OBSTACLE AVOIDANCE CONTROL FOR A HUMAN-OPERATED MOBILE ROBOT

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

YASER MOHAMADI NAZARABAD

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

NOVEMBER 2019

Approval of the thesis:

OBSTACLE AVOIDANCE CONTROL FOR A HUMAN-OPERATED MOBILE ROBOT

submitted by **YASER MOHAMADI NAZARABAD** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. M. A. Sahir Arıkan
Head of Department, **Mechanical Engineering**

Prof. Dr. Erhan İlhan Konukseven
Supervisor, **Mechanical Engineering, METU**

Assist. Prof. Dr. A. Buğra Koku
Co-Supervisor, **Mechanical Engineering, METU**

Examining Committee Members:

Assist. Prof. Dr. Ali Emre Turgut
Mechanical Engineering, METU

Prof. Dr. Erhan İlhan Konukseven
Mechanical Engineering, METU

Assist. Prof. Dr. A. Buğra Koku
Mechanical Engineering, METU

Prof. Dr. Y. Samim Ünlüsoy
Mechanical Engineering, METU

Assist. Prof. Dr. Andaç Töre Şamiloğlu
Mechanical Engineering, Başkent University

Date: 29.11.2019

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Yaser Mohamadi Nazarabad

Signature:

ABSTRACT

OBSTACLE AVOIDANCE CONTROL FOR A HUMAN-OPERATED MOBILE ROBOT

Mohamadi Nazarabad, Yaser
Master of Science, Mechanical Engineering
Supervisor: Prof. Dr. Erhan İlhan Konukseven
Co-Supervisor: Assist. Prof. Dr. A. Buğra Koku

November 2019, 117 pages

In this study, the collaboration between human operator and a wheeled mobile robot in obstacle avoidance scenario is addressed. The tele-operation task is completed by integrating a force-feedback joystick to the human-robot system. The force-feedback joystick is able to apply forces on human operator and establish a bi-directional communication interface between the operator and the robot. Depending on levels of autonomy assigned to the robot, the operator and the robot are assigned with different roles during the navigation to a desired position in an unknown environment. A behavior-based control structure is used to formulate different cooperative control schemes by which the data provided by both the operator and the robot sensory channels are used to generate the optimal control inputs, resulting a safe travel to the goal with a minimum possible number of collisions and in the shortest possible time. The results of the study reveal that compared with manual and autonomous drive modes, cooperative control schemes make significant improvements in the overall performance of the system.

Keywords: Mobile Robot, Obstacle Avoidance, Human-Robot Interaction, Tele-operation, Force Feedback

ÖZ

İNSAN ETKİLEŞİMLİ BİR MOBİL ROBOTUN ENGELDEN KAÇINMA KONTROLÜ

Mohamadi Nazarabad, Yaser
Yüksek Lisans, Makina Mühendisliği
Tez Danışmanı: Prof. Dr. Erhan İlhan Konukseven
Ortak Tez Danışmanı: Dr. Öğr. Üyesi A. Buğra Koku

Kasım 2019, 117 sayfa

Bu çalışmada engelden kaçınma senaryosu kapsamında operatör ve tekerlekli robot arasındaki uzaktan etkileşim incelenmiştir. Uzaktan etkileşim, operatöre geri besleme kuvvet uygulayabilen joystiğin sisteme dahil edilmesiyle gerçekleştirilmiştir. Operatöre uygulanan geri beslemeli kuvvet sayesinde insan-robot arasında çift-taraflı bir iletişim ara yüzü kurulmuştur. Robot için belirlenen otonomi seviyesine bağlı olarak, operatör ve robot bilinmeyen bir bölgede konumlandırılmış bir hedefe yaklaşması esnasında farklı görevler üstlenmiştir. Bu bağlamda davranış tabanlı bir kontrol yapısı kullanılarak, çeşitli işbirliği kontrol yöntemleri geliştirilmiştir. Robotun belirlenen hedefe en kısa sürede ve en az çarpma ile gidebilmesi için, operatörden ve robotun algılayıcılarından gelen veriler işlenerek optimum kontrol girdileri oluşturulmuştur. Bu çalışmanın sonuçları, önerilen işbirliği kontrol mekanizmalarının, manuel ve otomatik sürüş modlarına göre sistemin genel performansı üzerinde daha belirgin iyileştirmelere yol açtığını göstermektedir.

Anahtar Kelimeler: Mobil Robot, Engelden Kaçınma, İnsan-Robot Etkileşimi, Uzaktan Kontrol, Kuvvet Geri Besleme

To my beloved sister, Tayyebeh, who was more like a mother to me than a sister,
who taught me to never give up even if all the odds are against me.

ACKNOWLEDGEMENTS

First of all, I would like to express my gratitude to Professor İlhan Konukseven for pushing me to my limits to get the best of my potential and Dr. Buğra Koku for being a friend more than just a mentor to me. Only thanks to the guidance of my advisors, I was able to shape my knowledge on my academic field.

Secondly, I want to thank Zafer Kılıç, Kemal Açıkgöz, Yashar Badienia and Musab Çağrı Uğurlu, and other members of Mechatronics labrotary at Middle East Technical University for their valuable technical support and sincere friendship.

I would also like to thank Timex Filtration and Water System staff, especially Mr. Koray Torun for providing me the opportunity to develop my career alongside with pursuing my academic goals.

It is also certain that completion of this thesis would not be possible without the participation of volunteers in the test stage and time they devoted to my research.

I am deeply grateful to my family, particularly my mother, Khadijeh Zarghami, for her unconditional love and my brothers Naser and Feridon Mohamadi and my sister, Leila Mohamadi, for their continuous encouragement. I also want to thank my nieces and nephews Amirreza, Amirhosein, Alisan and Nuray Mohamadi, Mortaza Kazemzadeh, Masoud and Sara Mirzaei for being a great source of motivation.

Finally, I would like to thank Arsalan Gholizadeh, Sanam Dehghan, Merve Uludağ, Seray Şeker, Salar Ahmadi and especially Khorrameh Hassanzadeh for not leaving me alone in my most difficult times. Thanks to these few companions, I managed to stay focused and motivated through the many years of my master’s studies.

TABLE OF CONTENTS

CHAPTERS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

Application Programming Interface (API)

Artificial Intelligence (AI).

Artificial Intelligence Center (AIC)

Berkeley Software Distribution (BSD)

Component Object Model (COM)

Controllable Degree of Freedom (CDOF)

Degrees of Freedom (DOF)

Digital to Analog Converter (DAC)

Direct Current (DC)

Dynamic Data Exchange (DDE)

Electronic Speed Control (ESC)

Finite State Acceptor (FSA)

Fuzzy Logic Control (FLC)

General Purpose Input/Output (GPIO)

Haptic Device API (HDAPI)

Haptic Library API (HLAPI)

Human-Robot Interaction (HRI)

Infrared (IR)

Input/Output (I/O)

Internet protocol (IP)

Inter Process Communications (IPC)

Level of Autonomy (LOA)

Light Emitting Diode (LED)

Lithium – Polymer (Li-Po)

Long Term Support (LTS)

Open Graphics Library (OpenGL)

Open Source Computer Vision (OpenCV)

Personal Digital Assistant (PDA)

PHANTOM Device Drivers (PDD)

Position-Sensitive Device (PSD)

Pulse Width Modulation (PWM)

Radio Control (RC)

Remote Procedure Call (RPC)

Revolutions per Minute (RPM)

Robotic Operating System (ROS)

Single Board Computers (SBC)

Stanford Research Institute (SRI)

Three Dimensional (3D)

Time to Live (TTL)

Transistor-Transistor Logic (TTL)

Transmission Control Protocol (TCP)

Two Dimensional (2D)

Universal Serial Bus (USB)

Unmanned Aerial Vehicles (UAV)

User Datagram Protocol (UDP)

Wheeled Mobile Robot (WMR)

CHAPTER 1

INTRODUCTION

1.1. Landmarks in the History of Robotics

Despite its evolution over centuries and a controversial history full of fictionalization, the word *robot* had not been used until the early 20th Century. In 1920, Karel Capek (1890–1938) wrote a play in which a series of man-made workers perform the tasks of humans and relieve their hardships. In 1942, Isaac Asimov (1920–1992) used the term *robotics* for the first time to address the technology that deals with the robots' design, construction, and operation tasks. In 1985 Asimov revised his famous "Three Laws of Robotics" to include the so-called "Zeroth Law". This list of laws is now accepted by the public as follows:

Zeroth Law: A robot may not injure humanity, or, through inaction, allow humanity to come to harm.

First Law: A robot may not injure a human being, or, through inaction, allow a human being to come to harm unless this would violate a higher order law.

Second Law: A robot must obey the orders given to it by human beings, except where such orders would conflict with a higher order law.

Third Law: A robot must protect its own existence, as long as such protection does not conflict with a higher order law.

Joseph F. Engelberger, widely considered as the "Father of Robotics," formed a team of engineers to develop the first industrial robot in 1961. The robot was designed to unload high-temperature parts from a die casting machine.

The technology expansion after World War II resulted in a series of developments such as emerge of the first digital computer which paved the way for more noble breakthroughs. In 1966, the Artificial Intelligence Center (AIC) was founded at Stanford Research Institute (SRI). SRI has promoted the area of robotics through contributions to mobile and autonomous robots, machine vision and learning, computer graphics, Artificial Intelligence (AI), engineering tools, computer languages, and etc.

In 1959, Planet Corporation developed the first robot that was commercially available. This robot was controlled by a number of switches and cameras. In Norway, a 1964 labor shortage inspired a wheelbarrow producer to manufacture the first prototype of Trallfa robot, which was used in painting wheelbarrow parts.

Space exploration has also been greatly influenced by robotics, in many different ways, such as flyby probes, landers, rovers, atmospheric probes, and robot arms. All of these devices can be remotely operated and have aimed a common goal that is to remove mankind from difficult or hazardous environments.

One of the areas that is appealing public's opinion is the use of Unmanned Aerial Vehicles (UAV)'s in military and civilian applications. A UAV or drone is a plane that is operated remotely with no human pilot onboard, flying at a specific altitude above an area to collect information. Drones had been used by the U.S. in the Balkans in 1999 in a limited range, but it was during the engagement in Afghanistan that the drones were armed with anti-tank missiles and directly engaged in warfare. In November 2002, a Predator UAV fired a Hellfire missile destroying a car carrying a number of suspected al Qaeda associates. Since then, UAV's have continued to play roles in civilian applications as well. Amazon's Prime Air project is a good example of this. Prime Air is an item delivery system designed to safely get packages to customers in a relatively short time using drones.

The ability of high-precision operations in industrial settings gave the medical scientists hopes that in the near future robots will help medical staff in various forms.

The first medical robot started its work in a Danbury hospital in 1988. It was able to navigate the hospital and deliver supplies and medications.

Although there have been many applications and researches, the field of robotics is believed to still be in its infancy and there will be considerable growth in the following years. [1]

1.2. Mobile Robots

Robotics has accomplished its greatest success and reputation in the area of industrial manufacturing by means of robot arms, or manipulators. However, these commercial robots all suffer from lack of a fundamental advantage: mobility. Despite manipulators, a mobile robot is able to travel in the whole specified environment and apply its skills wherever needed.

The urge to mobile robots is mainly due to need for reaching dangerous and hostile environments, where tele-operated systems have gained popularity. On the other hand, some commercial robots operate not where it is impossible for humans to go but rather they share the same environment with humans. These robots are essential not for reasons of their mobility but because of their autonomy, therefore, the ability to maintain a sense of position and to navigate without human intervention is crucial.

The design of mobile robots involves the integration of a number of different areas of knowledge, which makes mobile robotics as one of the most interdisciplinary fields. The successful design of a mobile robot may include some or all of the following bodies of knowledge: Mechanics and Kinematics, Dynamics and Control Theory, Signal analysis, Computer Vision, Computer Algorithms, Information Theory, Probability Theory and etc. [2]

1.3. Human-robot Interaction

A common approach to consider the concept of autonomy in the field of robotics is to describe to what extent the human and robot interact with each other and to what degree each is capable of completing the pre-specified tasks independently.

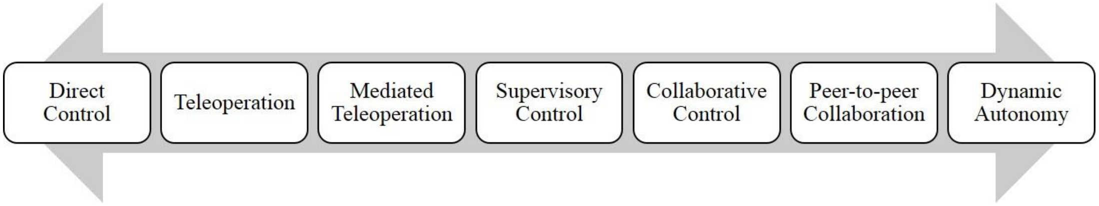


Figure 1.1. Levels of autonomy with emphasis on human interaction [3]

Figure 1.1 illustrates the major levels of autonomy with an emphasis on human-robot interaction. With respect to the level of autonomy for a mobile robot, it ranges along a spectrum with two extreme points: fully manual and fully autonomous. Anything between these thresholds requires both the involvement of a human operator and the necessary autonomy skills for the robot. This initiates the need for a relatively recent field: Human-Robot Interaction (HRI). [4]

Interaction, by definition, needs communication between the robot and human operator which can be categorized into two major types:

1. Remote interaction, in which human and the robot are separated. Remote interaction with mobile robots is also often called tele-operation or supervisory control.

2. Proximate interaction, where the robot is located in human operator's line of sight.

To address the kinds of problems encountered in HRI, one should also define the roles that robots can be assigned to. The following are the most common roles provided by the literature.

Supervisor, Operator, Mechanic, Peer, Bystander, Mentor, and Information Consumer.

Table 1.1. classifies the most common types of human-robot interactions for some contemporary robotics applications.

Table 1.1. *Roles and proximity patterns in some robotics applications. [3]*

Application Area	Remote / Proximate	Role	Example
Search and rescue	Remote	Human is supervisor or operator.	Remotely operated search robots
	Proximate	Human and robot are peers.	Robots support unstable structures
Assistive robotics	Proximate	Human and robot are peers, or robot is tool.	Assistance for the blind, and therapy for the elderly
	Proximate	Robot is mentor.	Social interaction for autistic children
Military and police	Remote	Human is supervisor.	Reconnaissance, demining
	Remote or Proximate	Human and robot are peers.	Patrol support
	Remote	Human is information consumer.	Commander using reconnaissance information
Edutainment	Proximate	Robot is mentor.	Robotic classroom assistant
		Robot is mentor.	Robotic museum tour guide
		Robot is peer.	Social companion
Space	Remote	Human is supervisor or operator.	Remote science and exploration
	Proximate	Human and robot are peers.	Robotic astronaut assistant
Home and industry	Proximate	Human and robot are peers.	Robotic companion
	Proximate	Human is supervisor.	Robotic vacuum
	Remote	Human is supervisor.	Robotic construction

Since robots are ideal for tasks that are monotonous, dirty or dangerous, the level of autonomy may also vary based upon the context in which the robot is used. Tele-operation is when a human operator controls the robot and performs localization and cognition as the robot executes motion and this system has no autonomy. A robot may be semi-autonomous or fully autonomous based upon the level of interaction with a human. In semi-autonomous control, the human controls the robot at certain times dependent upon the task. There are two types of semi-autonomous control: supervisory and shared. In supervisory control, the human may assign some tasks to the robot to complete autonomously and only observe the progress and intervene only whenever necessary. In shared control, the human assigns tasks to the robot but may also interrupt with input such as perception, additional instructions or to cancel execution on a regular basis. In fully autonomous control, a robot performs its own perception, planning and acting independently without human input. [3]

It is obvious that the communication between the operator and the robot is an indispensable part of semi-autonomous systems. And there have been a great number of tools and devices that have been used for this purpose. From the very basic method of keyboard stroke to the cutting-edge technology of virtual reality devices, all function as means to provide a real-time communication link between the two agents. One particular tool that has started to attract the interest of researchers in the field of semi-autonomous mobile robots in the recent years is the haptic devices. Haptics is a term used to describe the sense of touch and movement and the mechanical interactions involving these two. Haptic devices have started to be frequently used remotely in virtual reality or in tele-operation scenarios to invoke the sense of physical presence for the operator or supervisor. The haptic system enables humans to interact with real or virtual objects and environments by means of mechanical, sensory, motor, and cognitive devices. The haptic system may also use the virtual objects or environments to simulate properties such as stiffness, texture, or mass with haptic feedback [5], [6]. Thanks to its ability to provide physical interaction between humans and the robot, haptic devices are capable of establishing a bi-directional communication bridge between the two agents. A haptic device is not only able to issue motion commands generated by the operator to the robot, it can also provide the operator with necessary information about the robot, its surrounding and condition.

Haptic interface is also an integral part of this thesis study and its role in tele-operation task of wheeled mobile robot is investigated and its capabilities are put to the test.

1.4. Scope of the thesis

The main purpose of this thesis study is to investigate the effects of different roles human operator takes in tele-operation of the wheeled mobile robot. Depending on the level of autonomy assigned to the mobile robot on one hand and the role designated to the operator on the other hand, following levels of autonomy are defined:

- Fully manual control
- Fully automated control
- Supervisory control
- Assistive control
- Dynamic shared control

The obstacle avoidance and go-to-goal tasks of the human-robot system are examined in each of the four scenarios. The performance of the system is measured via objective criteria such as number of failures, time to accomplish the task, and the task load on the operator. This study aims to explore the advantages of each level of autonomy over the others and to find a possible optimal selection of roles for both the robot and the operator. The limitations the system may suffer under each condition is also in the scope of this study.

In order to evaluate the hypotheses of this work, a wheeled mobile robot is developed using a BeagleBone Black single board computer and equipped with the necessary hardware to communicate effectively with both a central computer and the human operator. The front-facing camera is used to stream the visual data while the range sensor is used to provide the data corresponding the distance and orientation of the obstacles to the robot.

The haptic device is used in the shared control of the robot in order to provide the operator with force feedback based on position and distance of the robot to the obstacles. The haptic control of the mobile robot in manual, supervisory and shared control schemes is implemented and its contribution to the performance of the system is evaluated.

The integration of main hardware components, development of software tools, and design of control algorithms are also in the scope of this thesis.

1.5. Outline of the thesis

The first chapter of this thesis is devoted to providing a brief introduction to the fundamental concepts discussed in the scope of the study. The second chapter includes the overview of a number of influential research studies which paved the way for the investigation of the hypothesis presented in this work. The third chapter deals with the structure enabling the components of the setup communicate with both each other and the operator. The control architecture used in the study is also presented in this chapter. The detailed work regarding the hardware components and their integration into the experimental setup is discussed in chapter 4. The fifth chapter is dedicated to providing the major findings of the study and the explanation about their implications. This study work is concluded in the sixth chapter, in which potential future opportunities to carry out further research work on the topic is also covered.

CHAPTER 2

LITERATURE REVIEW OF RELATED WORKS

Before exploring the methodologies of this study, a number of prominent articles published in the field are briefly reviewed in order to be familiar with their achievements and the challenges they have faced. These works have provided the author of this thesis with the necessary insight to the subject and may provide the reader with a general understanding of the path this study is going to follow.

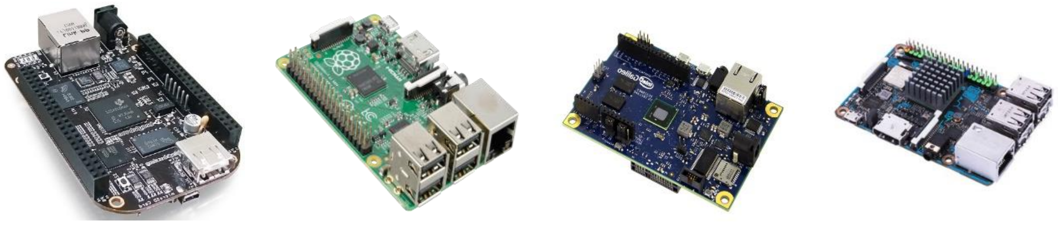
2.1. Mobile Robots

The emergence of the field of Mobile Robotics can be traced back to the advent of Shakey, which was developed at Stanford Research Institute (STRI) in the 1970s, and is considered to be the first AI-inspired robot [7].

As Shakey was being developed, the NASA space program was also working on using AI techniques to build planetary rovers to go to the moon [3].

Mobile robots can be used in many different research applications such as: social (entertainment) robots, service robots, androids, humanoids, multirobot systems, microrobots, nanorobots and etc [8]. However, based on the configuration and working environment, mobile robots can be categorized in four major groups: Wheeled Mobile Robots (WMR’s), Legged mobile robots, Flying robots and Underwater robots [3].

One key difference between mobile robots and other robots is that mobile robots deal with the problem of sensing, planning, navigating and achieving tasks in large-scale potentially unstructured environments [9], [10].



BeagleBone Black R3 Raspberry Pi 3 Intel Galileo Asus Tinker

Figure 2.1. Some common single board computers

2.1.1. Hardware

In order to analyze the performance of a mobile robot, it is necessary to examine its primary hardware components. Since a robot is an autonomous system that must sense, plan, move and act; it must have sensors for perception, a controller for planning and effectors that act on the world.

2.1.2. Controllers

The controller, or control board, is considered as the brain of the robot and performs the cognitive functions. A robot’s controller is usually a computer which can come in different types, sizes and capabilities. The earliest mobile robots used larger on-board computers to fulfill the cognitive tasks. Laptops, netbooks, and computers with compact main boards are among the best choices for this purpose.

As the development of technology, much smaller computers, called Single Board Computers (SBC) became widespread. A single board computer has a preprogrammed output based upon an electrical input; it can store, upload and run programs in an operating system. An SBC can also be defined as a computer- on- a- chip, with a “thinking” processing unit, memory, and means to connect with the outside world. Single board computers are the ideal form of robot cognition because they are simple, cheap, and easy to use. Some of the most common boards largely used in robotic research projects are BeagleBone Black, Raspberry Pi, Intel boards, which are shown in Figure 2.1. These small, inexpensive platforms are able to read, perform necessary computational processes and turn data into an output. By using different programming languages, the user can communicate with the SBC and either send instructions to it or extract information. The key elements in choosing the appropriate platform for a specific task include data- processing architecture, programming language, shape, size, Input/Output (I/O) pins and processing speed.

Recently, some smartphones, tablets, and PDAs, especially those with the Android operating system, have also been used to accomplish the cognition tasks. The disadvantage of these devices is the limitations inherent in their design as products that are made for something other than robot control. Their programming tools are not designed to control real-world devices, so developing a robot application tends to involve a lot of compromises [11].

While simple robotic systems are controlled by a single central processor, complicated robotics systems may have basic low-level control hardwired and high-level functionality programmed. In addition, complicated systems may have distributed control where some communication is via wireless, Bluetooth or radio modem. However, it should be noted that there are tradeoffs between these two options because on-board computation may be limited by weight and power consumption but offline processors may have bandwidth limitations [3].

2.1.2.1. Sensors

Sensors are frequently used to acquire data from the robot and its surrounding. Sensors are necessary parts for navigation, localization and mapping tasks. Sensors can be classified based on the data they measure. Some well-known sensors are proximity, range, motion, speed, position, and orientation. Table 2.1 provides some common sensors and their application [3].

Sensor arbitration is also a common way to get more precise information about the world. In this case, multiple sensors are combined to provide more complex and more reliable source of data.

There are three types of sensor arbitration: competing, complementary and coordinated. Sensors that return the same type of data are redundant or competing such

Table 2.1. *Some common sensors used in mobile robots*

Sensor	Class	Details
Gyroscope	Heading	uses angular acceleration to measure orientation
Compass	Heading	uses the earth's magnetic field to measure heading
Encoder	Motion/Speed	measures motor shaft turns to estimate position or velocity
Doppler radar	Motion/Speed	transmits and measures electromagnetic or sound waves to estimate velocity
GPS	Position	uses a constellation of satellites to estimate absolute position
Bumper	Proximity	uses a tactile switch to indicate the presence of an object
Infrared	Range	emits and measures reflected infrared light to determine distance to objects
Sonar	Range	emits a ping and measures sound from an object to estimate distance to objects
Laser	Range	emits highly amplified and coherent radiation and measures range using time of flight
Camera	Vision	captures images to provide information about the robot's environment

as a robot with two sonar rings. Sensors that are logically redundant and return identical percepts but with different modalities or levels of processing such as infrared and sonar are known as sensor fusion. For instance, if the robot finds a colored ball, the camera may be used to identify its color, infrared to find its location and laser to identify its size and shape. Coordinated sensors use a sequence of sensors to acquire a percept such as firing individual sonar on sonar ring at different times. [9], [10], [3]

2.1.2.2. Actuators

An actuator is a mechanism that enables the mobile robot to complete an action or movement. The most popular actuator for robots is a Direct Current (DC) motor because it is inexpensive, small in size and has a high energy output. It is possible to use Pulse Width Modulation (PWM) to drive the moto at desired speeds. Stepper motors are also used for applications that require high accuracy in position. In order to create high position accuracy with a DC motor, it must be converted to a servo motor by adding sensor feedback via an encoder to measure wheel position. A servo motor may also have a control circuit such as a Proportional–Integral–Derivative (PID) controller to adjust the wheels' motion to minimize the error between desired and actual positions. The key differences between DC and stepper motors are the accuracy and that the DC motor has high torque at low speeds and the stepper motor has the highest torque at high speeds. Stepper motors typically use a simpler open loop control as opposed to closed loop control for servo motors [3].

Some key factors in choosing the right motor for a specific application are: operating voltage, current draw, speed, stall and running torques [11].

2.1.3. Locomotion

Locomotion refers to the way that a robot moves from one place to another and is one of the major characteristics of mobile robots [3]. Ground locomotion is generally achieved via one of the main three systems: wheels, tracks and legs. Recently, there have been a number of hybrid methods which use two or three locomotion systems [12].

Regarding locomotion, there are a number of concepts that the reader should distinguish:

Navigation (path planning); the operation concerning with getting the robot to a goal [13].

Trajectory (motion) planning; dealing with following a path where the robot may need to find or follow the shortest, safest or more efficient path with a time constraint [14]. And

Odometry or path integration; a means of implementing dead reckoning to determine a robot's position based upon the robot's prior position and the current heading and velocity [15].

