Optimal Real-Time Tracking and Path Planning System for Omni-directional Mobile Robots

Eyad Almasri

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical and Electronic Engineering

Eastern Mediterranean University
February 2022
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Ali Hakan Ulusoy
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Doctor of Philosophy in Electrical and Electronic Engineering.

Assoc. Prof. Dr. Rasime Uyguroğlu
Chair, Department of Electrical and Electronic Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Doctor of Philosophy in Electrical and Electronic Engineering.

Prof. Dr. Mustafa Kemal Uyguroğlu
Supervisor

Examining Committee

1. Prof. Dr. Hasan Demirel
2. Prof. Dr. Ahmet Denker
3. Prof. Dr. Mehmet Siraç Özerdem
4. Prof. Dr. Mustafa Kemal Uyguroğlu
5. Asst. Prof. Dr. Ahmet Ünveren

ABSTRACT

Autonomous mobile robots have attracted growing interest due to their paramount significance in numerous applications including engineering, logistics, research and medical fields. The mobile robots are classified based on the nature of mobility into holonomic as well as nonholonomic. Omnidirectional wheeled mobile robot (OMRs) has concerned superior consideration due to its maneuvering proficiency as it is able to maneuver in any selected path directly from any position.

Producing the optimal trajectories for omnidirectional mobile robots so that it enhances the comprehensive performance of physical properties or decreases the consumption of the resources where the constraints system remains preserved among the most important research topics in the field of the holonomic mobile robot.

Due to the motion and energy proficiency of the omnidirectional wheeled mobile robots (OWMRs), it utilized to validate and test the efficiency and the competence of new methods functioning in dynamic environment and involving advanced activities such as trajectory planning optimization, moving ball estimation and obstacle avoidance.

In this thesis, first, a complete and compact mathematical model for the symmetrical annular-shaped omnidirectional wheeled mobile robot (SAOWMR) including the kinematic and dynamic aspects is derived and certified. This general mathematical model offers an occasion to perform research, experiments, and comparisons on the omnidirectional mobile robots that utilize two, three, four, five, or even more omnidirectional wheels without the necessity to switch models or derive a new model.

Then, a new computationally effective method based on the complete model of a SAOWMR with *n* omnidirectional wheels is developed so that improvements in the trajectory planning optimization (TPO) is achieved. Moreover, the trajectory planning optimization proposed method has been experienced in collision-free navigation by combining the problem of optimal control with the motion planner and the path constraints. Furthermore, in this thesis, ball tracking and estimating of the ball position is accomplished concerning the dynamic environment based on the Multi Sensor Fusion of the Kalman Filter. Additionally, a new method is proposed concerning the moving ball interception relative to the direction of the shoot so that the utilized motion planner based on the Artificial Potential Field method.

Experimental tests and simulations are offered pointing to confirm the efficiency and effectiveness of the proposed methods. Moreover, All the experiments in this thesis done based on the use of SAOWMR with three omnidirectional orthogonal wheels since it has full omnidirectional motion proficiency and provides three independent motion directions on a flat surface so that it maneuvers and navigates in tight spaces.

Keywords: Mathematical Modeling; Symmetrical Configuration; Omnidirectional Wheeled Mobile Robots (OWMRs); Motion Planning (MP); Holonomic Motion (HM); Obstacle Avoidance (OA); Multi Sensors Fusion (MSF); Kalman Filter (KF); Artificial Potential Field (APF); Moving Ball Interception (MBI)

ÖZ

Otonom mobil robotlar, mühendislik, lojistik, araştırma ve tıp alanları dahil olmak üzere çok sayıda uygulamada büyük önem taşımaları nedeniyle artan bir ilgi görmüştür. Mobil robotlar, hareketliliğin doğasına göre holonomik ve holonomik olmayan olarak sınıflandırılır. Çok yönlü mobil robotlar (OMR'ler), herhangi bir konfigürasyondan herhangi bir yöne hareket edebilme manevra yetenekleri nedeniyle üstün ilgi görmüştür.

Çok yönlü mobil robotlar için en uygun yörüngeleri üretme ve buna bağlı olarak fiziksel özelliklerin genel performanslarını iyileştirme veya kısıtlı sistemlerde kaynakların tüketimini azaltma, holonomik mobil robot alanındaki en önemli araştırma konuları arasında yer almaktadır.

Yıllık RoboCup yarışması, çok yönlü mobil robotların kullanılabileceği önemli bir fırsattır. Futbolcu çok yönlü mobil robotlar, çok yönlü mobil robotun optimum yörünge oluşturma, hareketli top durdurma ve engellerden kaçınma gibi gelişmiş faaliyetleri gerçekleştirebilmesi için dinamik ortama uygun etkili gerçek zamanlı yöntemler kullanır.

Bu tezde, ilk olarak, simetrik halka şeklindeki çok yönlü tekerlekli mobil robot (SAOWMR) için kinematik ve dinamik yönleri içeren kompakt ve eksiksiz bir matematiksel model türetilmiş ve doğrulanmıştır. Bu genel matematiksel model, modelleri değiştirmeye veya yeni bir model türetmeye gerek kalmadan iki, üç, dört, beş ve hatta daha fazla çok yönlü tekerlek kullanan çok yönlü mobil robotlar üzerinde araştırma, deney ve karşılaştırmalar yapmak için bir fırsat sunar.

Ardından, n çok yönlü tekerlekleri olan bir SAOWMR'nin tam modeline dayanan yeni bir etkili hesaplama yöntemi geliştirildi ve buna bağlı olarak yörünge planlama optimizasyonunda (TPO) iyileştirmeler elde edildi. Dahası önerilen yörünge planlama optimizasyonu yöntemi, optimal kontrol problemini hareket planlayıcı ve yol kısıtları ile birleştirerek çarpışmasız navigasyonda deneyimlenmiştir. Ayrıca bu tezde, Kalman Filtresinin Çoklu Sensör Füzyonuna dayalı olarak RoboCup ortamına ilişkin top takibi ve top pozisyonunun tahmini yapılmıştır. Ek olarak, kullanılan hareket planlayıcının Yapay Potansiyel Alan yöntemine dayalı olarak, atış yönüne göre hareketli top durdurma ile ilgili yeni bir yöntem önerilmiştir.

Önerilen yöntemlerin verimliliğini ve etkinliğini doğrulamak için deneysel testler ve simülasyonlar sunulmuştur.

Anahtar Kelimeler: Matematiksel Modelleme; Simetrik Yapılandırma; Çok Yönlü Tekerlekli Mobil Robotlar (OWMR'ler); Hareket Planlama (MP); Holonomik Hareket(HM); Engelden Kaçınma (OA); Çoklu Sensör Füzyon (MSF); Kalman Filtresi (KF); Yapay Potansiyel Alan (APF); Hareketli Top Durdurma (MBI)

To:

My Parents, thank you for your support, and unwavering belief in me. Without you, I would not be the person who I am today.

My Love Hiba, thank you for your love and constant support, for all the early morning and the late nights, for keeping me sane over the difficult times, thank you for being my best friend. I owe you everything.

My dear friend Dr. Ahmad Al Ahmad, thank you for being my real brother.

ACKNOWLEDGMENT

I would love to record my sincere thanks and gratitude to Prof. Dr. Mustafa Kemal Uyguroğlu for his encouragement and endless support of my Postgraduate studies. I gratefully acknowledge his supervision, guidance, patience, inspiration and invaluable advice he has provided me throughout this process. I would like to appreciate the opportunity given to me to be under his supervision. I would always be thankful and grateful to him for the rest of my life.

I would love to express my deepest gratitude to the Chair of the Electrical and Electronic Engineering Department Assoc. Dr. Rasime Uyguroğlu for her endless help, care and dedication. I am very grateful for all the support that she has extended to me during my postgraduate studies.

I would also like to extend my sincere gratitude to Prof. Dr. Hasan Demirel and Asst. Prof. Dr. Ahmet Unveren for their thoughtful comments and efforts towards improving this study.

I would also like to express my sincere thanks to all professors of the Electrical and Electronic Engineering Department for their support and dedication.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

D_i	Directional Vector perpendicular to the Motor Shaft of the *i*th Wheel.
E	Total Required Energy of the Omnidirectional Wheels' DC Motors.
F_i	The Force produced by the *i*th DC Motor.
J	The Mass Moment of Inertia of the SAOWMR.
k_τ	Torque constant of the DC motor.
L	The Radius of the SAOWMR.
m	The Mass of the SAOWMR.
n	Number of Omnidirectional Orthogonal Wheels of the SAOWMR.
P_{0n}	The Positional Vector of the *i*th Omnidirectional Wheel.
p_i	The Power Generated by the DC Motor attached to the *i*th Wheel
R	Armature Resistance of the DC motor.
r	The Radius of the Omnidirectional Wheel.
t_f	Total Maneuvering Time.
$u_{\max}$	The Maximum Input Voltage can be provided to the DC Motors
u_o	The Maximum Input Voltage applied to the Wheels DC Motors
u_{oi}	The Maximum Input Voltage supplies the *i*th DC motor
u_i	Voltage implemented on the input terminal of the *i*th Wheel.
v_i	Translational Velocity of the *i*th Omnidirectional Wheel.
ω_i	Angular Velocity of the *i*th Omnidirectional Wheel.
x	The Projection of the Center of Mass of the SAOWMR on the x-axis.
y	The Projection of the Center of Mass of the SAOWMR on the y-axis.

$\alpha_{\max}$	The Maximum Acceleration Amplitude
α, β	Constants specified using the Motor parameters.
γ	Positive weight states the trade-off of the E and the t_f
θ	Angle associates the Orientation of the SAOWMR.
φ	The Angle confined by adjacent Omnidirectional Wheels.

LIST OF ABBREVIATIONS

CDIB	Constrained dynamic inversion-based
CSMC	Coupled Sliding Mode Control
EKF	Extended Kalman Filter
FAHP	Fuzzy Analytic Hierarchy Process
KF	Kalman Filter
IACO	Improved ant colony optimization
LQT	Linear Quadratic Tracking
MPC	Model Predictive Controller
MPFTC	Model Predictive Fault-Tolerant Control
NMPC	Nonlinear Model Predictive Control
OCP	Optimal Control Problem
OMR	Omnidirectional Mobile Robot
OWMR	Omnidirectional Wheeled Mobile Robot
PID	Proportional-Integral-Derivative
SAOWMR	Symmetrical Annular Omnidirectional Wheeled Mobile Robot
SA	Simulated Anneling
SMC	Sliding Mode Control
TPO	Trajectory Planning Optimization
VO	Velocity Obstacles

Chapter 1

INTRODUCTION

1.1 Research Background and Motivation

Autonomous mobile robots have concerned growing interest due to their paramount significance in numerous utilizations including medical and industrial fields. The mobile robots are often categorized based on the type of mobility into holonomic and nonholonomic. Omnidirectional mobile robots (OMRs) have involved superior interest due to its maneuvering proficiency. OMRs can adjust its motion direction without requiring to apply intermediate steps compared to the traditional non-holonomic mobile robots [1]. Due to the energy proficiency and the low complexity of the omnidirectional wheeled mobile robots (OWMRs), it performs better than other OMRs such as the legged robots in regular terrain [2,3].

Omnidirectional wheels or what became famous as special wheels are extremely common for OMRs. Omnidirectional wheels intended to achieve traditional motions in the direction orthogonal to the shaft of the motor and sliding motions along the motor shaft providing the mobile robot extra flexibility to track complicated trajectory.

Omnidirectional wheeled mobile robot (OWMR) based on its motion proficiency is able to use effective real-time methods suitable for dynamic environment for path planning and trajectory generation so that the mobile robot can performing unconventional activities, such as path tracking, moving ball interception and obstacle avoidance.

Finding the optimal trajectory of the mobile robot considered as the search procedure for the optimal path of the robotic system so that it enhances the performance of the robot’s properties or reduces the consumption of the resources or traveling time where the system of constraints remains conserved. The trajectory optimization problem can be model mathematically as optimal control problem. optimal control methods handles the problem of finding the input control (such as the input voltage) for a specified system so that some optimality criteria such as the velocity, acceleration, voltage saturation are achieved.

1.2 Thesis Objectives

The main objectives of this thesis are as follows:

1. To generalize a compact and complete mathematical model including kinematic and dynamic models for the symmetrical annular-shaped omnidirectional wheeled mobile robot (SAOWMR) with n omnidirectional wheels.
2. To develop a new computationally effective method based on the compact model of a SAOWMR with n omnidirectional wheels so that improvements achieved in trajectory planning optimization.
3. To solve the problem of obstacle avoidance optimal trajectory planning for SAOWMR with n omnidirectional wheels by integrating the optimal control problem with the motion planner and the path constraints.
4. To utilize the Multi Sensor Fusion techniques in the field of the omnidirectional wheeled mobile robot soccer performing ball tracking and estimating of the ball position in dynamic environment.

5. To perform moving ball interception in dynamic environment so that a mobile robot located behind the ball relative to the shooting direction can manage the matched shooting configuration and shoot the ball to the goal.

1.3 Thesis Contribution

The main contribution of this thesis comprises as following:

1. A complete mathematical model for the SAOWMR with n omnidirectional orthogonal wheels is achieved and proved. This general mathematical model provides an occasion to perform studies, investigations, and evaluations on the omnidirectional wheeled mobile robots that utilize two, three, four, six, or even more omnidirectional wheels without the requisite to switch models or develop a new model.
2. A new computationally effective method in trajectory planning optimization (TPO) is developed concerning the compact mathematical model of SAOWMR with n omnidirectional orthogonal wheels. The developed procedure based on approximating the control input vector U applied to the terminals of the DC motors relevant to the robot's omnidirectional wheels so that the robot able to maneuver from a primary to a final configuration of boundary conditions with the optimality of total required energy E and maneuvering time t_{f} while a system of constraints remains maintained.
3. A new method proposed on the basis of Artificial Potential Field (APF) method to solve the obstacle avoidance optimal trajectory planning problem concerning the SAOWMR with n omnidirectional wheels. The proposed method obviously illustrates the fulfillment of the robot dynamic system and the constraints of the path, as well as the fluency of the motion control.

4. A global multi-agent sensor fusion achieved using the Kalman Filter in the context of the mobile robot soccer performing ball position estimation and tracking in dynamic environment.
5. A new method proposed regarding the moving ball interception relative to the direction of the shoot so that the omnidirectional wheeled mobile robot can maneuver and estimate the optimal configuration of shooting to the goal on the basis of the Artificial Potential Field method.

1.4 Thesis Outline

The remainder of the thesis is organized as follows:

1. Chapter 2 discusses several applications for the omnidirectional wheeled mobile robot in numerous fields. Moreover, it focuses on the structures and the functions of different designs of the omnidirectional wheels utilized in the literature. The topology and the mobility performance of the omnidirectional wheeled mobile robot are also considered. Several previously published works are discussed concerning the control and trajectory generation of the OWMRs. In the present of obstacles, several motion planning techniques mentioned regarding the OWMRs. Furthermore, the chapter also consider the navigational characteristics and the localization strategies of the mobile robots.
2. Chapter 3 provides a compact and complete mathematical model for the SAOWMR with n omnidirectional wheels including the kinematic and the dynamic model. This development gives an opportunity to conduct research, experiments, and comparisons on omnidirectional mobile robots that have two, three, four, six, or even more omnidirectional wheels without the need to switch models or derive a new model.

3. Chapter 4 targets to develop a novel computationally effective method on the basis of the compact model of the SAOWMR with n omnidirectional wheels so that enhancements concerning the TPO can be attained.
4. Chapter 5 simulates, tests and discusses the methodology proposed in the Chapter 4 of solving the optimal control problem so that three-wheeled omnidirectional mobile robot will be chosen to perform the experiments since it has complete omnidirectional motion proficiency and provides three independent motion directions on a flat surface. The experimental results will be compared and discussed with those generated by the constrained dynamic inversion-based (CDIB) method.
5. Chapter 6 aims to solve the problem of obstacle avoidance optimal trajectory planning for SAOWMR with n omnidirectional wheels by integrating the problem of optimal control with the motion planner and the path constraints.
6. Chapter 7 utilizes the multi-agent global sensor fusion based on the Kalman Filter in the context of the omnidirectional wheeled mobile robot soccer performing ball tracking and estimating of the ball position in dynamic environment.
7. Chapter 8 discusses the moving ball interception relative to the direction of the shoot that the mobile robot located behind the ball relative to the shooting direction. The motion planner considered in this chapter based on the Artificial Potential Field method.
8. Chapter 9 summarizes the contents of the thesis providing some conclusion, as well as, discusses the possible scope for future work.

Chapter 2

LITERATURE REVIEW

Omnidirectional wheeled mobile robots (OMRs) have concerned superior interest based on its maneuvering proficiency. OMR can adjust its direction of movements without requiring to apply middle way steps compared to the traditional non-holonomic mobile robot. Due to the energy proficiency and the low complexity of the omnidirectional wheeled mobile robots (OWMRs), it performs better than other OMRs as for example the legged robots in even platform.

2.1 OWMRs Applications

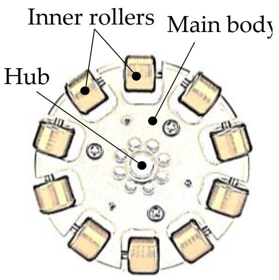
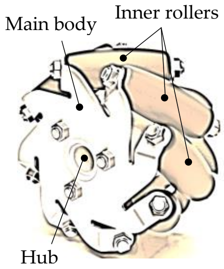
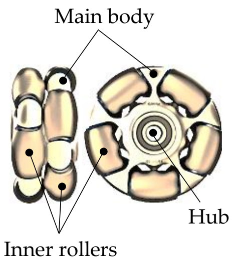
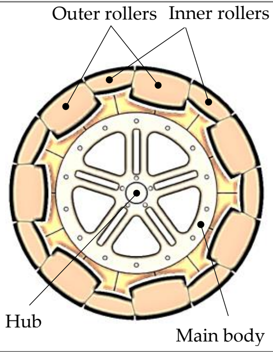
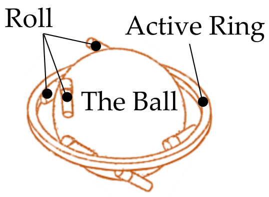
There are numerous applications for the OWMR in various fields. In regards to the industrial field, the authors in [4] considered an OWMR with four Mecanum wheels to smoothly move industrial's equipment. In [5], an industrial robot drilling system for aerospace manufacture is proposed to use so that the Mecanum wheels are utilized for the robot's movements in crowded environment. Enhanced tracking and localization method for an omnidirectional wheeled industrial mobile robot is developed in [6] so that the high positional accurateness requisite meet. Moreover, it is proposed in [7] to utilize the six Mecanum wheeled mobile robot to hold weighty load. Regarding to the medical field, the authors in [8] developed an omnidirectional based robot complements the healthcare professionals so that it carries the medical supplies and records electronic health data. Furthermore, the study proposed in [9, 10] utilizes three orthogonal wheels OMR to maneuver indoor as a personal assistant taking care and provide helpful and social services to aged people. Moreover, an omnidirectional wheeled based robotic nurse assistant is proposed in [11] so that it improves the efficiency of the health care providers can deliver. This assistant can enhance a nurse's functioning environments by off-loading most physically demanding duties so it is reducing the possible injury to the patient. In regards to the space research and outdoor applications, the research in [12] provides strategies for proposing design of OWMR working in rough outdoor environment so that its geometrical constraints applied on the wheels and the platform besides the affection on the terrain unevenness and the connection between the wheel and the ground are investigated. In [13], a new odometry improvement technique for an OMR that has extraordinary mobility in irregular ground platform proposed so that it is able to decrease the error on the position estimation due to the slippage of the wheel. The study investigated in [14] suggests usage of active split offset caster that consists of four sets of mobility modules so that the omnidirectional mobile robot able to navigate and achieve full mobility in rough terrain. Furthermore, an OWMR with flexible chassis and orthogonal wheels in [15] utilized in the field of climbing and high maneuverability. Additionally, the small and middle sized of the annual RoboCup league competition is an instance of where OMR can be utilized. The structure of the soccer robot in [16] is an OWMR utilizes three special orthogonal wheels skilled to perform innovative activities, for instance goal keeping and ball passing. In [17], the authors proposed an approach concerning omnidirectional wheeled based robot soccer so that grid-based path planning and polynomial trajectory generator utilized to achieve a real-time motion planning in dynamic environment. Regarding dealing with hazardous environments. The paper in [18] develop a remote control system of a navigational internet-based for the OWMR such that the mobile robot be able to maneuver safety in highly risk environments.

2.2 Omnidirectional (Special) Wheels

Omnidirectional special wheels are very popular for Omnidirectional mobile robot. Omnidirectional wheels invented to achieve conventional motions with the direction orthogonal to the shaft of the motor and slipping motions along the motor’s shaft. This special motions provide extra flexibility to the mobile robot in the congested environment [19].

