssef, V. Suriani, D. Nardi, and D. D. Bloisi, "A deep learning approach for object recognition with NAO soccer robots," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2017.
- [274] M. A. Fahami, M. Roshanzamir, and N. H. Izadi, "A reinforcement learning approach to score goals in RoboCup 3D soccer simulation for nao humanoid robot," in *2017 7th International Conference on Computer and Knowledge Engineering, ICCKE 2017*, 2017.
- [275] T. Belpaeme *et al.*, "Multimodal Child-Robot Interaction: Building Social Bonds," *J. Human-Robot Interact.*, 2013.
- [276] S. Shamsuddin *et al.*, "Initial response of autistic children in human-robot interaction therapy with humanoid robot NAO," in *Proceedings - 2012 IEEE 8th International Colloquium on Signal Processing and Its Applications, CSPA 2012*, 2012.
- [277] J. P. M. Vital, M. S. Couceiro, N. M. M. Rodrigues, C. M. Figueiredo, and N. M. F. Ferreira, "Fostering the NAO platform as an elderly care robot," in *SeGAH 2013 - IEEE 2nd International Conference on Serious Games and Applications for Health, Book of Proceedings*, 2013.
- [278] J. Han, N. Campbell, K. Jokinen, and G. Wilcock, "Investigating the use of Non-verbal Cues in Human-Robot Interaction with a Nao robot," in *3rd IEEE International Conference on Cognitive Infocommunications, CogInfoCom 2012 - Proceedings*, 2012.
- [279] I. Cohen, R. Looije, and M. A. Neerincx, "Child's Perception of Robot's Emotions: Effects of Platform, Context and Experience," *Int. J. Soc. Robot.*, 2014.
- [280] K. Jokinen and G. Wilcock, "Multimodal Open-Domain Conversations with the Nao Robot," in *Natural Interaction with Robots, Knowbots and Smartphones*, 2014.
- [281] R. Andreasson, B. Alenljung, E. Billing, and R. Lowe, "Affective Touch in Human-Robot Interaction: Conveying Emotion to the Nao Robot," *Int. J. Soc. Robot.*, 2018.

- [282] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016.
- [283] J. Redmon and A. Farhadi, "YOLO v.3," *Tech Rep.*, 2018.
- [284] G. A. Bekey, *Autonomous robots : from biological inspiration to implementation and control*. 2005.

Appendix.

Platform & Simulator

Nao is a humanoid robot that was designed by a French robotic company called Aldebaran in 2005 [247]. It is a human-like, a medium-sized and a fully programmable bipedal robot that attracted the robotics community for education and research. In 2006, Nao had been selected for the Robocup competition replacing AIBO robot, the dog-shape. In 2008, Robocup received 80 Nao platforms for the competition. Nao has spread rapidly. For instance, it is used as a research platform in more than 450 universities. In addition, over 200 secondary schools worldwide were equipped with Nao robots in 2012 [248]. In 2015, the Aldebaran company was owned by Softbank and renamed as Softbank Robotics. There are many versions that have been improved through the years until launching the 6th Version in 2018 [17]. In this chapter, a brief overview of Nao will be introduced by focusing on the hardware, software, adopted simulator and Nao research area.

Nao Humanoid Robot – Hardware

Nao is a human-like, a medium-sized with a height of 58 cm, and a fully programmable to perform many tasks autonomously. The adopted version for validating this work is Nao V4 (H25), i.e. twenty-five degrees of freedom (D.O.F). Twenty-five D.O.F means that the robot consists of 25 different motors for controlling different actuators. They are distributed into two legs, two hands with arms, pelvis and head as shown in Figure 1.

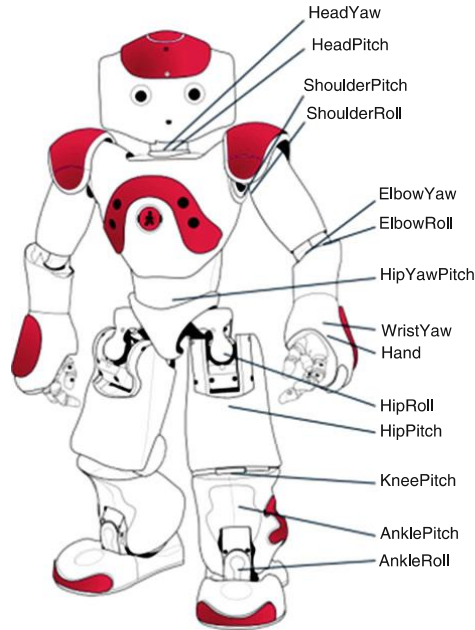


Figure 1: Nao's 25 actuators [247].

It is equipped with many proprioceptive and exteroceptive sensors. Proprioceptive sensors acquire internal information of the robot, while exteroceptive sensors acquire external information of the environment. Figure 2 shows different types of sensors in the standard version of Nao. The detailed specification of two sonar and 2D cameras will be explained since these are the employed sensors in our projects.