The advantages of this method are that it is self-contained, i.e. it always provides an estimate of position, and the position can be found anywhere along curved paths. The disadvantages are that the position error grows and require accurate measurement of wheel velocities over time [16].

2.1.4. Kinematics

Robot Kinematics is the dynamic model of how a robot moves. Kinematics is a fundamental characteristic of mechanical behavior of the robot required for its design and control stages. Mobile robot kinematics is used for both position or pose estimation (path planning) and motion estimation (trajectory planning).

Pose estimation includes the robot's orientation as well as its position. Since mobile robots are unbounded to their environment, there is no way to directly measure the robot's position, therefore, it must be integrated over time, which in turn, leads to inaccuracies in the estimations [17].

Forward Kinematics involves using robot wheel velocities to estimate the robot's motion, position, or orientation at future time. This is the same as dead reckoning and is prone to accumulation error. Inverse kinematics involves determining wheel velocities in order for a robot to achieve a desired motion, position or orientation [3].

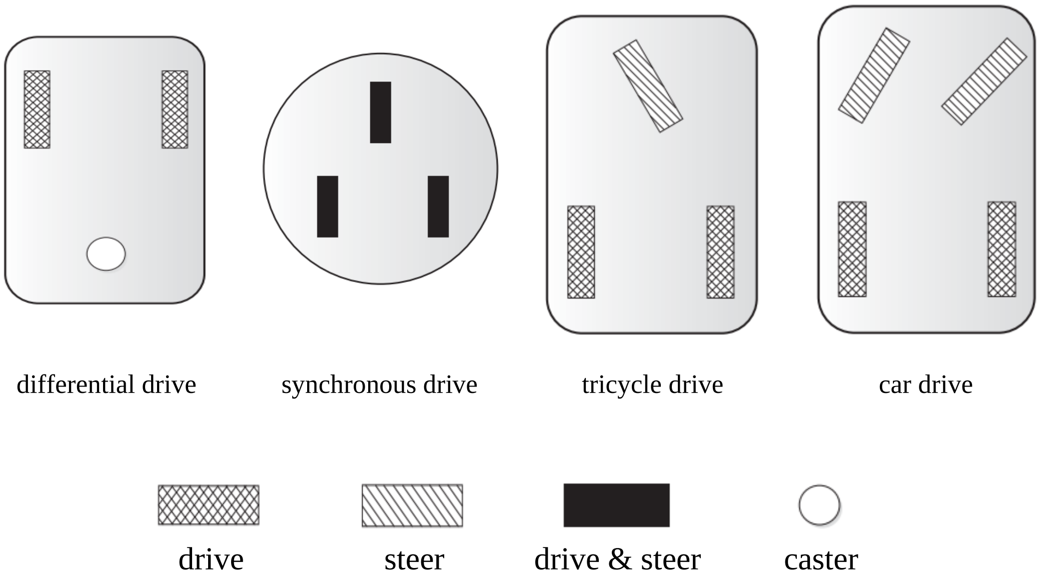


Figure 2.2. Wheeled mobile robot classification

As shown in Figure 2.2, based on the wheels' system, mobile robots are classified into four main groups: differential drive, synchronous drive, tricycle drive and car drive (Ackermann or kingpin steering) [18].

Differential drive is the most common type of wheeled mobile robot configuration. It is one of the simplest designs regarding programming and locomotion [19]. It has two wheels driven independently on a common axis and one or two caster wheels. The casters are just for support and do not affect the kinematic model. This configuration lets the robot perform straight drive, spin and turn maneuvers. For instance, if the drive wheels run at the same speed, the robot moves forward or in reverse. If the wheels move in opposite directions at the same speed, the robot spins in place. If the robot wheels move at different speeds, the robot turns or moves along a curve. However, differential drive robots are very sensitive to differences in wheel velocities, so it is usually hard to make the robot drive in a real straight line. Other disadvantages of differential drive are that it suffers from lateral movement or wheel slippage which is a problem for using odometry to estimate position [20].

One other major concept in mobile robots is the distinction between holonomic and non-holonomic mobile robots which is tied to the Degrees of freedom (DOF) of the robot. DOF is the minimum number of coordinates that can completely specify the motion of a mechanical system. A Controllable Degree of Freedom (CDOF) means that there is an actuator for every DOF. A holonomic robot has the same number of CDOF and DOF, whereas robots with less CDOF than total DOF are non-holonomic. Based on this definition, a differential drive robot is non-holonomic [3].

2.1.5. Control

It was previously discussed that the robot's brain makes the decisions about how to control the robot to achieve some goals or complete some tasks. The simplest control strategy is robot reflex [21]. This low level control is generally carried out with feedback control. Examples of robot tasks that require low level control include stable locomotion, monitoring battery level, obstacle avoidance and wall following. However, high level control generally require robot reasoning [22]. This method is more complex, flexible and general. In addition, it is necessary to have some type of control architecture or method for organizing how the robot uses information to act. Examples of robot tasks requiring high level control include navigation, foraging, localization, exploration or mapping [3].

Control strategies may also be categorized in terms of the types of tasks: maintenance or achievement. For example, maintenance control deals with an ongoing task such as to remain balanced or avoid collisions. Achievement control, however, is responsible for accomplishing some goals such as navigation to a distinctive place in the environment, localization or to find an object [3].

Feedback or closed loop control has been also used for many years in industrial systems and is a method for designing a system to achieve, track and/or regulate to a desired state with minimum error. The plant is the system under control, in this case the robot. The error for a feedback control system is based upon finding the difference between the system's desired state and actual state [23].

Feedback control is typically referred to as the engineering approach to robot control. This method is only applicable to low level control where there is no cognition or reasoning. On the other hand, control architectures use the Artificial Intelligence (AI) approach to accomplish high level tasks [9], [24].

A control architecture is a set of rules and constraints for organizing a robot's control system. There are three robot primitives used to describe the organization of the robot's control architecture: sense, plan, and act. The most prevalent control architectures are deliberative (hierarchical), reactive, hybrid and behavior-based [3].

The deliberative architecture is the oldest control architecture and it is based upon classical AI. It is an expert system which has a top to down design and has all of the answers before the task initiates. The deliberative mechanism uses a very detailed and accurate world model and considers all of the possible outcomes before taking any action. The advantage of deliberative control is that it provides an ordered relationship between sensing, planning and acting. Its disadvantage is that it takes a long time to form a fully detailed and accurate world model, search it for the optimal solution and then create the plan as well as update the plan after the robot acts. The problem is that a robot needs a rapid response time to an open dynamic world. Furthermore, there must be a closed world assumption that everything the robot needs to plan to act is available in the representation. However, retaining all this information can cause a memory shortage or it may not be computationally implemented [3].

Brooks [25] stated that the world was its own best model and this type of planning was a way of avoiding figuring out what to do next, thus reactive control was created. Reactive systems do not use any world model. They are more adaptive because they achieve tasks by using reflexes. They are also faster because they operate in the present with an immediate response. Reactive control is based upon the stimulus response reflex in the context of motor control. It operates very quickly and is effective in a dynamic unstructured environment. In this type of system, there is no complex computation; only short fast pre-computed responses. Arkin [26] stated that this

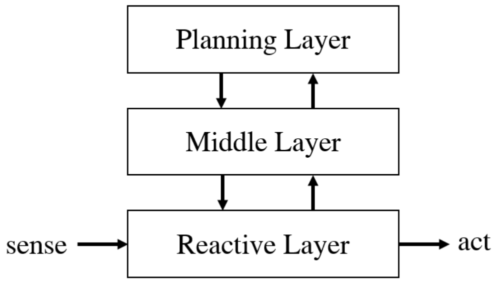


Figure 2.3. Three-layer hybrid Control Architecture

stimulus-response pair was typically modulated by attention and determined by intention. Attention is used to prioritize tasks and to focus the robot sensor resources. Intention is used to determine which rules should be active based upon the robot's goals and objectives. Some of the limitations of reactive control include minimal state, no memory, no learning and no internal models [3].

Hybrid control uses components of the deliberative and reactive control systems. Hybrid control was created to deal with limitations in reactive and deliberative control by integrating some planning and knowledge with a purely reactive system. There are three modules or layers in hybrid control including the reactive, planner and middle layer. The middle layer deals with behavior management or performance monitoring, the reactive layer handles local control tasks and the deliberative layer handles global control tasks. Some of the disadvantages of hybrid control are based upon the fact that the middle layer is difficult to design or debug. It is also difficult to determine the division of task responsibilities between the planning and reactive layer [26].

Behavior-based control is similar to reactive control because it is fast and reflexive but it also has the capability for learning. A behavior is defined as a close coupling between perception and action. Behaviors achieve and/or maintain particular goals. They may have different levels of complexity and time but they all can be combined in a behavior-based control. For example, obstacle avoidance would be instantaneous while homing would take a little longer but they can both be operational at the same time. Other examples of robot behaviors are exploration, goal-oriented, collision avoidance, path following and etc.

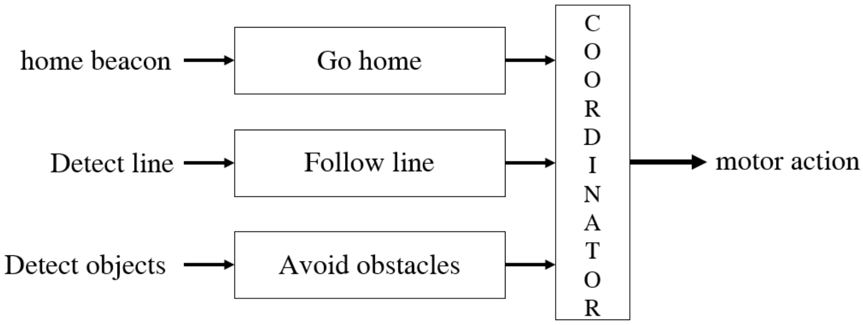


Figure 2.4. Behavior stimulus-response diagram

A mathematical definition for a behavior or functional notation is given by Equation (2.1):

$$b(s) = r \tag{2.1}$$

where b represents the behavior, s the stimulus to the robot and r is the robot's response [26].

A stimulus-response diagram is used to show the sequence of behaviors to achieve a particular task. Figure 2.5. presents an example of a stimulus-response diagram.

The coordinator is responsible for creating an emergent behavior considering all possible individual behaviors. Behavior arbitration is the method of selecting one behavior to execute. This method is also referred to as competitive method because only one behavior can win. Priority-based coordination and action-selection coordination are two common approaches in competitive method.

Behavior fusion is the method of combining multiple behaviors to execute. Behavior fusion is also referred to as the cooperative method since more than one behaviors work together and may create a final behavior. Voting-based coordination and vector summation methods are two common examples of cooperative coordination. [26], [9] and [25].

2.2. Obstacle Avoidance

One of the fundamental requirements of a mobile system is the ability to interact with the physical objects in its surrounding. The robot needs to navigate from a known position to a new desired location and/or orientation while avoiding any contact with any other fixed or moving objects.

Collision or obstacle avoidance is largely accepted as a key part of motion planning. [14] defines motion planning as computing collision-free paths among possibly moving obstacles, coordinating the motion of one or several robots and reasoning about uncertainty in order to build reliable sensor-based motion strategies.

It is evident that special sensors are needed to cope with collision avoidance problem in order to provide a mobile robot with sufficient data to improve the robot’s awareness of its environment and ultimately allow it to move towards the desired location in a safe and realistic way. [27]

Some key considerations for such sensors are presented by [28]: field of view, minimum and maximum range, accuracy, resolution, operation in real-time, ease of interpreting data, redundancy, simplicity, power consumption and size.

There are a number of significant works covering the field of motion planning and collision avoidance. [29] studies the problem of planning secure trajectories for computer-controlled manipulators with two moving arms and multiple Degrees of Freedom (DOF).

In his book, The Complexity of Robot Motion Planning, J. Canny discusses the fundamental issues related to the complexity of motion planning and applies high-powered mathematical techniques to handle the major challenge of finding a collision-free path for a jointed robot in an environment accommodating multiple obstacles [30].

[14] is also one of the best known classic sources which provides detailed explanations of a number of widely-used algorithms until 1990, and continues to be one of the best sources of information on algorithms aiming to solve motion planning and collision avoidance problems.

There are several works about summarizing and classifying different obstacle avoidance methods in the literature. [31] surveys the works on motion planning algorithms for point and rigid robots, as well as manipulators which function in stationary and time-varying, environments containing movable objects. The authors provided a comparison table of the algorithms that was commonly used until 1992.

[32] is another survey on collision detection algorithms which classifies the methods based on robots' geometric models represented by a collection of polygons.

Collision detection algorithms are also briefly presented in [33], the authors discuss several object representations, the different phases in collision detection process, and the factors of an efficient algorithm in addition to the significant developments of the role of graphics hardware in collision detection.

In a more recent study, [34] provides a summary of the major developments in the field of motion planning devised since 1992, focusing on the problem of trajectory planning for the point robots with differential constraints. The authors also compare different approaches based on their degree of how safe, complete, optimal, precise, and computationally complex they are.

The following chapters are devoted to present a brief introduction of a number of prevailing methods which are universally accepted in the field of motion planning and obstacle avoidance. Furthermore, some state of the art works in the literature are introduced and their key features are discussed.

2.2.1. Configuration Space

In [35], the author presented an algorithm aimed to compute the configuration space obstacles where the objects are polygons or polyhedra. This method utilizes characterizing the position and orientation of an object as a single point in a configuration space In this space, each coordinate represents a degree of freedom in the position and/or orientation of the object.

A configuration of an object is defined as a set of independent parameters sufficient to express the position of every point of the object in space. The configuration of a system of rigid bodies having a total of n degrees of freedom can also be identified by an n-tuple. Finally, the configuration space for a system of rigid objects is the space of all its possible configurations and is, therefore, an n-dimensional space bounded by upper and lower limits on each of the degrees of freedom. Configurations of this system are divided into three categories: prohibited (or forbidden), safe (or free) and contact configurations. In prohibited configurations objects collide or overlap and therefore must be avoided. In safe configurations, there is no overlap or contact. Finally, in contact configuration, two or more objects touch each other in one or more places. The Configuration space (C-space) map for a particular problem is a classification of every configuration of the system as one of those three categories. The prohibited regions in such a map are known as C-space obstacles, the safe regions as free space, and the boundary between the two as the contact surface [36].

[37] presented the mathematical properties of configuration space and the associated algorithms for applying those properties into the process of identifying obstacles in configuration space for an industrial robot.

[36] provides an overview of techniques that can be used to map the global C-space of a single robot in a static environment. The authors then discussed the fundamental issues regarding how the robot and its environment are modeled. A range of schemes used to represent a C-space map for mobile robots and manipulators is also discussed in this study. This survey eventually provides tables that list some 50 selected mapmaking papers.

2.2.2. Visibility Graph and Voronoi Diagram

Visibility Graph and Voronoi Diagram are two renowned skeleton-based approaches that are commonly used for 2-Dimensional (2D) obstacle detection problems. The visibility graph is a collection of lines in the free space which connect the edges of an object to those of another [38].

[39] states that this method guaranties the generation of the shortest possible collision-free path that might involve going near obstacles. [40] proposed an algorithm to find the shortest-path map in linear time instead of computing each shortest-path map

separately. The authors use vertex of each polygon-shaped object and use their maps as a modification to the neighboring one.

The problem of applying visibility diagram method in mobile robots is that the robot moves in a considerable close neighborhood of the obstacles. If the robot is required to keep distance from the obstacles however, the nearest-site Voronoi diagram can be used to guarantee a collision-free path [31].

[41] presents an algorithm for the construction of generalized Voronoi diagrams which merges two arbitrary (standard) Voronoi diagrams by employing a linear time algorithm. The generalized Voronoi diagram method may use efficient algorithms for computing the furthest-site diagrams described by the geodesic metric inside a simple polygon [42]. The problem of computing the furthest-site Voronoi diagram is defined as: "Given a finite collection of points, for each point identify the element in the collection that is maximally distant from it [43]." A study for planning a collision-free path for a rectangular shaped robot in a 2D environment clustered with polygonal obstacles is presented in [44]. This paper showed that the application of generalized Voronoi diagrams in motion planning application results a fast, safe and smooth translation in the plane. In [45] , Fast Marching Method is applied to the Voronoi diagram to obtain the shortest possible path including both global and local planning requirements. They managed to achieve smooth and safe and trap-free paths owing to the integration of real time sensory information. The optimality of Voronoi method is studied in [46]. In this paper, the Voronoi diagram algorithm is used to generate an optimal path connecting an origin to a target point in an environment where simple disjoint polygonal obstacles are present. The path is proven to be optimal concerning its length and safe distance from obstacles. The calculated path is shown to be also superior to one obtained from visibility graph approach in respect of its speed and smoothness.

2.2.3. Cell Decomposition

Cell decomposition method is probably the motion planning approach that has been studied more than any other method. In this method, the robot’s free space is divided into cells in such a way that an obstacle-free path is found between any two configurations of the robot. Then, a connectivity graph is constructed in order to represent the adjacency relation between the cells. The graph’s nodes are the cells that are extracted from the free space. Two nodes are connected by a link if two corresponding cells are adjacent. The result of the search in this graph is a series of cells called a channel, which in turn, can lead to emergence of a collision-free path [14].

In [47] a quadtree cell representation and hierarchical search method are used to automatic path planning in mobile robot applications.

[48] and [49] also used cell decomposition method to find a conditional shortest path in an unknown environment. These studies achieved a robust and optimal path by combining grid-based path planning techniques and quad-tree-based techniques.

[50] argued that in the worst case scenario, the motion planning based on cell decomposition does not find a path, but it would never find a path that would cause a collision. This means that exact cell decomposition method is guaranteed to find a free path as long as it is coupled with an appropriate search algorithm [14].

Finally in [51], cell decomposition method and probabilistic sampling fused to handle path planning problems for mobile robots that navigate in maze-like environments. The so called probabilistic cell decomposition gave satisfactory results for a variety of implementations.

If the search based on cell decomposition does not result any free path, either the resolution of the decomposition is insufficient or no obstacle-free path exists between the robot’s initial and goal configuration [14].

2.2.4. Potential Field

In this terminology, the robot is considered as a point in configuration space moving under the effect of a virtual potential force generated by the configuration of both the goal and the obstacles. Simply, the goal produces a potential force attractive the robot while the obstacles produce repulsive potential pushing the robot away. At every point, potential fields are combined through superposition of these two forces to create a global field that the robot uses to navigate. Then the direction of this total force is considered as the optimal direction of motion which the robot is assumed to move [14].

This method of path planning may also be referred to as Schema theory which is in direct contrast to subsumption control architecture. Schema theory is used for navigation tasks where the sensorimotor mappings are represented by vector-based potential fields to create goal-oriented movements. The coordination of the behaviors is through a cooperative method by using vector addition. The vectors represent the behaviors and the vector summation produces an emergent behavior. Each potential is a vector with a magnitude and direction so that the force imparted on the robot is inversely proportional to the distance squared.

$$Force \propto \frac{1}{Distance^2} \tag{2.2}$$

Figure 2.7. provides of a simplified example of a robot using the potential field method to navigate from a start point to a goal [3].

[52] applied this obstacle avoidance scheme to robot arm mechanisms. Using visual sensing in the COSMOS system for a PUMA 560 robot, the authors demonstrated real-time collision avoidance when moving obstacles are present. Khatib also claimed that using potential field approach as a low-level control strategy together alongside a high-level planning technique results a promising solution to path planning problems.

Compared with the previous obstacle avoidance techniques, potential field methods are generally considered to be more efficient. However, their major disadvantage is the chance that the robot falls in local minima of the potential function other than the goal configuration [14]. This phenomenon occurs because of the fact that the artificial potential field approach for real-time control is local rather than global. [53].

Another key problem arises when in real-time obstacle avoidance for fast mobile robots, where sensor data inaccuracy exists and the robot is required to navigate continuously without stopping in front of obstacles. This problem is addressed in [54]. The authors proposed Virtual Force Field, which is the combination of Certainty Grids for obstacle representation, and Potential Fields for navigation. They successfully managed to resolve local minima and oscillatory robot motion problems.

In order to eliminate the local minima problem in cluttered environments, [53] developed a new formulation for potential fields called harmonic functions. They integrated the harmonic potential method in the control task of both a bar-shaped mobile robot and a 3DOF manipulator.

Finally, [55] presented an artificial potential field method which establishes an analogy between the path planning problem and a steady-state heat transfer with variable thermal conductivity. In this simulation, the optimal path is similar to the heat flow with minimal thermal resistance. It is illustrated that applying variable thermal conductivity concept decreases the complexity of geometrical domain presentation significantly. The path generated using this strategy is free from local minima, smooth, optimal and collision-free.

2.3. Human-Robot Interaction

In this section, some noble works in Human-Robot Interaction (HRI), Tele-operation and Robot Force Feedback control are presented. The focus on selecting papers is such that they have had significant contribution to the foundation of this dissertation.

To get a clear idea of the application of force control in robotics fields until 1997, [56] provides a review of 75 papers in the field of robot force control and presents their differences and application conditions. [4] is another influential survey paper with an emphasis on human factors, robotics, cognitive psychology, and design.

One of the essential concepts in HRI is the Level of Autonomy (LOA). Levels of autonomy is used to describe the degree to which the robot can operate on its own [57]. The ultimate goal of introducing LOA is to enhance the mutual human-machine interaction and consequently improve the overall performance of the system by taking into account both operator's experience and the technological capabilities and limitations [58]. [59] introduced a 10-step LOA taxonomy that describes various ways in which the main functions can be assigned to a human operator and a computer to achieve the desired task performance. 10 levels of LOA include:

- Manual Control
- Action Support
- Batch Processing
- Shared Control
- Decision Support
- Blended Decision Making
- Rigid System
- Automated Decision Making
- Supervisory Control and
- Full Automation.

In the same paper, the performance of a human-machine system in completing various tasks was evaluated and then compared under different LOA conditions. It was seen that in Shared Control, Decision Support and Blended Decision Making cases the system had the minimum collapses and collisions. However, when the main objective was the time to recovery from a failure, Action Support and Shared Control demonstrated superior performances.

In [60], an experiment conducted to obtain the optimum LOA for Large-Scale Robotic Teams scenario. They introduced three levels of LOA as adaptive autonomy, adjustable and mixed initiative. The results of this work proved that mixed initiative, where the agent and supervisor collaborate to achieve the best perceived level of autonomy, resulted in better performance than two other cases. Finally, in [58], the authors proposed five different LOAs, to investigate their effectiveness for promoting tele-robot system performance and operator situation awareness, as well as decreasing work load imposed on the operator. The results indicated that the performance during automation failure was the highest when the system functioned at lower and intermediate LOAs where the involvement of human control of system functions was greater.

In the next two chapters, a summary of the prominent works regarding the major concerns of HRI in robot manipulators and mobile robots are presented and their key features are discussed. The papers are provided in a chronological order in order to provide a clear understanding of the main breakthroughs in the area.

2.3.1. HRI in Manipulators

Computer assisted surgery is one of the pioneer fields in which the HRI in robot arms and manipulators are extensively studied. Contribution of tele-operated robotic workstation for surgical applications requiring fine control of force and movement is the main subject of [61]. [62] proposed an approach called Acceleration Tracing Orientation Method to compensate the disturbance without applying algorithm of the inverse dynamics. Then the virtual compliance control and the hybrid control were implemented and tested in a three-DOF robot manipulator. In [63] the fundamental concerns of stability and performance of haptic interaction devices are discussed. By extending the idea of virtual coupling network- an artificial link between the haptic display and a virtual world- both the impedance and admittance models of haptic interaction were elaborated. The authors introduced a virtual coupling network between the mechanical device and the virtual environment in order to ensure the stability of the combined haptic interface in the case of arbitrary passive human operator and environmental impedances. Use of a virtual spring, and local contact force control is addressed in [64]. The performance of the force control is improved by an on-line stiffness estimation mechanism where the objects are in contact with the robot arm. This modular control structure for tele-operation improved the overall robustness of the system and desired performance of providing a human operator with continuous force feedback. This enables the operator to only control the translational motion of the arm through tele-operation whereas the other DOFs are autonomously managed by the robot on its own without compromising the outcome of the task. The problem of detecting and reacting to possible collisions between a robot manipulator and a human is addressed in [65]. Ensuring safety to the human during physical interaction is the main objective of this paper. A mechanical test platform was built in order to quantitatively analyze and compare the different reaction strategies. The experimental results prove that the proposed collision detection and reactions method work in sufficiently reliable way as contact forces are reduced to below a level which is dangerous to humans. Finally, the topic of [66] is the physical interaction between robot arms and humans as well as the exchange of objects. This study presents a control system to fulfill grasping action while maintaining a balance between safe HRI and effective task execution. The experiment is conducted by taking into the account safety, reliability and efficiency criteria.

2.3.2. HRI in Mobile Robots

Tele-operated mobile robot systems have been developed in order to allow a human operator to perform complex tasks in remote environments [67]. Their ability to be operated remotely in performing various tasks in inaccessible environments can be considered as the most prominent feature of remotely manipulated mobile robots. In [57], Thomas B. Sheridan defined tele-operators as general purpose submersible work vehicles controlled remotely by human operators and with sensors, power and propulsive actuators for mobility, and mechanical hands and arms for manipulation and possibly a computer for a specific degree of control autonomy. In this study, he focuses on undersea tele-operation concerning the human operator and the man-machine interface. One of the cases that tele-operated mobile robots play crucial roles is the situation in which the environment is too hostile for a human being to be. In

[68], J. Abouaf reports the challenges scientists had in completing tasks in Chernobyl nuclear power plant after catastrophic blast in 1986. The development of the robot called Pioneer was a breakthrough in many fields including tele-operation. The problem of obstacle avoidance for a powered wheelchair using force feedback joystick was addressed in [69]. It was demonstrated that the driving performance of experienced wheelchair users was considerably improved when the proposed force feedback algorithm of the joystick was activated to push the wheelchair away from obstacles.