Different structural styles of omnidirectional special wheels are illustrated in the Figure 1. In [20], the conception of the omnidirectional wheel suggested for the first time by J. Grabowiecki. In Figure 1a, an omnidirectional wheel constructed by a centric body and several passive small rollers enclosing the wheel circumference so that the wheel able to slip with the motor’s shaft direction. This special wheel usually called orthogonal wheel since the axes of its inner rollers are orthogonal to the wheel shaft. The actual usage of this scheme of the omnidirectional wheels lead to numerous concerns regarding the platform motion accuracy and constancy such as the slippage and vibrations due to the unstable connection between the ground and the wheel. Considerable attempts were completed successfully by scientists to produce improved schemes of the orthogonal wheels to accomplish greater enhancements. In the research paper detailed in [21], an innovative scheme assembled by a Swedish engineer concerning an omnidirectional wheel named as Mecanum wheel, illustrated in Figure 1b, so that the sliding motion rollers are placed on angles in an endeavor to attain constant connection between the ground surface and the omnidirectional wheel. The early version of the OWMR utilizes four Mecanum wheels was constructed in the research paper presented in [22] and named as URANUS. Numerous researches proposed based on the concept of the Mecanum wheels for instance in [23-27]. The benefits and the weaknesses of the use of the OWMRs utilize the Mecanum wheels are presented in [28,29]. The usage benefits of the OWMRs with the Mecanum wheels concerned by the omnidirectional fulfillment that the Mecanum wheel offers, permitting the mobile robot to maneuver with complete mobility performance in a congested surroundings. Furthermore, it is verified that the Mecanum wheels holds superior performance regarding the balancing movements compared to the earlier sorts of the special wheels. On the other hand, several disadvantages showed by utilizing the Mecanum wheels. A mobile robot with the Mecanum special wheels are subjected to the slippage. Modeling and calculation of this slippage is a problematic issue. Moreover, the space among every two neighboring passive rollers gives rise to vertical and horizontal vibrations in the robot platform, that is obviously affect the performance of the omnidirectional wheeled mobile robots. The dual parallel coincident omnidirectional wheel concept, shown in the Figure 1c, proposed in [30] as an improved construction of the traditional orthogonal special wheel in order to enhance the performance of the OWMRs. The conception of the overlapping dual omnidirectional special wheel has reached a considerable attractiveness in academic and commercial sides so that hundreds of research works mentioned this scheme. In [31], one more improved design of the orthogonal special wheel, depicted in Figure 1d. It guarantees constant contacting concerning the wheel and the ground surface. This improved design suggested to attain enhancements in the stability performance by minimize the gaps of the adjacent rollers. In [32], an alternative design concerning the omnidirectional special wheel named as the ball wheel scheme, presented in Figure 1e. It utilizes rollers formed in two arrays of circled shaped and an active ring able to rotate about an axis through the center of ball wheel. It is confirmed that an OWMR that utilizes three ball wheels holds complete performance regarding the mobility.

Table 2.1: Categories of the omnidirectional special wheels and their traces

Type	Scheme	Trace	Details
Primitive Orthogonal Wheels		_ _ _ _	The basic scheme of the orthogonal special wheels holds several internal rollers circulating the wheel arrangement so that the axes of the rollers are perpendicular to the wheel's axis.
Mecanum Wheel		\\\\	An omnidirectional wheel special design called the Mecanum wheel so that its rollers are attached on angles in an effort to accomplish constant connection of the ground and the wheel.
Dual Overlapping Orthogonal Wheels		_‾_‾_	Overlapping dual orthogonal special wheel constructed as an improved type of the traditional orthogonal special wheel
Continuous Alternate Orthogonal Wheels		–––––	A superior scheme of the orthogonal special wheel intended on the base of the usage of inner and outer passive rollers in order to guarantee the constant connection among the ground and the wheel.
Ball Wheel		———	The mechanism of the ball wheel constructed based on the usage of two arrays of inner rollers amounted about two circles. Moreover, an active ring established able to rotate about the axis of the ball center.

2.3 Topology and Mobility Performance of the OWMRs

The circulation of the omnidirectional special wheels about the circumference of the mobile robot platform and its relation with the mobility performance is an important matter concerning the omnidirectional wheeled mobile robot. It is confirmed that the optimal configuration is the rectangular symmetrical configuration of four standard Mecanum special wheels [33]. Moreover, a formulation of three Mecanum wheels circulated uniformly on the rounded shape platform also can be utilized with complete mobility, as detailed in [34-35]. Regarding the OWMRs that utilize the orthogonal special wheels that roller axes are perpendicular to the shaft of the wheel, the widely held scheme is the rounded platform surrounded by three or four wheels. regarding the in height center of gravity problem, OWMRs with three omnidirectional special wheels meet in some cases stability problems [36-37]. Adding more omnidirectional wheels may handle the stability problem [1]. Occasionally, the need of enhancing the handling and transportation capability of an OWMRs is an another significant purpose to rise the total number of the omnidirectional special wheels [7].

2.4 Control and Trajectory Generation of the OWMRs

Various researches have been published based on modeling and controlling of the OWMRs. The authors in [38] proposed a new algorithm regarding the feedback kinematic controller of OMRs utilize four Mecanum wheels. In [39, 40], kinematics and dynamics model in addition to the mobility analysis were discussed concerning the OMRs that utilize three orthogonal special wheels. The method of resolved control method, PID controller method, fuzzy logic control model method, and the stochastic fuzzy servo method were utilized in [41] to improve the system of feedback control concerning OMRs that utilizes three orthogonal special wheels. A methodology of motion control built upon a linearized inverse kinematic model proposed in [42]. In

[43], A computationally low-cost near-optimal method proposed to handle the control and trajectory generation problem concerning OMRs utilize three orthogonal special wheels. The research works mentioned in [44, 45] expressed the problem of optimal trajectory generation as a near-optimal minimal time trajectory that the solution built upon a relaxed optimal control problem (OCP).

Numerous research works proposed recently provides considerable volume of contributions and originality concerning the OMRs that use omnidirectional special wheels. In [46], the authors presented a kinematics and dynamics novel model based on the principle of virtual work concerning OMR that utilize four Mecanum wheels. In [47], A new energy modeling technique allows the mobile robot to compute and make a prediction of the consumption of energy. The forward and inverse kinematics concerning the OMR that utilizes six Mecanum wheels simulated and detailed in [48]. In [49], a control method proposed based on the usage of a 32 bits ARM microcontroller that the kinematic modeling system counted as a feedback. A model predictive fault-tolerant control (MPFTC) system intended in [50] to enhance the performance of the OWMR that utilize the four Mecanum wheels so that the mobile robot properly exploit the inherited actuation redundancy. A fuzzy logic of type-2 controller proposed to use in [51] in order to control the speed of the motor concerning the omnidirectional wheels so that the mobile robot performance keeps stable. The research in [52] focuses on solving the trajectory tracking problem concerning the OWMRs on uneven terrain based on the use of dual closed-loop strategy control. proportional derivative (PD) controller along with the dynamic optimization approach proposed to use in [53] concerning the OMRs that the error of tracking and the consumption of energy will be minimized.

In [54-57], fuzzy controller designed to permit tracking the selected trajectory concerning the OWMRs. To accomplish point stabilization along with the trajectory generation, a Model Predictive Controller (MPC) algorithm proposed to use in [58]. In [59], the authors proposed a hybrid method constructed by Sliding Mode Controller (SMC) along with back stepping method to control the OWMRs so that the kinematic and dynamic model utilized. To attain an acceptable tracking controller for OWMRs, a Coupled Sliding Mode Control (CSMC) method was suggested to use in [60]. Concerning the trajectory tracking of OMRs in the existence of uncertainty, adaptive robust control was proposed in [61]. In [62], flatness-based scheme platform proposed to use for OMRs so that the control input from the simulation model was reallocated to the hardware of the mobile robot. Moreover, a Kinect-based vision system utilized for the feedback in order to simulate the model. linear quadratic tracking based controller along with PID controller were proposed in [63] concerning the OMRs utilize four omnidirectional special wheels to achieve tracking of the trajectory. Constrained dynamic inversion-based (CDIB) method designed based on incorporation of the vehicle dynamics proposed in [64] aiming to achieve the optimality of the trajectory generation concerning OWMRs. In [65], A new control system was suggested on the basis of the mixed reality of the path planning concerning the OWMRs. In [66], the velocity tracking problem proposed regarding the OWMR by using a new control law voltage-based along with graphical feedback. In [67], an improved ant colony optimization (IACO) suggested in order to find the optimality of the path concerning the OWMR.

Table 2.2: Types of the controllers applied on the OWMRs

Refs	OWMR scheme	System Control Method	Details
[38]	Symmetrical rectangular configuration OWMR with four standard Mecanum wheels	Kinematic-based feedback control system	Kinematic feedback control system applied by calculate the robot configuration in real time so that the position error will be calculated then the control algorithm executed to compute the actuator velocity.
[42]	Symmetrical rectangular configuration OWMR with four standard Mecanum wheels	Model Predictive Controller (MPC)	The controller expects the system states on the basis of the prediction horizon aiming to solve the consequential open loop optimal control problem (OCP) online
[50]	Symmetrical rectangular configuration OWMR with four standard Mecanum wheels	Model Predictive Controller	The paper considers an actuation fault so that the wheel is not able to receive the signal. A Non-linear Model Predictive Controller is utilized to properly utilize the actuation redundancy inherited of the Omni-directional mobile robot and provides solution for multiple combinations of actuation faults.
[51]	Symmetrical rectangular configuration OWMR with four standard Mecanum wheels	Type-2 fuzzy logic controller	The research utilized fuzzy logic controller and kinematic equations in order to control the speed of the wheels so that the robot can follow a specific trajectory
[53]	Symmetrical OWMR with three dual orthogonal wheels	Cartesian space proportional–derivative control	The controller works along with the dynamic optimization method in order to minimize the error while the tracking and the energy consumption considered concurrently along the trajectory.
[54]	Symmetrical OWMR with three dual orthogonal wheels	Fuzzy logic controller	The fuzzy logic controller aims to control the linear and the angular velocity of the omnidirectional mobile robot on the basis of the information of the robot’s configurations

[55]	Symmetrical OWMR with three dual orthogonal wheels	Fuzzy controller	Fuzzy controller designed based on the mathematical modeling of the omnidirectional mobile robot then the control system design is applied for the linear and angular velocity.
[57]	Symmetrical rectangular configuration OWMR with four standard Mecanum wheels	Fuzzy controller	The suggested principle of fuzzy-based control approach aims to involve the motion controller and the obstacle avoidance fuzzy controller. The outputs of these two controllers propose to control the robot wheels.
[58]	Symmetrical OWMR with three dual orthogonal wheels	Model Predictive Controller	A model predictive control is intended aiming to achieve point stabilizing and trajectory tracking
[59]	Symmetrical OWMR with four orthogonal wheels	Sliding Mode Control	hybrid method suggested to control the OWMRs by combining the sliding mode control along with back stepping technique.
[60]	Symmetrical rectangular configuration OWMR with four standard Mecanum wheels	Coupled Sliding Mode Control	Coupled sliding mode control approach was suggested to use aiming to achieve a satisfactory tracking control for OWMRs.
[61]	Symmetrical OWMR with three dual orthogonal wheels	Adaptive controller	Adaptive robust controller was suggested concerning the trajectory generation of the OMR in the existence of uncertainty.
[62]	Symmetrical rectangular configuration OWMR with four standard Mecanum wheels	flatness-based controller	flatness-based scheme platform proposed to use for OMRs so that the control input from the simulation model was reallocated to the hardware of the mobile robot. A Kinect-based vision system utilized for the feedback in order to simulate the model.
[63]	Symmetrical OWMR with four orthogonal dual wheels	PID controller and linear quadratic tracking controller	PID controller and linear quadratic tracking-based controller applied for OWMR in order to achieve trajectory tracking

2.5 Motion Planning of the OWMRs in the Occurrence of Obstacles

The trajectory generation problem in the existence of the obstacles has been discussed in various research studies concerning the OWMRs. In [68], the method proposes to plan a path reference meets the obstacle avoidance settings on the basis of the Bézier curves, at that time the robot dynamic constraint has to be resolved as a time minimum trajectory generation problem. In [69], the optimal trajectory generation problem in the existence of static obstacles expressed as a constrained nonlinear optimal control problem. Moreover, a stated problem solved using a direct method of numerical solution. In [70], the researchers recommended a new hybrid technique established on the basis of integration the global deliberate approach along with local reactive approach so that the mobile robots solve the motion planning problem in the existence of the dynamic obstacles. A model predictive control strategy designed in [71] based on the potential field method so that OWMR with three orthogonal wheels can navigate while avoiding the obstacles. The research in [5] proposed to use a vision-based navigation concerning OWMR that utilize four Mecanum wheels so that the robot able to maneuver and avoid collisions with obstacles. In [72] it is proposed to hybridize A* algorithm, the fuzzy hierarchy procedure along with the bio-inspired controller so that the mobile robot achieves efficient time execution performance in both static and dynamic operational settings. To achieve dynamic obstacle avoidance, the velocity obstacles (VO) method is proposed to utilized along with the nonlinear model predictive control (NMPC) in [73]. In [74], it is suggested to combine the type-2 fuzzy logic with the adaptive controller so that the OWMRs able to achieve trajectory tracking and avoid the obstacles. The authors in [75], suggested to use fuzzy based method concerning three special wheeled OMR so that the robot be able to track the path while avoiding some obstacles. In [76], swarm intelligence optimization technique proposed to use concerning mobile robot path planning in the existence of dynamic and static obstacles.

2.6 Localization and Navigational Characteristics of the Mobile Robots

Localization analysis and Navigational characteristics concerning the mobile robots are indispensable sides so that numerous recent researches provide contributions to this field. The research work proposed in [77] suggested a new human-aware robot navigation method concerning the mobile robots in household environment so that the robot able to maneuver based on the user predilection. In [78], a vision based navigation method is proposed concerning the mobile robot so that scene classification reassigned from visual navigation problem. A multimodal deep learning method proposed in [79] to ensure the navigation in static indoor environment concerning the mobile robots. In [80], a probabilistic method designed based on sensor fusion for compound mobile robots in both outdoor and indoor environments to determine the position and orientation of the mobile robot. The research proposed in [81], suggested to use 2D LiDAR so that an autonomous mobile robot can maneuver through fenced yield so that it performs numerous phenotyping responsibilities in the agriculture industry environment. In [82], the authors subjected to solve the problem of navigation concerning the mobile robots in non-line-of-sight condition using a new localization method. The work done in [83] aims to improve the tracking of the mobile robots by utilizing stereo camera. In [84], the authors proposed to improve the localization performance concerning of the mobile robots by utilizing a neural network data fusion method. A light emitting diode based system intended and proposed to use in [85] aiming to enhance the localization and the communication system concerning the mobile robot so that the Kalman Filter was utilized to attain an enhanced the expectation of the mobile robot’s configuration.

Chapter 3

MATHEMATICAL MODELING OF THE SYMMETRICAL ANULAR OMNIDIRECTIONAL WHEELED MOBILE ROBOT (SAOWMR)

3.1 SAOWMR Topology and Characterization

A SAOWMR is an omnidirectional wheeled mobile robot with complete omnidirectional movement abilities if it possesses more than two omnidirectional wheels so that the robot can transport instantly concerning any path from any position. The movability benefit of this OWMR is due to the utilization of the orthogonal special wheels that perform dual motions with the direction of the shaft of the motor and with the direction orthogonal to the shaft of the motor permitting the wheels to proceed concurrently in the dualistic directions.

The SAOWMR holds numerous omnidirectional special wheels circulated commonly about a circular shaped platform. Although numerous models of the orthogonal omnidirectional special wheels are offered to be applied, the continuous alternate orthogonal omnidirectional wheels were chosen to utilize in this research to perform reliable effectiveness as they offer continuous trail concerning the surface of the wheel and the ground. The aforementioned orthogonal special wheel contains of a center handling a number of interior rollers and larger exterior rollers organized consecutively around the boundary of the wheel so it is able to freely slip in the motor direction of the motor shaft. The angle concerning the axis of the roller and the axis of the wheel is 90°. The construction and the arrangements of the mentioned continuous alternate special wheel is demonstrated in the Figure 3.1.

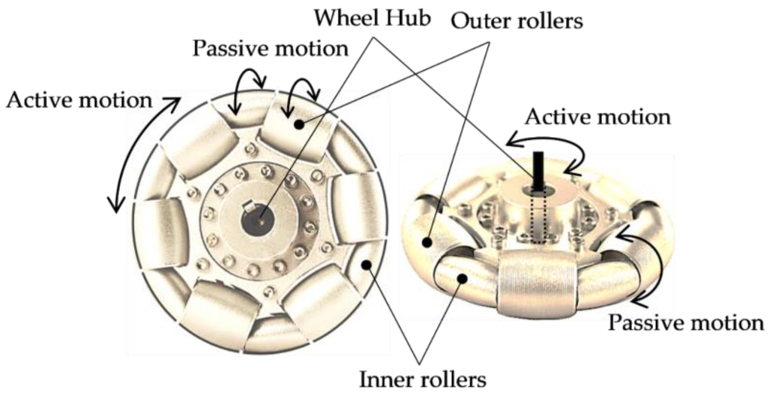


Figure 3.1: The arrangements of the continuous alternate special wheel

The SAOWMR achieves the holonomicity of motion that the rotation and translation movements occur simultaneously and independently on the basis of the usage minimum of three omnidirectional special wheels involved to the individual actuators. However, increasing the number of omnidirectional wheels be able to resolve some of the stability issues, as presented in [1]. Furthermore, the need of upholding the loud and the capacity of the transportation of the robot is also an essential purpose to increase the total number of the omnidirectional special wheels [7]. A SAOWMR with just two omnidirectional wheels is yet able to transform from one position to another by dividing the movement into rotation and translation.

Figure 3.2 illustrates the arrangement of the SAOWMR. In this omnidirectional wheeled mobile robot, n number of omnidirectional wheels circulated about the platform of the robot so that the angle concerning any two contiguous wheels is φ. The direction of rotation is considered positive if it is measured as the rotation in the counter-clockwise direction. Additionally, the radius of the SAOWMR is mentioned in Figure 3.2 as L.

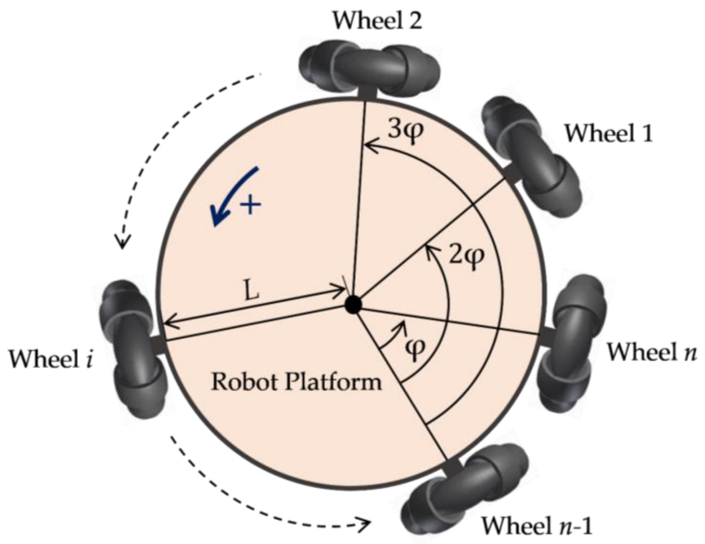


Figure 3.2: The arrangement of the SAOWMR with n omnidirectional wheels

An actual model of a SAOWMR utilizes three omnidirectional wheels demonstrated in the Figure 3.3.

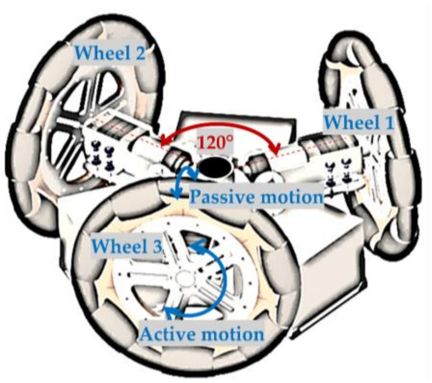


Figure 3.3: Actual model of a SAOWMR utilizes three omnidirectional wheels

A SAOWMR holds remarkable symmetrical configuration as its geometric character be able to be separated into corresponding fractions that are structured in a symmetrical manner. Numerous categories of symmetry can be detected obviously regarding to the

SAOWMR such as rotational, reflexive, scale symmetry and translational. Additional clearly identified symmetrical characteristic is that the total number of the axes of symmetry of the SAOWMR is identical to the total number of its wheels. The topological designs of SAOWMRs shown in Figure 3.4.