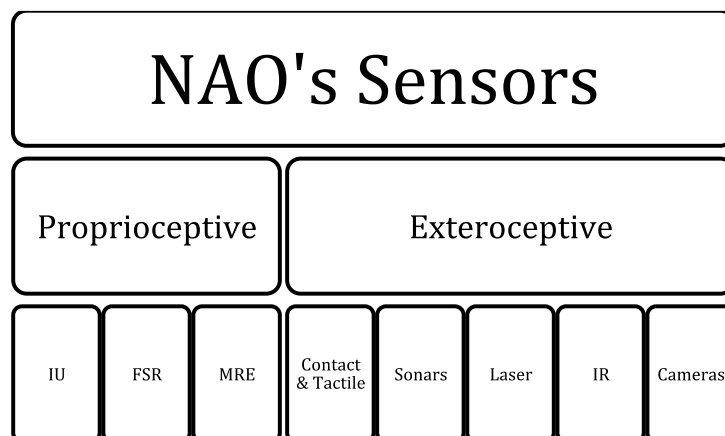


Figure 2: Nao main sensors.

Sonar:

Sonar or *ultrasonic* sensor is one of the most common sensors in robotics. It operates by emitting a cone of ultrasound to an obstacle and receiving a reflected echo signal from that obstacle in order to measure the distance to it. This distance is computed by using time-of-flight (TOF) of the sound signal and the known speed of sound, therefore sonars are regarded as a time-of-flight ranging sensor. Ultrasonic sensors are employed in robotics for three different objectives: obstacle avoidance, sonar mapping and object recognition based on a sequence of echoes [249]. Even though sonars are regarded as a popular sensor in robotic due to its low cost, low power consumption, low computational effort and its light-weight, it still has many limitations and disadvantages [249]. One of these limitations is that sonars sometimes do not receive a reflected sound from an uneven surface, which causes a deflection of the signal. Another limitation is the multiple reflections, which means that the reflected echo hits another obstacle and then is received late by sonars to measure longer and incorrect distances. In addition, objects with soft material, e.g. pillows, absorb the signal from sonars; thus, sonars are not able to detect them [249].

Nao robot consists of two sonars, in which each has an emitter and receiver that are located in the chest of Nao. They are able to detect and measure the distance to obstacles in meters within the ultrasound range, which is the range of 0.25 m to 2.55 m and an effective cone 60° . The existence of two sonars gives a capability for Nao robot to detect obstacles in a wider range instead of using one sonar with the same range of the two sonars that may lead to poor information.

Camera:

The vision system in robotics is the most powerful sense, as it has been developed to mimic human eyes in many robotic applications. It gives users a sufficient amount of data about the environment and allows an intelligent interaction in a dynamic environment. The vision system in robotics can be classified into stereo-based and monocular-based techniques. Stereo-based techniques acquire a clear computation of depth of objects in a 3D environment by overlapping images from two cameras, but they require a high cost at runtime. Monocular-based techniques, in contrast, use only one camera to deal with 2D images, which means it cannot provide immediate information of

depth. Nonetheless, Monocular-based techniques have advantages of low cost at runtime [250].

The vision system of the Nao humanoid robot consists of two identical cameras. The first one is mounted on Nao's forehead in order to scan the horizon, while the second one is mounted on the Nao's mouth in order to scan close entities on the floor, see Figure 5-3 in Chapter 5. Therefore, the vision system on Nao is considered a monocular-based system although it has two cameras since there is no data that can overlap between them. The two cameras give a *640x480* resolution at *30* frames per second (fps).

Nao Humanoid Robot – Software

Nao is a fully programmable humanoid robot. It is designed to be accessible when used by programmers at various levels from unskilled to experts, and to be stress-free for learning the robotics fundamentals by students at different levels both at schools and universities. To develop functions that control the robot properly, the user needs to know and understand all compatible software with the Nao humanoid robot, and how they are correlated to each other. Nao consists of three related software: in-robot software, out-robot software and programming tools. In-robot software refers to software that runs in the robot's motherboard to accomplish autonomous behaviors. The main software that stores all modules is called Naoqi, which runs under the robot operating system called OpenNao. On the other hand, out-robot software means that all well-matched software with Nao installed in computers, such as Aldebaran's software (Choregraphe & Monitor) or third-party software (e.g. Webots). Finally, programming tools point to the ability of users to create their own code, which is achievable by coding in Choregraphe and/or using available SDKs (Software Development Kits) in at least 8 languages with the identical API's modules. See Figure 3.

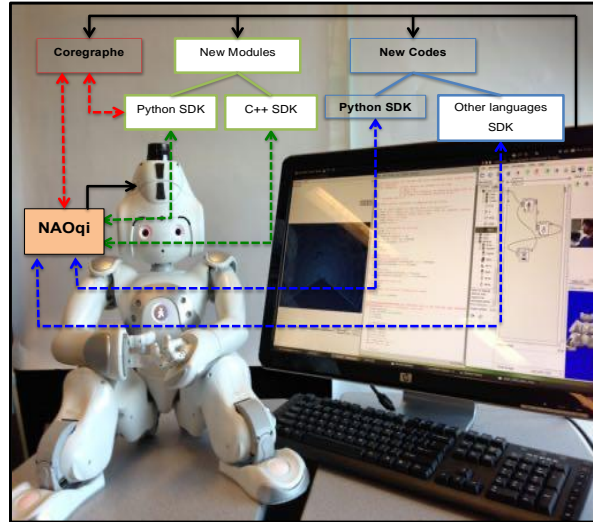


Figure 3: Software interaction within Nao robot [251].