In [67], a haptic interface is used to improve the operator's understanding of the mobile robot's workspace. Particularly, a virtual interaction force is generated based on obstacles in the close vicinity of the mobile vehicle in order to prevent dangerous collisions, so that a safe navigation task can be completed. In addition, owing to the passivity of the overall system, its stability is guaranteed. In a similar study, [70], the problem of tele-operating a mobile robot using shared autonomy is addressed. While an on-board controller performs obstacle avoidance task, the operator uses a haptic device to set the desired forward and rotational speed. Sensors on the robot are used to collect obstacle range data. This range information is converted into forces that projects to the operator's hand via the haptic probe. This strategy proved to significantly improve the human-robot performance in several ways including reduced number of collisions with obstacles. Use of the Internet in order to establish a platform to control a mobile robot by the use of a force feedback joystick is the main objective of [71] where the main focus is on multiple cooperating rovers. Again in [72] the performance of a force feedback joystick in a powered wheelchair is evaluated. The test subjects, disabled people, stated that there are some significant advantages of integrating a force feedback joystick into the wheelchairs, particularly while moving in corridors and passing through doors. In [73] the authors proposed a test of iterative usability to evaluate the performance of robot autonomy and examine the advantages of mixed-initiative control in real-world search and detection tasks. The results of this work indicated that performance is optimal when the operator focuses on the search and rescue task and provides only high-level direction commands to the robot. Rescue robots was also the topic of [74]. This paper discusses the issues encountered in the field of HRI until 2004. The results demonstrate the mechanism by which HRI improves the performance of the robotic system. [75] also presents a new methodology in robot control. Using brain-machine interaction, the authors managed to perform the tasks of navigation and visual exploration between remote places by means of the internet. In this study, the operator watches a real-time video stream provided by the robot camera combined with virtual reality parameters. Then he/she focuses on a target area to navigate the robot to. The results shows that all human subjects could successfully perform the navigation tasks with a high level of reliability. [76] uses the Force-Feedback Joystick control to train infant users with special needs to perform safe and purposeful drive of a mobile robot to explore the unknown environment. In this experiment, if the infant steers the joystick away from the desired direction, the joystick applies a force to the driver's hand and causes the vehicle move toward the desired path.

It can be concluded in this chapter that selecting the correct hardware components for the mobile robot is crucial for both optimizing the robot's performance and determining the suitable obstacle avoidance algorithm. In addition, the study of the prominent research works reveal that the role of human operator in control of semi-autonomous mobile robot, contribution the operator and the robot make to the system, and the extent to which these two hold authority over the task play significant role on the performance of the human-machine system.

Last by not least, different research studies show that the control algorithm plays essential role in achieving the human-robot system's objectives, including performance and security.

The next chapter explains the approach this work takes to develop the control algorithm which is capable of addressing above-mentioned issues.

CHAPTER 3

CONTROL ARCHITECTURE

As discussed earlier, the control architecture, which is one of the main contributions of this study, is one of the most important parts of a mobile robotic system. The necessary fundamental concepts regarding various control mechanisms used in the literature and their strengths are provided in the first two chapters. This chapter, in particular, provides a more detailed explanation of the control architectures developed and used in this thesis study after presenting the kinematic model of the wheeled mobile robot. The chapter covers the behavior-based control method which acts as building block for the cooperative system and the different levels of autonomy in cooperative control context.

3.1. Robot Kinematic Model

In order to assign proper behaviors to a mobile robot, or simply control it, the first requirement is acquiring a model by which the behavior of the robot can be explained and more importantly, predicted. One of the most common models in the literature is called Differential Drive Mobile Robot model which includes two wheels on the right and left sides and one or two caster wheel(s). The driven wheels on the sides can be controlled to turn independently at different speed rates and in different directions, which also enables the robot to move in any direction and with any velocity.

Depending on the complexity of the model to be used, various parameters need to be available in order to develop the model. However, for the sake of simplicity, one simple, yet powerful model, which requires only two parameters, is considered to be used in this study. One major parameter that needs to be known is the track of the mobile robot, L, which is the distance between two wheels of the robot. The other

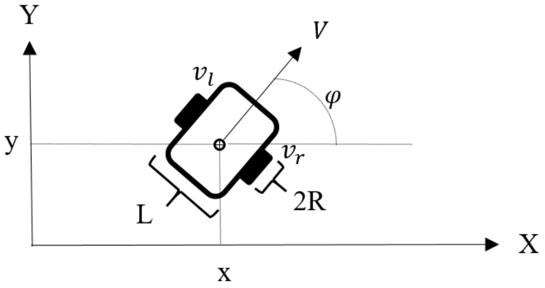


Figure 3.1. Mobile robot schematic

required component, R, is the radius of the wheels. The schematic of the robot is shown in Figure 3.1. The inputs to the system, i.e. the signals needed to control the motion of the robot, are the velocities at which the left and right wheels rotate. These rotational velocities are denoted by v_l and v_r respectively. In order to relate these inputs to the global coordinates of the robot *x* and *y*, and the orientation of the robot with respect to the *x* axis, φ we use the following equation [77]:

$$\dot{x} = \frac{R}{2}(v_r + v_l)\cos\varphi \tag{3.1}$$

$$\dot{y} = \frac{R}{2}(v_r + v_l)\sin\varphi \tag{3.2}$$

$$\dot{\varphi} = \frac{R}{2}(v_r - v_l) \tag{3.3}$$

Despite the fact that Equations (3.1) to (3.3) provide the necessary information about the motion of the robot, they fail to be feasible for generating motion control input. Therefore, by introducing V as the forward velocity of the robot, we get:

$$\dot{x} = V\cos\varphi \tag{3.4}$$

$$\dot{y} = V\sin\varphi \tag{3.5}$$

$$\dot{\varphi} = \omega \tag{3.6}$$

Where,

$$V = \frac{R}{2}(v_r + v_l) \tag{3.7}$$

And ω is the rotational (angular) velocity of the robot.

Using (3.7):

$$\frac{2V}{R} = v_r + v_l \tag{3.8}$$

And using (3.3):

$$\frac{\omega L}{R} = v_r - v_l \tag{3.9}$$

Now, we can solve for v_r and v_l as:

$$v_r = \frac{2V + \omega L}{2R} \tag{3.10}$$

$$v_l = \frac{2V - \omega L}{2R} \tag{3.11}$$

Since R and L are known, constant parameters, the desired control inputs, V and ω can easily be mapped to v_r and v_l, which are indeed the actual control inputs to the robot.

3.2. Behavior-based Control

In the conventional approach in Artificial Intelligence, a great focus has been put on precise modeling of both the mobile robot and its environment. Also planning is an indispensable part of the conventional robot control, particularly in applications where optimality is among the major concerns. However, the environment the robot operates in can be unpredictably dynamic and no matter how accurately the robot perceives its environment, and no matter how sophisticated its model and control algorithm are, the robot cannot react efficiently to complicated and unpredicted stimulus that it is exposed to. Therefore, unlike industrial robots (where the environment is static and optimality is inherently crucial), in the case of mobile robots, allocating a great amount of computation effort on modeling and planning is in many cases neither feasible nor sensible. Brooks [25] stated that the world is its own best model and this type of planning was a way of avoiding figuring out what to do next. Such ideologies paved the way to establishing more innovative control approaches such as reactive and hybrid control architectures [3]. The key factor in this approach is developing a library consisting of useful controllers which the robot uses frequently. These controllers can be regarded as behaviors and the robot can switch between these behaviors whenever there is a change in the environment or in the priority in robot's objectives etc.

In the following sections of this chapter, the main components of the behavior-based control methodology are discussed. The ideas presented in this schema serve as building blocks for the cooperative control of the mobile robot, which is explained at the end of this chapter.

3.2.1. Reactive Control

Reactive control was proposed by Rodney Brooks and was modeled after animal behavior. This control approach is based on responses to the external stimulus in the context of motor control. The main advantage of this method is the fact that it operates very quickly and is effective in a dynamic unstructured environment. In this type of system, instead of complex computation, only short, fast, and pre-computed responses are executed. The complete set of rules for the system is defined at design time, i.e. there is a default response for every single event that is within the scope of the robot's objectives. [26]

There are five characteristics of an architecture based upon the reactive paradigm [9]:

- Animals are a motivation for the collection of rules
- Robots are situated agents
- Rules are the building blocks for robot actions and the overall behavior is emergent

- All rules are based upon local ego centric sensing
- All tasks can be decomposed into component or primitive rules.

Although there is no planning, memory, learning or internal models in reactive control, the robot has the potential to become more intelligent by adding more rules.

This system can also be further developed by using layers to represent increasing levels of competency. The individual layers that work simultaneously on individual objectives, are task-achieving behaviors and used to represent the control system in a vertical layout versus the horizontal decomposition of traditional AI control. The motivation for this is to develop intelligent autonomous mobile robots with multiple sensory input channels, multiple objectives, and high robustness and extensibility. As mentioned in the beginning of this chapter, this control system does not afford and also does not need a world model due to the belief that the world is its own best model.

In this study, two behaviors are considered, their rules are developed and implemented in the robot's control architecture: Go-to-goal and Avoid obstacles.

3.2.1.1. Go-to-goal

This rule indicates that the robot has to move toward the goal's position until the distance between these two is negligible, i.e. the go-to-goal task is fulfilled. The robot achieves this objective by adjusting its orientation in a way that the goal is positioned in front of the robot.

The robot is able to accomplish this task by processing the image obtained from the frontal camera, computing the relative normalized angle of the goal to the central axis of the robot, θ_1, and rotating toward the goal until $\theta_1 = 0$.

The idea in reactive control paradigm is to link the external stimulus to the robot's motor actions. In this case, the objective is to directly couple the low-level input control variables, the speed rate of the mobile robot to the orientation and distance of the goal with respect to the robot. In other words, to achieve a set of functions mapping $d_1$and θ_1 to v_r and v_l .

The approach used in this part of study is inspired by the use of Virtual Force Field (VFF) [78], in which, similar to other reactive control methodologies, the goal is assumed to generate an attractive force vector and exert it on the robot. The magnitude of this force is proportional to the square of the distance of the goal to the robot. The direction of the force is in along the line connecting the robot to the goal [79].

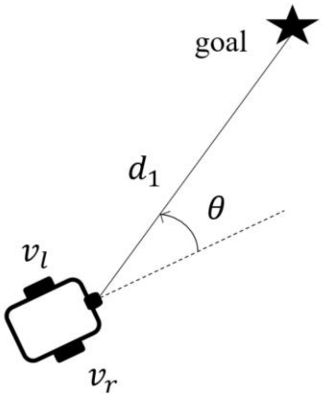


Figure 3.2. Go-to-goal behavior

If the vector representing the attractive force of the goal is denoted by $\vec{V}_1$:

$$\vec{V}_1 = V_1 \hat{r}_1 \tag{3.12}$$

Where, $\hat{r}_1$ is the unit vector linking the position of the center of the range sensor (and the camera) on the robot to the position of the goal.

The magnitude of the attractive force is proportional to the square of d_1, the distance of the goal to the robot, as shown in Figure 3.2:

$$V_1 = K_1 d_1^2 \tag{3.13}$$

Where, K_1 is a constant scaling factor.

The attractive force can be decomposed to its components, one along the direction of motion of the robot and one perpendicular to it:

$$V_1^V = V_1 \cos{(\theta_1)} \tag{3.14}$$

$$V_1^{\omega} = V_1 \sin(\theta_1) \tag{3.15}$$

V_1^V is the component of the goal attraction force along the direction of motion of the robot and V_1^{ω} rotational component of the attractive force. Rewriting Equations (3.14) and (3.15):

$$V_1^V = K_1 d_1^2 \cos(\theta_1) \tag{3.16}$$

$$V_1^{\omega} = K_1 d_1^2 \sin(\theta_1) \tag{3.17}$$

Using the kinematic model of the robot, a pure reactive control approach is realized here. Assuming V_{ref} and ω_{ref} to be the desired forward and rotational velocity of the robot, they can be related to V_1^V and V_1^{ω} respectively:

$$V_{ref} = K_1' \, V_1^V = K_1' K_1 d_1^2 \cos(\theta_1) \tag{3.18}$$

$$\omega_{ref} = K_1'' \, V_1^{\omega} = K_1'' K_1 d_1^2 \sin(\theta_1) \tag{3.19}$$

Therefore, these reference velocities, V_{ref} and ω_{ref} can be considered as multiples of V_1^V and V_1^{ω} respectively. By merging the constant coefficients as:

$$K_1^V = K_1' K_1 \tag{3.20}$$

$$K_1^{\omega} = K_1'' K_1 \tag{3.21}$$

$$V_{ref} = K_1^V d_1^2 \cos(\theta_1) \tag{3.22}$$

$$\omega_{ref} = K_1^{\omega} d_1^2 \sin(\theta_1) \tag{3.23}$$

Equations (3.22) and (3.22) show how the robot simultaneously reacts to the deviation from the orientation and distance of the goal by changing the values of ω_{ref} and V_{ref}.

Now these higher level control input can be translated into low level control inputs exerted on the robot wheels. Using Equations (3.10) and (3.11):

$$v_r = d_1^2 \frac{2K_1^V \cos(\theta_1) + LK_1^{\omega} \sin(\theta_1)}{2R} \tag{3.24}$$

$$v_l = d_1^2 \frac{2K_1^V \cos(\theta_1) - LK_1^{\omega} \sin(\theta_1)}{2R} \tag{3.25}$$

Where, L is the track of the robot and R is the radius of the wheels

3.2.1.2. Obstacle Avoidance

While in the previous behavior, the data from frontal camera determines the motion of the robot to the goal, in obstacle avoidance behavior, the data from laser range sensor is utilized to issue the motion commands. The key concept in this behavior is to move away from the closest obstacle in a way that no collision takes place. Once the data from the range sensor is processed and the orientation and distance of the closest obstacle with respect to the robot, θ_0 and d_0 are computed, a repulsive force is generated and applied to the robot. Figure 3.3. demonstrates how the robot behaves (reacts) when faced with an obstacle in its range.

Similar to the go-to-goal behavior, the idea in reactive obstacle avoidance is to link the perception of the obstacle to the robot's motor actions. In this case, however, the objective is to directly couple the speed rate of the mobile robot to the orientation and distance of the obstacle with respect to the robot. In other words, to achieve a set of functions mapping $d_0$and θ_0 to v_r and v_l .

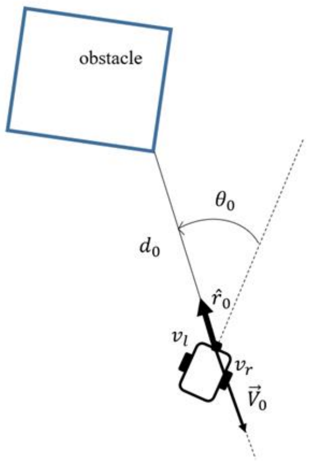


Figure 3.3. Obstacle avoidance behavior

The approach used in this part of study is again based on the use of Virtual Force Field (VFF) [The vector field histogram-fast obstacle avoidance for mobile robots], in which, the obstacle is assumed to generate a repulsive force vector and exert it on the robot. The magnitude of this force is proportional to the inverse of the square of the distance of the obstacle to the robot. The direction of the force is in along the line connecting the robot to the goal in the opposite direction of the obstacle [Path planning for autonomous mobile robot using the Potential Field method]. (Figure 3.3)

If the vector representing the repulsive force of the obstacle is denoted by $\vec{V}_0$:

$$\vec{V}_0 = -V_0 \hat{r}_0 \tag{3.26}$$

Where, $\hat{r}_0$ is the unit vector linking the position of the center of the range sensor (and the camera) on the robot to the position of the obstacle.

The magnitude of the repulsive force can be expressed as:

$$V_0 = K_0 \frac{1}{d_0^2} \tag{3.27}$$

Where, K_0 is a constant scaling factor.

The repulsive force can also be decomposed to its components, one along the direction of motion of the robot and one perpendicular to it:

$$V_0^V = -\, V_0 \cos{(\theta_0)} \tag{3.28}$$

$$V_0^\omega = -\, V_0 \sin{(\theta_0)} \tag{3.29}$$

V_0^V is the component of the goal attraction force along the direction of motion of the robot and V_0^ω rotational component of the repulsive force. Rewriting Equations (3.28) and (3.29):

$$V_0^V = -K_0 \frac{1}{d_0^2} \cos{(\theta_0)} \tag{3.30}$$

$$V_0^\omega = -K_0 \frac{1}{d_0^2} \sin{(\theta_0)} \tag{3.31}$$

Using the kinematic model of the robot, a pure reactive control approach is utilized here. Assuming V_{ref} and ω_{ref} to be the desired forward and rotational velocity of the robot, they can be related to V_0^V and V_0^ω respectively:

$$V_{ref} = K_0' \, V_0^V = -K_0' K_0 \frac{1}{d_0^2} \cos(\theta_0) \tag{3.32}$$

$$\omega_{ref} = K_0'' \, V_0^\omega = -K_0'' K_0 \frac{1}{d_0^2} \sin(\theta_0) \tag{3.33}$$

Therefore, these reference velocities, V_{ref} and ω_{ref} can be considered as multiples of to V_0^V and V_0^ω respectively. By merging the constant coefficients as:

$$K_0^V = K_0' K_0 \tag{3.34}$$

$$K_0^\omega = K_0'' K_0 \tag{3.35}$$

$$V_{ref} = -K_0^V \frac{1}{d_0^2} \cos(\theta_0) \tag{3.36}$$

$$\omega_{ref} = -K_0^\omega \frac{1}{d_0^2} \sin(\theta_0) \tag{3.37}$$

Equations (3.36) and (3.37) show how the robot simultaneously reacts to the orientation and distance of the closest obstacle by changing the values of ω_{ref} and V_{ref}.

Now these high level control inputs can be translated into low level control inputs exerted on the robot wheels. Using Equations (3.10) and (3.11):

$$v_r = -\frac{2K_0^V \cos(\theta_0) + LK_0^\omega \sin(\theta_0)}{2Rd_0^2} \tag{3.38}$$

$$v_l = -\frac{2K_0^V \cos(\theta_0) - LK_0^\omega \sin(\theta_0)}{2Rd_0^2} \tag{3.39}$$

Where, L is the track of the robot and R is the radius of the wheels.

3.2.1.3. Behavior Arbitration

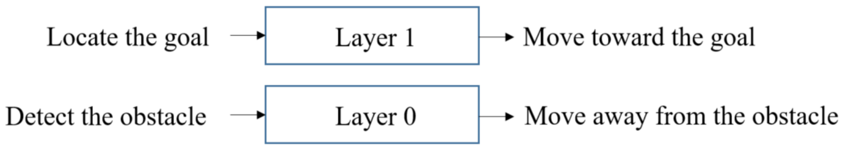


Figure 3.4. Layered reactive control

Behavior arbitration is the method of selecting one behavior to execute. A competitive behavior arbitration method, called behavior subsumption, can be used as one alternative. In subsumption, a fixed hierarchy among the layers are used and no two behaviors could be active at the same time. This method is also referred as behavior decomposition.

Figure 3.4. shows the layered structure of the reactive control system in subsumption approach. The zeroth layer corresponds to the obstacle avoidance and has inherently higher priority compared with the first layer, which represents the go-to-goal behavior.

Since the two layers are considered to be entirely independent, it is obvious that conflicts between objectives corresponding to these layers are inevitable. Therefore, a solution for handling conflicting stimuli, which impose conflicting motion commands to the robot, has to be developed. Figure 3.5 represents two most common behavior arbitration methods.

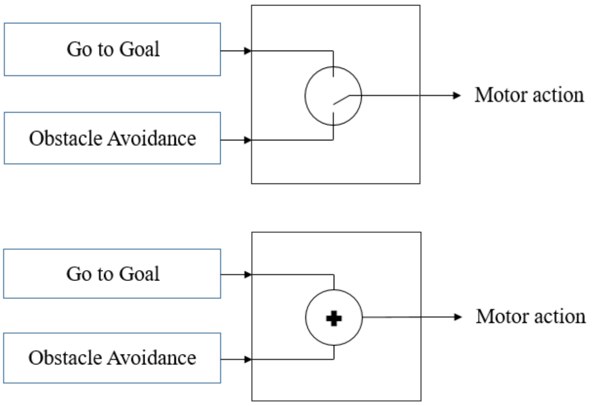


Figure 3.5. Behavior decomposition vs behavior fusion

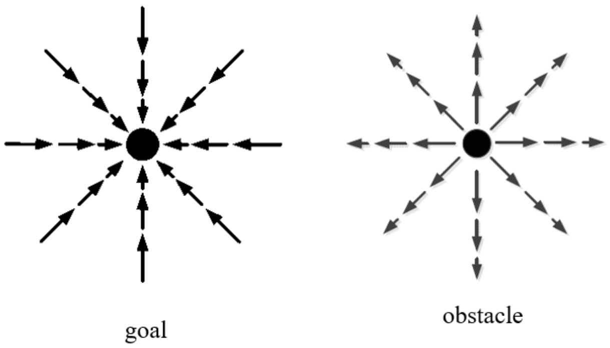


Figure 3.6. Attractive (goal) and repulsive (obstacles) points in potential field

In order to overcome the drawbacks of competitive behavior arbitration method, in this section a cooperative behavior arbitration method is presented. Unlike switching between behaviors, this method enables multiple behaviors work together and create an emergent behavior. Since more than one behavior is combined to generate overall action, this method is referred to as behavior fusion. In this study, the tool that is used to fuse go-to-goal and obstacle avoidance behaviors is Motor Schema method. Proposed by Ron Arkin [19], motor schema is a method suggesting that path planning and navigation are essentially a collection of behaviors. Motor schema uses the vector-based potential fields to create goal-oriented movements. The coordination of the behaviors is then achieved by using vector addition. Therefore, this type of control is sometimes referred to as the potential fields method. In this approach, as demonstrated in Figure 3.6, the goals are attractors and the obstacles are repulsors. The potential fields are combined through superposition to create a global field that the robot uses to navigate. The vectors represent the behaviors and the vector summation produces an emergent behavior. Each potential is a vector with a magnitude and direction generated from goal or obstacle located in a relatively small vicinity of the robot and applied on the robot at any instance of the operation.

If the repulsive effect of a i^{th} obstacle is denoted by $\vec{v}_i$, and the attractive effect of the goal by $\vec{v}_g$, the overall effect of the goal and obstacles on the robot is the vector

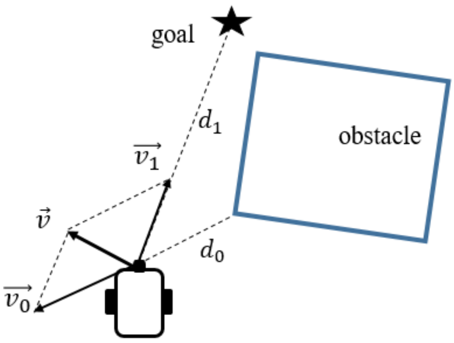


Figure 3.7. Superposition of the effects of the goal and an obstacle

summation of all attractive and repulsive effects applied from the goal and all obstacles detected in a predefined range of the robot:

$$\vec{v} = \vec{v}_g + \sum_{i=1}^{n} \vec{v}_i \tag{3.40}$$

It should be noted that in this study, instead of a series of obstacles, only the closest object is selected to be considered in obstacle avoidance behavior; therefore, using the expressions in the previous section:

$$\vec{v} = \vec{V}_0 + \vec{V}_1 \tag{3.41}$$

Where, $\vec{V}$ is the overall motion vector resulted by both the goal and the obstacle effect. The schematic representation of the motion vector and its components are shown in Figure 3.7. Writing its forward and rotational components:

$$V = -V_0 \cos(\theta_0) + V_1 \cos(\theta_1) \tag{3.42}$$

$$\omega = -V_0 \sin(\theta_0) + V_1 \sin(\theta_1) \tag{3.43}$$

using Equations (3.22), (3.23), (3.36) and (3.37):

$$V = -K_0^V \frac{1}{d_0^2} \cos\,(\theta_0) + K_1^V d_1^2 \cos(\theta_1) \tag{3.44}$$

$$\omega = -K_0^V \frac{1}{d_0^2} \sin(\theta_0) + K_1^V d_1^2 \sin(\theta_1) \tag{3.45}$$

Relating to the low-level wheel speed commands:

$$v_r = \frac{1}{R}\left[-K_0^V \frac{1}{d_0^2}\cos(\theta_0) + K_1^V d_1^2 \cos(\theta_1)\right] + \frac{L}{2}\left[-K_0^V \frac{1}{d_0^2}\sin(\theta_0) + K_1^V d_1^2 \sin(\theta_1)\right] \tag{3.46}$$

And

$$v_l = \frac{1}{R}\left[-K_0^V \frac{1}{d_0^2}\cos(\theta_0) + K_1^V d_1^2 \cos(\theta_1)\right] + \frac{L}{2}\left[K_0^V \frac{1}{d_0^2}\sin(\theta_0) - K_1^V d_1^2 \sin(\theta_1)\right] \tag{3.47}$$

Finally, in the case of presence of the motion commands from the haptic joystick, this effect can also be expressed in terms of potential fields vectors. If the motion command from the operator is shown by $\vec{V_j}$, based on the methodology of superposition in motor schema theory, the ultimate vector that determines the robot's motion can be shown as:

$$\vec{v} = \vec{V_0} + \vec{V_1} + \vec{V_j} \tag{3.48}$$

Now that different behaviors for the robot are defined, these micro scale rules are going to be used as building blocks for a macro scale architecture in which different roles are defined for the robot and the human operator. Based on the roles of these two, different levels of autonomy are presented and the performance of the robot is investigated under each circumstance.

3.2.2. Fuzzy Reactive Control

The motivation for using Fuzzy Logic Control (FLC) in navigation of wheeled mobile robot is to make use of human expert knowledge in making decisions under unpredicted circumstances. A human operator is able to drive any vehicle in virtually any kind of environment by integrating his perception of the physical environment with his subjective judgement of the conditions (antecedents), without needing the exact information about locations and sizes of objects in the environment Fuzzy reactive control for wheeled mobile robots .

The theory of fuzzy logic systems is inspired by human capability to operate according to perception-based information [80]. Rule-based fuzzy logic provides a scientific formalism for reasoning and decision-making in cases where uncertainties and unpredictability are prevalent. Fuzzy logic provides a formal methodology for representing and implementing the human expert’s knowledge and perception-based actions. Using the fuzzy logic framework, the strengths of human reasoning and decision making can be formulated by a set of simple IF (antecedent)–THEN (consequent) rules, coupled with easily understandable and natural linguistic representations. Therefore, fuzzy rule-based systems are capable of generating actions based on perceptions. The linguistic variables in the IF-THEN rule set are defined by fuzzy sets in accordance with user-defined membership functions. The main advantages of a fuzzy navigation strategy lie in the ability to extract heuristic rules from human experience, and to mitigate the need for an analytical model of the process [81].