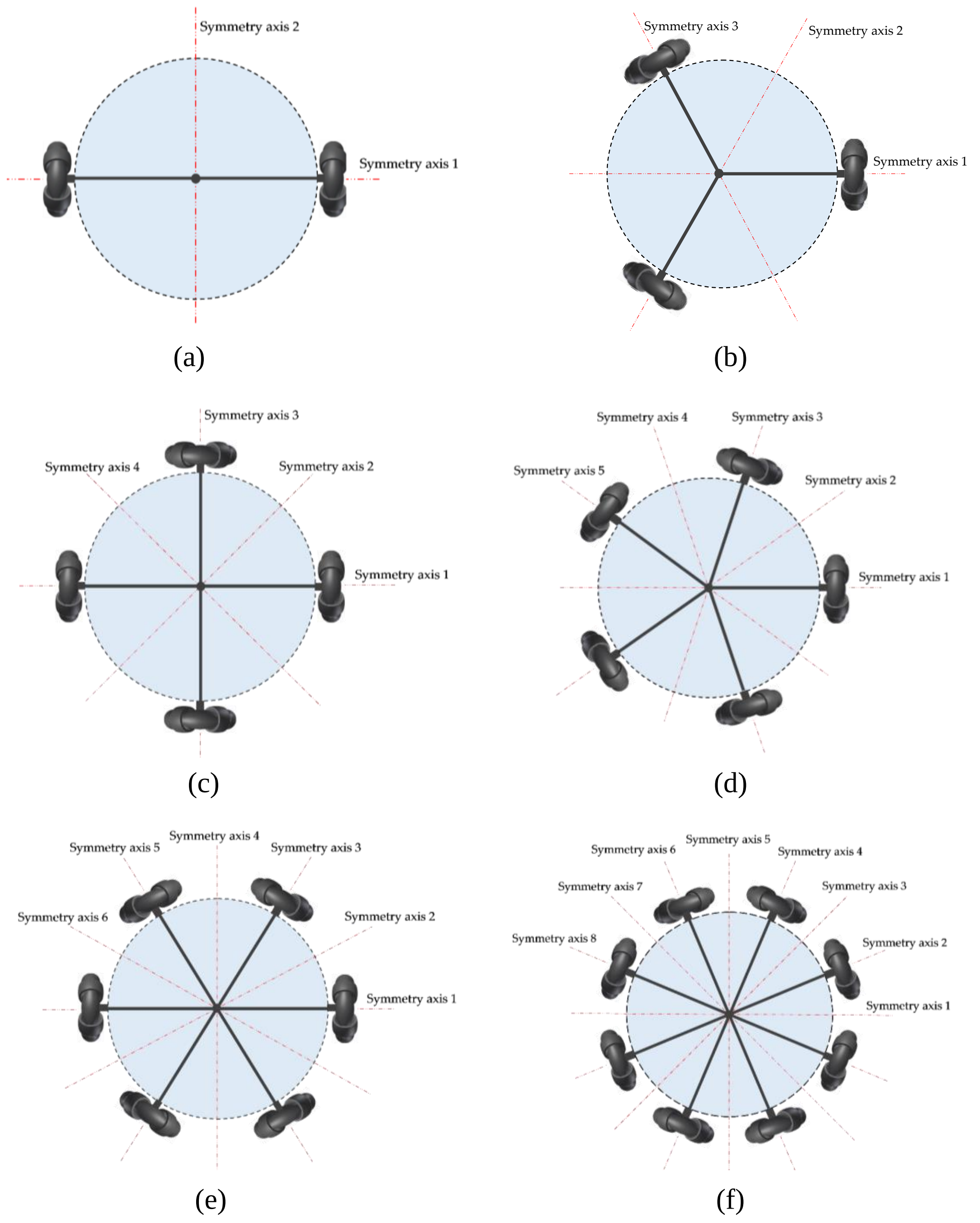


Figure 3.4: The topological models of the SAOWMRs: (a–f) are respectively the SAOWMR along with two, three, four, five, six, and eight special wheels

3.2 Geometry of the SAOWMR

Figure 3.5 illustrates the general configuration of the SAOWMR considered in this research. The robot assembly consist of n special orthogonal wheels compatibly arranged at φ degrees on the circumference of the mobile robot's platform. In this concern, the angle φ rely on the total number of the provided orthogonal wheels, for instance, φ matches 180° if the SAOWMR has only two omnidirectional wheels and 90° if the mobile robot holds four omnidirectional wheels. The motor shaft and its wheel has the identical axes of rotation. Subsequently, translation velocities $v_1, v_2, \cdots, v_n$ are distributed on n orthogonal wheels respectively with the directions $\mathrm{D}_1, \mathrm{D}_2, \cdots, \mathrm{D}_n$ as shown in the Figure 3.5. Furthermore, angular velocity of the ith wheel is given as ω_i. Furthermore, the direction of the counter-clockwise is identified to be the positive direction of rotation.

Moreover, a local system of reference frame $\left(\mathrm{o}_l, \mathrm{x}_l, \mathrm{y}_l\right)$ is attached to the robot's center of gravity so that the axis x_l is selected to be along the rotational axis of the first special wheel. Additionally, n systems of reference frames is assigned evenly to n special orthogonal wheels so that the rotational shaft of the ith wheel corresponds with the x_i axis of the frame $\left(\mathrm{o}_i, \mathrm{x}_i, \mathrm{y}_i\right)$. Furthermore, a global reference frame system $\left(\mathrm{o}_f, \mathrm{x}_f, \mathrm{y}_f\right)$ is assigned in order to identify the position and the orientation of the mobile robot so that the axes x_f and x_l and are related using the angle θ. The radius of the SAOWMR is given as L. The vector P_{0i} is the vector position of the ith wheel drive associated with the reference frame $\left(\mathrm{o}_l, \mathrm{x}_l, \mathrm{y}_l\right)$. $\left(P_{0_x}, P_{0_y}\right)$ is the projection point of the center of mass of the mobile robot on the global reference frame $\left(\mathrm{o}_f, \mathrm{x}_f, \mathrm{y}_f\right)$.

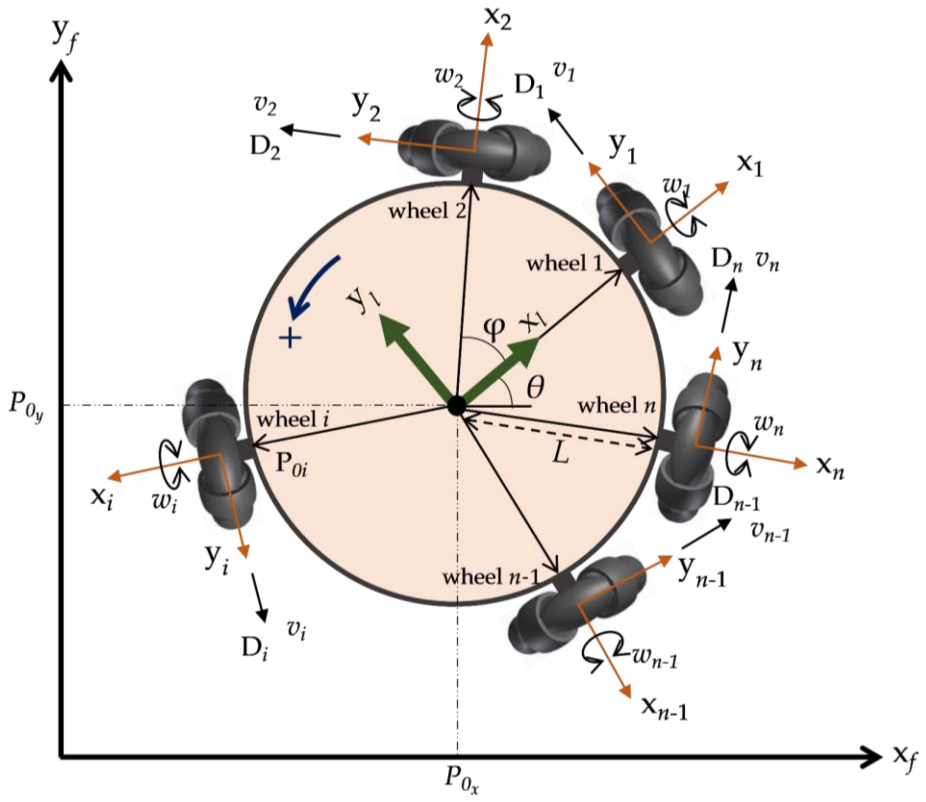


Figure 3.5: The schematic of the SAOWMR with n wheels

3.3 Kinematic Analysis of the SAOWMR

The comprehensive transformation matrix relates the ith wheel reference frame (o_i, x_i, y_i) and the reference frame associated with the mass center of gravity of the mobile robot (o_l, x_l, y_l) can be specified as following:

$$ {}^{l}\mathbf{T}_i = \begin{bmatrix} \cos((i-1)\varphi) & -\sin((i-1)\varphi) & 0 & L\cos((i-1)\varphi) \\ \sin((i-1)\varphi) & \cos((i-1)\varphi) & 0 & L\sin((i-1)\varphi) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1} $$

In Equation (3.2), P_{0i} is representing the positional vector of the ith wheel center considered in the reference frame (o_l, x_l, y_l).

$$ {}^{l}\mathrm{P}_{0i} = {}^{l}\mathbf{T}_{i} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} L\cos\left(\left(i-1\right)\varphi\right) \\ L\sin\left(\left(i-1\right)\varphi\right) \\ 0 \\ 1 \end{bmatrix} \tag{3.2} $$

The unit directional vector D_i associated with the i^{th} omnidirectional wheel in the local system of reference frame $\left(\mathrm{o}_l, \mathrm{x}_l, \mathrm{y}_l\right)$ is provided as

$$ {}^{l}\mathrm{D}_{i} = {}^{l}\mathbf{R}_{i} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -\sin\left(\left(i-1\right)\varphi\right) \\ \cos\left(\left(i-1\right)\varphi\right) \\ 0 \end{bmatrix} \tag{3.3} $$

R given in Equation (3) is the rotation portion of the homogeneous transformation matrix **T**. Moreover, the local system of reference frame $\left(\mathrm{o}_l, \mathrm{x}_l, \mathrm{y}_l\right)$ associated with the global system of reference frame $\left(o_f, \mathrm{x}_f, \mathrm{y}_f\right)$ by the general transformation matrix clarified in following equation

$$ {}^{f}\mathbf{T}_{l} = \begin{bmatrix} \cos\left(\theta\right) & -\sin\left(\theta\right) & 0 & x \\ \sin\left(\theta\right) & \cos\left(\theta\right) & 0 & y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4} $$

$\left(x, y\right)$ is the point projection of the center of gravity of the mobile robot in the global system of reference frame $\left(o_f, \mathrm{x}_f, \mathrm{y}_f\right)$. Moreover, the unit directional vector D_i can be given in terms of the frame $\left(o_f, \mathrm{x}_f, \mathrm{y}_f\right)$ as follows:

$$ {}^{f}\mathrm{D}_{i} = \begin{bmatrix} -\sin\left(\left(i-1\right)\varphi+\theta\right) \\ \cos\left(\left(i-1\right)\varphi+\theta\right) \\ 0 \end{bmatrix} \tag{3.5} $$

In the subsequent equation, the position vector P_{0i} of the ith wheel drive unit is considered regarding the global reference as follows:

$$ {}^{f}\mathrm{P}_{0i} = \begin{bmatrix} L\cos\left(\theta+(i-1)\varphi\right)+x \\ L\sin\left(\theta+(i-1)\varphi\right)+y \\ 0 \\ 1 \end{bmatrix} \tag{3.6} $$

The velocity of the ith omnidirectional wheel drive unit calculated by taking the derivative of the equation (3.6) as follows:

$$ {}^{f}\mathrm{v}_{i} = \frac{d}{dt}\left({}^{f}\mathrm{P}_{0i}\right) = \begin{bmatrix} -\dot{\theta}L\sin\left(\theta+(i-1)\varphi\right)+\dot{x} \\ \dot{\theta}L\cos\left(\theta+(i-1)\varphi\right)+\dot{y} \\ 0 \end{bmatrix} \tag{3.7} $$

Equation (3.7) can be reworded to be in matrix format as follows:

$$ {}^{f}\mathrm{v}_{i} = \begin{bmatrix} {}^{f}v_{ix} \\ {}^{f}v_{iy} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -L\sin\left(\theta+(i-1)\varphi\right) \\ 0 & 1 & L\cos\left(\theta+(i-1)\varphi\right) \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{3.8} $$

Equation (3.9) expresses the expanded format of equation (3.8) concerning a SAOWMR along with n omnidirectional wheels

$$ \begin{bmatrix} {}^{f}v_{1x} \\ {}^{f}v_{1y} \\ \vdots \\ {}^{f}v_{ix} \\ {}^{f}v_{iy} \\ \vdots \\ {}^{f}v_{nx} \\ {}^{f}v_{ny} \end{bmatrix} = \mathbf{A} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -L\sin(\theta) \\ 0 & 1 & L\cos(\theta) \\ \vdots & \vdots & \vdots \\ 1 & 0 & -L\sin\left(\theta+(i-1)\varphi\right) \\ 0 & 1 & L\cos\left(\theta+(i-1)\varphi\right) \\ \vdots & \vdots & \vdots \\ 1 & 0 & -L\sin\left(\theta+(n-1)\varphi\right) \\ 0 & 1 & L\cos\left(\theta+(n-1)\varphi\right) \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{3.9} $$

The relation between the center of mass velocities and the linear velocity components of the robot’s wheels in terms of the global system of reference frame expressed using the matrix **A** mentioned in equation (3.9). The inverse of the matrix **A** known as the Jacobian matrix stated in the Equation (3.10).

$$\mathbf{J} = \frac{1}{n}\begin{bmatrix} 1 & 0 & \cdots & 1 & 0 & \cdots & 1 & 0 \\ 0 & 1 & \cdots & 0 & 1 & \cdots & 0 & 1 \\ -\frac{\sin(\theta)}{L} & \frac{\cos(\theta)}{L} & \cdots & -\frac{\sin(\theta+(i-1)\varphi)}{L} & \frac{\cos(\theta+(i-1)\varphi)}{L} & \cdots & -\frac{\sin(\theta+(n-1)\varphi)}{L} & \frac{\cos(\theta+(n-1)\varphi)}{L} \end{bmatrix} \tag{3.10}$$

The relation between the velocities of the center of mass of the robot and the drive units' velocity components is resulting using the Jacobian Matrix **J** in equation (3.10) as follows:

$$\begin{cases} \dot{x} = \frac{1}{n}({}^{f}v_{1x} + {}^{f}v_{2x} + \cdots + {}^{f}v_{ix} + \cdots + {}^{f}v_{nx}) \\ \dot{y} = \frac{1}{n}({}^{f}v_{1y} + {}^{f}v_{2y} + \cdots + {}^{f}v_{iy} + \cdots + {}^{f}v_{ny}) \end{cases} \tag{3.11}$$

For the sake of finding the individual velocities assigned to the wheels of the mobile robot, the projection of the velocity specified in equation (3.7) should be identified along the unit vector D$_i$ as follows:

$$v_i = \left[{}^{f}\mathrm{v}_i\right]^{\mathrm{T}}\, {}^{f}\mathrm{D}_i = \dot{\theta}L - \dot{x}\sin\left(\theta + (i-1)\varphi\right) + \dot{y}\cos\left(\theta + (i-1)\varphi\right) \tag{3.12}$$

Equation (3.13) expresses the expanded form of the equation (3.12) concerning a SAOWMR with *n* wheels

$$\begin{bmatrix} v_1 \\ \vdots \\ v_i \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} -\sin(\theta) & \cos(\theta) & L \\ \vdots & \vdots & \vdots \\ -\sin(\theta+(i-1)\varphi) & \cos(\theta+(i-1)\varphi) & L \\ \vdots & \vdots & \vdots \\ -\sin(\theta+(n-1)\varphi) & \cos(\theta+(n-1)\varphi) & L \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{3.13}$$

In equation (3.13), the $n\times 3$ matrix characterizes the matrix of the inverse Jacobian so that it relates the center of mass velocity components of the mobile robot along with the translation velocities of the omnidirectional wheels. The kinematic system presented in equation (3.14) acquired by the Jacobian matrix in addition to the relevance of the translational velocity v_i and the angular velocity ω_i of the wheel's shaft

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \frac{r}{n} \begin{bmatrix} -2\sin(\theta) & \cdots & -2\sin(\theta+(i-1)\varphi) & \cdots & -2\sin(\theta+(n-1)\varphi) \\ 2\cos(\theta) & \cdots & 2\cos(\theta+(i-1)\varphi) & \cdots & 2\cos(\theta+(n-1)\varphi) \\ 1/L & \cdots & 1/L & \cdots & 1/L \end{bmatrix} \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_i \\ \vdots \\ \omega_n \end{bmatrix} \tag{3.14}$$

The radius of the omnidirectional special wheel stated in equation (3.14) as r. The verification of the kinematic model specified in this chapter accomplished by comparing it with that presented in [45].

3.4 Dynamic Model of the SAOWMR

In equation (3.15), with no-slip condition, the force produced by the i^{th} wheel DC motor involved to the i^{th} omnidirectional special wheel is stated as following:

$$F_i = \alpha u_i - \beta v_i \tag{3.15}$$

u_i and v_i presented in equation (3.15) are respectively the voltage implemented on the terminal input of the DC motor attached to i^{th} omnidirectional wheel drive and the translation velocity of the i^{th} omnidirectional orthogonal wheel. α and β are constants specified using the geometry of the omnidirectional wheels and the motor parameters as stated in the equations (3.16) and (3.17)

$$\alpha = \frac{k_\tau}{Rr} \tag{3.16}$$

$$\beta = \frac{k_\tau^{\,2}}{Rr^2} \tag{3.17}$$

k_τ and R are correspondingly the torque constant and the armature resistance of the DC motor omnidirectional wheel. As a result of applying Newton's laws, the stability equations of motion can be stated as follows:

$$\begin{cases} \sum_{i=1}^{n} F_i \,{}^f\mathrm{D}_i = m\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} \\ \sum_{i=1}^{n} L F_i = J\ddot{\theta} \end{cases} \tag{3.18}$$

m and J stated in equation (3.18) are correspondingly the mass and the mass moment of inertia of the mobile robot. By using the equations (3.5) and (3.15), equation (3.18) can be modified to be as follows:

$$\alpha\mathbf{Q}\begin{bmatrix} u_1 \\ \vdots \\ u_i \\ \vdots \\ u_n \end{bmatrix} - \beta\mathbf{Q}\begin{bmatrix} v_1 \\ \vdots \\ v_i \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} m\ddot{x} \\ m\ddot{y} \\ (J/L)\ddot{\theta} \end{bmatrix} \tag{3.19}$$

With

$$\mathbf{Q} = \begin{bmatrix} -\sin(\theta) & \cdots & -\sin((i-1)\varphi+\theta) & \cdots & -\sin((n-1)\varphi+\theta) \\ \cos(\theta) & \cdots & \cos((i-1)\varphi+\theta) & \cdots & \cos((n-1)\varphi+\theta) \\ 1 & \cdots & 1 & \cdots & 1 \end{bmatrix} \tag{3.20}$$

By substituting the equation (3.13) into the equation (3.19) yields

$$\mathbf{Q}\begin{bmatrix} u_1 \\ \vdots \\ u_i \\ \vdots \\ u_n \end{bmatrix} = \frac{1}{\alpha}\begin{bmatrix} m\ddot{x} \\ m\ddot{y} \\ (J/L)\ddot{\theta} \end{bmatrix} + \frac{\beta}{\alpha}\begin{bmatrix} n\dot{x}/2 \\ n\dot{y}/2 \\ nL\dot{\theta} \end{bmatrix} \tag{3.21}$$

By replacing the matrix **Q** given in equation (3.20) into the equation (3.21), the stability equations of motions of the SAOWMR are characterized as following:

$$\begin{aligned}
&\sum_{i=1}^{n} -\sin\left(\left(i-1\right)\varphi+\theta\right)u_i = \frac{n\beta}{2\alpha}\dot{x}+\frac{m}{\alpha}\ddot{x} \\
&\sum_{i=1}^{n} \cos\left(\left(i-1\right)\varphi+\theta\right)u_i = \frac{n\beta}{2\alpha}\dot{y}+\frac{m}{\alpha}\ddot{y} \\
&\sum_{i=1}^{n} u_i = \frac{n\beta L}{\alpha}\dot{\theta}+\frac{J}{\alpha L}\ddot{\theta}
\end{aligned} \tag{3.22}$$

Through presenting the vector of state Z=[$x \quad y \quad \theta$]^{T} and the vector of the control input U=[$u_1 \quad \cdots \quad u_i \quad \cdots \quad u_n$]^{T}, equation (3.22) can be specified in compact form as following:

$$\mathbf{M}\ddot{\mathrm{Z}}+\mathbf{A}\dot{\mathrm{Z}}=\mathbf{Q}\mathrm{U} \tag{3.23}$$

In Equation (3.23), the matrices M and A are stated in (3.24) as follows:

$$\mathbf{M}=\begin{bmatrix} \frac{m}{\alpha} & 0 & 0 \\ 0 & \frac{m}{\alpha} & 0 \\ 0 & 0 & \frac{J}{\alpha L} \end{bmatrix}, \mathbf{A}=\begin{bmatrix} \frac{n\beta}{2\alpha} & 0 & 0 \\ 0 & \frac{n\beta}{2\alpha} & 0 \\ 0 & 0 & \frac{n\beta L}{\alpha} \end{bmatrix} \tag{3.24}$$

The verification of the dynamic model detailed in this chapter done by comparing it with that presented in [64] and in [44] regarding the OWMR that utilizes three omnidirectional orthogonal special wheels and four omnidirectional wheels respectively.

Chapter 4

OPTIMAL TRAJECTORY GENERATION METHODOLOGY FOR THE SAOWMR

Finding the optimal trajectory of the mobile robot considered as the process of search for the ideal trajectory of the robotic system that enhances the robot's properties performance, decreases the resources consumption or traveling time where the system of constraints remains preserved. In terms of the mathematic formulation, trajectory optimization problem can be model as optimal control problem. Some alternate formulations generally solved using metaheuristic algorithms. Optimal control methods handles the problem of finding the control for a specified system so that some optimality criteria are achieved. This chapter discuses a new computationally efficient method on the basis of the compact and complete model of the SAOWMR so that enhancements in the optimization of the trajectory planning can be attained.

The optimal trajectory planning problem concerning the SAOWMR that utilizes *n*-omnidirectional orthogonal wheels formulated in this chapter as time-optimal control problem. The proposed procedure of solving this optimization problem specified by approximating the vector of the control input U applied on the DC motors terminals associated with the robot's omnidirectional orthogonal wheels so that the mobile robot able to maneuver from a primary to final positions of boundary conditions with the optimality of total required energy E and maneuvering time t_f while the system of constraints remains preserved.

4.1 Control Problem Formulation

In this chapter, the trajectory planning optimization problem expressed as a time-optimal control problem. The solution of this problem done by approximating the vector of the control input U applied to the motors' terminals relevant to the omnidirectional wheels with the optimality of total required energy E and the maneuvering time t_f. Subsequently, the robot be able to maneuver from a primary position to an ending position of boundary conditions.