There are many preprogrammed modules with their functions within Naoqi API that were useful for the practical experiments in our project. Table 1 shows the predefined functions used within the python programs of our work.

Table 1: Useful predefined functions from the Naoqi API for our project.

Objectives	Modules	Functions	Descriptions
Motion	ALMotion	wakeup()	Nao wakes up and its motors set on.
		setStiffnesses()	Sets the stiffness of Nao's joints.
		moveTo()	Nao moves to the given translation and direction.
	ALRobotPosture	goToPosture()	Nao goes to the predefined posture.
Audio	ALTextToSpeech	say()	Nao says the specified string.
Vision and Sensor	ALVideoDevice	getImagesRemote()	Retrieves the latest images from the video source.
	ALPhotoCapture	takePicture()	Take one picture with a given resolution.
	ALVideoDevice & ALSonar	subscribe()	Launching sensors in a hardware level.
Core	ALMemory	getData()	Retrieving data.

Webots Simulator

Since using simulators in robotics is very important for developers to test their work safely before any real-time experiments, we adopted Webots for testing our work. Webots is a development environment for robotics that is led by Cyberbotics company and became an open-source simulator in 2018 [14]. Several types of robotic platforms and sensors can be simulated in Webots. The main reason for adopting Webots over other robotics simulators is that during the time we have started our work with Nao, i.e. when it was owned by Aldebaran, the two companies cooperated to release a special light version of Webots called “Webots for Nao”. This version supported the Naoqi simulator, in which the user using Naoqi API instead of Webots APIs while testing the program within Webots virtual environments. Unfortunately, this cooperation was over in the last few years. In order to avoid the big changes in our codes, we addressed this problem by interfacing Webots with the Naoqi simulator within Linux operating system adopting the work in [252]. There is another reason for sticking with Webots is that we created a new virtual environment model for Nao within an apartment as shown in Figure 6-13 in Chapter 6. This model was launched for the public in Webots version R2019 b.

Nao’s Research Areas

The Nao robot is one of the humanoid robots in the market that has sufficient sensors and actuators with a relatively low-cost. For that reason, it has been spread out in the world, in which there are nearly 13,000 robots since 2006 have been used in different sectors, especially in research and education [140]. There are several research areas that adopted a Nao robot for virtual and real experiments. Mobile robot navigation is one of the central problems in robotics research, in which Nao has been employed for addressing different subproblems of navigation. The odometry and single-camera of Nao have been used to address the localization problem of navigation [253], [254], whereas the same problem has been solved by detecting a 2D bar code [255]. Mapping is another problem of robotic navigation that addressed by Nao through the nearest neighbour algorithm [256], cognitive model [257] and image processing for extracting and distinguishing between walls and floors [258]. Furthermore, localization and mapping have been addressed simultaneously, i.e. it is called SLAM, and applied on Nao by adopting a probabilistic approach [251], [259], or a bio-inspired approach [260].

Another basic navigation task is an obstacle avoidance that considered for Nao by designing a fuzzy logic controller [261], [262]. Integrating Nao with other platforms or devices and fuse their sensors is another approach of addressing the navigation problem, such as integrating Google-Glass [263], a car-like platform [264], or an external laser scanner [265]. The other crucial research area is object recognition using Nao's camera. Recognizing a QR code or Nao Mark is an example of this research area that can be used for specific task detection such as open-door detection [266]. The other example is the face recognition as shown in [267] that applied simultaneous recurrent network for Nao. Object detection and localization have been addressed on Nao in [268] through an image processing method and a monocular vision ranging method. Whereas the spectrum segmentation algorithm was applied for object detection based on color in [269]. Studying Nao kinematics is an important research area especially for object grasping [270]–[272]. Many researchers have been focusing on adopting the beforementioned approaches specifically for the robotic soccer competition in the RoboCup [80], such as [273], [274].

Interestingly, education is the largest market for Nao as it is a really good platform that attracts students in order to teach robotics or other science areas. Therefore, the area of Human-Robotic Interaction HRI is one of the important fields for Nao applications, thanks to the look of Nao. One of the common applications is the interaction with children having a disease such as diabetes [275] or autism [276]. Another application of interaction is that Nao assists and accompanies elder people. [277] suggested an architecture that integrates Nao with other health sensors that measure needed body parameters. There are also different forms of interaction with a human, such as a non-verbal communication [278], recognizing emotions while interacting [279], spoken interaction [280] or tactile communication for emotion recognition [281]. Since Nao might be the first robot to enter into our houses [247], our project focused on implementing the proposed behavioristic system for navigation on the Nao robot.