The reactive fuzzy controller, proposed and applied in this study, is composed of two fuzzy logic controllers and a coordinator to merge the two behaviors. Similar to the previous section, go-to-goal fuzzy controller makes the robot approach the target assuming there are no obstacles in the environment; Fuzzy obstacle controller, on the other hand, generates obstacle avoidance commands based on obstacle information when the robot detecting obstacles via its range sensor. Behavior coordination is used

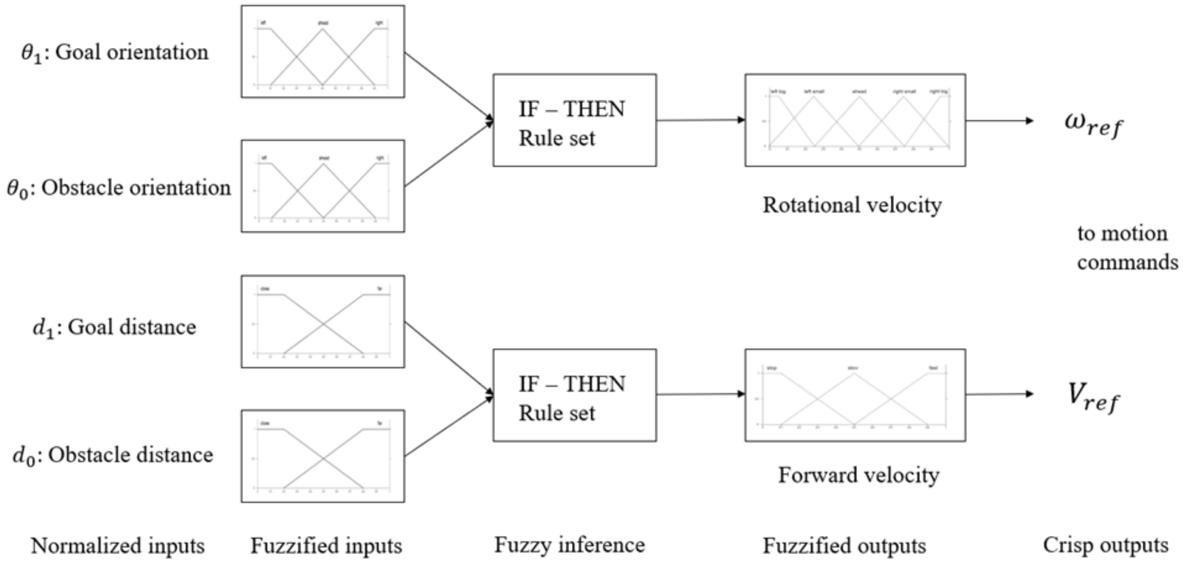


Figure 3.8. Fuzzy Reactive Control Interface

to generate robot's ultimate control command. The general overview of the Fuzzy Logic Controller and its different stages are shown in Figure 3.8.

As the first step in designing the FLC, the membership functions of the input variables for go-to-goal task is introduced. As shown in Figures Figure 3.9 and Figure 3.10, distance and orientation of the goal with respect to the robot are selected as the input variables. Three different values for the orientation, i.e. right, left and ahead and two values for the distance, far and close, are introduced. The parameters associated with the orientation of the closest obstacle for three trapezoidal Mamdani-type membership functions are as follows.

left: [-0.5 -0.1 0.1 0.5]
ahead: [0.1 0.5 0.5 0.9]
right: [0.5 0.9 1.1 1.5]

where the trapezoidal membership functions of the distance of the obstacle are:

close: [-0.8 -0.2 0.2 0.8]
far: [0.2 0.8 1.2 1.8]

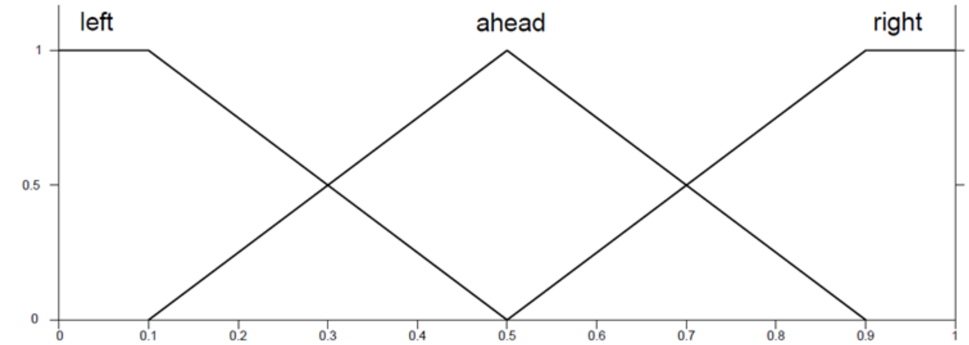


*Figure 3.9.*Orientation input membership functions

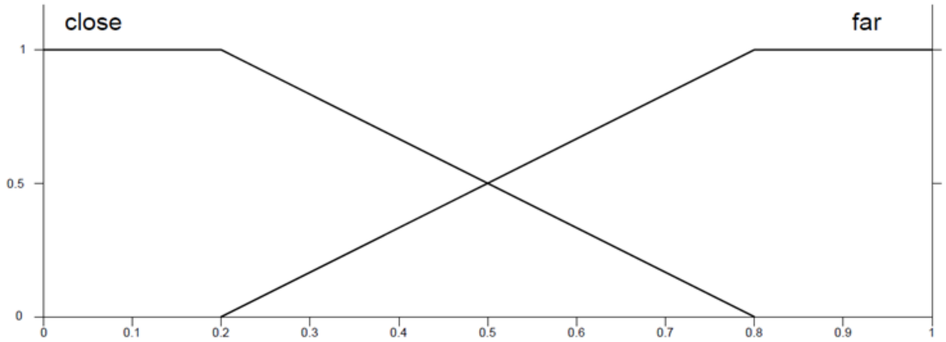


Figure 3.10. Distance input membership functions

The same configuration of membership functions can also be used for the obstacles orientation and distance of the closest obstacle with respect to the mobile robot:

Once the membership functions of the input variables are constructed, the membership functions of the outputs can also be defined. The membership functions of the reference rotational velocity ω_{ref}, and forward velocity V_{ref} are shown in Figures 3.11 and 3.12.

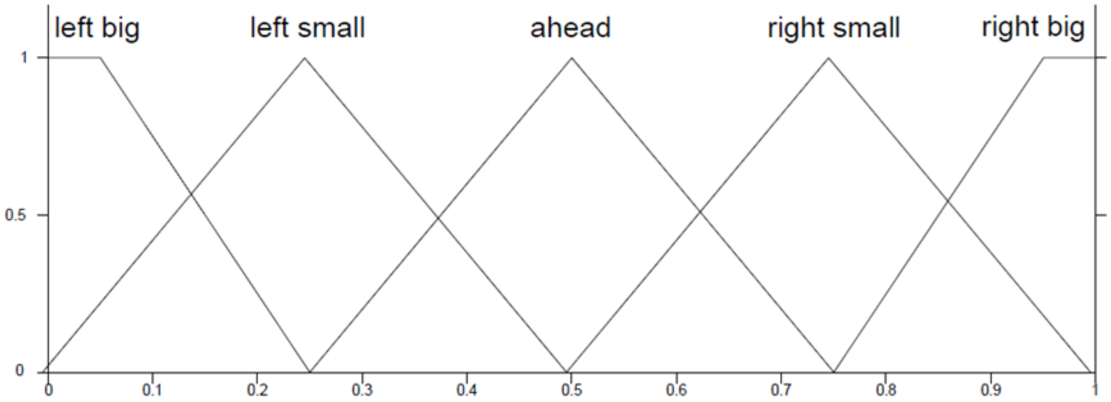


Figure 3.11. Angular velocity output membership functions

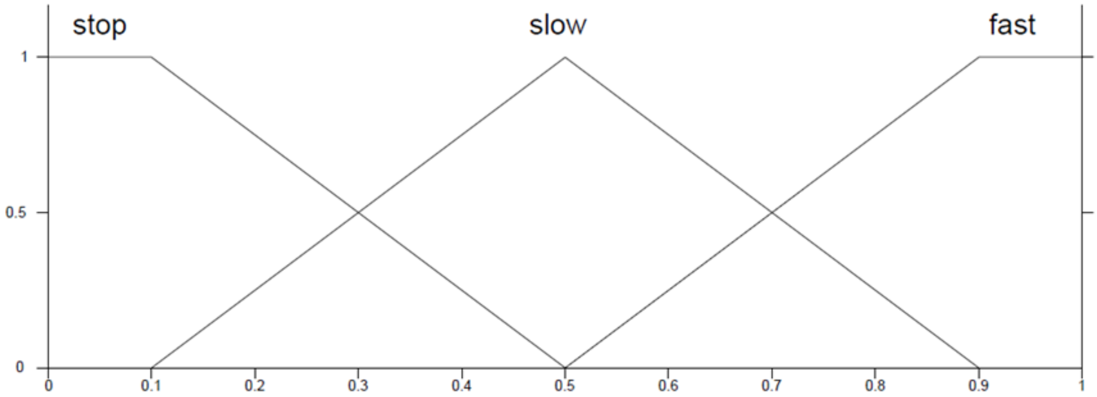


Figure 3.12. Forward velocity output membership functions

The parameters corresponding triangular and trapezoidal membership functions of the rotational velocity are as follows:

left big: [-0.25 -0.05 0.05 0.25]
left small: [0 0.25 0.5]
ahead: [0.25 0.5 0.75]
right small: [0.5 0.75 1]
right big: [0.75 0.95 1.05 1.25]

and the membership function parameters for forward velocity are:

stop: [-0.5 -0.1 0.1 0.5]
slow: [0.1 0.5 0.5 0.9]
fast: [0.5 0.9 1.1 1.5]

Once the membership functions of all input and output variables are defined, the rule base for the FLC can now be introduced.

The following set is the rule base associated with rotational velocity:

1) IF *goal is on the left & obstacle is on the left*, THEN *turn is right small*.
2) IF *goal is on the left & obstacle is ahead*, THEN *turn is left big*.
3) IF *goal is on the left & obstacle is on the right*, THEN *turn is left big*.
4) IF *goal is ahead*, THEN *turn is ahead*.
5) IF *goal is on the right & obstacle is on the left*, THEN *turn is right big*.
6) IF *goal is on the right & obstacle is ahead*, THEN *turn is right big*.
7) IF *goal is on the right & obstacle is on the right*, THEN *turn is left small*.

The following list is the rule base associated with forward velocity:

1) IF *obstacle is close*, THEN *stop*.
2) IF *goal is close & obstacle is far*, THEN *move slow*.
3) IF *goal is far & obstacle is far*, THEN *move fast*.

The structure by which the fuzzy rules are inferred, can be explained by the following example. Assuming the normalized values for the input variables are obtained as follows:
Orientation of the goal w.r.t the robot, $\theta_1 = 0.3$ and orientation of the closest obstacle w.r.t the robot, $\theta_0 = 0.6$

Since the operator linking the antecedents for two input variables is AND operator, the minimum value for mapping into the output is selected.

The overall output set is obtained by adding all individual output fuzzy sets associated with each fuzzy rule.

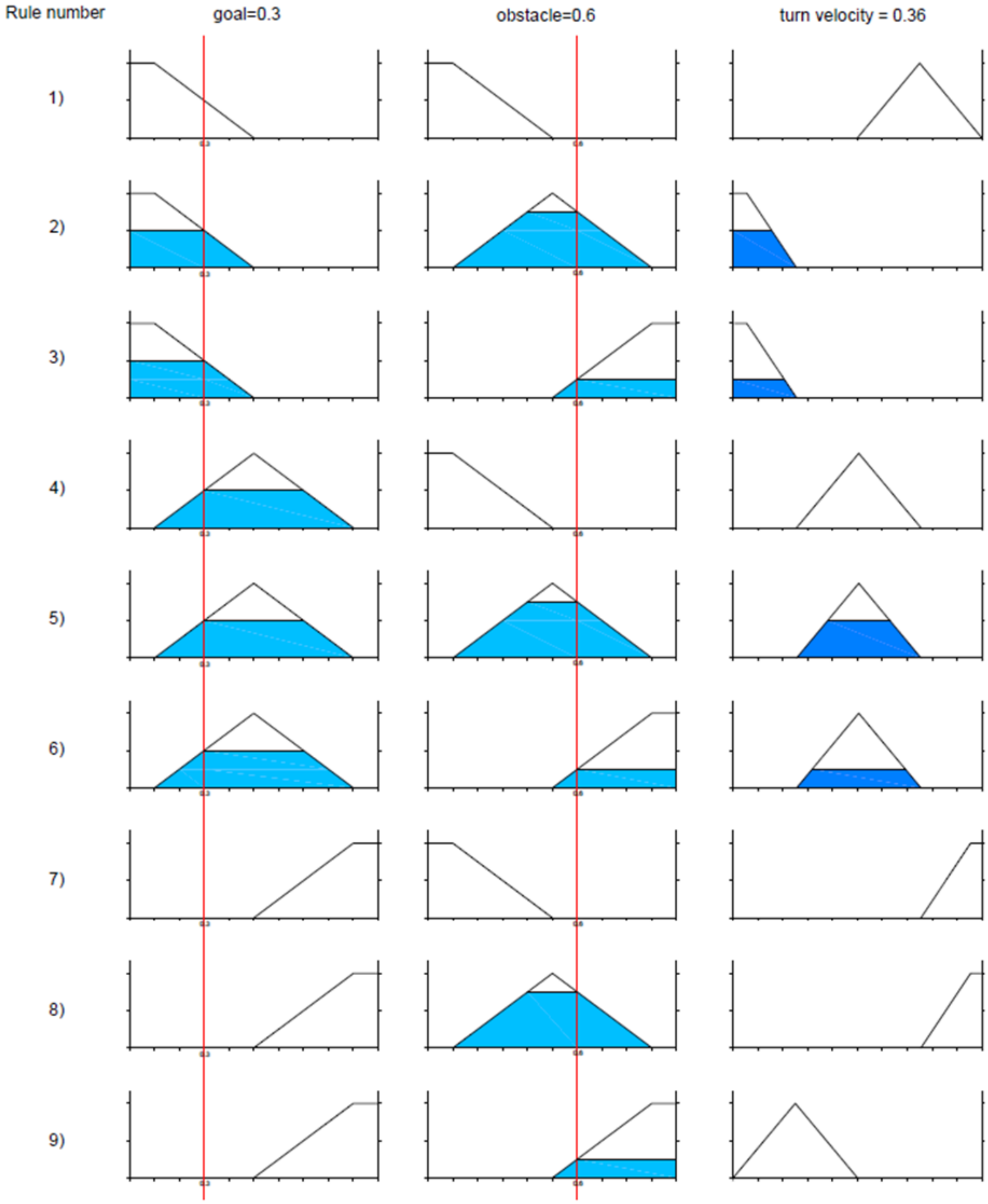


Figure 3.13. Fuzzy inference example

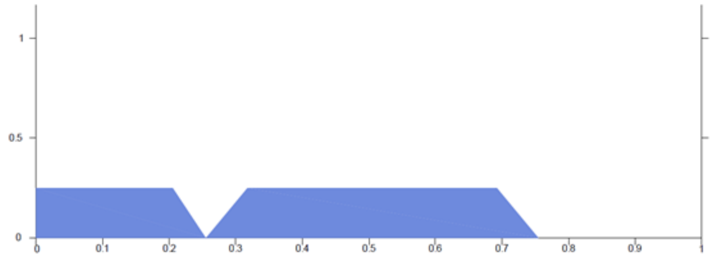


Figure 3.14. Example fuzzy inference output

In order to issue the motion commands, however, the output of the FLC need to be translated from fuzzy sets to numerical or crisp values. This process is called defuzzification and different approaches may be used for this purpose. In this study, in order to obtain the crisp value, Centroid (Center of Gravity) Method is used. The relation for calculating the crisp output is provided in Equation (3.49):

$$\omega_{ref} = \frac{\sum_{i=1}^{n} x_i A_i}{\sum_{i=1}^{n} A_i} \tag{3.49}$$

Where ω_{ref} is the rotational velocity, x_i is the center of area corresponding to the i'th section and A_i the area corresponding to the i'th section. And n is the number of the sections the output area is divided to. In the above example, $n = 3$, and:

$$\omega_{ref} = \frac{\sum_{i=1}^{3} x_i A_i}{\sum_{i=1}^{3} A_i} \tag{3.50}$$

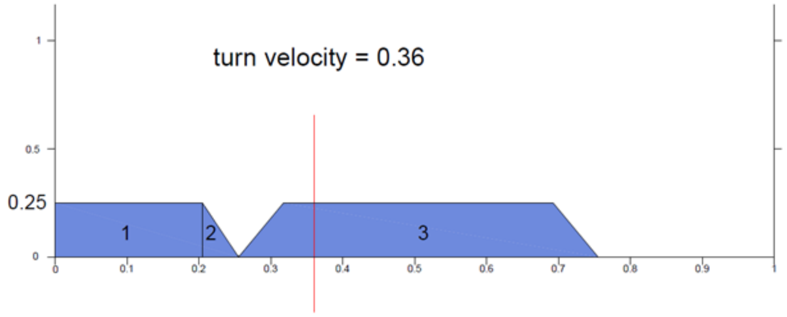


Figure 3.15. Center of area defuzzification method

The implementation of the FLC is realized by using Scikit-Fuzzy toolbox. Scikit-Fuzzy is an open source collection of fuzzy logic algorithms intended for use in the SciPy Stack, written in the Python computing language. This provides reliable tools for introducing membership functions, generating rule base, different defuzzification methods and all other necessary options to design and implement the Fuzzy Logic Controller.

Finally, the crisp values for the reference forward and angular velocity of the robot, V_{ref} and ω_{ref} are transferred to low-level motion commands, i.e. the rotational velocities of the robot wheels:

$$v_r = \frac{2V_{ref} + L\omega_{ref}}{2R} \tag{3.51}$$

$$v_l = \frac{2V_{ref} - L\omega_{ref}}{2R} \tag{3.52}$$

3.3. Cooperative Control

As its name suggests, the key concept in the cooperative control is to establish a reliable coordination between the human operator and the mobile robot aiming to accomplish a number of predefined mutual tasks with benefitting from both the operator and robot’s perception and decision-making capabilities. In this study, using the haptic device interface, the cooperative approach to robot control has been used to accomplish the go-to-goal and obstacle avoidance tasks of the robot. The core element in the proposed control strategy is adopting the behavior-based control method to construct the foundations necessary to tackle the aforementioned tasks. In the following sections, the components of this control strategy are explained in details.

3.3.1. Manual Control

As the name suggests, manual control assigns no task or authority to the mobile robot. The operator is in complete charge of controlling the robot. The human operator addresses both go-to-goal and obstacle avoidance behaviors. This means that the operator is not only required to get the robot to the final destination, but he also has to navigate the robot through a safe and collision-free path. The only perception channel for the operator is the frontal camera, which provides him with continual stream of the image of the area where the robot is heading to.

Figure 3.16. shows the overall layout of the system in the context of manual control. It is clear that there is no feedback data from the haptic joystick device to the operator. Therefore, the operator does not feel any force effect resulted by the obstacles. And the haptic device functions only as a regular joystick issuing motion commands in only one direction. The operator also has no idea about the nearby object except for those visible by the frontal camera, due to the fact that the range sensor has left idle in this scenario.

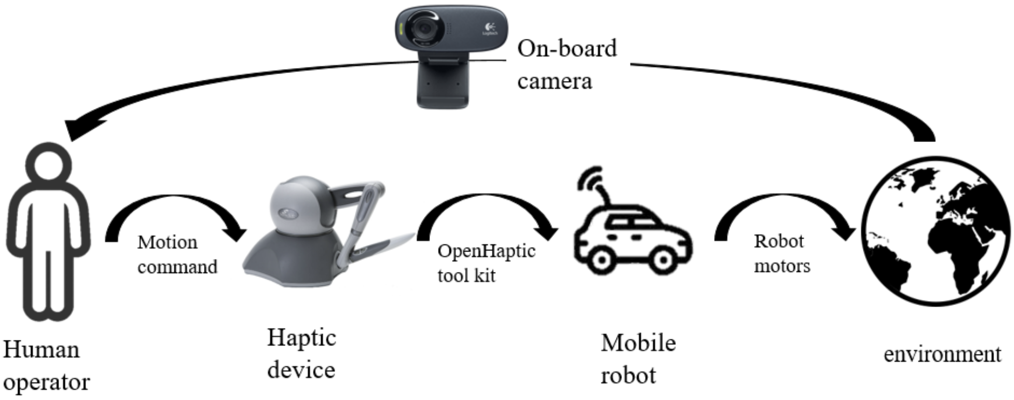


Figure 3.16. Layout of the system in manual control

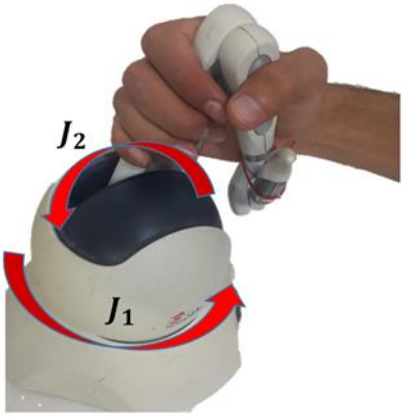


Figure 3.17 Phantom Omni device and its major joints

In order to issue motion commands to the robot, the coordinates of the joints of the haptic device is mapped into forward and rotational velocities, V_{ref} and ω_{ref}. As shown in Figure 3.17, two joints are utilized to control the robot: J_1 , the joint linking the fixed bottom base of the device to the rotating base; and J_2, the joint linking the first arm of the device to the rotating base. The other joints are tied fixed because neither their coordination have any effect on the motion of the robot, nor the force feedback exerted on the device has any components on those links.

In this study, throttle and pivot method for controlling differential mobile robots is applied to control the robot. In this scenario, the rotational velocity ω_{ref} is considered to be proportional to the coordinate of the first joint, J_1; which means based on the direction and magnitude by which the joystick is turned clock-wise (counter clock-wise), the robot also turns in clock-wise (counter clock-wise) direction:

$$\omega_{ref} \;= K_j^{\omega}\, J_1 \tag{3.53}$$

Where K_j^{ω} is the scaling factor mapping the joint coordinate value to rotational velocity.

The forward velocity is also considered to be proportional to the coordinate of the second joint, J_2; which means based on the direction and magnitude by which the joystick is pushed forward (backward), the robot moves in forward (backward) direction:

$$V_{ref} \; = K_j^V \, J_2 \tag{3.54}$$

Where K_j^V is the scaling factor mapping the joint coordinate value to forward velocity.

Hence, using (3.10) and (3.11), the rotational speeds of the right and left wheels can be computed:

$$v_r = \frac{2K_j^V \, J_2 + LK_j^{\omega} \, J_1}{2R} \tag{3.55}$$

$$v_l = \frac{2K_j^V \, J_2 - LK_j^{\omega} \, J_1}{2R} \tag{3.56}$$

In manual control the control system operates in an open-loop layout and all data stream (except the camera’s visual data) is in one direction, i.e. from the operator to the robot. It is also evident that the task load on the operator is considerably high in this situation because he is in full control of the robot and receives no assistance from the robot in sense of action or perception.

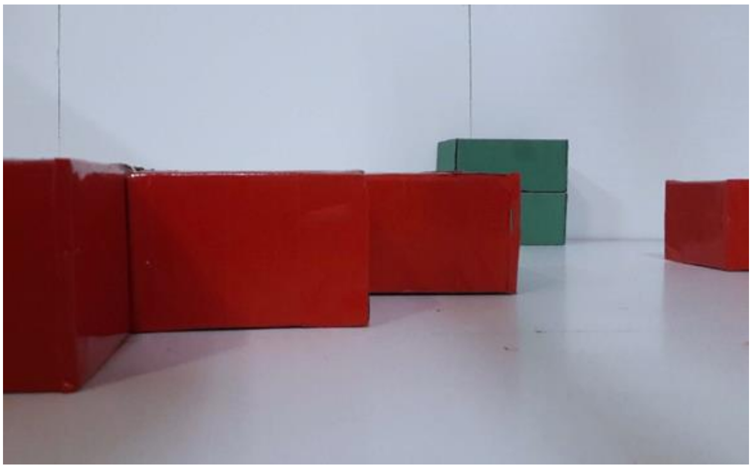


Figure 3.18. Frontal camera view

3.3.2. Fully-automated Control

This configuration is exactly opposite to the manual control in the level of autonomy spectrum. At this level, the mobile robot is in full charge of all operations and is responsible for both go-to-goal and obstacle avoidance behaviors. The robot implements all the actions including detecting obstacles and the goal, generating motion commands using the reactive control approach and navigating toward the goal while avoiding collisions with obstacles. In this situation, human operator is entirely out of the control loop and cannot interfere with any task at any stage of the operation. This level of autonomy is the representative of a fully automated system where the presence of a human operator is unnecessary. Figure 3.19 demonstrates the configuration of the system in this scenario.

In this part of the study, two distinct methodologies are proposed and applied in autonomous drive paradigm: reactive control based on force vector field and fuzzy reactive control.