The time-optimal control problem can be modeled as following. Assuming that the dynamic equation of motion of the SAOWMR that utilizes *n*-omnidirectional special wheels detailed in the previous chapter as follows:

$$\mathbf{M}\ddot{Z}+\mathbf{A}\dot{Z}=\mathbf{Q}U \tag{4.1}$$

This control system is subjected to four boundary conditions due to the motion of the mobile robot from the primary to the ending configurations.

$$Z(t=0)=Z_o=\begin{bmatrix}x_o\\ y_o\\ \theta_o\end{bmatrix},\ \dot{Z}(t=0)=\dot{Z}_o=\begin{bmatrix}\dot{x}_o\\ \dot{y}_o\\ \dot{\theta}_o\end{bmatrix},\ Z(t=t_f)=Z_f=\begin{bmatrix}x_f\\ y_f\\ \theta_f\end{bmatrix},\ \dot{Z}(t=t_f)=\dot{Z}_f=\begin{bmatrix}\dot{x}_f\\ \dot{y}_f\\ \dot{\theta}_f\end{bmatrix} \tag{4.2}$$

System of constraints can arise as a result of several causes. However, two constraints possess a greater importance in the robotics implementations. Maximum acceleration amplitude of the center of mass constraint and input voltage saturation constraint placed on the of the omnidirectional wheels' DC motors given respectively in equations (4.3) and (4.4) as follows

$$\ddot{x}^2+\ddot{y}^2\le\alpha_{\max}^2 \tag{4.3}$$

$$u_o\le u_{\max} \tag{4.4}$$

Moreover, $u_{\max}$ and $\alpha_{\max}$ are respectively limitation applied on the wheel's DC motor and the acceleration amplitude. Furthermore, u_o is the maximum voltage applied on the input of omnidirectional wheels DC motors terminals considered by utilizing the subsequent equation:

$$u_o = \max\left\{\left|u_{o1}\right|, \ \cdots \ ,\left|u_{oi}\right|, \ \cdots \ ,\left|u_{on}\right|\right\} \tag{4.5}$$

u_{oi} given in equation (4.5) expresses the maximum input voltage of the input terminal of the ith DC motor corresponding to the ith omnidirectional special wheel.

The following optimization problem expressed in the equation (4.6) states the previously explained optimal control problem concerning SAOWMR that utilizes n-omnidirectional wheels maneuvering from a primary position to a final position of boundary conditions.

$$\begin{aligned}&\text{Min} \quad \mathrm{C}\left(t_{\mathrm{f}}\right) = t_{\mathrm{f}} + \gamma \mathrm{E} \\ &\text{Subject to:} \begin{cases} \mathbf{M}\ddot{\mathrm{Z}} + \mathbf{A}\dot{\mathrm{Z}} = \mathbf{Q}\mathrm{U} \\ \mathrm{Z}\left(t=0\right) = \mathrm{Z}_{\mathrm{o}}, \ \dot{\mathrm{Z}}\left(t=0\right) = \dot{\mathrm{Z}}_{\mathrm{o}}, \ \mathrm{Z}\left(t=t_{\mathrm{f}}\right) = \mathrm{Z}_{\mathrm{f}}, \ \dot{\mathrm{Z}}\left(t=t_{\mathrm{f}}\right) = \dot{\mathrm{Z}}_{\mathrm{f}} \\ \ddot{x}^2 + \ddot{y}^2 \le \alpha_{\max}^2 \\ u_o \le u_{\max} \end{cases}\end{aligned} \tag{4.6}$$

γ symbolized in equation (4.6) as a positive weight clarifying the trade-off between the total energy required E and the robot maneuvering time t_{f}.

4.2 Optimal Trajectory Generation Methodology

This section focuses on solving the problem of the optimal trajectory generation stated in the equation (4.6) so that the SAOWMR with n-omnidirectional traveling from a preliminary position to a final position of the boundary conditions.

4.2.1 Reference Trajectory

It is proposed to use the third-order polynomial in order to characterize the reference trajectory that fulfills the requirement of the specified four boundary conditions corresponding to the Equation of motion of the SAOWMR stated in equations (4.1) and (4.2). The third-order polynomial trajectory corresponding to the boundary conditions given in the following equation:

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} t^3 + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} t^2 + \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} t + \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} \tag{4.7}$$

The polynomial coefficients stated in equation (4.7) are calculated based on the boundary conditions in equation (4.2) to be as follows:

$$\begin{cases} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \dfrac{1}{t_{\mathrm{f}}^{2}} \left(-\dfrac{2}{t_{\mathrm{f}}} \left(\begin{bmatrix} x_{\mathrm{f}} \\ y_{\mathrm{f}} \\ \theta_{\mathrm{f}} \end{bmatrix} - \begin{bmatrix} x_{\mathrm{o}} \\ y_{\mathrm{o}} \\ \theta_{\mathrm{o}} \end{bmatrix} \right) + \begin{bmatrix} \dot{x}_{\mathrm{o}} \\ \dot{y}_{\mathrm{o}} \\ \dot{\theta}_{\mathrm{o}} \end{bmatrix} + \begin{bmatrix} \dot{x}_{\mathrm{f}} \\ \dot{y}_{\mathrm{f}} \\ \dot{\theta}_{\mathrm{f}} \end{bmatrix} \right) \\ \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \dfrac{1}{t_{\mathrm{f}}} \left(\dfrac{3}{t_{\mathrm{f}}} \left(\begin{bmatrix} x_{\mathrm{f}} \\ y_{\mathrm{f}} \\ \theta_{\mathrm{f}} \end{bmatrix} - \begin{bmatrix} x_{\mathrm{o}} \\ y_{\mathrm{o}} \\ \theta_{\mathrm{o}} \end{bmatrix} \right) - 2 \begin{bmatrix} \dot{x}_{\mathrm{o}} \\ \dot{y}_{\mathrm{o}} \\ \dot{\theta}_{\mathrm{o}} \end{bmatrix} - \begin{bmatrix} \dot{x}_{\mathrm{f}} \\ \dot{y}_{\mathrm{f}} \\ \dot{\theta}_{\mathrm{f}} \end{bmatrix} \right) \\ \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} \dot{x}_{\mathrm{o}} \\ \dot{y}_{\mathrm{o}} \\ \dot{\theta}_{\mathrm{o}} \end{bmatrix}, \quad \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} x_{\mathrm{o}} \\ y_{\mathrm{o}} \\ \theta_{\mathrm{o}} \end{bmatrix} \end{cases} \tag{4.8}$$

4.2.2 Dynamic Analysis of the SAOWMR Motion Planning

The equation motion specified in equation (4.1) corresponding to the SAOWMR utilizes *n* omnidirectional wheels' will be given in the following equality based on third-order polynomial trajectory model stated in equation (4.7).

$$\mathbf{M}\ddot{\mathrm{Z}} + \mathbf{A}\dot{\mathrm{Z}} = \mathbf{K} \begin{bmatrix} t^2 \\ t \\ 1 \end{bmatrix} = \mathbf{Q}\mathrm{U} \qquad t \in \begin{bmatrix} 0 & t_{\mathrm{f}} \end{bmatrix} \tag{4.9}$$

The matrix **K** specified in equation (4.9) given in equation (4.10) as follows

$$\mathbf{K} = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} = \begin{bmatrix} \left(\dfrac{3n\beta a_1}{2\alpha} \right) & \left(\dfrac{6ma_1 + n\beta b_1}{\alpha} \right) & \left(\dfrac{4mb_1 + n\beta c_1}{2\alpha} \right) \\ \left(\dfrac{3n\beta a_2}{2\alpha} \right) & \left(\dfrac{6ma_2 + n\beta b_2}{\alpha} \right) & \left(\dfrac{4mb_2 + n\beta c_2}{2\alpha} \right) \\ \left(\dfrac{3n\beta La_3}{\alpha} \right) & \left(\dfrac{6Ja_3 + 2n\beta L^2 b_3}{\alpha L} \right) & \left(\dfrac{2Jb_3 + n\beta L^2 c_3}{\alpha L} \right) \end{bmatrix} \tag{4.10}$$

Supposing that the SAOWMR with n >2 omnidirectional wheels, it is possible to isolate the input voltage vector U applied on the DC motors terminals utilizing the equation (4.9) as follows

$$\mathrm{U}=\begin{bmatrix}u_1\\ \vdots\\ u_i\\ \vdots\\ u_n\end{bmatrix}=\begin{bmatrix}f_{11} & f_{12} & f_{13}\\ \vdots & \vdots & \vdots\\ f_{i1} & f_{i2} & f_{i3}\\ \vdots & \vdots & \vdots\\ f_{n1} & f_{n2} & f_{n3}\end{bmatrix}\begin{bmatrix}t^2\\ t\\ 1\end{bmatrix}\quad t\in\left[0\quad t_{\mathrm{f}}\right] \tag{4.11}$$

The matrix elements of the equation (4.11) will be given as follows:

$$\begin{cases}f_{11}=\frac{1}{n}\left(-2k_{11}\sin\left(\theta\right)+2k_{21}\cos\left(\theta\right)+k_{31}\right)\\ f_{12}=\frac{1}{n}\left(-2k_{12}\sin\left(\theta\right)+2k_{22}\cos\left(\theta\right)+k_{32}\right)\\ f_{13}=\frac{1}{n}\left(-2k_{13}\sin\left(\theta\right)+2k_{23}\cos\left(\theta\right)+k_{33}\right)\\ f_{i1}=\frac{1}{n}\left(-2k_{11}\sin\left(\left(i-1\right)\varphi+\theta\right)+2k_{21}\cos\left(\left(i-1\right)\varphi+\theta\right)+k_{31}\right)\\ f_{i2}=\frac{1}{n}\left(-2k_{12}\sin\left(\left(i-1\right)\varphi+\theta\right)+2k_{22}\cos\left(\left(i-1\right)\varphi+\theta\right)+k_{32}\right)\\ f_{i3}=\frac{1}{n}\left(-2k_{13}\sin\left(\left(i-1\right)\varphi+\theta\right)+2k_{23}\cos\left(\left(i-1\right)\varphi+\theta\right)+k_{33}\right)\\ f_{n1}=\frac{1}{n}\left(-2k_{11}\sin\left(\left(n-1\right)\varphi+\theta\right)+2k_{21}\cos\left(\left(n-1\right)\varphi+\theta\right)+k_{31}\right)\\ f_{n2}=\frac{1}{n}\left(-2k_{12}\sin\left(\left(n-1\right)\varphi+\theta\right)+2k_{22}\cos\left(\left(n-1\right)\varphi+\theta\right)+k_{32}\right)\\ f_{n3}=\frac{1}{n}\left(-2k_{13}\sin\left(\left(n-1\right)\varphi+\theta\right)+2k_{23}\cos\left(\left(n-1\right)\varphi+\theta\right)+k_{33}\right)\end{cases} \tag{4.12}$$

The complete energy E concerning SAOWMR maneuvering from an initial configurations as $t=0$ to the ending configurations when $t=t_{\mathrm{f}}$ is stated using the subsequent equation

$$\mathrm{E}=\int_{t=0}^{t_{\mathrm{f}}}p_1\,dt+\cdots+\int_{t=0}^{t_{\mathrm{f}}}p_i\,dt+\cdots+\int_{t=0}^{t_{\mathrm{f}}}p_n\,dt \tag{4.13}$$

The power p_i produced by the DC motor of the i^{th} omnidirectional wheel will be given in equation (4.14) in terms of the parameters α and β. The derive of the equation (4.14) detailed in Appendix A.

$$p_i = \left| \frac{r}{k_\tau} \left(\alpha u_i^2 - \beta v_i u_i \right) \right| \tag{4.14}$$

4.2.3 Dealing with the Constraints System

In this research, a system of constraints raised by two constraints. Limitation on the acceleration amplitude of the center of mass of the SAOWMR and saturation constraint placed on the DC motor' input voltage of the omnidirectional wheels.

4.2.3.1 Acceleration Amplitude Constraint

The method of solution adopted in this section is to determine the numeric value of the complete steering time t_f so that the acceleration amplitude of the mass center of the mobile robot $\ddot{x}^2 + \ddot{y}^2$ critically reaches the limit α_max. Subsequently, a new domain of the total maneuvering time t_f generated in such a way that the acceleration constraint remains maintained.

Subsequently, the third-order polynomial trajectory model given in the equation (4.7) is utilized to produce the trajectory of the SAOWMR with n omnidirectional wheels, the acceleration of the mass center of the mobile robot is linear function of time. Moreover, the acceleration restriction given in relation (4.3) is stated based on the polynomial coefficients as follows:

$$\delta_1 t^2 + \delta_2 t + \delta_3 \le \alpha_\mathrm{max}^2 \qquad t \in \left[0 \quad t_\mathrm{f}\right]$$
$$\begin{cases} \delta_1 = 36\left(a_1^2 + a_2^2\right) \\ \delta_2 = 24\left(a_1 b_1 + a_2 b_2\right) \\ \delta_3 = 4\left(b_1^2 + b_2^2\right) \end{cases} \tag{4.15}$$

In equation (4.15), the quadratic function of time is determined as t changeable from 0 to the t_f. As the parabola of the quadratic function has an upward shape, δ_1 is positive, and the domain of the quadratic function is positive entity. Subsequently, just two points in the domain of the quadratic function identified by the boundary of the domain when $t = 0$ and $t = t_\mathrm{f}$ must be tested pointing to identify the maximum point of the function. To locate the critical value of the final steering time t_f regarding $\ddot{x}^2 + \ddot{y}^2$ getting the bound $\alpha^{\mathbf{2}}_{\max}$, the subsequent two equalities must be resolved separately:

$$\begin{aligned}
&\lambda_4\, t^4_{\mathrm{f}_{\mathrm{a}_1}} + \lambda_2 t^2_{\mathrm{f}_{\mathrm{a}_1}} + \lambda_1\, t_{\mathrm{f}_{\mathrm{a}_1}} + \lambda_\mathrm{o} = 0 \quad t_{\mathrm{f}_{\mathrm{a}_1}} \in \mathbb{R}^+\\
&\varsigma_4\, t^4_{\mathrm{f}_{\mathrm{a}_2}} + \varsigma_2\, t^2_{\mathrm{f}_{\mathrm{a}_2}} + \varsigma_1\, t_{\mathrm{f}_{\mathrm{a}_2}} + \varsigma_\mathrm{o} = 0 \quad t_{\mathrm{f}_{\mathrm{a}_2}} \in \mathbb{R}^+\\
&\begin{cases}
\lambda_4 = \varsigma_4 = \left(\dfrac{\alpha^2_{\max}}{4}\right)\\
\lambda_2 = -\left(\left(2\dot{x}_\mathrm{o} + \dot{x}_\mathrm{f}\right)^2 + \left(2\dot{y}_\mathrm{o} + \dot{y}_\mathrm{f}\right)^2\right)\\
\varsigma_2 = -\left(\left(\dot{x}_\mathrm{o}{}^2 - 2\dot{x}_\mathrm{f}{}^2\right) + \left(\dot{y}_\mathrm{o}{}^2 - 2\dot{y}_\mathrm{f}{}^2\right)\right)\\
\lambda_1 = 6\left(\left(x_\mathrm{f} - x_\mathrm{o}\right)\left(2\dot{x}_\mathrm{o} + \dot{x}_\mathrm{f}\right) + \left(y_\mathrm{f} - y_\mathrm{o}\right)\left(2\dot{y}_\mathrm{o} + \dot{y}_\mathrm{f}\right)\right)\\
\varsigma_1 = 6\left(\left(x_\mathrm{f} - x_\mathrm{o}\right)\left(\dot{x}_\mathrm{o} + 2\dot{x}_\mathrm{f}\right) + \left(y_\mathrm{f} - y_\mathrm{o}\right)\left(\dot{y}_\mathrm{o} + 2\dot{y}_\mathrm{f}\right)\right)\\
\lambda_\mathrm{o} = \varsigma_\mathrm{o} = -9\left(\left(x_\mathrm{f} - x_\mathrm{o}\right)^2 + \left(y_\mathrm{f} - y_\mathrm{o}\right)^2\right)
\end{cases}
\end{aligned} \tag{4.16}$$

Since the final steering time t_f is a real positive quantity, the equalities stated in equation (4.16) must be resolved to give one satisfactory solution for each equation. The satisfactory solution $t_{\mathrm{f}_\mathrm{a}}$ regarding the term $\ddot{x}^2 + \ddot{y}^2$ analytically reach $\alpha^{\mathbf{2}}_{\max}$ is specified as follows:

$$t_{\mathrm{f}_\mathrm{a}} = \max\left\{t_{\mathrm{f}_{\mathrm{a}_1}}, t_{\mathrm{f}_{\mathrm{a}_2}}\right\} \tag{4.17}$$

In order to ensure that the acceleration constraint remains maintained, the total maneuvering time t_f must be selected according to the following criteria

$$t_\mathrm{f} \geq t_{\mathrm{f}_\mathrm{a}} \tag{4.18}$$

Now, it is possible to replace the acceleration constraint in the relation (4.3) by the relation (4.18)

4.2.3.2 Control Saturation Constraint

The proposed method in this section to handle the control saturation constraint summarized by finding the value of the total steering time $\mathrm{t}_{\mathrm{f}_u}$ that the maximum input voltage of the omnidirectional wheels' DC motors u_{o} given in equation (4.5) critically reaches the limit $u_{\max}$. Subsequently, a new domain of the total steering time t_{f} generated so that the control saturation constraint remains maintained.

It is obvious that the nature of the input voltage function u_i given in equation (4.11) is nonconvex and nonlinear in common in the domain contained by zero and t_{f}. The traditional approaches of finding the highest value of a function will be unsuccessful in most of the cases to detect the maximum value of the input voltage function u_i and will fail to find the optimal solution $u_{\mathrm{o}i}$ and frequently held local maxima or minima. For this purpose, we suggest to utilize a numerical technique capable of approximating actively the global maximum solution $u_{\mathrm{o}i}$, such as the Simulated Annealing algorithm (SA).

Simulated Annealing algorithm is an optimization solver suggested to solve unconstrained and bound-constrained optimization problems. This method mimics the process of heating the solid material then gradually relief the temperature in order to reduce the defects. The simulated annealing algorithm generates a new point randomly based on probability distribution criteria with a scale relative to the temperature. Supposing that a minimization problem holds, the method tends to accept all the new solutions that lowering the fitness function. However, the method under particular probability accept solutions that raise the fitness function. Following this method gives a chance to avoid being stuck in local minima and globally explore for global solution.

The procedure tends to methodically decrease the temperature as the algorithm search for the optimal solution. As the temperature reduces, the method decreases the probability of accepting solutions that raise the fitness function so that the algorithm converges to the minimum solution.

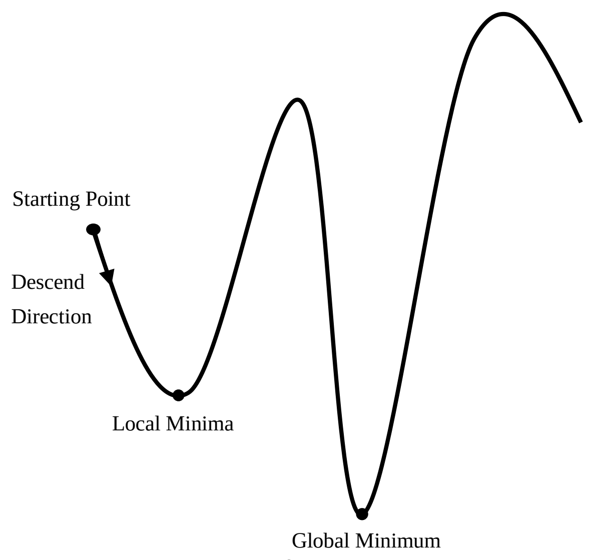


Figure 4.1: Local and Global Minimum of the Non-Linear and Non-Convex Function

Two central features describe SA algorithm. The first is the acceptance of the search step, constantly accept the solution those possess descending direction. Accept the solutions those having ascending direction only if it passes a random test. The second is the cooling schedule that the control of the acceptance frequency of the ascending steps takes place.

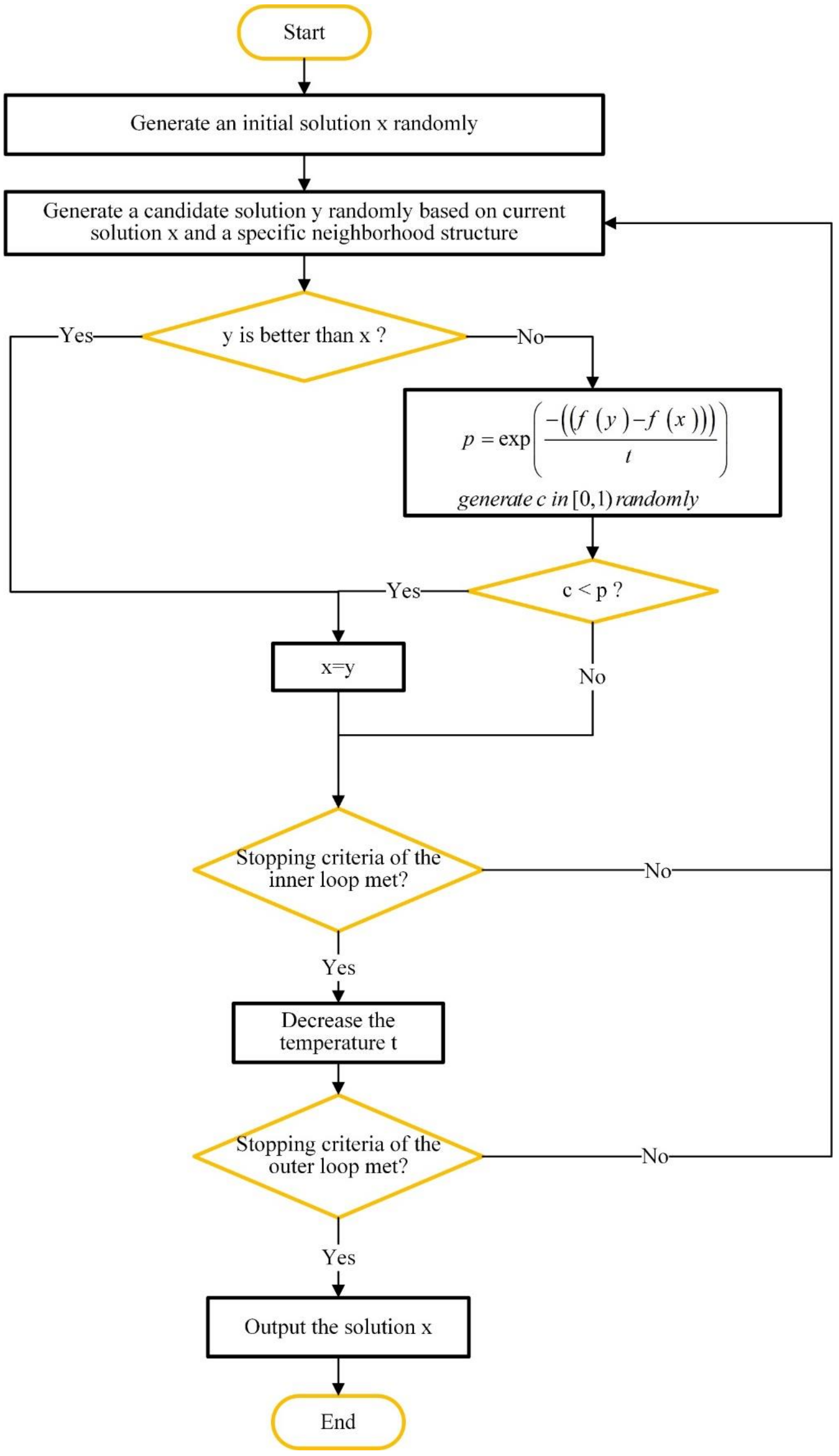

Start
Generate an initial solution x randomly
Generate a candidate solution y randomly based on current solution x and a specific neighborhood structure
y is better than x ?
Yes
No
$p = \exp\left(\frac{-((f(y)-f(x)))}{t}\right)$
generate c in [0,1) randomly
c < p ?
Yes
No
x=y
Stopping criteria of the inner loop met?
No
Yes
Decrease the temperature t
Stopping criteria of the outer loop met?
No
Yes
Output the solution x
End

Figure 4.2: Simulated Annealing Algorithm Flowchart

Regarding a specific total maneuvering time t_f, the maximum value of the input voltages u_o placed on the terminal inputs of the omnidirectional wheels DC motors is found by utilizing the equation (4.5).