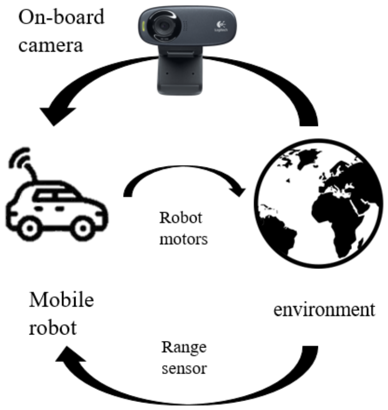


Figure 3.19. Layout of the system in full automation

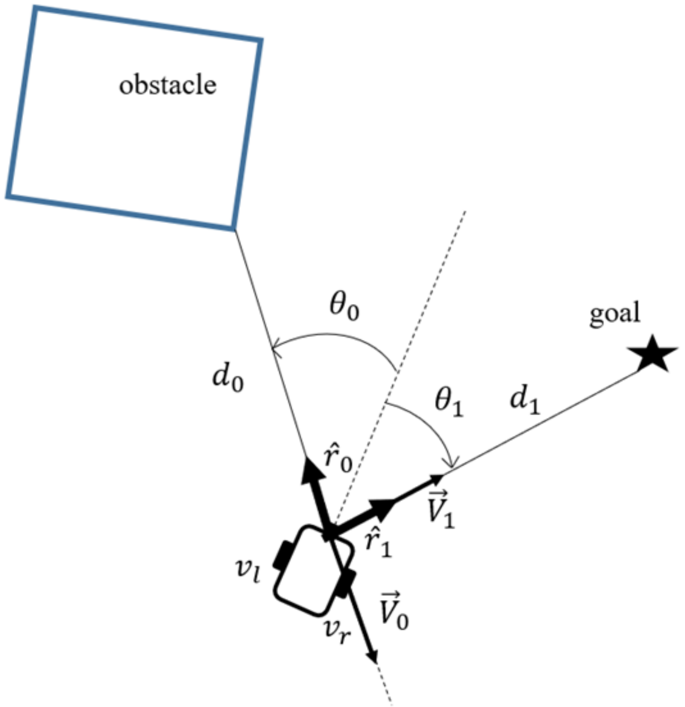


Figure 3.20. force vector affecting the robot

As discussed in the previous sections, the data from laser range sensor is used to detect the orientation and distance of the nearby obstacles to the robot. This data is then used to generate the vector corresponding the obstacle effect. The image from the frontal camera is also processed to provide the robot with relative position and distance of the goal with respect to the robot, which in turn, is used to generate the effect vector related to go-to-goal task in potential field methodology. The motion commands are then generated using the vector summation of the both behaviors and the robot moves towards the goal while avoiding the obstacles.

The principles of reactive control are discussed in section 3.2.1. Figure 3.20 illustrates the effects of goal and obstacle forces acting on the robot. The robot reacts to the external stimulus, i.e. the goal and the obstacles by generating force vectors associated with stimulus and producing the motion vectors by vector summation, which in turn, are transferred to low-level motion commands, i.e. wheels rotation speed:

$$\vec{v} = \vec{V}_0 + \vec{V}_1 \tag{3.57}$$

Equations (3.58) and (3.59) demonstrate the relationship by which the relative orientations and distances of the goal and the closest obstacles are related to the rotational speeds of the right and left wheels of the mobile robot:

$$\begin{aligned} v_r = & \frac{1}{R}\left[-K_0^V \frac{1}{d_0^2}\cos(\theta_0) + K_1^V d_1^2 \cos(\theta_1)\right] \\ & + \frac{L}{2}\left[-K_0^V \frac{1}{d_0^2}\sin(\theta_0) + K_1^V d_1^2 \sin(\theta_1)\right] \end{aligned} \tag{3.58}$$

And

$$\begin{aligned} v_l = & \frac{1}{R}\left[-K_0^V \frac{1}{d_0^2}\cos(\theta_0) + K_1^V d_1^2 \cos(\theta_1)\right] \\ & + \frac{L}{2}\left[K_0^V \frac{1}{d_0^2}\sin(\theta_0) - K_1^V d_1^2 \sin(\theta_1)\right] \end{aligned} \tag{3.59}$$

3.3.3. Supervisory Control

Unlike the previous levels of autonomy, where all the tasks were assigned to only one of the agents (either the robot or the human operator), in this configuration both the mobile robot and the operator share the tasks in a way that the accomplishing the tasks is due to the collaboration between the two. The key concept at this level is the fact that the mobile robot takes charge of go-to-goal behavior and the operator is assigned with obstacle avoidance behavior.

Figure 3.23. shows the overall layout of the system at this level of autonomy. The only sensory data that the robot consumes is the visual data coming from the frontal camera. The robot, after processing the image, which includes computing the relative orientation and distance of the goal, moves towards the goal's position. The operator, on the other hand, is fed with the data from both the camera and the range sensor data on computer screen. Although the operator is able to issue motion commands via the haptic device, he is supposed to monitor the system and intervene only at points where

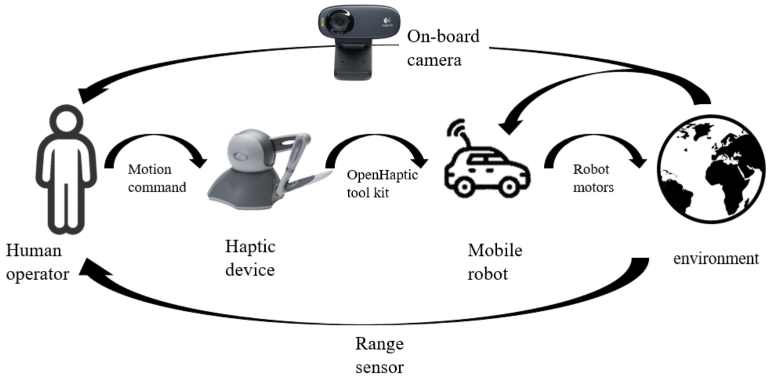


Figure 3.21. Layout of the system in supervisory control

a possible collision to the obstacles is foreseen. It should be noted that similar to the previous cases, at this stage the haptic device acts only as a regular joystick and does not exert any force feedback on the operatorThe overall force vector acting on the robot can be written as:

$$\vec{v} = \vec{V}_1 + \vec{V}_j \tag{3.60}$$

Where $\vec{V}_1$ is the force exerted on the robot due to the attractive effect of the goal and $\vec{V}_j$ is the force vector exerted by the haptic device (joystick). Using the vector superposition method presented in section 3.2.1, the overall reference forward and angular velocities can be computed:

$$V_{ref} = K_1^V d_1^2 \cos(\theta_1) + K_j^V J_2 \tag{3.61}$$

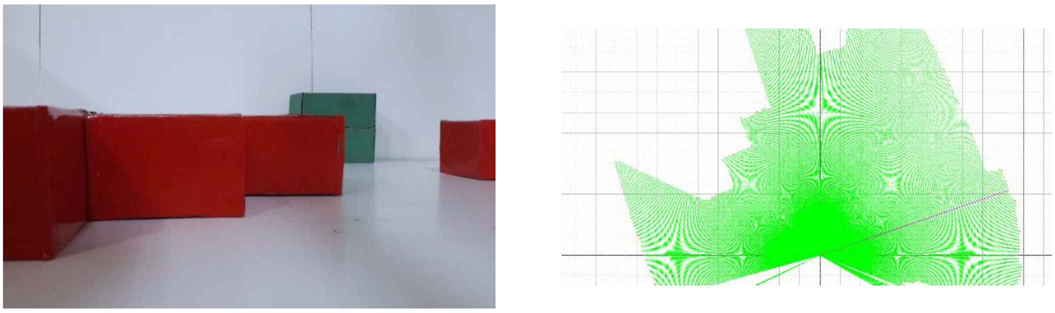
$$\omega_{ref} = K_1^\omega d_1^2 \sin(\theta_1) + K_j^\omega J_1 \tag{3.62}$$

Where, d_1 is the relative distance of the goal to the robot and θ_1 is its relative orientation. J_1 is the coordinate of the first (turn) joint and J_2 the coordinate value of the second (throttle) joint. K_1^V, K_1^ω, K_j^V, and K_j^ω are the constant scaling factors.

The wheels speed can be expressed as:

$$v_r = \frac{1}{R}\left(K_1^V d_1^2 \cos(\theta_1) + K_j^V J_2\right) + \frac{L}{2R}\left(K_1^\omega d_1^2 \sin(\theta_1) + K_j^\omega J_1\right) \tag{3.63}$$

$$v_l = \frac{1}{R}\left(K_1^V d_1^2 \cos(\theta_1) + K_j^V J_2\right) - \frac{L}{2R}\left(K_1^\omega d_1^2 \sin(\theta_1) + K_j^\omega J_1\right) \tag{3.64}$$



(a) Frontal camera view (b) Range sensor stream

Figure 3.22 Visual stream in manual control

3.3.4. Assistive Control

Figure 3.23. illustrates the overall configuration of the system in assistive control scenario. At this level of autonomy, both the robot and the human operator are assigned wit almost equal share of the overall tasks and can generate motion commands. While the go-to-goal task is assigned exclusively to the operator, the robot is only responsible for avoiding obstacles. Therefore, the two major behavior are divided between the two agents.

Not only the level of autonomy is shifted to the middle of the spectrum in this scenario, but there are also a number of other major differences which should be taken into the account. First, unlike the previous cases, the haptic joystick now fulfills both the task of issuing motion commands to the robot and exerting force feedback to the operator.

Second, the haptic device performs the vector summation process through its mechanical joints. Once the vector corresponding the obstacles is obtained, it directly applies the force on the haptic device.

Figure 3.23 demonstrates the repulsive effect of the obstacle acting on the robot and Figure 3.24 shows the joints of the haptic device used for issuing motion commands by the operator.

If the force vector associated with the obstacle effect is represented by $\vec{V}_0$, where:

$$\vec{V}_0 = -K_0 \frac{1}{d_0^2} \hat{r}_0 \tag{3.65}$$

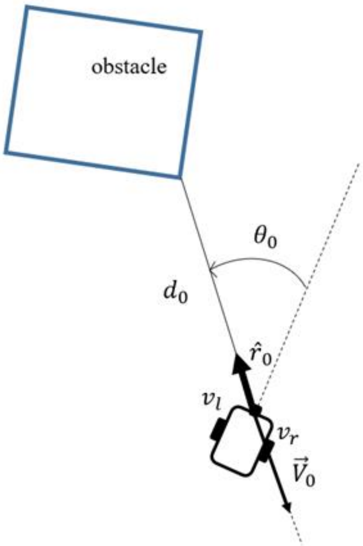


Figure 3.23. Feedback force effect by obstacle effect

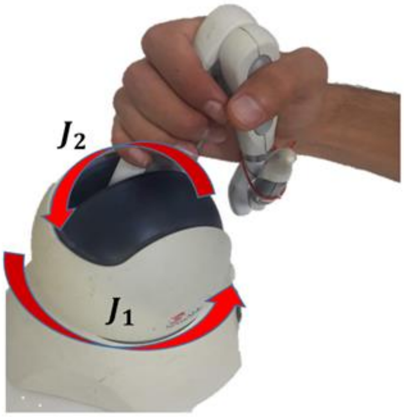


Figure 3.24. Actuated joints

The repulsive force of the obstacle, $\vec{V}_0$ can again be decomposed to two components, one on direction of motion and the other perpendicular to it:

$$V_0^V = -K_0 \frac{1}{d_0^2} \cos(\theta_0) \tag{3.66}$$

$$V_0^\omega = -K_0 \frac{1}{d_0^2} \sin(\theta_0) \tag{3.67}$$

These force components are mapped to the components of forces acting on the operator via the haptic joystick device. In this case, two force components are applied on the device: One through joint 1, whose magnitude is proportional to V_0^ω and; the other through joint 2, whose magnitude is proportional to V_0^V:

$$F_1 = K_\omega V_0^\omega \tag{3.68}$$

$$F_2 = K_V V_0^V \tag{3.69}$$

K_ω and K_V are positive constant factors scaling the magnitudes of forward and rotational components of the force vector associated with the obstacle effect to the magnitudes of the forces applied on the haptic device joints respectively. It should also be noted that the directions of these forces are identical, i.e. creating the force effect in the direction from obstacle to the robot.

Rewriting based on d_0 and θ_0:

$$F_1 = -K_\omega K_0 \frac{1}{d_0^2} \sin(\theta_0) \tag{3.70}$$

$$F_2 = -K_V K_0 \frac{1}{d_0^2} \cos(\theta_0) \tag{3.71}$$

When this vector is added with the force applied by the operator, the final coordinates of the haptic joystick is mapped to the motion commands and issued to the robot. Therefore, one significant variation in this scenario is the mechanical bound in haptic device that functions as the behavior arbitrator. Use of this mechanical bound provides a number of advantages which are explained in the next chapters.

Similar to manual control scenario, the motion commands can be computed as follows.

$$\omega_{ref} = K_j^\omega J_1 \tag{3.72}$$

Obtaining the wheels velocities:

$$v_r = \frac{2K_j^V J_2 + LK_j^\omega J_1}{2R} \tag{3.74}$$

$$v_l = \frac{2K_j^V J_2 - LK_j^\omega J_1}{2R} \tag{3.75}$$

$$V_{ref} = K_j^V J_2 \tag{3.73}$$

Where K_j^V and K_j^ω are the scaling factor mapping the joint coordinate values to forward and angular velocities of the robot, respectively.

The division of the behaviors between the robot and the operator is another key feature in this control mechanism. The robot, using the data from range sensor handles the local task i.e. obstacle avoidance, while the operator uses the visual data stream from the frontal camera and is in charge of global task of going to the goal. This separation of tasks provides the operator with the opportunity to plan ahead and navigate through the paths that provides a higher level of efficiency to the overall system.

3.3.5. Dynamic Shared Control

In this thesis study, various types of control frameworks at different levels of autonomy for the control of mobile robot have been introduced. These methods range from pure manual operation to fully autonomous drive. Despite all the differences in the way by which the human operator and mobile robot share the autonomy, there is one key factor that remains valid for all levels of autonomies: the contribution level of each agent remains constant throughout the whole operation of the system. As seen in previous sections and as it will be demonstrated by test experiments, each level of autonomy provides major strengths and suffer from certain drawbacks. The idea in this section is to propose a specific cooperation scheme in which the performance of the human-machine system is maximized (or at least improved) by adjusting the contribution of the two agents in a way that their highest potentials are exploited.

A dynamic shared control technique is introduced to solve the problems associated with the limited contribution of the agents to the tasks. It proposes a satisfactory mechanism for autonomy level adjustment that covers more than only a single point on autonomy spectrum. The use of dynamic shared autonomy enables the level of autonomy to change during the robot's operation by introducing gains to the motion commands from both the operator and the robot itself. These gains, which are between 0 and 1, with vote zero meaning no contribution, and one representing the full control on the system. This method employs the concept of multi-value rather than simple binary value of the previous methods, in which the contribution gains were either zero or one, with no value between.

If the gain corresponding to the authority level of the robot is represented by G_r , and the gain associated with the contribution of the operator is denoted by G_o, the reference forward and rotational velocities can be expressed as:

$$V_{ref} = G_r\left[-K_0^V \frac{1}{d_0^2} \cos\,(\theta_0) + K_1^V d_1^2 \cos(\theta_1)\right] + G_o\big(K_j^V\, J_2\,\big) \tag{3.76}$$

$$\omega_{ref} = G_r\left[-K_0^V \frac{1}{d_0^2} \sin(\theta_0) + K_1^V d_1^2 \sin(\theta_1)\right] + G_o\big(K_j^\omega\, J_1\,\big) \tag{3.77}$$

Where

$$0 \leq G_r, G_o\ \leq 1 \tag{3.78}$$

And

$$G_r + G_o = 1 \tag{3.79}$$

Note that if

$$G_r = 0 \tag{3.80}$$

Equations (3.76) and (3.77) change to the motion control inputs of the manual drive; whereas if

$$G_o = 0 \tag{3.81}$$

The shared control strategy changes to fully autonomous control.

In order to assign the gains for the two effectors, the forward velocity of the mobile robot is taken into the account. It is seen that in situations where the forward velocity of the robot reduces, the robot is inclined to fail to initiate the proper motion command due to cancelling out the go-to-goal and obstacle avoidance behaviors. One major instance is the local minima where the force vectors associated with go-to-goal and obstacle avoidance behaviors grow similar in magnitude and act in opposite directions. In such cases, the operator has the ability to intervene and apply motion command through the haptic device. Hence, more authority can be assigned to the operator. The gain related to the operator, therefore is expressed as the following:

$$G_o = \left|\frac{V_{ref}}{V_{max}}\right| \tag{3.82}$$

Where, V_{max} is the maximum forward velocity the robot can hypothetically achieve. And V_{ref} is the reference forward velocity.

On the other hand, as the forward velocity of the robot increases, more authority can be assigned to the robot because it can be concluded that the robot does not suffer from contradicting stimulus to the extent that go-to-goal and obstacle avoidance force vectors do not hinder the robot from taking decisive actions. Therefore, more authority can be assigned to robot and consequently more task load can be spared from the operator. Hence, the gain associated with the operator's contribution can be expressed as:

$$G_1 = 1 - \left|\frac{V_{ref}}{V_{max}}\right| \tag{3.83}$$

Robot and operator gains are shown as functions of V_{max} in Figure 3.25.

It should be noted that similar to the assistive control, the feedback force generated by the obstacle effect is exerted on the operator as explained in previous section.

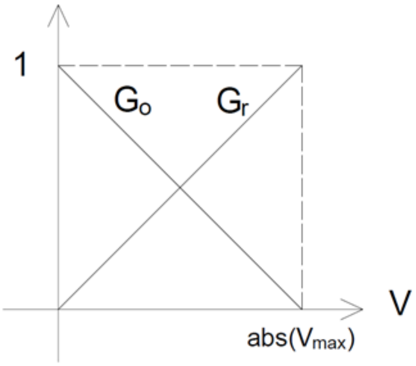


Figure 3.25. Robot and operator gains in dynamic shared control

CHAPTER 4

EXPERIMENTAL SETUP

In this chapter, the major components of the experimental setup and the fundamental connections between them are presented. The setup includes the wheeled mobile robot, the haptic interface, and the software platform used as a communication link between the robot and the haptic device. Two computers are used to fulfill the computation and communication tasks. The workstation is the one responsible for incorporating the processes requiring more hardware and software performance. The operator also is in contact with the workstation through both the haptic device and the camera’s stream. The computer on the robot, on the other hand, is devoted to performing lower-level tasks that are needed to be carried out immediately. A BeagleBone Black is used to accomplish this task.

The hardware and software components are specifically selected and developed in a way that their individual operation and mutual interaction maximizes the reliability of the system and its speed of reaction to both execution commands and environmental stimulus.

The role and specifications of each major components of the experimental setup are discussed in details in the following sections.

4.1. Development of the Mobile Robot

Mobile robots in research areas appear in numerous configurations. They can be either pre-fabricated or custom made. The one used in this study belongs to the latter category. It has been manufactured by assembling the required components on a circular fiberglass chassis. The components of the mobile robot are selected in a way that they all aim maintaining the optimal balance between performance on one hand and size and weight considerations on the other hand.

The following chapters are devoted to providing brief descriptions of the major components used in constructing the mobile robot.

4.1.1. Controller

As the brain of the mobile robot, a control board has been used to fulfill all the required processing tasks. The unit which can completely achieve this task has to possess some key features such as sufficient speed of computation, flexibility to link with other robotic components and the small size. For this purpose, BeagleBone Black R3 development platform has been selected because it includes all the mentioned specifications besides being low-cost, open source and community-supported. Figure 4.1. illustrates the major components of BeagleBone Black R3.

Using an AM335x 1GHz ARM® Cortex-A8 microprocessor, it provides a more powerful processing capacity compared with its alternatives. Some other features of the controllers include 512MB DDR3 RAM, 4GB 8-bit eMMC on-board flash storage which can hold the operating system, a 3D graphics accelerator, a NEON floating-point accelerator and two PRU 32-bit microcontrollers [82].

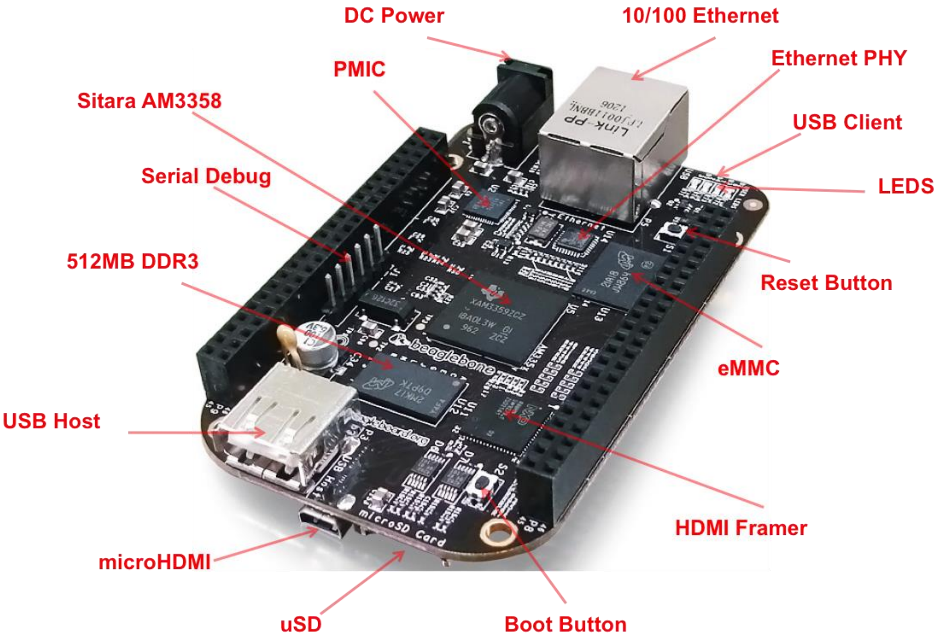


Figure 4.1. Beaglebone Black R3

Figure 4.2. ProFuse 7.4 V Li-Po Battery and 5V 3A Voltage Regulator Card

BeagleBone Black (BBB) ships with the Debian GNU/Linux™ in onboard eMMC Flash to start evaluation and development. A number of other operating systems are also supported.

Another powerful tool that the BBB is equipped with, is the General Purpose Input/Output (GPIO) pins which can be either inputs or outputs. These pins can significantly facilitate the board's communication capabilities with a vast number of electronic interfaces.

To power up the system, there are multiple options such as a 5V mini-USB connector on-board, through a 5V DC Adapter Jack or a using a light-weight battery. In this work, in order to provide the system with the maximum mobility, a single 7.4 V Lithium – Polymer (Li-Po) battery is used as the main source for the whole system, including BBB. The battery has a capacity of 2800mah and can generate current output up to 25A.

In order to regulate the battery's output voltage to the point that is safe to the working devices, an LM2596-5V regulator card has been used. This card is able to maintain a discharge rate of 3A, which is crucial to sustain the performance of all units on the robot, without any failures resulted from power shortage.

4.1.2. Wireless Connection

Another powerful feature of BBB is its Universal Serial Bus (USB) host. This port is used to connect the microcontroller to a USB hub, which in turn provides the connection to a number of essential components. Additionally, since the board itself

Figure 4.3. 7 Port Slimline USB 2.0 Hub and TP-Link Wireless Adapter

cannot provide enough current to all connected USB devices, the hub has to separately be connected to a power source.

One of the essential units which has to be connected to the board through USB is the wireless module. In this work, TP-Link Wireless Nano USB Adapter has been used. There are two main reasons to select this adapter. First, there are a limited number of wireless modules that operate stably with BBB and Nano series are one group of them. Second, the small size and low power consumption of this device makes it perfect to be utilized in such projects because of the limitations in weight, size and power consumption.

Once the wireless module has been connected to BBB via the USB hub, it allows the mobile robot establish a robust communication with the workstation through Wi-Fi’s wireless local area networking based on the IEEE 802.11 standards. This guarantees the mobility of the overall system besides enhancing its flexibility.

4.1.3. Motor Controller

The next major component of the mobile robot is the motor controller. A Motor Controller is a device that acts as intermediary between the microcontroller (BeagleBone Black), batteries and the motors. A motor controller serves to control the performance of the motors. One major reason to use a motor controller is the fact that the BBB cannot provide the amount of current required by the motors depending on the torque they have to provide.

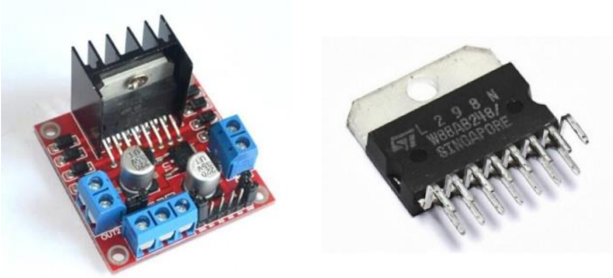


Figure 4.4. L298N Dual Motor Controller Module

In this study, an L298N Dual Motor Controller Module is selected to be used in the experimental setup. This controller is amongst the most popular general-purpose motor controllers for brushed, DC motors. This motor controller from Tronixlabs Australia®, is based on the L298N heavy-duty dual H-bridge controller, which can be used to either drive two DC motors at up to 2A each, with a voltage between 5 and 35V DC - or a single stepper motor. The controller has fast short-circuit protection diodes, and a built-in heat sink to keep the L298N operating without difficulty. An onboard 5V regulator is another feature accompanied the module.

The L298N is a high voltage, high current, dual full bridge driver designed to accept standard Transistor-Transistor Logic (TTL) levels and drive inductive loads. This module uses Dual H-Bridge Motor Controller. An H-Bridge is a circuit that can drive a current in either polarity and be controlled by Pulse Width Modulation (PWM). PWM is a means in controlling the duration of an electronic pulse, which makes it possible to control the speed and direction of the motion of the robot.

4.1.4. Range Sensor

One of the most important tasks of an autonomous mobile robot is to acquire reliable knowledge about its environment. This is usually completed by taking measurements using sensors and then extracting meaningful information from those measurements.

The fundamental characteristics of the sensors has been discussed in Chapter 1. A more comprehensive discussion of most common sensors used in mobile robots and the strategies for extracting information from the sensors are provided in [28].

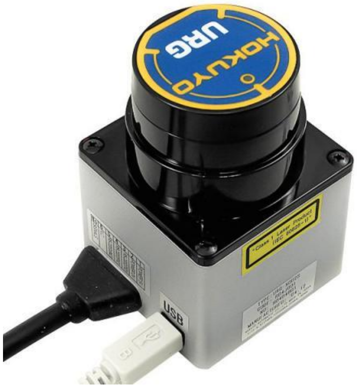


Figure 4.5. Hokuyo URG-04LX Scanning Laser Rangefinder

Since the detection of obstacles is a major part of this study, the range sensor has to be integrated into the mobile robot platform. One of the pioneer producers of scanning rangefinders is Hokuyo [83]. Along with its relatively low price, Hokuyo's URG-04LX-UG01 is an ideal choice for students and researchers who are involved in autonomous robotics studies.