Subsequently, the final maneuvering time t_{f_u} regarding the maximum value of input voltage u_o meeting the maximum bound u_{max} able to be found by applying a simple root-finding bisection method. It is exceedingly suggested to utilize equation (4.19) as a starting solution of the search in order to converge to the solution in minimal calculation time. More information concerning the derive of the equation (4.19) given in the Appendix B.

$$t_{f_o} = \frac{3\beta}{n\alpha u_{max}} \left(\sqrt{\left(x_f - x_o\right)^2 + \left(y_f - y_o\right)^2} + \left(\theta_f - \theta_o\right) L \right) \tag{4.19}$$

In order to ensure that the control input saturation constraint remains maintained, the total maneuvering time t_f must be selected according to the following criteria

$$t_f \geq t_{f_u} \tag{4.20}$$

Now, it is possible to replace the control input saturation constraint in the relation (4.4) by the relation (4.20).

Then, the lower bound $t_{f_{min}}$concerning the free variable t_f so that the system of constraint remains preserved can be determined based on finding t_{f_a}that the acceleration $\ddot{x}^2$+ $\ddot{y}^2$ critically meet the limit $\alpha^{\mathbf{2}}_{max}$ and t_{f_u}that the maximum input voltage u_o meet the limit u_{max} by utilizing the subsequent equation:

$$t_{f_{min}} = \max\left\{t_{f_a}, t_{f_u}\right\} \tag{4.21}$$

4.2.4 Optimization Process

The trajectory optimization approach suggested in this thesis to find solution for the optimization problem stated in equation (4.6) able to be considered as selecting the trajectory identified by choosing t_f from a set of an acceptable domain constrained by minimum and maximum bound on t_f. Hence, the optimization problem specified in equation (4.6) is decreased to the search in one dimension for the optimum value of the final steering time t_f so that it minimizes the cost function C as following:

$$\mathrm{Min} \quad \mathrm{C}(t_\mathrm{f}) = t_\mathrm{f} + \gamma \mathrm{E} \qquad t_\mathrm{f} \in \left[t_{\mathrm{f}_{\min}} \quad t_{\mathrm{f}_{\max}} \right] \tag{4.22}$$

The solution of the optimization problem stated in equation (4.22) able to be done by specifying the feasible domain for the total steering time t_f that ensures the system of constraints. A lower bound $t_{\mathrm{f}_\mathrm{min}}$ concerning the free variable t_f ensures that the system of the constraints stated in equation (4.6) remains sustained can be determined based on finding $t_{\mathrm{f}_\mathrm{a}}$that the acceleration $\ddot{x}^2+\ddot{y}^2$ critically meet the limit $\alpha_\mathrm{max}^{\mathbf{2}}$ and $t_{\mathrm{f}_\mathrm{u}}$that the maximum input voltage u_o meet the bound u_max and then by applying the equation (4.21). An upper bound $t_{\mathrm{f}_\mathrm{max}}$ could be selected based on experience (a proper OWMR proceeding movements on a 3 m^2 playing field could accomplish most of the transpositions in just a few seconds). To solve for a suitable solution concerning the optimality of the total maneuvering time t_f that minimizes the cost C in equation (4.22), a simple golden-section search technique could be straightforwardly applied since the energy is a monotonic function regarding the free variable t_f changing within the lowest and highest limits. More specifics concerning the optimization process given in chapter 5.

4.2.5 Kinematic Saturation

The fact that the SAOWMR can correctly reacts to the orders requires that the velocities of the mass center of gravity being contained by their ranges depending on the saturation of the actuators. To get hold of the kinematic saturation, the voltage profile u_i applied to the terminal of the ith omnidirectional wheel DC motor must not violate the bound u_max, and the boundary conditions of the third order polynomial trajectory given in equation (4.2) must fulfil the following two equations:

$$\left|\begin{array}{l}\left(\left(2c_3-c_5\right)\sin\left(\theta_o+\left(i-1\right)\varphi\right)\right)\dot{x}_o+\left(\left(c_5-2c_3\right)\cos\left(\theta_o+\left(i-1\right)\varphi\right)\right)\dot{y}_o+\left(\left(Lc_5-2c_4\right)\right)\dot{\theta}_o+\\ \left(c_3\sin\left(\theta_o+\left(i-1\right)\varphi\right)\right)\dot{x}_f+\left(-c_3\cos\left(\theta_o+\left(i-1\right)\varphi\right)\right)\dot{y}_f+\left(-c_4\right)\dot{\theta}_f+\\ \left(-c_1\sin\left(\theta_o\right)\left(x_f-x_o\right)+c_1\cos\left(\theta_o\right)\left(y_f-y_o\right)+c_2\left(\theta_f-\theta_o\right)\right)\end{array}\right|\le u_{\max} \tag{4.23}$$

$$\left|\begin{array}{l}\left(-c_3\sin\left(\theta_f+\left(i-1\right)\varphi\right)\right)\dot{x}_o+\left(c_3\cos\left(\theta_f+\left(i-1\right)\varphi\right)\right)\dot{y}_o+\left(c_4\right)\dot{\theta}_o+\\ \left(\left(-2c_3-c_5\right)\sin\left(\theta_f+\left(i-1\right)\varphi\right)\right)\dot{x}_f+\left(\left(2c_3+c_5\right)\cos\left(\theta_f+\left(i-1\right)\varphi\right)\right)\dot{y}_f+\left(Lc_5+2c_4\right)\dot{\theta}_f+\\ \left(c_1\sin\left(\theta_f+\left(i-1\right)\varphi\right)\left(x_f-x_o\right)-c_1\cos\left(\theta_f+\left(i-1\right)\varphi\right)\left(y_f-y_o\right)-c_2\left(\theta_f-\theta_o\right)\right)\end{array}\right|\le u_{\max} \tag{4.24}$$

With

$$c_1=\frac{12m}{\alpha n t_f^2}\ ,\ c_2=\frac{6J}{L\alpha n t_f^2}\ ,\ c_3=\frac{4m}{\alpha n t_f}\ ,\ c_4=\frac{2J}{\alpha n t_f L}\ ,\ c_5=\frac{\beta}{\alpha} \tag{4.25}$$

4.3 Special Case (SAOWMR with Two Orthogonal Wheels)

In this case, the SAOWMR illustrated in Figure 3.5 is equipped with just two omnidirectional wheels correspondingly organized at 180° on the border of the mobile robot. The equation of motion stated in Equation (3.23) be able to be modified concerning *n*=2 and φ=180° to be in the following form:

$$\begin{bmatrix}-\sin\left(\theta\right) & \sin\left(\theta\right)\\ \cos\left(\theta\right) & -\cos\left(\theta\right)\\ 1 & 1\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix}=\begin{bmatrix}\frac{m}{\alpha}\ddot{x}+\frac{\beta}{\alpha}\dot{x}\\ \frac{m}{\alpha}\ddot{y}+\frac{\beta}{\alpha}\dot{y}\\ \frac{J}{\alpha L}\ddot{\theta}+\frac{2\beta L}{\alpha}\dot{\theta}\end{bmatrix} \tag{4.26}$$

It is obvious that the system stated in equation (4.26) is an overdetermined. Subsequently, the process of estimating an accepted solution concerning the control input vector so that the mobile robot concurrently achieves translational and rotational is not in general prospective.

In this section, it is suggested to separate the congregation movements into rotational motions and translational motion to resolve the optimal control problem given in equation (4.6) so that the mobile robot is capable to transport with the optimal trajectory from the initial position to the final position. It is supposed that the equation of motion of the mobile robot stated in equation (4.26) is restricted by the boundary conditions specified in equation (4.27):

$$Z(t=0)=\begin{bmatrix} x_o \\ y_o \\ \theta_o \end{bmatrix},\ Z(t=t_f)=\begin{bmatrix} x_f \\ y_f \\ \theta_f \end{bmatrix} \tag{4.27}$$

The approach suggested to find a solution regarding the problem of the optimal control stated in equation (4.6) so that the mobile robot be able to fulfills the terminal conditions in equation (4.27) is established on accomplishment of three sequential movements: First, the mobile robot performs a rotational motion around a hub perpendicular to the surface of the robot and corresponding to the robot's center of mass so that the robot modifies its orientation from θ_o to θ_v. At that instant, translational movements proceeds so that the mobile robot varies its position from (x_o, y_o) to (x_f, y_f). Lastly, additional rotational movement carries out so that the mobile robot modifies its orientation from θ_v to θ_f. The midway mobile robot orientation θ_v is specified in equation (4.28).

$$\theta_v = \tan^{-1}\left(-\frac{x_f - x_o}{y_f - y_o}\right) \tag{4.28}$$

4.3.1 Rotational Motion

It is supposed that the mobile robot proceeding rotational movements modifies its coordination from θ_o to θ_v. The third-order polynomials able to be utilized in order to express the movement as follows:

$$\theta = -\frac{2}{t_f^{\,3}}(\theta_v - \theta_o)t^3 + \frac{3}{t_f^{\,2}}(\theta_v - \theta_o)t^2 + \theta_o \tag{4.29}$$

The equation system of motion based on the equations (4.26) and (4.29) concerning two orthogonal wheels acting synchronized and analogous movements can be stated as follows:

$$u_1 = u_2 = -\frac{6\beta L}{\alpha t_f^{\,3}}\left(\theta_v - \theta_o\right)t^2 + \left(\frac{6\beta L^2 t_f - 6J}{\alpha t_f^{\,3} L}\right)\left(\theta_v - \theta_o\right)t + \frac{3J}{t_f^{\,2}\alpha L}\left(\theta_v - \theta_o\right) \qquad (4.30)$$

The extreme input voltage mentioned as u_o concerning the terminal inputs of the DC motors is considered merely using the subsequent formula:

$$u_o = \left|\left(\frac{\left(3\beta^2 L^4 t_f^{\,2} + 3J^2\right)}{2\alpha\beta L^3 t_f^{\,3}}\right)\left(\theta_v - \theta_o\right)\right| \qquad (4.31)$$

Subsequently, the minimum bound $t_{f_{\min}}$ concerning the free variable t_f regarding the extreme input voltage u_o attaining the extreme limit $u_{\max}$ is calculated by resolving the subsequent equation:

$$\left(\frac{u_{\max}}{\left(\theta_f - \theta_o\right)}2\alpha\beta L^3\right)t_{f_{\min}}^3 - \left(3\beta^2 L^4\right)t_{f_{\min}}^2 - 3J^2 = 0 \quad t_{f_{\min}} \in \mathbb{R}^+ \qquad (4.32)$$

Supposing that the total steering time $t_{f_{\min}}$ is a positive and real number. Subsequently, the equation (4.22) has to be resolved to give just one suitable solution.

4.3.2 Translational Motion

It is supposed that the mobile robot carrying out translational movements modifies its location from (x_o, y_o) to (x_f, y_f). In this case, the angle coordination of the mobile robot fulfills the equation (4.23). To represent the motion, the third-order polynomial able to be utilized to express the movement as follows:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -\frac{2}{t_f^{\,3}}\left(x_f - x_o\right) \\ -\frac{2}{t_f^{\,3}}\left(y_f - y_o\right) \end{bmatrix} t^3 + \begin{bmatrix} \frac{3}{t_f^{\,2}}\left(x_f - x_o\right) \\ \frac{3}{t_f^{\,2}}\left(y_f - y_o\right) \end{bmatrix} t^2 + \begin{bmatrix} x_o \\ y_o \end{bmatrix} \qquad (4.33)$$

supposing that two omnidirectional wheels carrying out synchronized motion concerning the translational motion of the mobile robot. In this case, the equation of motion is derived on the basis of the Equations (4.26) and (4.33) to be as follows

$$u_1 = -u_2 = \left(\frac{3\beta\eta}{t_f{}^3\alpha}\right)t^2 - \left(\frac{(3\beta t_\mathrm{f} - 6m)\eta}{\alpha t_\mathrm{f}{}^3}\right)t - \left(\frac{3m\eta}{\alpha t_\mathrm{f}{}^2}\right) \tag{4.34}$$

With

$$\eta = (x_\mathrm{f} - x_\mathrm{o})\sin(\theta_\mathrm{v}) - (y_\mathrm{f} - y_\mathrm{o})\cos(\theta_\mathrm{v}) \tag{4.35}$$

In Equation (4.34), it is shown that wheel DC motors supplied by power sources with different polarities. The extreme input voltage u_o concerning the terminal inputs of the wheels' DC motors can be derived just by utilizing the following equation:

$$u_\mathrm{o} = \left|\left(\frac{-9\beta^2 t_\mathrm{f}{}^2 - 36m^2}{12\beta\alpha t_\mathrm{f}{}^3}\right)\eta\right| \tag{4.36}$$

The total steering time $t_{\mathrm{f_u}}$regarding the extreme input voltage u_o getting the highest bound u_max able to be calculated by resolving the following equality so that one suitable solution only acquired:

$$\left(12\frac{u_\mathrm{max}}{\eta}\beta\alpha\right)t_{\mathrm{f_u}}^3 + (9\beta^2)t_{\mathrm{f_u}}^2 + (36m^2) = 0 \qquad t_{\mathrm{f_u}} \in \mathbb{R}^+ \tag{4.37}$$

To manage the total steering time $t_{\mathrm{f_a}}$ regarding $\ddot{x}^2 + \ddot{y}^2$ getting the bound α_max^2, the subsequent equality should be solved:

$$\left(\frac{\alpha_\mathrm{max}^2}{4}\right)t_{\mathrm{f_a}}{}^4 + 6\big((x_\mathrm{f} - x_\mathrm{o}) + (y_\mathrm{f} - y_\mathrm{o})\big)t_{\mathrm{f}_a} - 9\left((x_\mathrm{f} - x_\mathrm{o})^2 + (y_\mathrm{f} - y_\mathrm{o})^2\right) = 0 \qquad t_{\mathrm{f_a}} \in \mathbb{R}^+ \tag{4.38}$$

Accordingly, the lower bound $t_{\mathrm{f_{min}}}$ in Equation (4.22) ensures that the system of constraints provided in equation (4.6) remains sustained.

$$t_{\mathrm{f_{min}}} = \max\left\{t_{\mathrm{f_a}}, t_{\mathrm{f_u}}\right\} \tag{4.39}$$

Chapter 5

VALIDATION, EXPERIMENTAL PROCESS, COMPARISON AND DISCUSSION OF THE REPRESENTATIVE TRAJECTORIES

In this chapter, the methodology proposed in the previous chapter of solving the problem of the optimal control stated in equation (4.6) will be simulated and tested. Moreover, a SAOWMR with three omnidirectional wheels will be chosen to perform the experiments since it possesses complete omnidirectional motion proficiency and arrange for three independent motion directions on a flat surface. To demonstrate the efficiency of the method proposed, the outcomes will be discussed and compared with those generated by the constrained dynamic inversion-based (CDIB) method stated in [64].

5.1 Constrained Dynamic Inversion Based (CDIB) Method

CDIB method aims to solve the trajectory optimization problem based on relaxing the terminal conditions as an alternate to the directly managing the restriction applied on the wheels' DC motors input voltage. Consequently, it is only requisite that the predefined configuration at the end of the trajectory being close enough to the actual ending configuration. The aim of the CDIB method is to find solution of the trajectory optimization problem stated in the following equation:

$$\text{Min} \quad C(t_f) = t_f + \zeta\, e_{\text{terminl}}$$

$$\text{Subject to:} \begin{cases} \mathbf{M}\ddot{Z} + \mathbf{A}\dot{Z} = \mathbf{Q}U \\ Z(t=0) = Z_o,\ \dot{Z}(t=0) = \dot{Z}_o,\ Z(t=t_f) = Z_f,\ \dot{Z}(t=t_f) = \dot{Z}_f \\ \ddot{x}^2 + \ddot{y}^2 \le \alpha_{\max}^2 \\ |u_o| \le u_{\max} \end{cases} \tag{5.1}$$

The positive weight ζ corresponds to the trade-off between the overall steering time t_f and the terminal error corresponding to the actual and the planned final configurations e_{terminal}. The closeness e_{terminal} is stated as follows:

$$e_{\text{terminl}} = \left\| Z(t=t_f) - Z_{\text{ref}}(t=t_f) \right\|_2 + \left\| \dot{Z}(t=t_f) - \dot{Z}_{\text{ref}}(t=t_f) \right\|_2 \tag{5.2}$$

CDIB method summarized as follows: Given the equation of motion of the OWMR illustrated in the equation (4.1). The control equation of motion able to be rewritten as following

$$U = \mathbf{Q}^{-1}\left(\mathbf{M}\ddot{Z} + \mathbf{A}\dot{Z}\right) = \mathbf{Q}^{-1}\, G(t) \tag{5.3}$$

Assuming that there is no constraint applied on the control voltage input, the robot will follow exactly the reference trajectory such as stated in equation (4.7) according to the following equation:

$$U_{\text{ref}} = \mathbf{Q}^{-1}\left(\mathbf{M}\ddot{Z}_{\text{ref}} + \mathbf{A}\dot{Z}_{\text{ref}}\right) = \mathbf{Q}^{-1}\, g(t) \tag{5.4}$$

Due to the actuator saturation, the constraint applied on the control voltage input, a Proportional-Differential tracker applied in order to keep track of the reference trajectory. The tracking error express the variance between the actual closed-loop trajectory and the reference trajectory given as follows:

$$e(t) = Z(t) - Z_{\text{ref}}(t) \tag{5.5}$$

A second-order dynamic could be imposed on the error in equation (5.5) as follows:

$$\ddot{e}(t) + \mathbf{C}_1\dot{e}(t) + \mathbf{C}_2 e(t) = 0 \tag{5.6}$$

The positive definite matrices $\mathbf{C_1}$ and $\mathbf{C_2}$ are presented to assure the stability. The reference trajectory $Z_{ref}(t)$ separated from equation (5.5) and substituted in equation (5.4) as follows

$$g(t)=\mathbf{M}\left(\ddot{Z}(t)-\ddot{e}(t)\right)+\mathbf{A}\left(\dot{Z}(t)-\dot{e}(t)\right) \tag{5.7}$$

Moreover, $\ddot{e}(t)$ could be isolated from equation (5.6) then substituted in equation (5.7) as follows:

$$g(t)=\mathbf{M}\ddot{Z}(t)+\mathbf{A}\dot{Z}(t)+\left(\mathbf{MC}_1-\mathbf{A}\right)\dot{e}(t)+\mathbf{MC}_2e(t) \tag{5.8}$$

Equation (5.8) able to be written as follows:

$$G(t)=g(t)+\left(\mathbf{A}-\mathbf{MC}_1\right)\dot{e}(t)+\mathbf{MC}_2e(t) \tag{5.9}$$

By substituting $G(t)$ in equation (5.3)

$$U=\mathbf{Q}^{-1}\left(g(t)+\left(\mathbf{A}-\mathbf{MC}_1\right)\dot{e}(t)+\mathbf{MC}_2e(t)\right) \tag{5.10}$$

According to the actuator constraint the closed-loop control law in equation (5.10) is not always achievable. We can insure that the actuator constraint always maintained by using the saturation function. Accordingly, the equation (5.10) rewritten as follow

$$U_{saturated}=\text{saturation}\left(\mathbf{Q}^{-1}\left(g(t)+\left(\mathbf{A}-\mathbf{MC}_1\right)\dot{e}(t)+\mathbf{MC}_2e(t)\right)\right) \tag{5.11}$$

Based on the equation (5.11), the error tracking evolution is stated as follow

$$\ddot{e}(t)=\mathbf{M}^{-1}\left(-\mathbf{A}\dot{e}(t)-g(t)+\mathbf{Q}U_{saturated}\right) \tag{5.12}$$

The error along the trajectory can be found by solving the nonlinear and non-autonomous differential equation clarifies in equation (5.12). The initial conditions for equation (5.12) given in (5.13) depending on the fact that in the mobile robot starting configuration there is no initial error.