Table 4.1. *Some key features of URG-04LX-UG01*

Power source	**5VDC±5%(USB Bus power)**
Light source	Semiconductor laser diode (λ=785nm), Laser safety class 1
Measuring area	20 to 5600mm
Accuracy	60 to 1,000mm : ±30mm, 1,000 to 4,095mm : ±3% of measurement
Angular resolution	Step angle : approx. 0.36°(360°/1,024 steps)
Scanning time	100ms/scan
Interface	USB2.0/1.1
Command System	SCIP Ver.2.0
Vibration resistance	10 to 55Hz, double amplitude 1.5mm each 2 hour in X, Y and Z directions
Impact resistance	196m/s2, Each 10 time in X, Y and Z directions
Weight	Approx. 160g

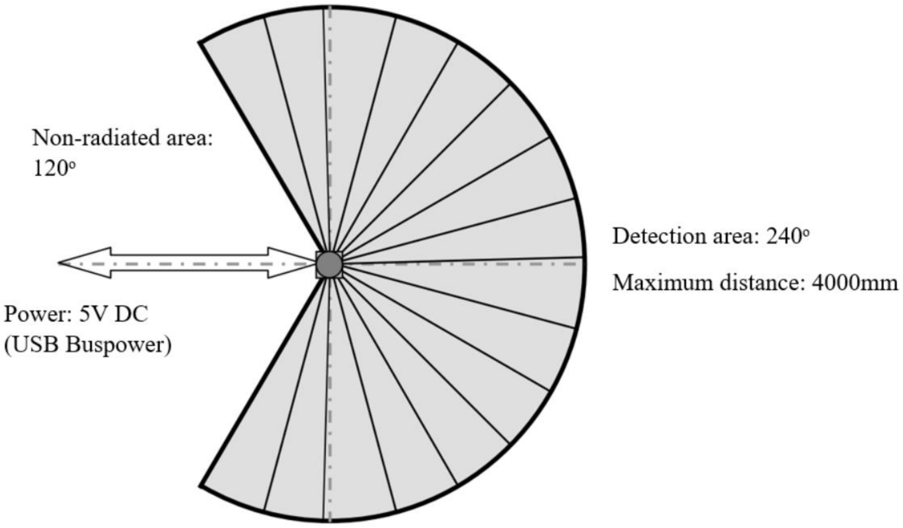


Figure 4.6. Detectable area of URG-04LX-UG01 for white Kent sheet

4.1.5. Camera

In this study, providing the operator with continuous real-time vision of the environment is one of the most important tasks that have to be addressed efficiently. In tele-operation scenario, the human operator needs the visual data to issue the appropriate direction commands to the mobile robot. Using overhead cameras in mobile robots for path planning and localization tasks of mobile robots has been investigated in many research works such as [84].

For this purpose, a Logitech's HD Webcam C270 is linked to the USB hub. Possessing a 3.0 megapixels lens, the selected camera is able to capture videos up to 1280×720 pixels. However, this resolution has been reduced to 320×240 due to the limitation of BBB process speed. Hence, the delay problem is handled successfully. The camera is mounted in front section of the robot and its capturing position can be modified thanks to pan, tilt, and zoom controls.

In order to provide a simple interface for capturing and viewing video from the camera, with a reliable compatibility with linux, *guvcview* package has been used. This interface aims establishes a control interface based on Gtk3 or Qt5, depending on

Figure 4.7. Logitech HD Webcam C270

the build configuration. In addition, to maximize the computational efficiency, Open Source Computer Vision (OpenCV) has been used. OpenCV is a library of programming functions mainly aimed at real-time computer vision. Thanks to OpenCV, video streaming task is both simplified and accelerated in a way that it maximizes the use of BBB's image processing capabilities [85], [86]. Being compatible with Python programming language and ROS, it provides valuable tools to enhance the real-time image processing performance. As the mobile robot moves around, it transmits video and audio over its Wi-Fi interface to the workstation and then to the operator in real time.

Figure 4.8. shows the final configuration of the mobile robot.

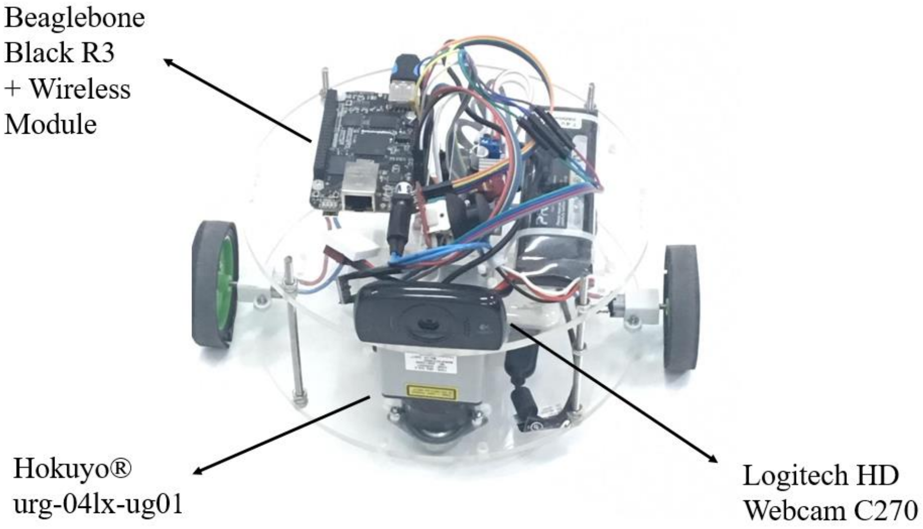


Figure 4.8. The mobile robot

4.2. Haptic Interface

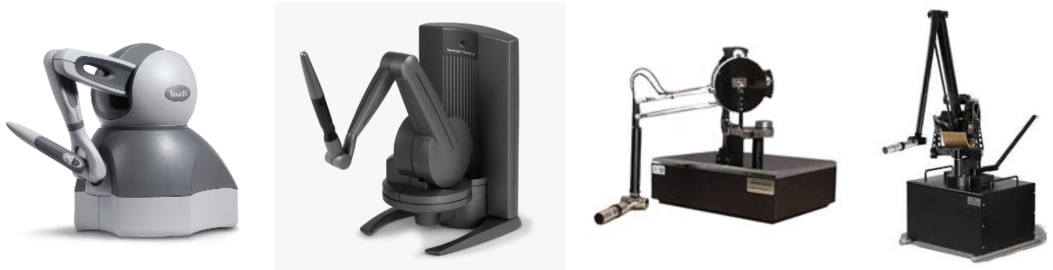
Haptics is the science of integrating the sense of touch and control into computer applications through force feedback. By using special input and output devices—called haptic devices—with a haptically enabled application, users can feel and manipulate virtual three-dimensional objects [87].

In this study, inexpensive Sensable Phantom Omni haptic device is used as a force-feedback joystick in order to transmit direction commands to the mobile robot and apply forces on human operator according to the distance to the existing obstacles in

the vicinity of the robot. The tele-operation scenario is, therefore, fulfilled by establishing a haptic interface between the operator and the robot

4.2.1. Phantom Omni Haptic Device

Among the rapidly expanding research areas for haptics, robotic and tele-operated are the ones that have been studied intensively in the recent years. Haptics is the term used to describe the science of integrating the sense of touch with computer applications using force feedback. By using particular input and output devices, which are known as haptic devices, the user can feel and manipulate virtual three-dimensional objects and environments with haptically-enabled applications. In other words, a haptic device is a motorized device that exerts force on the operator's hand, resulting the user to feel virtual 3D objects on the computer's screen.



Touch

Touch X

Phantom Premium 1.5 6DOF

Phantom Premium 3.0 6DOF

Figure 4.9. 3D Systems' haptic devices

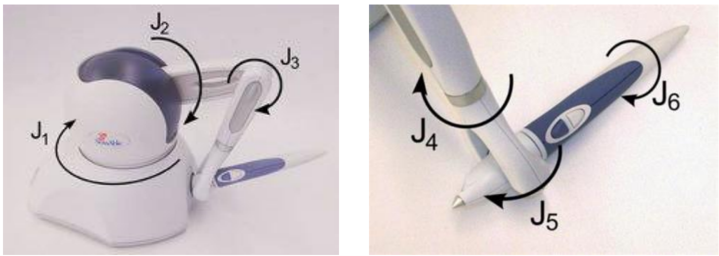


Figure 4.10. Phantom Omni Joints

In order to meet the need of academic and commercial researchers, the 3D Systems products are among the most well-reputed haptic devices. The most cost-effective haptic device available today is the Touch model. This mid-range professional haptic device evolved from research done by Thomas Massie and Dr. Kenneth Salisbury at MIT and had been known as PHANTOM Omni model until its producer company, SensAble Technologies sold the rights of the product to 3D Systems. Its portable design and compact footprint are additional features that make it a preferable choice to be used in this experimental setup as both the source of motion command and feedback force. Touch device possesses six degrees of freedom with positional sensing as well as three degrees of freedom in force feedback. As shown in Figure 3.12., J_1, J_2 and J_3 joints are active joints, which means they are actuated by motors and can be rotated according to the force commands. The remaining three joints, J_4, J_5 and J_6 are passive joints without any force exertion quality.

Table 4.2. *3D system Touch device's major specifications*

Force feedback workspace	> 160 W x 120 H x 70 D mm
Footprint	~168 W x 203 D mm
Weight (device only)	~1.8 kg
Nominal position resolution	~ 0.055 mm
Backdrive friction	<0.26 N
Maximum applicable force at orthogonal arms position	3.3 N
Continuous applicable force (24 hrs.)	> 0.88 N
Stiffness	X axis > 1.26 N/mm Y axis > 2.31 N/mm Z axis > 1.02 N/mm

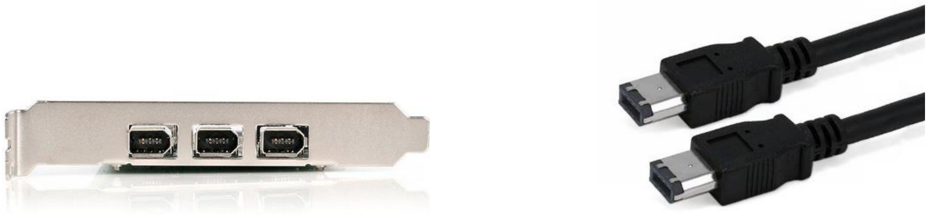


Figure 4.11. FireWire IEEE 1394 Port and Cable

The interface used in Touch device is IEEE 1394, commonly known as FireWire. This interface is a standard connection type used for many other different kinds of electronic devices such as digital video cameras, printers and scanners, and external hard drives. FireWire connection’s plug-and-play feature as well as being hot-swappable makes it advantageous over some other kinds of interfaces. High speed of data transaction compared with its counterparts, such as USB2.0 also makes FireWire interface a plausible candidate for handling the communication between Touch and other linked hardware. However, these advantages come with a cost of compatibility issues. Unfortunately, FireWire interface is not fully compatible with the latest versions of operating systems and software. The details of the methods used to tackle these difficulties are discussed in the section devoted to the communication platform.

4.2.2. OpenHaptics Toolkit

The communication interface of a Phantom Omni Device is IEEE-1394 Fire Wire port and allows real-time programming when used in connection with OpenHaptics Toolkit. OpenHaptics toolkit allows developers to rapidly design and deploy haptic programs, do mash-ups into existing applications, try out new ideas, and create haptically enabled products.

QuickHaptics is an Application Programming Interface (API) that provides fast and easy tools to write new haptic applications or to add haptics to existing applications. Built-in geometry parsers and intelligent default parameters also makes it possible to set up haptics and graphics scenes with a minimal amount of code.

One of the major parts of OpenHaptics, which has been used in this study, is called QuickHaptics micro API. Its powerful library allows the simplification of the programming by implementing the following steps:

- Parsing geometry files from popular animation packages
- Creating graphics windows and initializing the environment
- Initializing one or multiple haptics devices Scene and camera design
- Mapping force and stiffness parameters to objects in the scene
- Setting up callback responses to interactions

Through an informed choice of default values for haptic and graphics parameters, QuickHaptics makes the programmer to be able to create a viable scene without the need for explicitly define the camera location, device space parameters, settings for various shape properties, or any other low-level programming.

Four primary functional classes of QuickHaptics micro API is presented in Table 4.3.

Table 4.3. *Major QuickHaptics micro API classes*

API Class	Description	Application
DeviceSpace	Workspace through which the haptic device can move	Generating a 2D plane portraying the robot, goal, and obstacles
QHRenderer	On-screen window that renders shapes from a camera viewpoint and lets the user feel those shapes with a haptic device	Enabling the operator to recognize the environment and the objects in it
Shape	Base class for one or more geometric objects that can be rendered both graphically and haptically	Using the range sensor data to render the shapes corresponding the robot and obstacles
Cursor	Graphical representation of the end point of the second link on the Touch device	Providing steering and velocity commands to the robot by the operator

4.3. Communication Platform

As a major part of this study, the processes running on different hardware components of the experimental setup need to sustain a reliable communication among themselves. In order to fulfill this requirements, a communication platform has to be established. One popular approach to tackle this situation is to use the Robot Operating System (ROS). However, due to a number of considerations, ROS has not been used in this experiment. First of all, ROS has proved to be reliable only on Linux platforms, while the haptic device operates on a machine running Windows operating system. Additionally, the Linux version of OpenHaptics does not support the FireWire connection for the device used in the setup. Therefore, the robotic ecosystem would not be complete if ROS were the selected ecosystem. On the other hand, migrating to windows is almost impossible because the BeagleBone Black operates with Debian systems. As a final resolution, a platform must be constructed in such a configuration that is firstly, independent of the operating system, and secondly, can handle the programs written in both Python and C++ programming languages. One plausible candidate to tackle these limitations is to use sockets. In this study, a communication platform is constructed by using the core functionalities of sockets. In the following chapters, the description of this platform is discussed.

4.3.1. Socket Programming

A socket is a communications connection point (endpoint) that can be named and addressed in a computer network. Sockets use Application Program Interfaces (APIs) to establish communication links between remote and local processes on wither same or different machines. Sockets are frequently used for client and server communication which runs one server on a system and the clients on the same or other machines. The clients connect to the server, exchange information, and then disconnect. [88]

There are two types of programs in a socket structure: server and client. The server is a process that provides the service to a number of clients and needs to always be running. The server also needs to acquire a fixed Internet Protocol (IP) address, a 32-bit address that is assigned to the machine and refers to the internet layer of the overall network architecture of the internet. The clients, on the other hand, are programs or processes that request services from a single server. They can connect the server via its IP address and start sending or receiving the data which they require. [89]

If the client knows the IP address of the host on which the server runs, it can contact that machine; however, an integer number between 1024 and 65535, called port number has also be assigned to allow the client to identify the particular server process running on the host machine.

Among all protocols that send packets (bits of data) over IP traffic, Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are most commonly used. Although their functioning principles are similar, there are also significant differences that need to be taken into the account. TCP is a connection-oriented protocol, which means a message makes its way across the internet from one computer to another, whereas UDP is a connectionless protocol meaning that once a program sends a load of packets to another, the connection is not used anymore. TCP protocol guarantees that the data transferred remains intact and arrives in the same order in which it was sent. Therefore, it is more suited for applications that require high reliability, and transmission time is relatively less critical. UDP, on the other hand, has no inherent order as all packets are independent of each other. If ordering is required, it has to be managed by the application layer. Consequently, it is more suitable for applications that need fast and efficient transmission, where missing or corruption of a single message has no significant effect on the process . This is one of the main reasons this protocol has been extremely popular among game developers and gamers. UDP's stateless nature is also useful for servers that answer small queries from huge numbers of clients. To sum up, UDP’s light weight makes it plausible for being applied in sending messages over IP and establishing a fast communication between different programs and processes in this project. However, despite all these strengths, UDP has one drawback that needs to be tackled in order to be utilized in the

Figure 4.12. A screenshot of the Cube game

experimental setup. The packets sent by this protocol have are limited to a theoretical field size of 65,535 bytes (8 byte header + 65,527 bytes of data). Since larger data packets might be sent during different stages of the operation of the mobile robot, particularly by the laser range sensor, the mentioned limitation has to be eliminated.

4.3.2. ENet

As discussed in the previous section, UDP does not include any sequencing, connection and bandwidth management, and more importantly, suffers from limitations on the size of data. Therefore, a reliable enhancement is needed to overcome such limitations. The solution is presented by ENet, initially for a multiplayer first person shooter game called Cube. [90]

ENet is a uniform protocol constructed over UDP in such a way that the most useful features of both UDP and TCP as well as some additional features are achieved.

Rather than a single byte stream, ENet presents connections as multiple, properly sequenced packet streams that simplify the transfer of different types of data. This feature is extremely valuable because various types of data are simultaneously sent and received by different devices such as laser range sensor, motor driver, haptic device and etc.

Furthermore, it ensures packets are delivered exactly in the order they are sent. The importance of this feature arises from the nature of real-time applications. Especially in the case of control of mobile robots, both the commands sent to the robot and the feedback data sent by it have to be properly sequenced in order to be interpreted properly by the robot and the operator.

ENet also presents optional reliability of data delivery because all sent packets can be verified by the receiver. Although this feature is missing in the original UDP architecture, ENet is able to send and deliver packets of any size. Larger packets are segmented into many smaller packets of proper size, and fused on the receiver in order to recover the original packet.

Last but not least, ENet operates reliably on Windows and Linux platform providing a Berkeley Software Distribution (BSD) sockets interface. ENet library has a small and stable code base that can easily be extended to support other platforms and integrates easily. This is another major factor that facilitates the communication between processes on BeagleBone Black and the ones on the Windows machine.

4.3.3. Protobuf

Although ENet protocol makes the communication between Windows and Linux machines possible, it falls short in situations where the processes use different programming languages. To tackle this obstacle, there is a need for another powerful tool which also needs to be both lightweight and adaptable to ENet and UDP protocols.

For this purpose, protocol buffers, or shortly protobuf has been used. Protocol buffers is a method for serializing structured data in a way that the communication is independent from both the language and the platform. Protocol buffers were initially developed at Google to deal with an index server request/response protocol. Now, it has turned into the common language between the employees at Google. In this method, once the structure of the data is defined, a special source code is generated to easily write and read the structured data to and from a variety of data sources by using various languages such as Python, Java, C++ and C#.

No matter which programming language is selected, usıng protobuf consists of the following 3 steps:

1. Defining the format of the message using a ‘.proto’ file
2. Running the protocol buffer compiler, ‘protoc’ to compile the ‘.proto’ file
3. Using the language-based protocol buffer API to write and read messages

To have serialization mechanism usable from both Python and C++ is a necessary feature for the communication protocol used in this study. Since various clients are written in either Python or C++, protocol buffers has to be implemented in order to establish the reliable communication between these processes.

4.3.4. OpenHaptics

As discussed earlier, the Touch haptic device is responsible for two major tasks:

- Generate robot’s motion commands based on the coordinates of the device’s joints
- Apply force on the operator based on the proximity and orientation of the obstacles

In order to establish a reliable communication between the Touch haptic device and other components in the setup, OpenHaptics® Developer Edition is used. The OpenHaptics toolkit supports the range of 3D Systems, including Touch device. This toolkit facilitates adding haptics and true 3D navigation to a wide range of applications. OpenHaptics is patterned after the Open Graphics Library (OpenGL) API making it familiar to graphics programmers and enabling integration with OpenGL applications. Using this toolkit, existing OpenGL code for specifying geometry is integrated with OpenHaptics commands to simulate haptic properties.

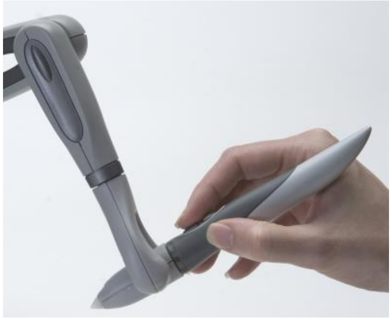


Figure 4.13. Haptic Interface Point

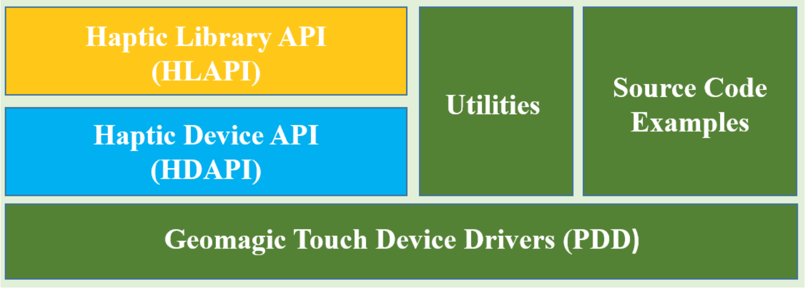

Haptic Library API (HLAPI)
Haptic Device API (HDAPI)
Utilities
Source Code Examples
Geomagic Touch Device Drivers (PDD)

Figure 4.14. OpenHaptics Toolkit

This toolkit includes the Haptic Device API (HDAPI), the Haptic Library API (HLAPI), utilities, PHANTOM Device Drivers (PDD), and source code examples. The HDAPI provides low-level access to the haptic device, enables rendering forces directly from the URG sensor data and provides convenient utility feature, in this case the feedback force. The HLAPI also allows significant reuse of existing OpenGL code and greatly simplifies the coding and debugging without the need to construct the script from the scratch.

As briefly stated earlier, the following code examples from OpenHaptics library have been used to complete the chain of processes necessary for the operation of the experimental setup as it has predefined.

- QueryDevice:

Originally checks the state of the gimbal button and gets the position of the device, retrieves the information from the haptic device, and prints it to the console. This script has been modified to get the current location of the device continually and publish the position data in the form of a vector of three doubles. This routine allows the device to provide information about the current location of the stylus, this data then is sent to haptic_send script via pipe structure to be mapped into meaningful motion commands.

- CommandMotorDAC:

This source code example commands force values using Digital to Analog Converter (DAC) directly to the motors of the Touch device via a keyboard menu interface. This script has also been modified in a way that it continually applies the force inputs coming from haptic_receive based on the proximities and orientations of the obstacles retrieved by Hokuyo URG laser range sensor. This data is also transmitted to the process through pipe (or data pipeline-a set of data processing elements connected in series, where the output of one element is the input of the next one.)

4.3.5. Interprocess Communications

Interprocess Communications (IPC) are the mechanisms provided by the Windows operating system that enables communications and data sharing between applications. Different types of IPC either facilitate the labor division among various processes within a machine or among two or several computers on a single network. Although clipboard and Component Object Model (COM) are two most common IPC mechanisms, there are many more IPC methods that applications can benefit. Dynamic Data Exchange (DDE), File Mapping, Pipes and Remote Procedure Call (RPC) are other common IPC methods that are frequently used in numerous applications. Even Windows Sockets can be categorized as one of the IPC methods. [89]

The IPC method which is used in this project is called Mailslot. This one-way communication approach is specifically advantageous over named pipes and sockets where applications are expected to transmit a relatively small number of relatively short messages. A mailslot, by definition, is a pseudo file that resides in memory and can be accessed by standard file functions. The data in a mailslot message is also limited to 424 bytes when sent between computers. Mailslots are also temporary,

Table 4.4. *Major processes operating in the experimental setup*

Process	Hardware	Machine	Language	Function
urg_send	Hokuyo URG sensor	BeagleBone Black (Linux)	Python	Detects and sends the coordinates of the obstacles
haptic_receive	-	Workstation (Windows)	C++	Receives the coordinates of the obstacles and generates the force input for the haptic joystick
urg_plot	-	Workstation (Windows)	Python	Receives the coordinates of the obstacles and plots the figure
CommandMotorDAC	Phantom Omni	Workstation (Windows)	C++	Applies the force on the joints of the haptic joystick
QueryDevice	Phantom Omni	Workstation (Windows)	C++	Publishes the coordinates of the joints of the haptic joystick
haptic_send	-	Workstation (Windows)	C++	Receives the coordinates of the joints of the haptic joystick and generates motor commands
motor_receive	L298N Motor Controller	BeagleBone Black (Linux)	Python	Receives motor commands and drives the dual motor via PWM

which means when all handles to a mailslot are closed, the mailslot and all the data it contains are deleted.

One major consideration, however, is that similar to UDP Sockets, mailslots also broadcast messages using datagrams- small packets of information that the network sends along a channel. Therefore, mailslot communication offers no confirmation of receipt and there is no guarantee that a specific packet of data has been received.

The objective of IPC in the communication platform used in this project is to establish the communication between the core processes in OpenHaptics and the ENet network. Mailslots have been used in order to enable the transmission of haptic device's joint angles from QueryDevice process to haptic_send, which in turn, converts this data to corresponding motor commands for the mobile robot and publishes it to the ENet network. Similarly, Mailslot IPC method makes it possible for haptic_receive to send the data related to the coordinates of the obstacles to CommandMotorDAC process, which is responsible for applying force on the haptic device's motors.

This chapter presented the actual experimental setup that is particularly designed, fabricated and developed in order to implement the control algorithm and carry out the experimental tests aiming to evaluate the performance of the proposed method in action. Major hardware components of the wheeled mobile robot, the haptic interface, and more importantly, the platform enabling reliable communication between the key elements are all covered in this chapter. The thesis continues by discussing the test procedure and the obtained results in the next chapter.

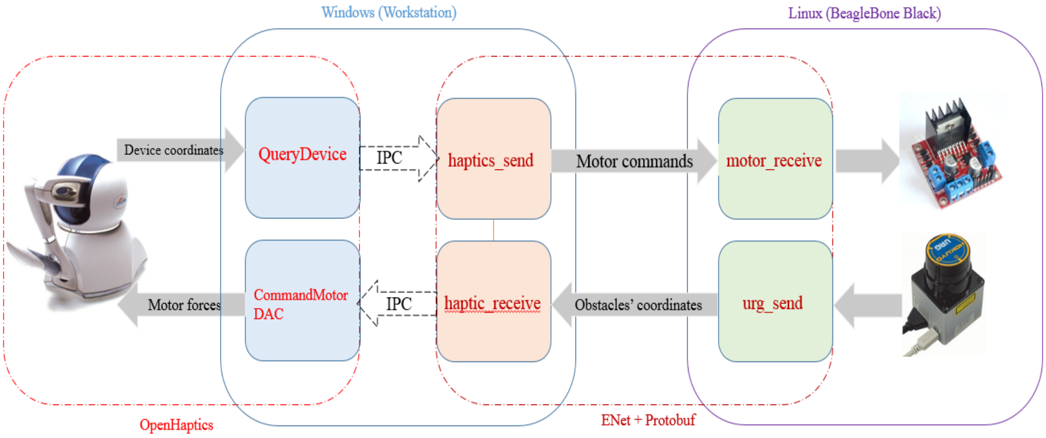


Figure 4.15. General overview of the communication platform

CHAPTER 5

TESTS AND RESULTS

The human-robot system has to be put to the test in order to evaluate the performance of the system's configuration and effectiveness of the proposed control algorithm. As discussed earlier, the relative superiority of different levels of autonomy over one another can be revealed by conducting experiments that aim to measure objective criteria such as time to accomplish the tasks, number of failures, and task load on human operator. This chapter is devoted to addressing these results. The test platform is first introduced in this chapter and is followed by an overall description of the test procedure. The numeric results are presented and their implications are discussed. A brief explanation of the task load measurement is also covered in this chapter.

5.1. Test Platform and Scenarios

The test platform is constructed with a 165x165 cm wooden plate. A Microsoft lifecam HD 3000 camera is situated 250cm above the platform and is tasked with recording the motion of the mobile robot on the platform. The goal is represented by the green box and the red boxes, of size 15x15x10 cm, function as the obstacles. The obstacles are also made of soft material in order to prevent any possible damage to the mobile robot in case of collisions. The robot is situated on opposite side to the goal on the platform and is positioned directly facing an obstacle at the beginning of the test trial. The layout of the test platform is organized in three different configurations representing three distinct test scenarios:

5.1.1. Scenario A

The aim in this configuration is to evaluate the performance of the system where the contradicting stimulus from both the goal and the obstacle have similar strength and

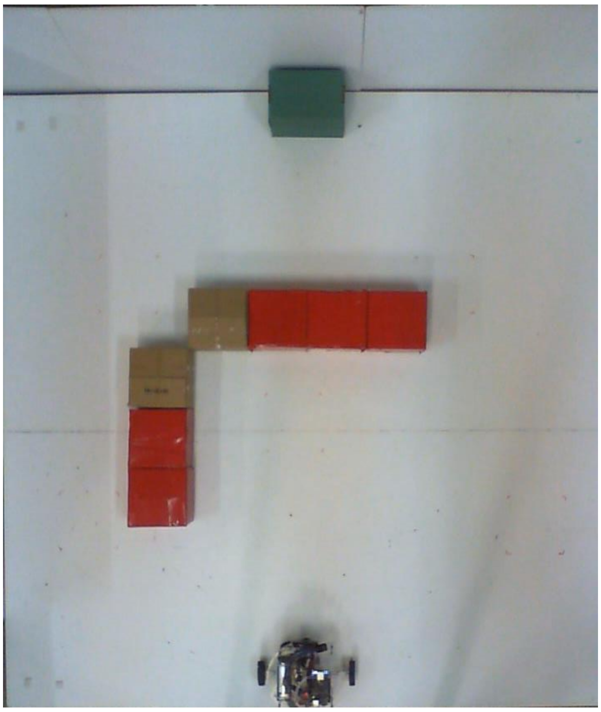


*Figure 5.1.*Test platform in test scenario A

prevent the robot to take a decisive action. The situation in which the force vectors associated from the goal and the obstacles cancel each other out is known as local minima in the literature. As seen in Figure 5.1., the robot is placed directly in front of the goal and a wall of obstacles block the direct path towards the goal. The only possible solution is for the robot is to maneuver around the wall, which needs evaluation of the environment beyond the capability of pure reactive approach. This situation becomes clear specially in pure reactive control theme in which oscillatory motion occurs in robot’s motion.

5.1.2. Scenario B

This configuration is designed to challenge the human-robot system in narrow passages where the limited distance between the obstacles make it difficult for the robot to navigate towards the goal position without hitting the obstacles or changing the direction too many times. This situation becomes clear particularly in pure reactive control theme due to instantaneous changes in direction and magnitude of obstacle

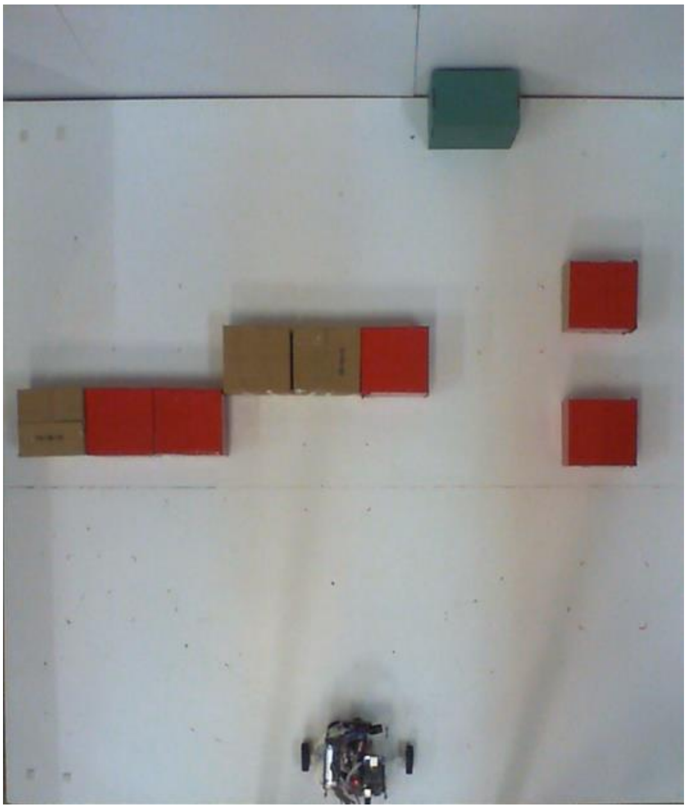


Figure 5.2. Test platform in test scenario B

avoidance force vector acting on the robot, which results in fluctuations in the robot's direction. Also as seen in Figure 5.2., all alternative passages are blocked intentionally in order to force the robot to navigate through the narrow passage towards the goal.

5.1.3. Scenario C

In this scenario no extreme challenge is predefined for the system. The robot and the goal are situated on opposite corners of the platform. This scenario, in particular, is able to provide a satisfactory comparison between the effectiveness of different control methodologies regarding the time spent on completing the task, number of failures and task load on operator.

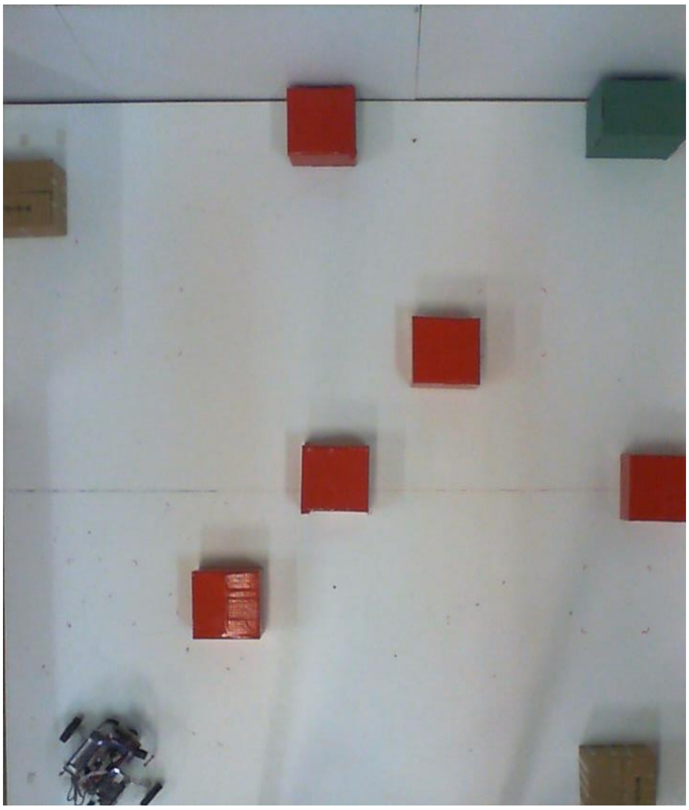


Figure 5.3. Test platform in test scenario C

It also needs noting that the configuration of the test platform is kept unchanged for all participant, and in all autonomous, manual, supervisory, assistive and shared control scenarios. This has been done in order to provide an unbiased and objective evaluation for the performance and effectiveness of the all methods that are compared in this study. Another major fact that should be taken into account is that the participants have no access to the image captured by the aerial camera during the test.

5.2. Procedure

Twelve volunteers, comprising both undergraduate and graduate students participated in the testing process. The participants were chosen from the individuals who had limited knowledge about mobile robot dynamics, locomotion and navigation. This was necessary in order to minimize the probable impact of the prior knowledge on the task performance. All subjects went through a short briefing session in which they were introduced with the overall objectives of the study without providing too much detail which might result in bias affecting their objectivity. The briefing period was then followed by a ten-minute-long familiarization session at the beginning of the experiment. At this stage, the participants were familiarized with the experiment equipment and were taught how to use the haptic device to control the mobile robot on the test platform. At least one intentional collision with obstacles was also simulated to establish a rigid understanding on the concept of avoiding and colliding with the obstacles. The subjects were also informed in advance that task failures such as collisions may also be a part of the test. This was done in order to ensure that they knew how to effectively behave in the case of obstacle collision and guarantee the continuation of the test process. It should also be noted that no information regarding the evaluation criteria was provided to the participant in order to prevent any undesired effect on the performance due to advanced preparation.

After the introduction and familiarization steps, each subject is asked to complete test trials for all three scenarios mentioned in the previous section with four distinct levels of autonomy discussed in Chapter 4, totaling 12 tests for each participant.

1) Fully manual control: The control algorithm is switched off and the data from the range sensor is not taken into consideration. Therefore, no feedback force is generated by the haptic device and applied to the operator. In this scenario, the operator is in full control of the robot and uses only the visual data stream from on-board camera to navigate the mobile robot through the obstacles on the test platform and get to the target position. This stage can be considered as a complete manual tele-operation task.

2) Supervisory Control: After completing the first part of the experiment, the participants are asked to monitor the robot’s navigation to the goal among the obstacles and intervene whenever they find necessary. The operators are fed with both laser range sensor data and the visual stream from the frontal camera. The robot handles the go-to-goal task and does not consider avoiding obstacles, because this task is assigned to the operator. Using the haptic device, he can manipulate the motion of the robot.

3) Assistive Control: At this stage, both the operator and the mobile robot contribute to the motion control task of the robot. In addition to the visual data from the frontal camera, the subjects' perception is also enhanced with the feedback force exerted on their hands based on the distance and orientation of the obstacles with respect to the laser range sensor.

4) Dynamic Shared Control: Similar to the previous control paradigm, both the operator and the mobile robot contribute to the motion control task of the robot consisting go-to-goal and obstacle avoidance. However, unlike assistive control, the share by which two agents contribute to the motion of the robot vary according to the forward velocity of the robot.

During the experiment, the trajectory which the robot followed alongside with its collision with obstacles were recorded by the bird-eye camera. In addition, after completing each trial, the participants were required to rate the tasks according to the workload they had felt. In the scope of this study, the task performance measures as well as the workload pressure data are collected and compared for the three scenarios for every volunteer. The test was carried out in a laboratory environment with minimum distraction and a satisfactory level of comfort for all subjects.

5.3. Workload Assessment

In order to subjectively rate the perceived workload of the tele-operation task, Task Load Index (NASA-TLX), developed by the Human Performance Group at NASA's Ames Research Center, was used. In this multi-dimensional rating procedure, the overall task workload is categorized into six criteria that are provided on a single page in the form of a questionnaire. The scales of NASA-TLX include: mental, physical, and temporal demand, performance, effort, and frustration. The participants of the experiment were asked to provide a rating for each of the five scales after completing their trials. Each scale ranges from zero to 21, resulting a total task load index of zero to 125. They also ranked their level of performance from zero to 21, [91].

5.4. Tests Results

In this section, the mean values of the recorded data for each autonomy level in all three test scenarios for all 12 participants are provided.

5.4.1. Number of Collisions

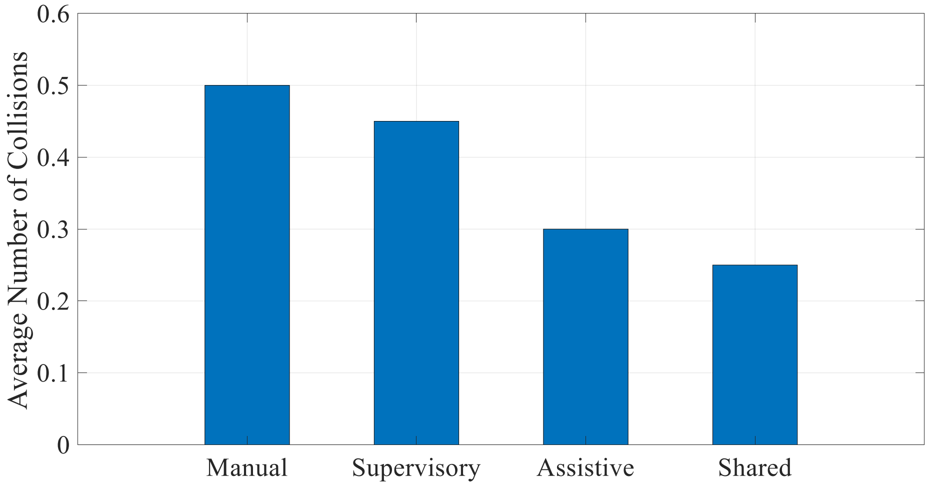


*Figure 5.4.*Average number of collisions in scenario A

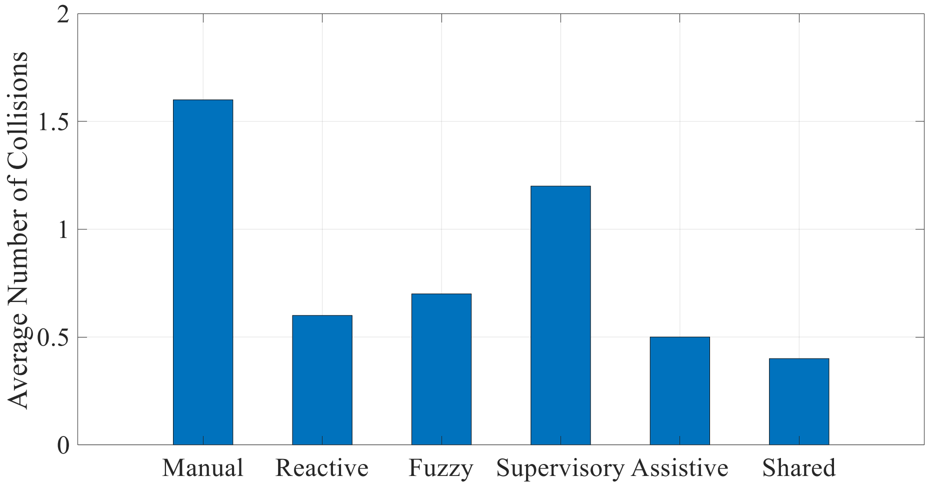


Figure 5.5. Average number of collisions in scenario B

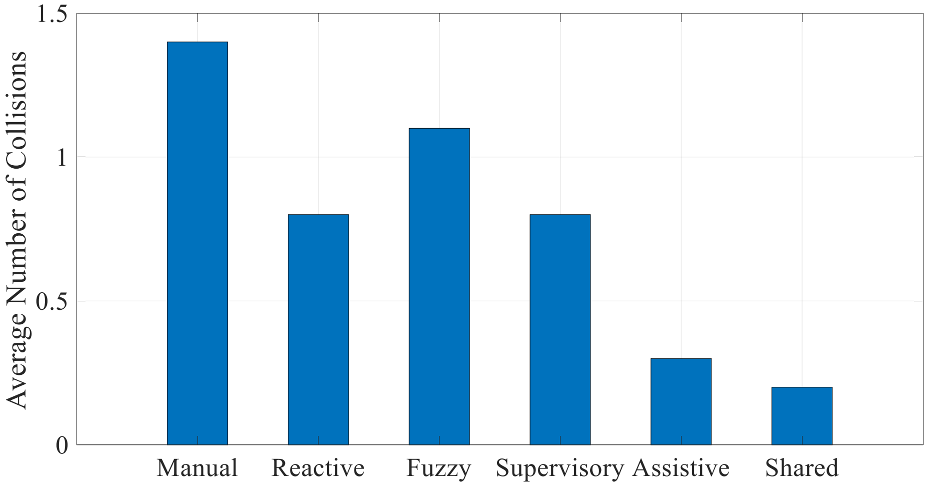


Figure 5.6. Average number of collisions in scenario C

Figures 5.4. to 5.6. show the average of the number robot hits obstacles for all 12 test sets for each scenario under all autonomy levels. When this figure is analyzed thoroughly, significant implications can be revealed.

First of all, it can be seen that the highest number of collisions occur in manual drive mode. This is mostly due to the fact that the operator fails at reacting to the situations where either the obstacle lies beyond the detection range of the frontal camera or the transmission of the data suffer from the lag in communication, particularly in relatively higher speeds. Conversely, when the complete Control of the system is designated to the robot itself, the number of failures decline. This can be explained considering the fact that the robot can perform the task in a more conservative fashion and avoids conducting risky maneuvers in the expense of spending more time finalizing the action.

More important result, however, is the performance of both fully manual and fully automated control compared with cooperative control approaches. It is clear that the system demonstrates much better reliability when both the robot and the operator are kept in the loop. This can be revealed from comparing the number of collisions in assistive and dynamic shared control with those of fully automated and fully manual control schemes. Therefore, it is undeniable that the contribution of human operator to the control of the system increases the level of security by decreasing the number of collisions with the obstacles.

In the case of supervisory control, however, the system is not able to reduce the number of collisions in Scenario B, where the narrow passage limits the freedom of motion for the robot. This can be attributed to the natural characteristics of human operator, who may not be accustomed to the task of assisting the robot as a secondary contributor. This argument needs to be further investigated, however.

Finally, it is also evident from Figure 5.3. that in Scenario C, although the supervisory control outperforms solo controllers, thanks to operator's intervention in critical situations, it falls behind the assistive and shared control when obstacle avoidance performance is considered. This can partly be attributed to fact that in the latter two control schemes, the tasks are divided between the operator and the robot in a way that the strengths of both agents are fully exploited. While the human operator focuses on global planning task and takes charge of go-to-goal behavior, the robot is designated with avoiding obstacles and thanks to utilizing the laser range sensor, it can quickly respond to the presence of obstacles in the immediate vicinity. Even in cases where the operator fails at detecting obstacles on the camera, the robot is capable of modifying the motion command considering the distance and orientation of the obstacle.

The biggest asset the system possesses in shared control, however, is the force feedback effect of the haptic device. In order to fully investigate the contribution the haptic interface makes to the system, one can simply compare the average number of collisions in fully manual control to assistive and shared control approaches. While in all three cases the human operator is responsible for navigating the robot to the goal, it is in the assistive and dynamic shared control that his perception from the surrounding via frontal camera is enhanced by presence of haptic device.

It can be seen that the effect of haptic feedback force on reducing the number of failures of the system is considerable. However, the positive contribution that the haptic device make on the overall system in the context of shared control is not limited to security of the system only.

5.4.2. Travel Time

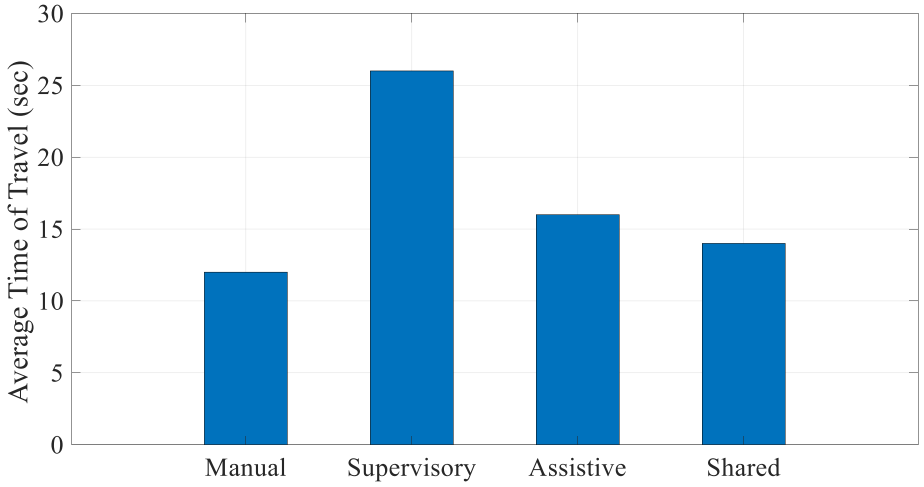


Figure 5.7 Average time of travel in scenario A

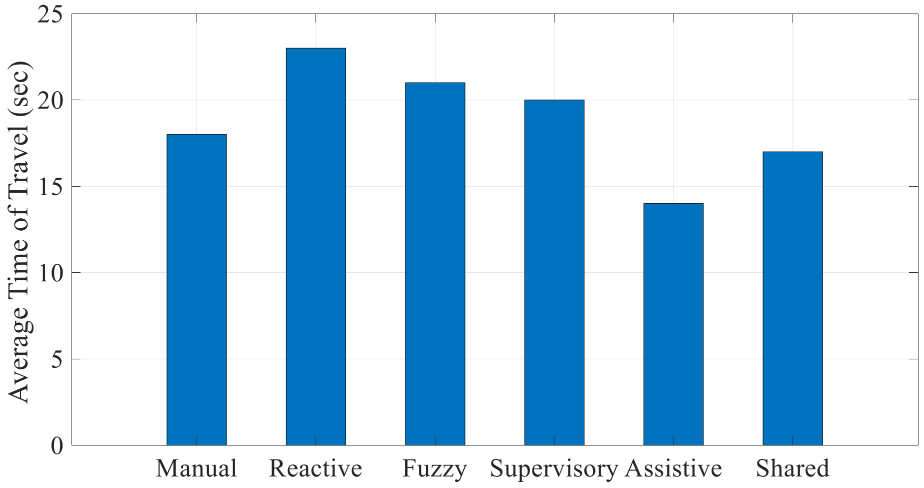


Figure 5.8. Average time of travel in scenario B

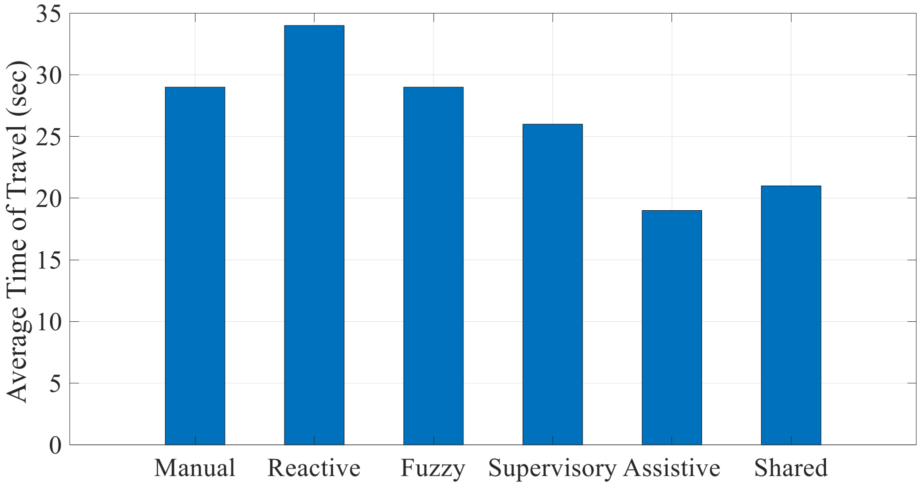


Figure 5.9. Average time of travel in scenario C

Figures 5.7. to 5.9. show the average time takes for the robot to get to the goal from start position for each test set. This figure also provides a number of significant facts.

First of all, it is obvious that in manual control it generally takes less time for the operator to get the robot to the final destination compared with automated and supervisory control approaches in both scenarios A and B. This difference in performance can be related to operator's ability in planning ahead and minimize the fluctuations in direction when encountered obstacles. Therefore, it is no surprise that the operator can select a faster path and despite some possible compromise in security (which was explained in previous section).

In case of cooperative control schemes, where both the operator and the wheeled mobile robot are entitled with some degree of authority, the evaluation of the system's performance becomes more complicated. In supervisory control, it is clear that the participant's intervene resulted a decrease in overall time of travel, mostly because the operators ability to predict the troublesome paths the robot may encounter and push the robot away from those routes. Despite increasing the safety of travel, since the humans have tendency to react to the approaching obstacles by exerting repulsive motion commands when they are not in high-level control of the robot, the system has to compensate for deviation from the shortest path; so, the decrease in the travel time in supervisory approach is not considerable compared with the other cooperative approaches.

In the case of assistive control, the improvement in travel time compared to manual, autonomous, and supervisory controls is significant. This enhancement can be explained by the fact that human operator, in this autonomy level, is in charge of go-to-goal behavior and does not concern about the local collision failures as much as he would in manual and supervisory schemes. In addition, since the humans are more skilled in planning, their lower speed in motion control is compensated and therefore, the difference the difference in performance with automated control is quite negligible.

Finally, in dynamic shared control, the time spent for completing the task has increased slightly compared with assistive control. This can be explained by considering the fact that in lower velocities, the operator is only able to exert limited effect on the robot and regardless of the path the robot intends to move, it automatically limits the operator’s motion command.

Contribution of assistive and shared control, however, is not limited to the number of collisions and travel time as another major criteria, task load on operator, also has to be investigated.

5.4.3. Task Load

Measuring the task load on operator is often considered to be more difficult compared with the other criteria due to its subjective nature. However, thanks to NASA’s TLX approach, a reliable measure for the task load can be obtained.

Figures 5.10. to 5.12. represent the TLX measure expressed by the participants in the study. The questionnaires are filled after each test trial for manual, supervisory, assistive and shared control by the volunteers.