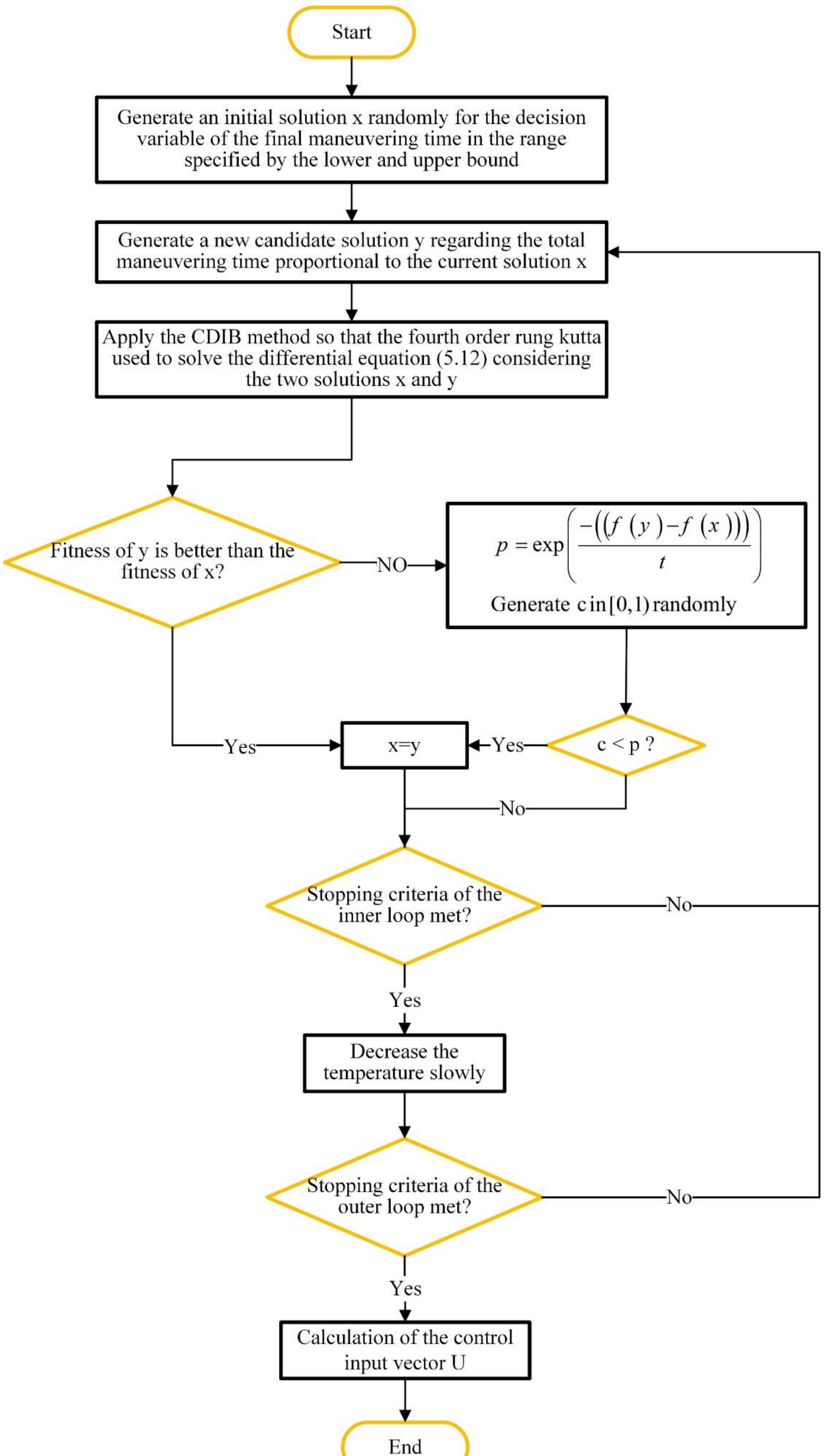
$$e(0)=\dot{e}(0)=0 \tag{5.13}$$

The 4th order Runge-Kutta discretization method can be implemented aiming to resolve the differential equation stated in equations (5.12) and (5.13). Once the differential equation problem gets solved, it is possible then to find the closed-loop trajectory given in equation (5.11). Furthermore, the velocities and the acceleration of the trajectory components could be generated by substituting the error along the trajectory $\mathrm{e}(t)$ in equation (5.5) and differentiate $\mathrm{Z}(\mathrm{t})$.

Successively, the dynamic optimization problem in (5.1) able to be expressed as a one dimensional search for the total maneuvering time t_{f} based on the inversion based dynamic control method as follows

$$\mathrm{Min} \quad \mathrm{C}\left(t_{\mathrm{f}}\right) = t_{\mathrm{f}} + \zeta\, e_{\mathrm{terminl}} \qquad t_{\mathrm{f}} \in \left[t_{\mathrm{f}_{\min}} \quad t_{\mathrm{f}_{\max}} \right] \tag{5.14}$$

A lower limit for the free variable t_{f} can be selected so that the acceleration constraints remains maintained. An upper bound $t_{\mathrm{f}_{\max}}$ could be selected based on experience (a proper OWMR proceeding movements accomplish most of the movements regarding initial and final configurations in just a few seconds). Furthermore, to perform the optimization process, the cost function can be calculated at homogenously spaced points in the minimum and maximum limit interval. However, it is noticeable that the fitness function in equation (5.14) is a non-linear and non-convex function. The conventional optimization methods may fail to find the global solution and typically held a local minimum. Subsequently, it is suggested to implement a smart technique to perform the optimization such as utilized in [86] that the Simulated Annealing (SA) algorithm used to perform the optimization process. The flowchart that illustrates the CDIB method with SA algorithm as an optimizer solver given in the following Figure



Start
Generate an initial solution x randomly for the decision variable of the final maneuvering time in the range specified by the lower and upper bound
Generate a new candidate solution y regarding the total maneuvering time proportional to the current solution x
Apply the CDIB method so that the fourth order rung kutta used to solve the differential equation (5.12) considering the two solutions x and y
Fitness of y is better than the fitness of x?
NO
$p = \exp\left(\frac{-((f(y)-f(x)))}{t}\right)$
Generate c in [0,1) randomly
Yes
x=y
Yes
c < p ?
No
Stopping criteria of the inner loop met?
No
Yes
Decrease the temperature slowly
Stopping criteria of the outer loop met?
No
Yes
Calculation of the control input vector U
End

Figure 5.1: Constraint dynamic inversion based method Flowchart

5.2 Numerical Values of Different Characteristics concerning the Representative Trajectories Experiment Process

Values of different characteristics stated to produce the optimal trajectory concerning the SAOWMR and its omnidirectional special wheels given in the Table 5.1. SAOWMR with 3 omnidirectional orthogonal wheels selected to perform the simulation experiments since it has complete omnidirectional movement ability and affords three independent motion directions on an even terrain.

Table 5.1: Values of the SAOWMR characteristics utilized to produce the trajectory

Parameter	**Description**	**Value**	**Unit**
n	Number of the omnidirectional wheels	3	**-**
φ	The angle concerning any two neighboring wheels	$2\pi/3$	**rad**
m	Mass of the mobile robot	2.45	**kg**
J	Mass moment of inertia of the mobile robot	0.00625	**Kgm^2**
L	The radius of the mobile robot	0.09	**m**
r	The radius of the mobile robot's wheels	0.02	**m**
$\alpha_{\mathbf{max}}$	Restriction applied on the acceleration amplitude	2	**ms^{-2}**
k_τ	The torque constant of the mobile robot	0.293	**Nm/A**
R	The resistance of the armature	1.465	**Ω**
$u_{\mathbf{max}}$	Restriction applied on the DC motor' terminal input	14.8	**V**
α	Characteristic constant	10	**NV^{-1}**
β	Characteristic constant	146	**Nsm^{-1}**

Two numerical tests proposed to compare the results acquired by the proposed method with those obtained by the CDIB technique. In Table 5.2, the boundary conditions stated concerning the two examined maneuvers given in SI units as follows:

Table 5.2: Configurations of the two examined maneuvers.

Maneuvers	Initial Alignments	End Alignments
Maneuver test 1	$\mathrm{Z}(0)=\begin{bmatrix}-1\\0\\\pi/4\end{bmatrix},\ \dot{\mathrm{Z}}(0)=\begin{bmatrix}0.1\\-0.5\\0.2\end{bmatrix}$	$\mathrm{Z}(t_\mathrm{f})=\begin{bmatrix}0.5\\-1.5\\-\pi/2\end{bmatrix},\dot{\mathrm{Z}}(t_\mathrm{f})=\begin{bmatrix}-0.8\\-0.1\\0.4\end{bmatrix}$
Maneuver test 2	$\mathrm{Z}(0)=\begin{bmatrix}-2.5\\1.7\\-\pi/2\end{bmatrix},\dot{\mathrm{Z}}(0)=\begin{bmatrix}-0.6\\0.5\\-0.6\end{bmatrix}$	$\mathrm{Z}(t_\mathrm{f})=\begin{bmatrix}-1.1\\0\\-\pi/6\end{bmatrix},\dot{\mathrm{Z}}(t_\mathrm{f})=\begin{bmatrix}-0.1\\0.8\\0.2\end{bmatrix}$

5.3 Validation Process of the Representative Trajectories produced by the Proposed Method

Solving the trajectory optimization problem stated in equation (4.6) based on the method proposed may be thought as an examination for the total maneuvering time t_f that minimizes the cost function C stated in equation (4.22) that a lowest limit $t_{\mathrm{f}_\mathrm{min}}$for the variable t_f ensures that the restriction system stated in equations (4.3) and (4.4) remains sustained. A simple golden-section search technique could be applied to solve for an appropriate solution concerning the optimum final time t_f that minimizes the cost function C stated in the Equation (4.22) since the energy is a monotonic function regarding the variable t_f changing inside a domain of minimum and maximum limits.

5.3.1 Dealing with the Constraints System

The trajectory optimization problem stated in equation (4.6) considers two constraints: one placed on the amplitude of the maximum acceleration of the center of mass of the SAOWMR and the other is saturation constraint placed on the control input voltages of the omnidirectional wheels' DC motors. This section aims to solve for the lower bound of the total steering time $t_{\mathrm{f}_\mathrm{min}}$ stated in equation (4.22) so that the system of constrains remains preserved concerning the two numerical test Maneuvers given in the Table 5.2.

Aiming to solve for the value of the complete steering time t_{f} so that the acceleration amplitude of the center of mass of the mobile robot $\ddot{x}^2+\ddot{y}^2$ critically reaches the limit $\alpha_{\max}$, the two equalities stated in equation (4.16) must be resolved to give one satisfactory solution for each equation. Regarding the maneuvers given in the Table 5.2, the solutions obtained and the acceptable solution given in the subsequent table. Matlab function named as "*acceleration_bound*" written and provided in Appendix C aiming to find the total maneuvering time so that the acceleration amplitude of the center of mass of the mobile robot $\ddot{x}^2+\ddot{y}^2$ critically reaches the limit $\alpha_{\max}$. The solutions obtained by solving the two equality in terms of the two provided experimental test given as follows

Table 5.3: The acceptable solution of the two testing maneuvers

Maneuvers	**Solutions obtained**		**Acceptable Solution**
Maneuver 1	$t_{\mathrm{f_{a1}}}$=2.5000	$t_{\mathrm{f_{a2}}}$=3.0656	$t_{\mathrm{f_a}}$=3.0656
Maneuver 2	$t_{\mathrm{f_{a1}}}$=3.8933	$t_{\mathrm{f_{a2}}}$=3.9087	$t_{\mathrm{f_a}}$=3.9087

The solution procedure proposed aiming to resolve the saturation of voltage constraint summarized by finding the numeric value for the overall steering time $\mathrm{t}_{\mathrm{f}_u}$ that the maximum voltage of the input terminal of the omnidirectional wheels' DC motors u_{o} given in the equation (5.15) critically reaches the limit $u_{\max}$ given in this section as 14.8 V.

$$u_o=\max\left\{\left|u_{o1}\right|,\quad\cdots\quad,\left|u_{oi}\right|,\quad\cdots\quad,\left|u_{on}\right|\right\}\tag{5.15}$$

u_{oi} stated in equation (5.15) represents the extreme voltage value supplies the input terminal of the ith DC motor corresponding to the ith omnidirectional wheel. As discussed in the previous chapter, the nature of the voltage function of the input terminal of the ith DC motor u_i is nonconvex and nonlinear in common in the domain contained by 0 and t_f. The Simulated Annealing algorithm is proposed to use in order to solve for the extreme solution $u_{\mathrm{o}i}$ of the voltage function u_i. The Simulated Annealing algorithm characteristic chosen to resolve the optimization problem expressed in the Table 5.4. In Appendix D, the Simulated Annealing algorithm written and modeled as a function named "*simulated_anl*".

Table 5.4: SA algorithm characteristics

Parameter	Value
Temperature reduction rule	Slow-decrease
Temperature update function	Exponential
Number of iterations per temperature	50
Maximum function evaluation	2500
Tolerance	10^{-3}

The final steering time $t_{\mathrm{f_u}}$ regarding the extreme voltage input of the terminals DC motors u_o getting the extreme bound u_max able to be found by applying root-finding bisection technique on the following equality within a range specified by $t_{\mathrm{f_{min}}} = 0$ and $t_{\mathrm{f_{max}}}$. Appendix E provides a Matlab function modeled based on root finding bisection method.

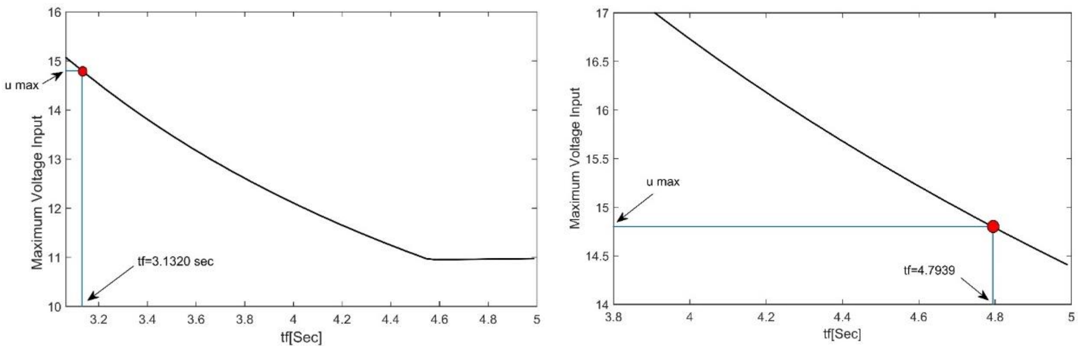
$$\mathrm{u_o}\left(t_\mathrm{f}\right)=\mathrm{u_{max}} \qquad t_\mathrm{f} \in \left[t_{\mathrm{f_a}} \quad t_{\mathrm{f_{max}}}\right] \tag{5.16}$$

The outcomes acquired by utilizing the method proposed to find $t_{\mathrm{f_u}}$ so that the restriction on the input terminals of the DC motor remains maintained given in the following table concerning the two given numerical test maneuvers. The proposed method to find $t_{\mathrm{f_u}}$ so that the maximum input voltage applied to the terminals of the DC motor reaching the extreme bound u_max written as Matlab function named " *maximum_point_control_input*" provided in the Appendix F

Table 5.5: The solutions obtained by solving the saturation constraint based on utilizing the Simulation Annealing algorithm and the Bisection method

Maneuvers	t_{f_a}	t_{f_u} **Obtained Value**	t_{f_u} **Actual Value**	**Number of Iterations**	**Execution Time**
Maneuver 1	3.0656	3.1313	3.1320	7	1.4657
Maneuver 2	3.9087	4.7943	4.7939	6	1.1457

The following figures validate the method of finding the overall steering time so that the extreme input voltage u_o getting the extreme bound $u_{\max}$



(**a**) (**b**)

Figure 5.2: Validation the method of finding the overall steering time regarding the extreme input voltage u_o getting the extreme limit u_{max}. (a) and (b) respectively consider the maneuver1 and maneuver 2

Based on the development in this section, the domain of total steering time t_f so that the constraints system still preserved concerning the maneuver1 and maneuver 2 given in the equations (5.17) and (5.18) respectively

$$t_f \geq 3.1320 \tag{5.17}$$

$$t_f \geq 4.7939 \tag{5.18}$$

5.3.2 Optimization Process

The optimal trajectory generation approach proposed to resolve the optimization problem stated in equation (4.6) able to be considered as selecting the trajectory stated by choosing t_f from a set of an acceptable domain constrained by minimum and maximum limit on t_f. However, the problem specified in equation (4.6) is decreased to the one-dimensional search for the optimum numeric value for the complete steering time t_f that reduces the cost function C expressed as follows:

$$\mathrm{Min} \quad \mathrm{C}(t_\mathrm{f}) = t_\mathrm{f} + \gamma \mathrm{E} \qquad t_\mathrm{f} \in \left[t_{\mathrm{f}_\mathrm{min}} \quad t_{\mathrm{f}_\mathrm{max}} \right] \tag{5.19}$$

The solution of the optimization problem stated in equation (5.19) able to be done by specifying the domain for the complete steering time t_f that ensures the system of constraints. A lowest limit $t_{\mathrm{f}_\mathrm{min}}$ concerning the variable t_f ensures that the system of the constraints remains sustained. An upper bound $t_{\mathrm{f}_\mathrm{max}}$ could be selected based on experience (a proper OWMR proceeding movements on a 3 m^2 playing field could accomplish most of the transpositions in just a few seconds). A positive weight clarifying the trade-off between the total energy required E and the robot maneuvering time t_f given in (5.19) as γ and will be given a value of 2. The minimization problem in (5.19) concerning the two numerical test maneuvers given in (5.20) and (5.21) respectively.

$$\mathrm{Min} \quad \mathrm{C}(t_\mathrm{f}) = t_\mathrm{f} + 2\mathrm{E} \qquad t_\mathrm{f} \in \left[3.1313 \quad 7 \right] \tag{5.20}$$

$$\mathrm{Min} \quad \mathrm{C}(t_\mathrm{f}) = t_\mathrm{f} + 2\mathrm{E} \qquad t_\mathrm{f} \in \left[4.7943 \quad 7 \right] \tag{5.21}$$

To find a suitable solution concerning the optimality of the total maneuvering time t_f that reduces the cost C in (5.20) and (5.21), a golden-section search technique could be straightforwardly applied since the energy is a monotonic function regarding the variable t_f changing contained by the lowest and highest limits.

Figure 5.3 demonstrates the estimation of the energy E and the sample space of the fitness function stated in (5.20) and (5.21) with $\gamma = 2$ considering the two movements stated in Table 5.3 calculated at equivalently spaced points in the domain bounded by minimum and maximum limits concerning the variable t_f. Furthermore, the optimal solutions acquired by applying the golden-section search technique with margin of 10^{-3} concerning the two maneuvers are expressed in the Table 5.6. The proposed method modeled using Matlab programing language and provided in Appendix G.

Table 5.6: the solution obtained by the golden section search method concerning the maneuver 1 and 2

Maneuvers	$t_{f_{\min}}$	$t_{f_{\max}}$	**Optimal Final Time t_f [sec]**	**E [Joule]**	**Code Execution Time [sec]**
Maneuver 1	3.1313	7	4.5103	2.4688	0.9054
Maneuver 2	4.7943	7	5.7563	3.8798	1.0827

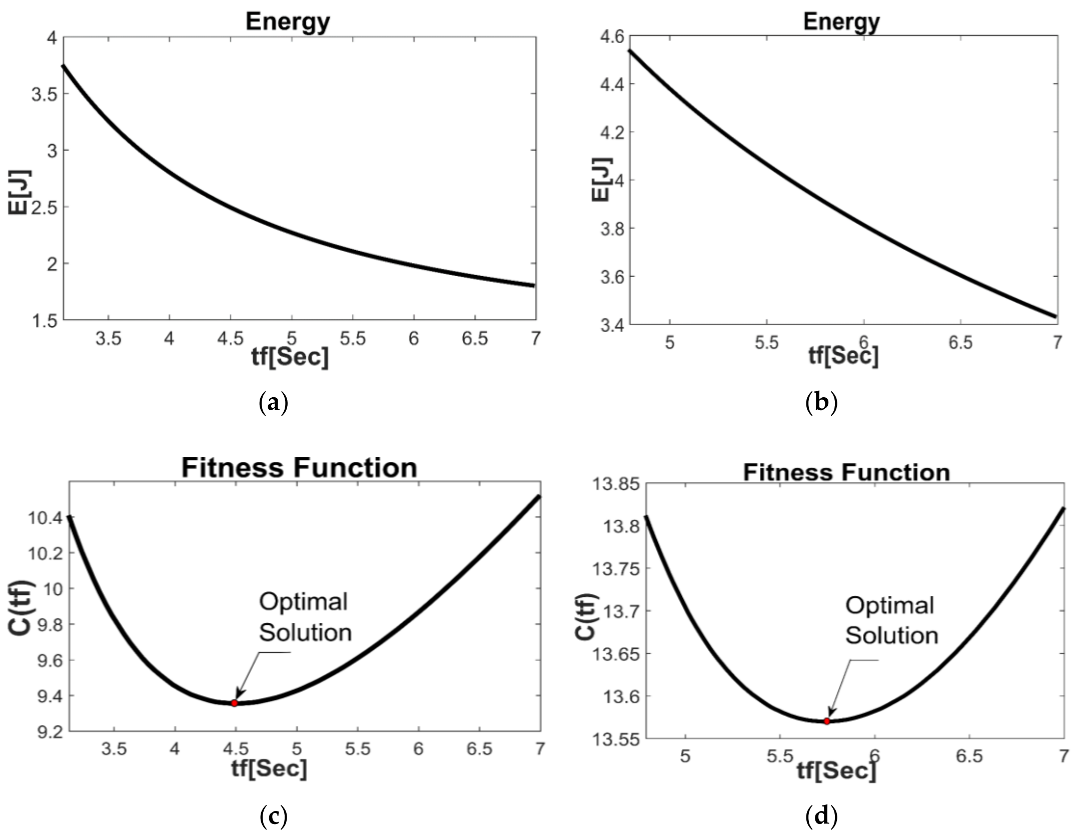


Figure 5.3: Energy and fitness function assessment and illustration of the optimum solutions considering the two movements: (a, b) are respectively the energy assessment at regularly spaced points for the variable t_f considering maneuver 1 and maneuver 2; (c, d) respectively are the fitness function stated in Equation (5.19) with $\gamma = 2$ evaluated at homogeneously spaced points to show the convergence to the optimum solution concerning the maneuver 1 and maneuver 2.

The outcomes demonstrated in Figure 5.5 and the Table 5.6 show obviously that the method proposed meets to the comprehensive minimum concerning the optimization problem stated in equation (5.1). Figures 5.4 and 5.5 illustrate the optimal trajectories produced by the method proposed concerning maneuver 1 and maneuver 2.

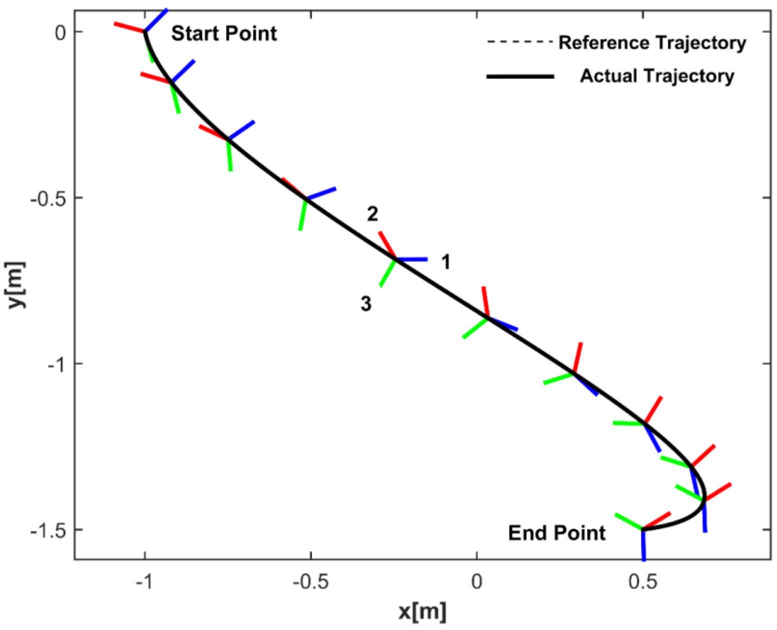


Figure 5.4: The trajectory produced by the method proposed concerning the movement 1

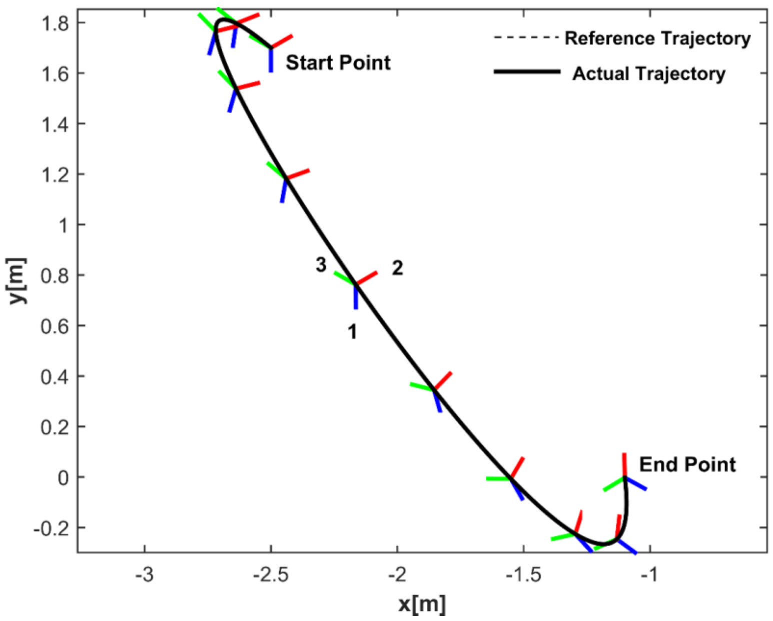


Figure 5.5: The trajectory produced by the method proposed regarding the movement 2

The optimum trajectories produced by the method proposed concerning the movement 1 and movement 2 demonstrated respectively in the Figure 5.6 and Figure 5.7 demonstrate a successful process in following the reference trajectory without producing any tracking error.

5.4 Comparison Analysis

To demonstrate the efficiency of the method proposed in the chapter 4, the results will be compared and discussed with those generated by the constrained dynamic inversion-based (CDIB) method stated in [64].

The CDIB technique used to solve the time-optimal control problem based on relaxing the terminal conditions as an alternate to the direct managing of the constraint applied on the voltage input of the terminals of the DC motors. Consequently, it is just requisite that the predefined configuration at the end of the trajectory being enough close to the actual ending configuration. The accuracies required concerning the CDIB method are stated in Table 5.7.

Table 5.7: Values utilized to generate the trajectory regarding CDIB method.

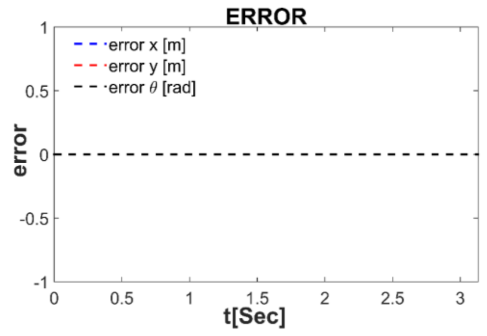
Parameter	**Description**	**Value**	**Unit**
$\begin{bmatrix} e_x \\ e_y \end{bmatrix}$	Chosen distance accurateness	$\begin{bmatrix} 0.02 \\ 0.02 \end{bmatrix}$	m
e_θ	Chosen orientation accurateness	0.12	Rad
$\begin{bmatrix} e_{\dot{x}} \\ e_{\dot{y}} \end{bmatrix}$	Chosen linear velocity accurateness	$\begin{bmatrix} 0.02 \\ 0.02 \end{bmatrix}$	m/s
$e_{\dot{\theta}}$	Chosen angular velocity accurateness	0.12	Rad/s
ζ	Positive weight illustrating the trade-off between the steering time and the error	2	

The MATLAB programing language is employed in this thesis to simulate the algorithms and to obtain the results, and graphs. Moreover, the algorithm proposed and the CDIB technique evaluated concerning the time execution by utilizing the C language. The results attained by applying the proposed method and the CDIB technique will be evaluated in Table 5.6 concerning the optimum complete steering time, error at the terminal, and code time execution. The CDIB technique stated in [64] doesn't concern the energy consuming. Subsequently, the trade-off between the mobile robot total steering time t_f and the energy γ stated in equation (4.6) replaced by digit zero.

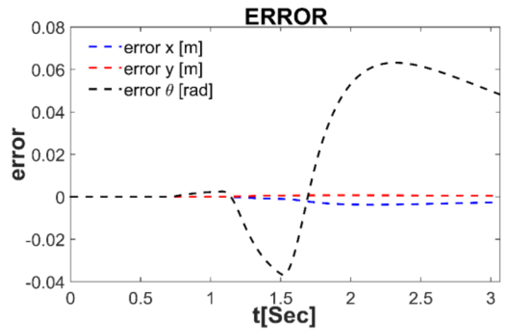
Table 5.8: Trajectory optimization process outcomes the two maneuvers 1 and 2

Movement	**Methodology**	**Optimum Final Time**	**Error at termination condition**	**Fitness Function**	**Code Time Execution (MATLAB)**	**Code Time Execution (C)**
Movement 1	Method Proposed	3.1320 s	0.0000	3.1320	2.1265 s	0.06947 s
	CDIB [64]	3.0656 s	0.0044	3.0744	14.6981 s	1.68790 s
Maneuver 2	Method Proposed	4.7938 s	0.0000	4.7938	1.9827 s	0.06891 s
	CDIB [64]	3.9087 s	0.0206	3.9499	15.0198 s	1.49827 s

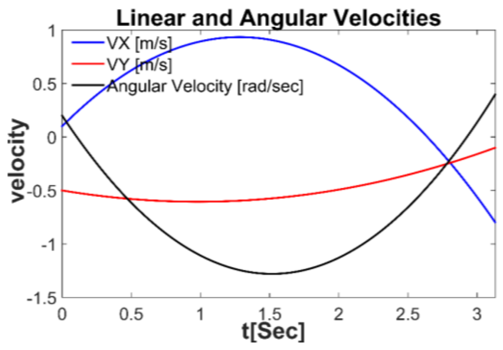
The method proposed in this thesis together with the CDIB technique were applied on a PC with CPU Intel CORE i5 2.53 GHz. The outcomes demonstrate obviously intensive enhancement regarding the code time execution with an improvement from nearby 1.68 s and 1.5 s respectively in movement 1 and 2, to nearly 0.07 s. Furthermore, it is obviously that the CDIB technique aims to reduce the error termination generated by violate the restriction applied on the voltage input. However, the method proposed aims to find the optimum solution without generating terminal errors.



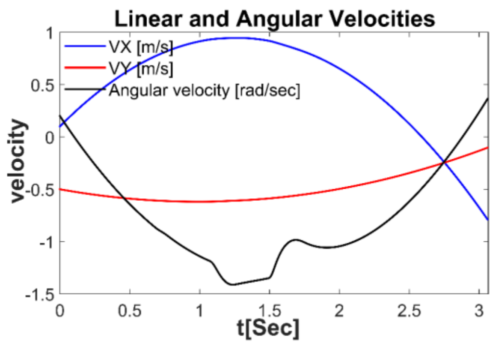
(**a**)



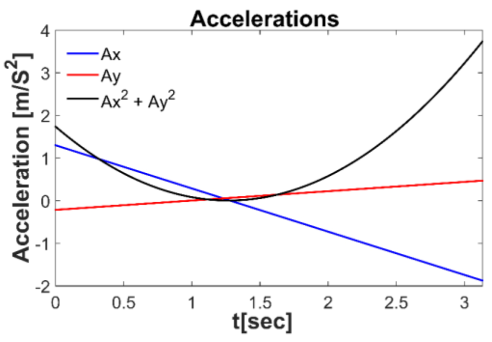
(**b**)



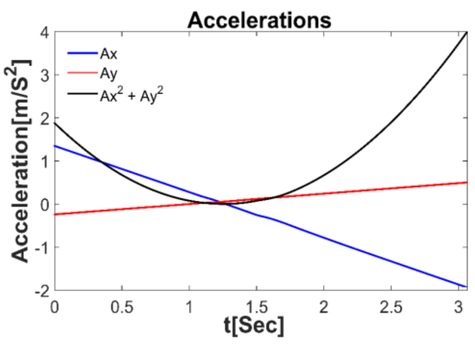
(**c**)



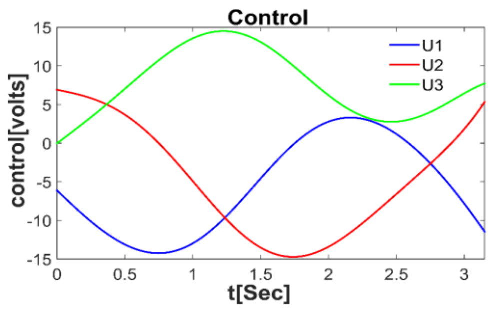
(**d**)



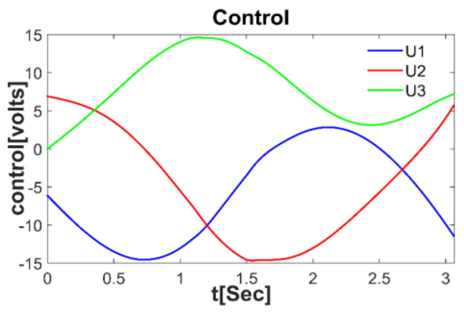
(**e**)



(**f**)

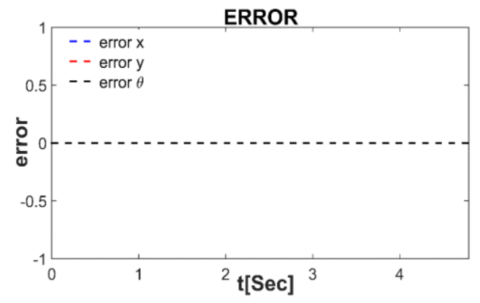


(**g**)

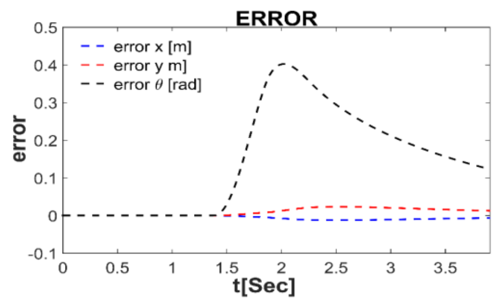


(**h**)

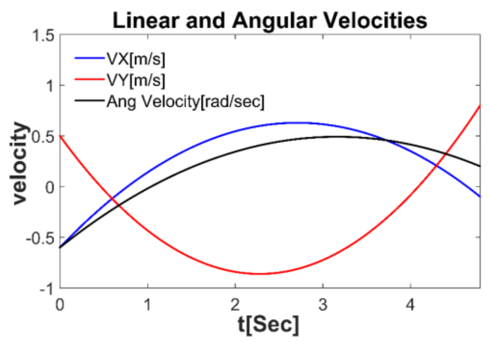
Figure 5.6: The figures illustrate a Comparison between the method and the CDIB technique concerning maneuver 1: (a, c, e, g) are respectively the evolution of the error alongside the trajectory, the time history of the angular and linear motion, the acceleration along the trajectory, and the voltage inputs produced by the method proposed,; (b, d, f, h) the evolution of the error alongside the trajectory, the time history of the angular and linear motion of the robot' center of mass, the acceleration of the robot' center of mass alongside the trajectory, and the voltage inputs produced by the CDIB technique, respectively.



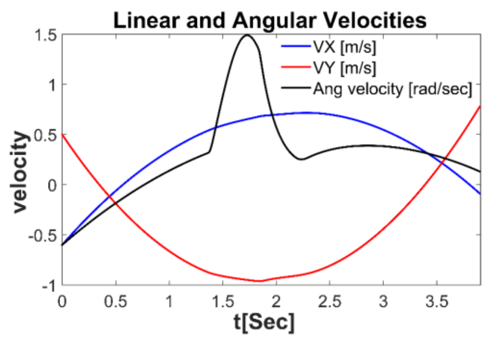
(**a**)



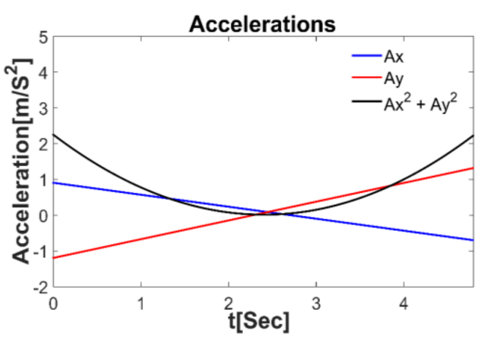
(**b**)



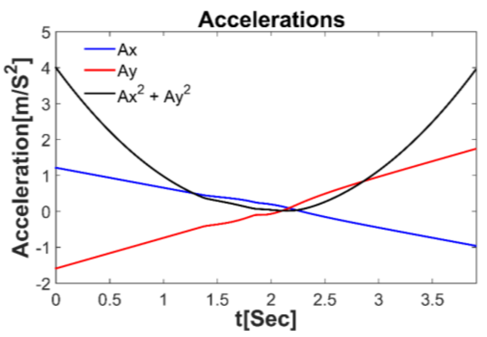
(**c**)



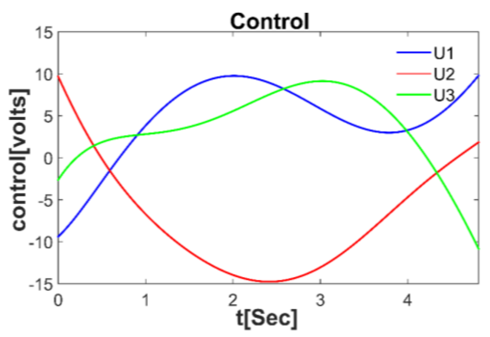
(**d**)



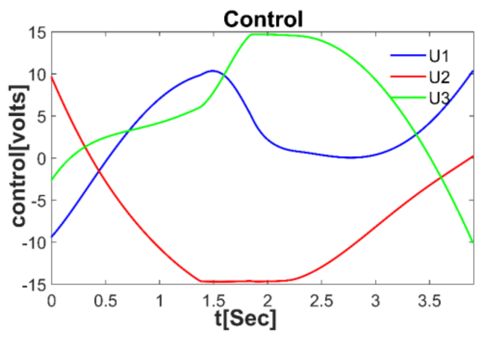
(**e**)



(**f**)



(**g**)



(**h**)

Figure 5.7: The figures illustrate a Comparison between the method and the CDIB technique concerning maneuver 2: (a, c, e, g) are respectively the evolution of the error alongside the trajectory, the time history of the angular and linear motion, the acceleration along the trajectory, and the voltage inputs produced by the method proposed,; (b, d, f, h) the evolution of the error alongside the trajectory, the time history of the angular and linear motion of the robot' center of mass, the acceleration of the robot' center of mass alongside the trajectory, and the voltage inputs produced by the CDIB technique, respectively.

Figure 5.6 and Figure 5.7 compare the results produced by proposed method with that acquired by the CDIB technique offered in [64] concerning the evolution of the error alongside the trajectory, the time history of the angular and linear motion of the mass center of the mobile robot, the acceleration of the mass center of along the trajectory, and the voltage inputs generated by the both methods concerning movements 1 and 2. The outcomes presented in Figure 5.6 and Figure 5.7 demonstrate that the proposed method effectively handle the acceleration restriction and the limitation on the voltage inputs presented in equation (4.6) regarding movement 1 and 2. The employment of the golden-section search technique to resolve the optimization problem stated in equation (4.22) was recommended since the energy E is a monotonic function concerning the variable t_f as presented in Figure 5.3.

Table 5.9 expresses the efficiency of employing the method proposed over the CDIB technique regarding the total energy required to move the mobile robot concerning the two movements stated in Table 5.2.

Table 5.9: Trajectory optimization process results regarding maneuver 1 and 2

Maneuvers	**Method**	**trade-off value**	t_f **[sec]**	**E [Joule]**
Movement 1	Proposed Method	$\gamma = 0$	3.1320	3.7029
		$\gamma = 2$	4.5103	2.4688
	CDIB [64]	-	3.0656	3.8327
Movement 2	Proposed Method	$\gamma = 0$	4.7938	4.4805
		$\gamma = 2$	5.7563	3.8798
	CDIB [64]	-	3.9087	5.4504

5.5 Study Discussion

This work discusses the problem of the trajectory planning optimization for the SAOWMR with n special wheels as the procedure of assessing the input voltage vector U placed on the DC motors terminals of the special wheels so that the mobile robot able to perform maneuver from one configuration to another one with the optimality of the final steering time and the total energy E while the system of the constraint still conserved. The proposed technique to resolve the problem of the trajectory optimization on the basis of selecting the optimal trajectory indicated by choosing the final steering time t_{f} over an acceptable range constrained by minimum limit $t_{\mathrm{f}_{\mathrm{min}}}$ and maximum limit $t_{\mathrm{f}_{\mathrm{max}}}$ so that the problem is simplified to the one dimension search for the t_{f}. A system of constraint can ascend due to several causes. However, two important restrictions have been treated in this study those are the restriction on the extreme amplitude acceleration and the restriction on the voltage input of the terminals of the DC motors. The minimum limit $t_{\mathrm{f}_{\mathrm{min}}}$concerning the variable t_{f} specified on the basis of resolving for the t_{f} so the constraints system stays sustained. Furthermore, the extreme limit $t_{\mathrm{f}_{\mathrm{max}}}$ concerning the variable t_{f} can be chosen on the basis of the understanding of the functioning background and the qualifications of the mobile robot. Consequently, a golden-section search method is applied to find the optimal solution for the optimization problem formed by the weight vector of the complete steering time t_{f} and the required energy E.

A SAOWMR that utilizes three special orthogonal wheels used to validate the efficiency of the trajectory planning optimization method proposed. The method proposed in this chapter compared with other highly efficiency trajectory optimization method known as constrained dynamic inversion-based (CDIB) technique to demonstrate the competence of our proposed method. By relaxing the terminal conditions of the constraint applied on the input voltage of the DC motors, the CDIB method solves the trajectory planning optimization on the basis that it is simply requisite at the end of the terminal configuration of the trajectory the actual configurations are sufficiently close to the primarily intended ending configuration. All the outcomes and plots in this thesis acquired by utilizing MATLAB programing language. Furthermore, C language utilized to achieve a judgement of the algorithm proposed and the CDIB technique on the basis of the complete execution time. Moreover, both the CDIB technique and the method proposed were applied on the identical PC with CPU Intel Corei5 3.2 GHz so that the tolerance at the termination configurations and the execution time of the code concerning the two approaches were considered. The outcomes in Table 5.8 obviously shows the efficiency of the method proposed in this thesis over the CDIB technique concerning the overall implementation time, presenting a great improvement was attained. Additionally, the trajectory formed by the method proposed finds to the optimal solution without generating terminal error comparing to the CDIB technique. One more benefit of utilizing the method proposed over the CDIB technique concerning the overall energy requisite to move the mobile robot from one configuration to another is stated in Table 5.9. The convergence of the method proposed to the optimal universal solution is clarified in Figure 5.3.