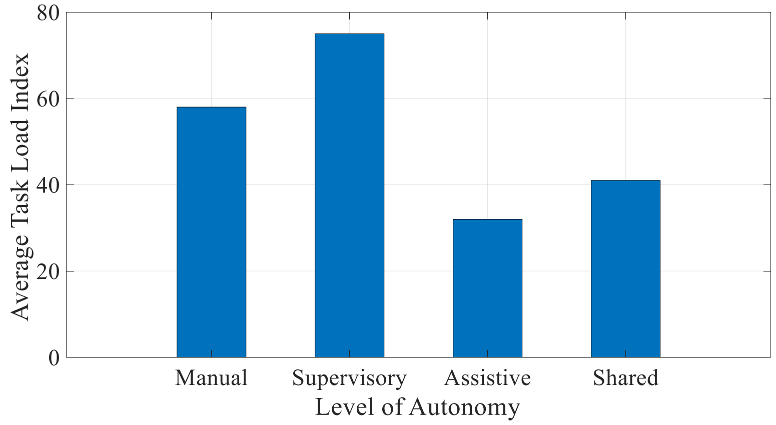


Figure 5.10 Average TLX in scenario A

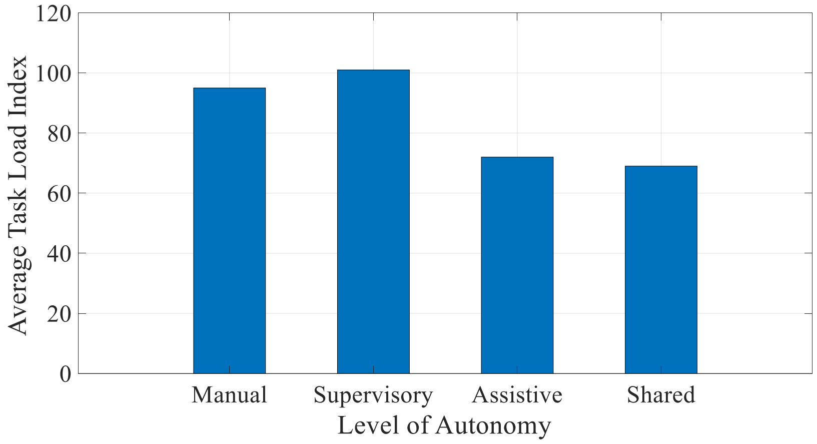


Figure 5.11. Average TLX in scenario B

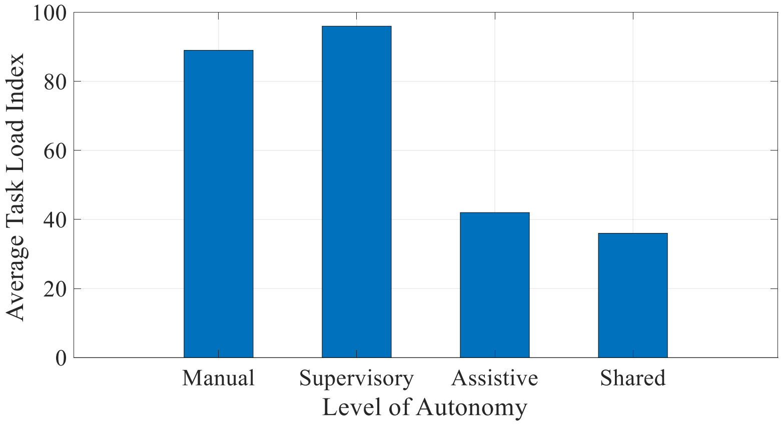


Figure 5.12. Average TLX in scenario C

to variety in participants' personalities, professions and academic backgrounds, each experienced different types and levels of task load.

In order to investigate the effect of level of autonomy on cognitive and psychological pressure felt by human operator in human-robot interaction context, the average TLX is obtained and shown in Figure 5.7.

Since manual control was the first trial carried out for each participant and is followed by supervisory control, one may predict that the task load which was felt by the participants might have been less intense for supervisory control. However, as it can be concluded from this figure, the participants felt the highest pressure in supervisory control. This can be related to the fact that the operator does not feel in control of the wheeled mobile robot on one hand and is responsible for keeping the robot safe during its travel to the goal on the other hand. Most participants expressed high level of pressure during completion of the test trial because they felt since they were supposed to interfere only in case of possible collision with obstacles, they had to stay alert in whole duration of the trial.

It is evident from Figure 5.7. that ask load index has decreased significantly in shared control scenario. There are a number of explanations for this considerable reduction. First, the participants stated that unlike supervisory control, they felt more relieved because they were aware that without their motion command, the robot would not take initiative to move towards the goal. In other words, except the cases where the robot is found to be too close to the obstacles, it did not undermine the operator's authority, which in turn, made him feel in command.

Second, the participants knew that the robot is capable of avoiding collision with obstacles on its own. Since obstacle avoidance behavior is designated to the robot, the operator was able to concentrate his cognitive and physical effort on go-to-goal task.

Last but not the least, the haptic interface was initiated in this scenario and made valuable contribution to the human-robot system, not only in sense of increasing performance and security of the operation, but also in decreasing the physical and psychological demand. The participants were assured that in case they failed in detecting any obstacle, they would be prompted by its presence through the feedback force applied on their hands through the haptic device. Integration of the haptic device into the potential field also provided a smooth navigation for the robot, which resulted decline in the number of times the operator had to act quickly in response to possible sudden changes in external stimulus.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1. Conclusion

This thesis study addresses the collaboration between human operator and mobile robot in obstacle avoidance and navigation tasks. The main objective of the study is to evaluate the effect of different levels of robot autonomy on the overall performance of the human-machine system. In this research work, two different automated control method, both based on behavior-based control, and three collaboration models, alongside with manual drive model are developed considering different roles for the human operator and the robot.

In order to enable the operator to carry out tele-operation task, the Phantom Omni haptic device is used. Application of haptic interface is another contribution of this study and plays significant role in enhancing the success rate of the system. Not only the haptic joystick acts as a tool to issue motion commands, it also provides the operator with continual force feedback about the relative location of the obstacles to the robot.

To implement the cooperative control of mobile robot, behavior-based control methodology is applied. Since the obstacle avoidance and go-to-goal tasks can be dealt with as two different robot behaviors, and the operator and robot as two intelligent agents capable of performing both behaviors, different autonomy models can be transformed into distinct robot control scenarios with unique definition of roles and levels of contribution for both agents.

In order to evaluate the effectiveness of each control scenario, and therefore the desired level of human contribution to the system, a physical wheeled mobile is constructed with a Beagle Bone Black and a laser range sensor as its key hardware components. A communication platform is also developed to maintain the continual interaction between both the agents and the major autonomous components.

The results of the experimental tests show that cooperative control scenarios provide superior outcomes in terms of efficiency and security when compared with fully manual and fully autonomous cases. In addition, it can be concluded that assistive control and dynamic shared control prove to be the more effective autonomy models compared with the other alternatives.

Finally, integrating the haptic device into motion control of the mobile robot in assistive control and dynamic shared control scenarios helped to decrease the mental and physical load on human operator. This finding holds significant importance because keeping the human operator in cooperative control tasks for long periods of time enables the overall system to benefit from the collaboration between human and robot and accomplish the mutual objectives more effectively.

6.2. Future Work

In order to further investigate the effect of cooperative control approaches, other levels of autonomy may be introduced. Action Support, Batch Processing, and Decision Support are three more possible control models that can be implemented in tele-operation scenario and be compared with current cases.

In addition, some other control methodologies can be used as either alternatives or complements for behavior-based control. Sliding Mode, Fuzzy Sliding Mode, and Adaptive Control approaches have all the potential to be integrated into the control architecture to further improve the performance of the tele-operated system.

Finally, more challenging test scenarios may be carried out to study the performance of human-robot system under extreme conditions such as using dynamic obstacles instead of static objects. Also, the distinction between the success rates of different levels of autonomy may become clearer if the robot is equipped with more advanced hardware components and software platforms.

REFERENCES

[1] T. R. Kurfess, *Robotics and Automation Handbook*, 1st ed. CRC Press, 2004.

[2] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, 2nd ed. MIT Press, 2011.

[3] C. A. Berry, *Mobile Robotics for Multidisciplinary Study (Synthesis Lectures on Control and Mechatronics)*. Morgan & Claypool Publishers, 2012.

[4] M. A. Goodrich and A. C. Schultz, "Human-Robot Interaction: A Survey," *Found. Trends Human-Computer Interact.*, vol. 1, no. 3, pp. 203–275, 2007.

[5] C. Hatzfeld and T. Kern, *Engineering Haptic Devices*. Springer Series on Touch and Haptic Systems, 2014.

[6] M. Mihelj and J. Podobnik, *Haptics for Virtual Reality and Teleoperation*, vol. 64. Dordrecht: Springer Netherlands, 2012.

[7] N. J. Nilsson, "Shakey the Robot," *Tech. Note 323*, 1984.

[8] T. Fong, I. Nourbakhsh, and K. Dautenhahn, "A survey of socially interactive robots," *Rob. Auton. Syst.*, vol. 42, no. 3–4, pp. 143–166, Mar. 2003.

[9] J. Borenstein and Liqiang Feng, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE Trans. Robot. Autom.*, vol. 12, no. 6, pp. 869–880, 1996.

[10] J. L. Jones, B. A. Seiger, and A. M. Flynn, *Mobile robots: Inspiration to Implementation*, 2nd ed. CRC Press, 1998.

[11] G. McComb, *Robot Builder's Sourcebook*, 4th ed. McGraw-Hill Education, 2011.

[12] F. Michaud *et al.*, "Multi-Modal Locomotion Robotic Platform Using Leg-Track-Wheel Articulations," *Auton. Robots*, vol. 18, no. 2, pp. 137–156, Mar. 2005.

[13] G. N. Desouza and A. C. Kak, "Vision for mobile robot navigation: a survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, pp. 237–267, 2002.

[14] J.-C. Latombe, *Robot Motion Planning*. Boston, MA: Springer US, 1991.

[15] S. Thrun, "Robotic Mapping: A Survey," *Science (80-.).*, vol. 298, Oct. 2002.

[16] Kok Seng Chong and L. Kleeman, "Accurate odometry and error modelling for a mobile robot," in *Proceedings of International Conference on Robotics and Automation*, 1997, vol. 4, no. April, pp. 2783–2788.

[17] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, “Monte Carlo Localization: Efficient Position Estimation for Mobile Robots,” *Proc. Natl. Conf. Artif. Intell.*, pp. 343–349, 1999.

[18] G. Campion, G. Bastin, and B. D’Andrea-Novel, “Structural properties and classification of kinematic and dynamic models of wheeled mobile robots,” in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, 1996, vol. 12, pp. 462–469.

[19] R. C. Arkin, “Motor Schema -- Based Mobile Robot Navigation,” *Int. J. Rob. Res.*, vol. 8, no. 4, pp. 92–112, 1989.

[20] G. Bayar, M. Bergerman, E. i. Konukseven, and A. B. Koku, “Improving the trajectory tracking performance of autonomous orchard vehicles using wheel slip compensation,” *Biosyst. Eng.*, vol. 146, pp. 149–164, 2016.

[21] G. Bekey and R. Tomovic, “Robot control by reflex actions,” *Proc. 1986 IEEE Int. Conf. Robot. Autom.*, vol. 3, pp. 240–247, 1986.

[22] M. P. Georgeff and A. L. Lansky, “Reactive reasoning and planning,” *Proc. sixth Natl. Conf. Artif. Intell.*, vol. 2, pp. 677–682, 1987.

[23] K. Ogata, *Modern control engineering*. Prentice Hall, 2001.

[24] J. Borenstein, H. R. Everett, L. Feng, and D. K. Wehe, “Mobile robot positioning: Sensors and techniques,” *J. F. Robot.*, vol. 14, no. 4, pp. 231–249, 1997.

[25] R. A. Brooks, “Intelligence without representation,” *Artif. Intell.*, vol. 47, no. 1–3, pp. 139–159, 1991.

[26] R. C. Arkin, *Behavior Based Robotics*. The MIT press, 1998.

[27] C. Everett, “Survey of collision avoidance and ranging sensors for mobile robots,” *Rob. Auton. Syst.*, vol. 5, no. 1, pp. 5–67, May 1989.

[28] A. Halme, “Sensors for Mobile Robots: Theory and Application [Book Review],” *IEEE Trans. Robot. Autom.*, vol. 12, no. 6, p. 922, Dec. 1996.

[29] S. M. Udupa, “Collision detection and avoidance in computer controlled manipulators,” in *IJCAI’77 Proceedings of the 5th international joint conference on Artificial intelligence*, 1977, pp. 737–748.

[30] J. F. Canny, *The complexity of robot motion planning*. The MIT press, 1987.

[31] Y. Hwang and N. Ahuja, “Gross motion planning---a survey,” *ACM Comput. Surv.*, vol. 24, no. 3, pp. 219–291, 1992.

[32] M. C. Lin and S. Gottschalk, “Collision detection between geometric models: a survey,” in *Proc. of IMA Conference on Mathematics of Surfaces*, 1998, pp.

1–20.

[33] F. Madera, “An Introduction to the Collision Detection Algorithms,” *Abstr. Appl. Mag.*, vol. 5, pp. 7–18, 2011.

[34] C. Goerzen, Z. Kong, and B. Mettler, *A survey of motion planning algorithms from the perspective of autonomous UAV guidance*, vol. 57, no. 1–4. Springer Science + Business Media, 2010.

[35] T. Lozano-Perez, “Spatial Planning: A Configuration Space Approach,” *IEEE Trans. Comput.*, vol. c–32, no. 2, pp. 108–120, 1983.

[36] K. D. Wise and A. Bowyer, “A Survey of Global Configuration-Space Mapping Techniques for a Single Robot in a Static Environment,” *Int. J. Rob. Res.*, vol. 16, no. 8, pp. 762–779, 2000.

[37] M. S. Branicky and W. S. Newman, “Rapid computation of configuration space obstacles,” in *Proceedings., IEEE International Conference on Robotics and Automation*, 1990, no. 2, pp. 304–310.

[38] T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, “Visibility of disjoint polygons,” *Algorithmica*, vol. 1, no. 1–4, pp. 49–63, 1986.

[39] T. Lozano-Pérez and M. A. Wesley, “An algorithm for planning collision-free paths among polyhedral obstacles,” *Commun. ACM*, vol. 22, no. 10, pp. 560–570, 1979.

[40] J. Hershberger, “An optimal visibility graph algorithm for triangulated simple polygons,” *Algorithmica*, vol. 4, no. 1–4, pp. 141–155, 1989.

[41] D. G. Kirkpatrick, “Efficient computation of continuous skeletons,” in *20th Annual Symposium on Foundations of Computer Science (sfcs 1979)*, 1979, pp. 18–27.

[42] F. Aurenhammer, “Voronoi Diagrams — A Survey of a Fundamental Data Structure,” *ACM Comput. Surv.*, vol. 23, no. 3, pp. 345–405, 1991.

[43] B. Aronov, S. Fortune, and G. Wilfong, “The furthest-site geodesic voronoi diagram,” *Discrete Comput. Geom.*, vol. 9, no. 1, pp. 217–255, 1993.

[44] O. Takahashi and R. J. Schilling, “Motion Planning in a Plane Using Generalized Voronoi Diagrams,” *IEEE Trans. Robot. Autom.*, vol. 5, no. 2, pp. 143–150, 1989.

[45] S. Garrido, L. Moreno, M. Abderrahim, and F. Martin, “Path Planning for Mobile Robot Navigation using Voronoi Diagram and Fast Marching,” in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 2376–2381.

[46] P. Bhattacharya and M. L. Gavrilova, “Voronoi diagram in optimal path

planning," in *4th International Symposium on Voronoi Diagrams in Science and Engineering (ISVD 2007)*, 2007, vol. 1, no. c, pp. 38–47.

[47] S. Kambhampati and L. Davis, "Multiresolution path planning for mobile robots," *IEEE J. Robot. Autom.*, vol. 2, no. 3, pp. 135–145, 1986.

[48] D. Z. Chen, R. J. Szczerba, and J. J. Uhran, "Planning conditional shortest paths through an unknown environment: a framed-quadtree approach," in *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, 1995, vol. 3, pp. 33–38.

[49] D. Z. Chen, R. J. Szczerba, and J. J. Uhran, "A framed-quadtree approach for determining Euclidean shortest paths in a 2-D environment," *IEEE Trans. Robot. Autom.*, vol. 13, no. 5, pp. 668–681, 1997.

[50] R. Mehrotra and D. M. Krause, "Obstacle free path planning for mobile robots," *Image Process. its Appl. 1989., Third Int. Conf.*, pp. 431–435, 1989.

[51] F. Lingelbach, "Path planning using probabilistic cell decomposition," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, 2004, vol. 1, pp. 467–472.

[52] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, 1986, vol. 2, pp. 500–505.

[53] J.-O. Kim and P. K. Khosla, "Real-time obstacle avoidance using harmonic potential functions," *IEEE Trans. Robot. Autom.*, vol. 8, no. 3, pp. 338–349, Jun. 1992.

[54] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst. Man. Cybern.*, vol. 19, no. 5, pp. 1179–1187, Sep. 1989.

[55] Yunfeng Wang and G. S. Chirikjian, "A new potential field method for robot path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, 2000, vol. 2, no. April, pp. 977–982.

[56] G. Zeng and A. Hemami, "An overview of robot force control," *Robotica*, vol. 15, no. 5, pp. 473–482, 1997.

[57] T. B. Sheridan and W. L. Verplank, *Human and Computer Control of Undersea Teleoperators*. Man-Machine Systems Laboratory, Massachusetts Institute of Technology, 1978.

[58] D. . B. Kaber, E. Onal, and M. R. Endsley, "Design of automation for telerobots and the effect on performance, operator situation awareness, and subjective

workload," *Hum. Factors Ergon. Manuf.*, vol. 10, no. 4, pp. 409–430, 2000.

[59] M. R. ENDSLEY, "Level of automation effects on performance, situation awareness and workload in a dynamic control task," *Ergonomics*, vol. 42, no. 3, pp. 462–492, Mar. 1999.

[60] B. Hardin and M. A. Goodrich, "On using mixed-initiative control," in *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, 2009, pp. 165–172.

[61] A. M. Sabatini, M. Bergamasco, and P. Dario, "Force feedback-based telemicromanipulation for robot surgery on soft tissues," in *Images of the Twenty-First Century. Proceedings of the Annual International Engineering in Medicine and Biology Society*, 1989, vol. 3, pp. 890–891.

[62] S. Komada and K. Ohnishi, "Force feedback control of robot manipulator by the acceleration tracing orientation method," *IEEE Trans. Ind. Electron.*, vol. 37, no. 1, pp. 6–12, Jun. 1990.

[63] R. J. Adams and B. Hannaford, "Stable haptic interaction with virtual environments," *IEEE Trans. Robot. Autom.*, vol. 15, no. 3, pp. 465–474, Jun. 1999.

[64] J. Park and O. Khatib, "A Haptic Teleoperation Approach Based on Contact Force Control," *Int. J. Rob. Res.*, vol. 25, no. 5–6, pp. 575–591, 2006.

[65] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger, "Collision Detection and Reaction: A Contribution to Safe Physical Human-Robot Interaction," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 3356–3363.

[66] D. Sidobre *et al.*, "Human – Robot Interaction," pp. 123–172, 2012.

[67] N. Diolaiti and C. Melchiorri, "Teleoperation of a mobile robot through haptic feedback," in *IEEE International Workshop HAVE Haptic Virtual Environments and Their*, 2002, pp. 67–72.

[68] J. Abouaf, "Trial by fire: teleoperated robot targets Chernobyl," *IEEE Comput. Graph. Appl.*, vol. 18, no. 4, pp. 10–14, 1998.

[69] J. L. Protho, E. F. LoPresti, D. M. Brienza, and D. Ph, "An Evaluation of an Obstacle Avoidance Force Feedback Joystick," *Proc. Resna Annu. Conf.*, vol. 20, pp. 447–449, 2000.

[70] N. Diolaiti and C. Melchiorri, "Haptic tele-operation of a mobile robot," in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2003, vol. 36, no. 17, pp. 521–526.

[71] H. Roth, K. Schilling, and O. J. Rösch, "Haptic Interfaces for Remote Control of Mobile Robots," *IFAC Proc. Vol.*, vol. 35, no. 1, pp. 177–182, 2002.

[72] a. Fattouh, M. Sahnoun, and G. Bourhis, “Force feedback joystick control of a powered wheelchair: preliminary study,” in *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, 2004, vol. 3, pp. 2640–2645.

[73] D. J. Bruemmer, R. L. Boring, D. A. Few, J. L. Marble, and M. C. Walton, “‘I call shotgun!’: an evaluation of mixed-initiative control for novice users of a search and rescue robot,” in *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, 2004, vol. 3, pp. 2847–2852.

[74] R. R. Murphy, “Human–Robot Interaction in Rescue Robotics,” *IEEE Trans. Syst. Man Cybern. Part C (Applications Rev.*, vol. 34, no. 2, pp. 138–153, May 2004.

[75] C. Escolano, J. Antelis, and J. Minguez, “Human brain-teleoperated robot between remote places,” in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 4430–4437.

[76] S. K. Agrawal, X. Chen, C. Ragonesi, and J. C. Galloway, “Training Toddlers Seated on Mobile Robots to Steer Using Force-Feedback Joystick,” *IEEE Trans. Haptics*, vol. 5, no. 4, pp. 376–383, 2012.

[77] M. J. Mataric, *The Robotics Primer*. The MIT Press, 2007.

[78] J. Borenstein and Y. Koren, “The vector field histogram-fast obstacle avoidance for mobile robots,” *IEEE J. Robot. Autom.*, vol. 7, no. 3, pp. 278–288, 1991.

[79] K.-M. Jung and K.-B. Sim, “Path planning for autonomous mobile robot using the Potential Field method,” *Int. J. Fuzzy Log. Intell. Syst.*, vol. 9, no. 4, pp. 315–320, 2009.

[80] L. A. Zadeh, “Fuzzy sets,” *Inf. Control*, vol. 8, no. 3, pp. 338–353, Jun. 1965.

[81] M. B. Montaner and A. Ramirez-Serrano, “Fuzzy knowledge-based controller design for autonomous robot navigation,” *Expert Syst. Appl.*, vol. 14, no. 1–2, pp. 179–186, 1998.

[82] R. Grimmett, *BeagleBone Robotic Projects*. Packt Publishing, 2013.

[83] Y. Okubo, C. Ye, and J. Borenstein, “Characterization of the Hokuyo URG-04LX laser rangefinder for mobile robot obstacle negotiation,” in *Unmanned Systems Technology XI*, 2009, vol. 7332, p. 733212.

[84] R. S. Rao, V. Kumar, and C. J. Taylor, “Planning and Control of Mobile Robots in Image Space from Overhead Cameras,” in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, no. April, pp. 2185–2190.

[85] A. Dadhich, *Practical Computer Vision- Extract insightful information from images using TensorFlow, Keras, and OpenCV*. Packt Publishing, 2018.

[86] J. Howse, *OpenCV Computer Vision with Python*. Packt Publishing, 2013.

[87] L. O. F. Liability *et al.*, “OpenHaptics Toolkit v.3.3 Programmer’s Guide.” .

[88] W. Gay, *Linux Socket Programming by Example*, 1st ed. Que Publishing, 2000.

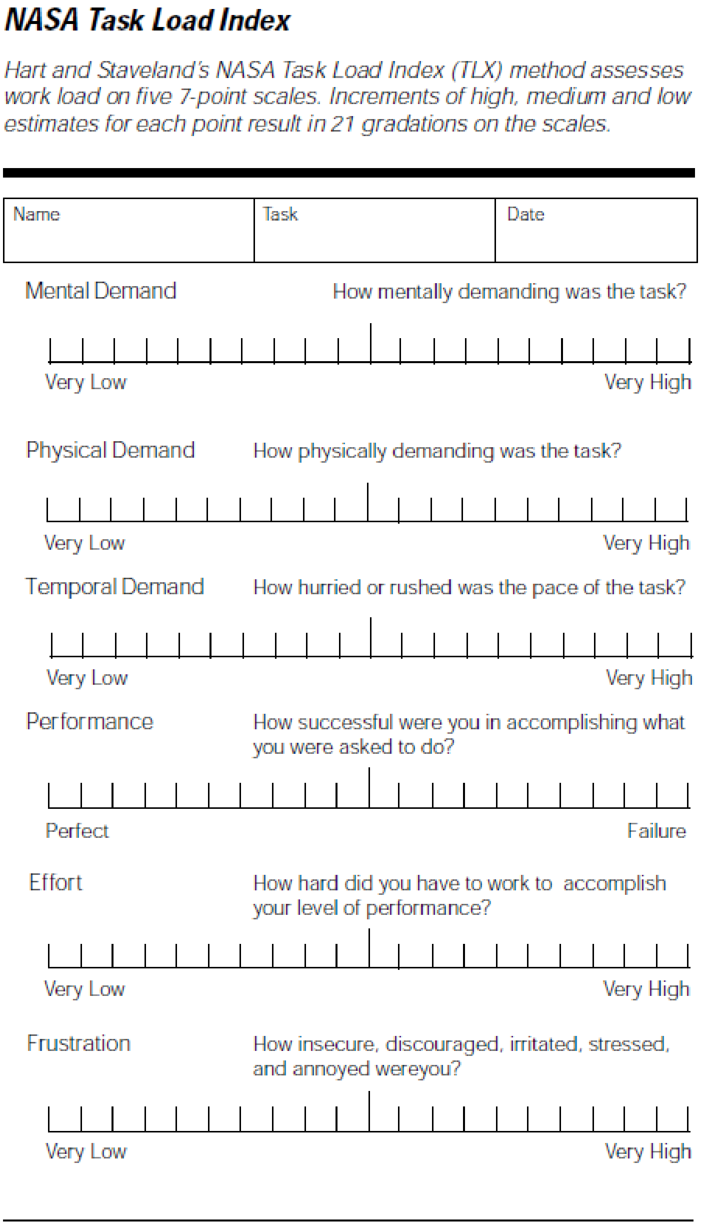
[89] A. Jones, *Network Programming for Microsoft Windows*. Microsoft Press, 1999.

[90] L. Salzman, “enet.bespin.org.” [Online]. Available: http://enet.bespin.org/index.html.

[91] S. G. Hart and L. E. Staveland, “Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research,” *Adv. Psychol.*, vol. 52, no. Human Mental Workload, pp. 139–183, 1988.

APPENDICES

A. NASA TLX Paper and Pencil Version



NASA Task Load Index

Hart and Staveland's NASA Task Load Index (TLX) method assesses work load on five 7-point scales. Increments of high, medium and low estimates for each point result in 21 gradations on the scales.

Name	Task	Date

Mental Demand — How mentally demanding was the task?

Very Low — Very High

Physical Demand — How physically demanding was the task?

Very Low — Very High

Temporal Demand — How hurried or rushed was the pace of the task?

Very Low — Very High

Performance — How successful were you in accomplishing what you were asked to do?

Perfect — Failure

Effort — How hard did you have to work to accomplish your level of performance?

Very Low — Very High

Frustration — How insecure, discouraged, irritated, stressed, and annoyed wereyou?

Very Low — Very High