To conclude this chapter, numerous points ensures the novelty and the efficiency of the method proposed over other techniques in the scope of trajectory planning optimization. First, the proposed method concerning SAOWMR that utilizes n omnidirectional special wheels. This makes this method appropriate for OWMR that apply three, four, five, or even more of those special wheels. Second, the method proposed in this chapter shows real-time performance that the optimum trajectory produced in less than 70 milliseconds. Also, the proposed method demonstrates a precision performance that the produced trajectory tends to be without terminal error. Moreover, getting the extreme bound of the input control of the DC motor' wheel is occasionally unwanted, the proposed method in this chapter allows the control of getting the extreme bound by utilizing of the factor γ expressed in Equation (4.6). Furthermore, the combination of proposed method of producing the optimal trajectory along with the path constraints such as obstacles is conceivable, as presented in the following chapter.

Chapter 6

OBSTACLE AVOIDANCE STRATEGY

6.1 Problem Statement

The problem in this chapter is expressed as follows: a SAOWMR that utilizes n omnidirectional special wheels at a preliminary configuration must attain the termination configuration whilst avoiding constant obstacles. This chapter aims to solve the problem of obstacle avoidance optimal trajectory planning for SAOWMR with n special wheels by integrating the problem of optimal control in equation (4.6) with the motion planner and the path constraints.

6.2 Motion Planning

In this section, the Artificial Potential Field (APF) technique will be applied as a motion planner concerning the SAOWMR that utilizes n omnidirectional special wheels. Artificial Potential Field Technique considered as one of the most important and conventional algorithms used for motion planning concerning the mobile robot. The concept of this algorithm is based on creating an APF that lead the mobile robot to the goal while avoiding obstacles exist in the robot environment. Two important ideas characterize this method: The Mobile Robot's starting point will be assigned a high potential and the goal will be assigned a low potential so that the robot maneuvers from the highest potential to the lowest potential. The obstacles will be given high potential so that repulsive force acting between the robot and the obstacles.

In order to model the artificial potential field, two different types of potential must be considered: Attractive potential lead the mobile robot to maneuver to the goal and repulsive potential assist the robot to avoid the obstacles. The attractive and repulsive potential stated as follows:

$$\mathrm{U}_{att}(\mathrm{q}) = -\sigma_{\mathrm{T}} \exp\left(-\mu_{\mathrm{T}}(\mathrm{q}-\mathrm{q}_{\mathrm{T}})^2\right) \tag{6.1}$$

$$\mathrm{U}_{rep}(\mathrm{q}) = \sigma_{\mathrm{o}} \exp\left(-\mu_{o}(\mathrm{q}-\mathrm{q}_{\mathrm{o}})^2\right) \tag{6.2}$$

In equations (6.1) and (6.2), $\mathrm{q}=(x,y)$ is the current mobile robot position, $\mathrm{q}_{\mathrm{o}}=(x_{\mathrm{o}},y_{\mathrm{o}})$ is the obstacle position, $\mathrm{q}_{\mathrm{T}}=(x_{\mathrm{g}},y_{\mathrm{g}})$ is the goal position, σ_{T} is the depth of the attractant, σ_{o} is the depth of the repellant, μ_{T} is the width of the attractant and μ_{o} is the width of the repellant. Regarding the case of multiple obstacles, the equation (6.2) given as follows:

$$\mathrm{U}_{rep}(\mathrm{q}) = \mathrm{U}_{rep\_1}(\mathrm{q}) + \cdots + \mathrm{U}_{rep\_n}(\mathrm{q}) \tag{6.3}$$

Combined field or combined potential will be arrangement of attractive and repulsive potential considered as follows:

$$\mathrm{U}_{total}(\mathrm{q}) = \mathrm{U}_{att}(\mathrm{q}) + \mathrm{U}_{rep}(\mathrm{q}) \tag{6.4}$$

Two stages describe the planning of movement of the mobile robot on the basis of the APF method: The Glide stage performed while no obstacles in the sensor rang, instruct the mobile robot to move in a straightforward line from the preliminary to the ending point and The Maneuver stage, it assists the mobile robot to go around the obstacle based on Artificial Particles (AP) placed around the mobile robot on a circle of radius c_t at equal angle intervals. The motion planning strategy in terms of the potential field method will be expressed in the following flowchart:

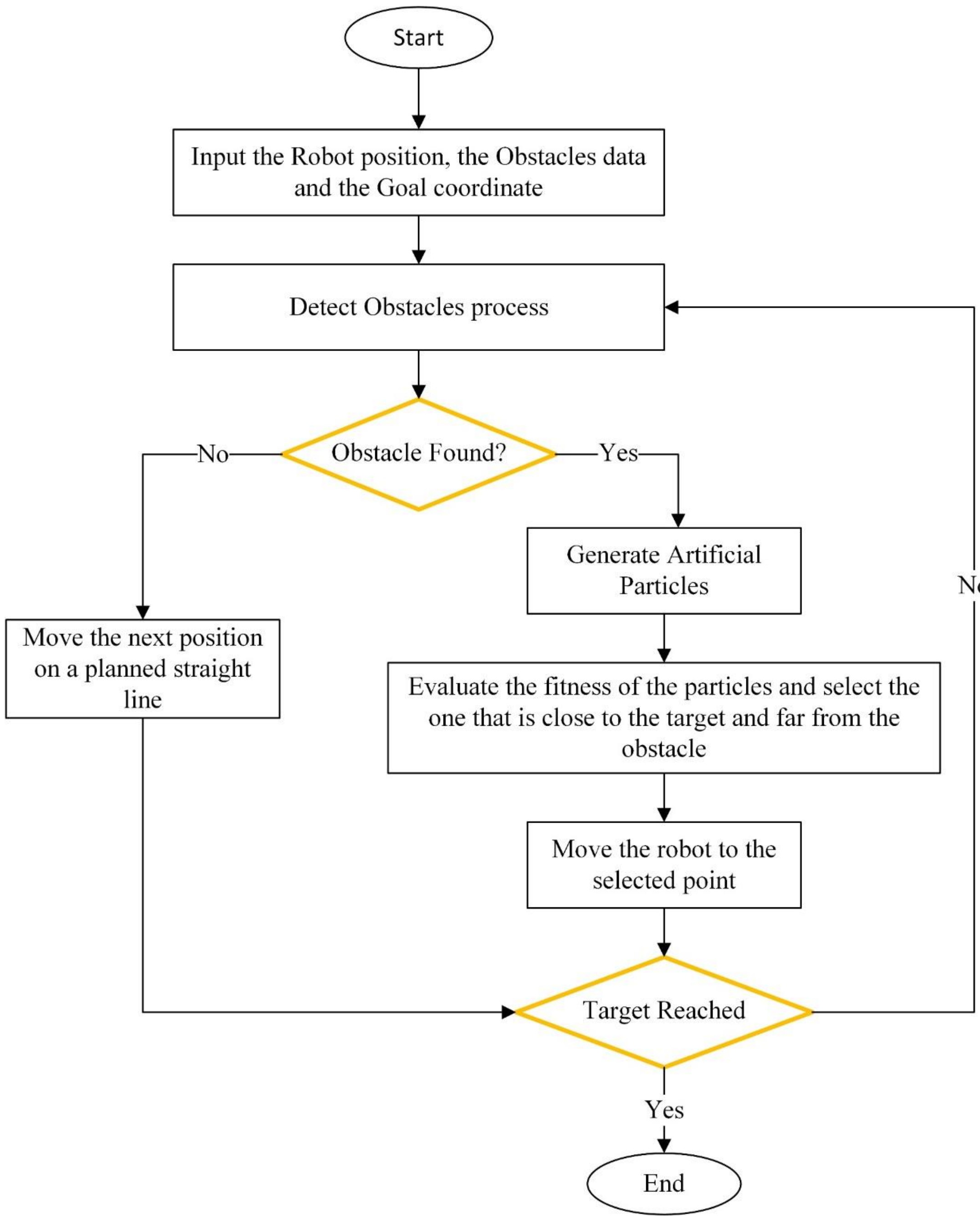

Start
Input the Robot position, the Obstacles data and the Goal coordinate
Detect Obstacles process
Obstacle Found?
No
Yes
Generate Artificial Particles
No
Move the next position on a planned straight line
Evaluate the fitness of the particles and select the one that is close to the target and far from the obstacle
Move the robot to the selected point
Target Reached
Yes
End

Figure 6.1: Artificial Potential Field Method working principle Flowchart

6.3 Trajectory Generation Optimization Approach

The procedure proposed in this chapter to resolve the optimal trajectory generation with obstacle avoidance concerning the SAOWMR with n omnidirectional wheels is summarized based on criteria given as follows:

Criteria 1: Two dimensional points generated and constructing collision free path between the robot' initial position and the target by utilizing the well-known APF method. The parameters concerning the attractive–repulsive potential chosen based on the dimensions of the mobile robots and the workplace.

Criteria 2: A convinced number of correspondingly organized via points will be selected along the path produced.

Criteria 3: As the applied path planning procedure handles just x and y components, the mobile orientation of the robot concerning the midpoints able to be simply identified using the constant distribution notion.

Criteria 4: To choose appropriate velocity values corresponding to the midpoints, the interpolation of the cubic spline concerning to the potential value of the complete steering time will be used so that the third-order polynomial characterizes the trajectory between the contiguous via points along the path.

Criteria 5: The total steering time from the preliminary position to the completion position can be assigned uniformly to the portions of the overall generated path on the basis of the via points distributed along the path. Consequently, with regard to the j-1 via points, the total maneuvering time $t_{\mathrm{f_T}}$ will be apportioned into j parts constructed by third-order polynomials as following:

$$t_{\mathrm{f_T}} = \sum_{\mathrm{c}=1}^{j} t_{\mathrm{f_c}} \tag{6.5}$$

Criteria 6: The optimal trajectory can be determined based on solving the subsequent optimization problem

$$\text{Min} \quad C\left(t_{f_T}\right) = t_{f_T} + \gamma E \qquad t_{f_T} \in \left[t_{f_{T_{\min}}} \quad t_{f_{T_{\max}}}\right] \tag{6.6}$$

Criteria 7: Equation (6.6) can be resolved by choosing the optimum complete steering time t_{f_T} over a set of an acceptable range constrained by minimum and maximum limit. The domain of t_{f_T} fulfills the constraint system of concerning the input control voltage and the acceleration detailed in equation (4.6) regarding to each section of the complete trajectories. The maximum voltage u_m placed on the terminals input of the DC motors concerning the complete trajectory considered utilizing the subsequent formula:

$$u_m = \max\left\{u_{m1}, u_{m2}, \cdots, u_{mj}\right\} \tag{6.7}$$

In equation (6.7), u_{mi} is the extreme voltage input functional on the terminal inputs of the DC motors concerning the section i of the complete trajectory. Moreover, u_{mi} able to be derived utilizing equation (4.5). The complete steering time $t_{f_{T_u}}$regarding the extreme input voltage u_m attaining the extreme bound $u_{\max}$ can be identified by applying the a root-finding bisection technique. Moreover, the complete steering time $t_{f_{T_a}}$ regarding $\ddot{x}^2+\ddot{y}^2$ attaining the bound $\alpha_{\max}^{\mathbf{2}}$ can also be calculated by applying the root finding bisection technique. The minimum limit $t_{f_{T_{\min}}}$concerning the free variable t_{f_T} can be calculated utilizing the following formula:

$$t_{f_{T_{\min}}} = \max\left\{t_{f_{T_u}}, t_{f_{T_a}}\right\} \tag{6.8}$$

Criteria 8: The optimum optimization problem stated in Equation (6.6) can be considered as a examine for the final time t_{f_T} that minimizes the cost function C. A lowest limit $t_{\mathrm{f_T}_{\min}}$ for the variable $t_{\mathrm{f_T}}$ ensures that the system of constraint still conserved. Extreme limit $t_{\mathrm{f_T}_{\max}}$is selected on the basis of experience. In order to find an suitable solution concerning the optimum final time $t_{\mathrm{f_T}}$ that reduces the cost function C in equation (6.6), a golden-section search technique can be certainly applied since the energy is a monotonic function concerning the variable $t_{\mathrm{f_T}}$ changing inside the minimum and maximum limits.

6.4 Numerical Simulations and Experiments

The simulation designated in this chapter exhibits the efficiency of the optimal obstacle avoidance trajectory generation proposed method. The test presented involves navigation from a preliminary position to the final position concerning a SAOWMR that utilizes three orthogonal special wheels while avoiding a system of obstacles. The SAOWMR characteristics given in the Table 5.1. The preliminary and target positions are expressed in Table 6.1.

Table 6.1: The configurations of the starting point and the target

Starting Point Configurations	**Ending Point Configurations**
$\mathrm{Z}(0) = \begin{bmatrix} 9 \\ 2 \\ 0 \end{bmatrix}, \dot{Z}(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$	$\mathrm{Z}(\mathrm{t_f}) = \begin{bmatrix} 1 \\ 8.5 \\ \pi \end{bmatrix}, \dot{\mathrm{Z}}(\mathrm{t_f}) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

A system of obstacles composed of four round-shaped of 0.09 m radius obstacles distributed in a 10 × 10 m field given in the following table:

Table 6.2: The Obstacles Configurations

Obstacle 1	**Obstacle 2**	**Obstacle 3**	**Obstacle 4**
(x_{o1},y_{o1})=(2,6)	(x_{o2},y_{o2})=(4,4)	(x_{o3},y_{o3})=(6,8)	(x_{o4},y_{o4})=(7,5)

6.4.1 Motion Planning using APF method

The Artificial Potential Field (APF) method will be utilized in this section as a motion planner concerning the SAOWMR with 3 omnidirectional wheels since it has full omnidirectional motion proficiency and provides three independent motion directions on a flat surface so that it maneuvers and navigates in crowded environment. The following table demonstrates the parameter used to generate the free-collide path using the Artificial Potential Field method.

Table 6.3: Artificial Potential Field Parameters

Number of the artificial particles	60
The radius of the circle constructing the artificial points	0.09
The depth of the attractant σ_T	1
The depth of the repellant σ_o	1
Width of the attractant μ_T	0.1
Width of the repellant μ_o	1.9

The Mobile Robot's starting point will be assigned a high potential and the goal will be assigned a low potential so that the robot maneuvers from the highest potential to the lowest potential. The obstacles will be given high potential so that repulsive force acting between the robot and the obstacles. The combined Artificial Potential Field which is an arrangement of attractive and repulsive potential concerning the considered numerical test and based on the given artificial potential field parameters in Table 6.3 will be demonstrated in the subsequent figure:

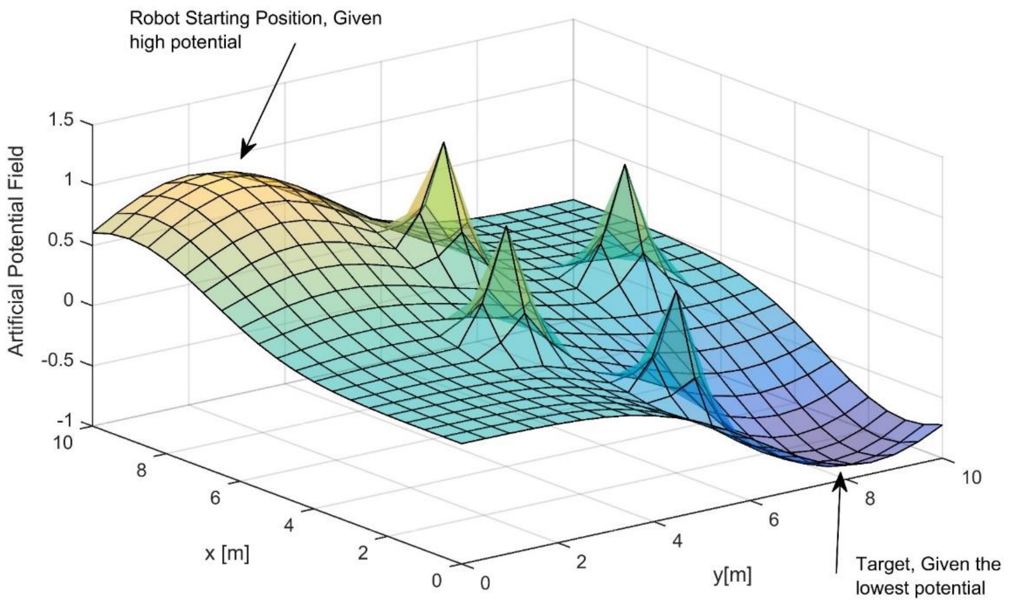


Figure 6.2: The combined Artificial Potential Field

The path produced on the basis of the APF method between the robot preliminary position and the target corresponding to the APF parameters in Tables 6.3 exposed in the subsequent figure. Matlab code provided in the Appendix H so that the APF method utilized to plan the motion between the initial and final configuration concerning the mentioned numerical experiment.

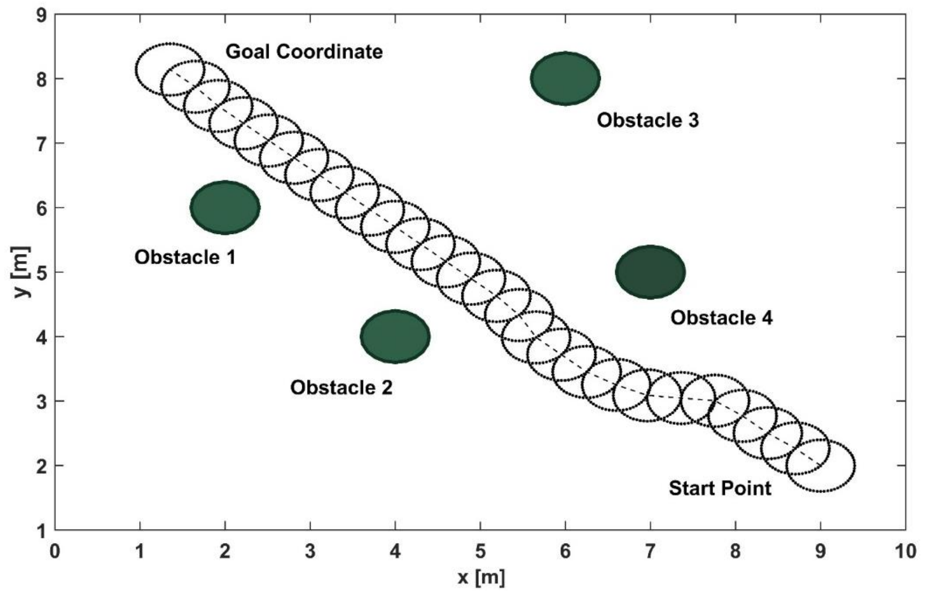


Figure 6.3: Motion Planning based on the APF method

A specific number of correspondingly organized via points will be selected along the produced path that guarantees avoiding the obstacles are demonstrated in the following figure:

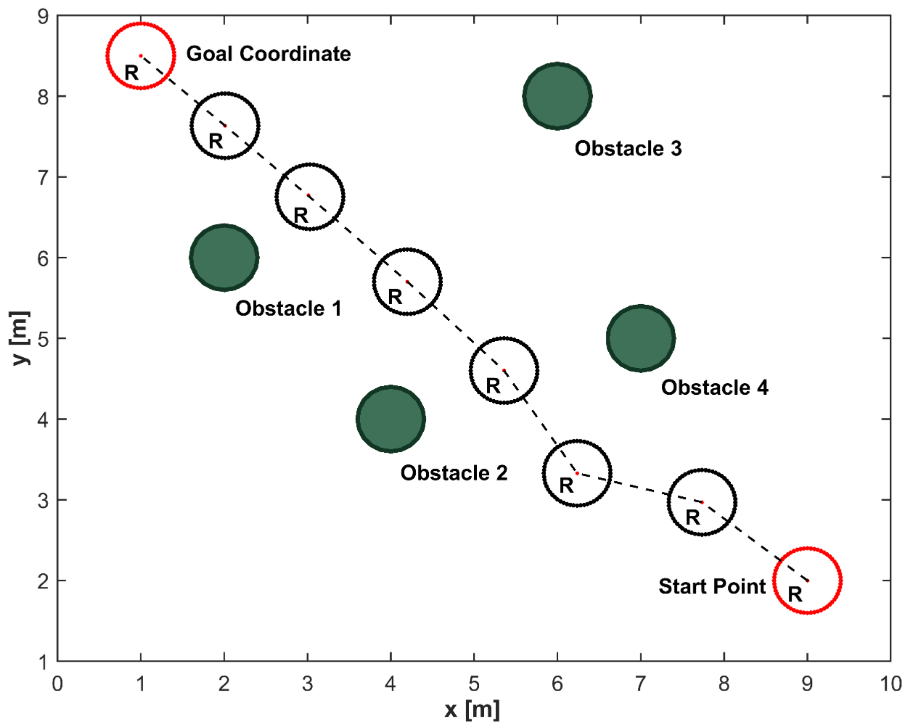


Figure 6.4: Organized path produced by selecting several via points along the path

The cubic spline interpolation concerning the potential value of the overall steering time will be used in order to assign proper velocity values at the midpoints. Hence, the third-order polynomial characterizes the trajectory between the midpoints along the produced path. Figures 6.5 and 6.6 validate noticeably how the proposed method solve optimal control problem so that the collision-free optimal trajectory will be produced. In Appendix I, matlab code written to generate the free collision optimal trajectory between the initial and the final configuration so that the optimal trajectory generation method will be integrated with the path constraints by utilizing the method proposed in this chapter.

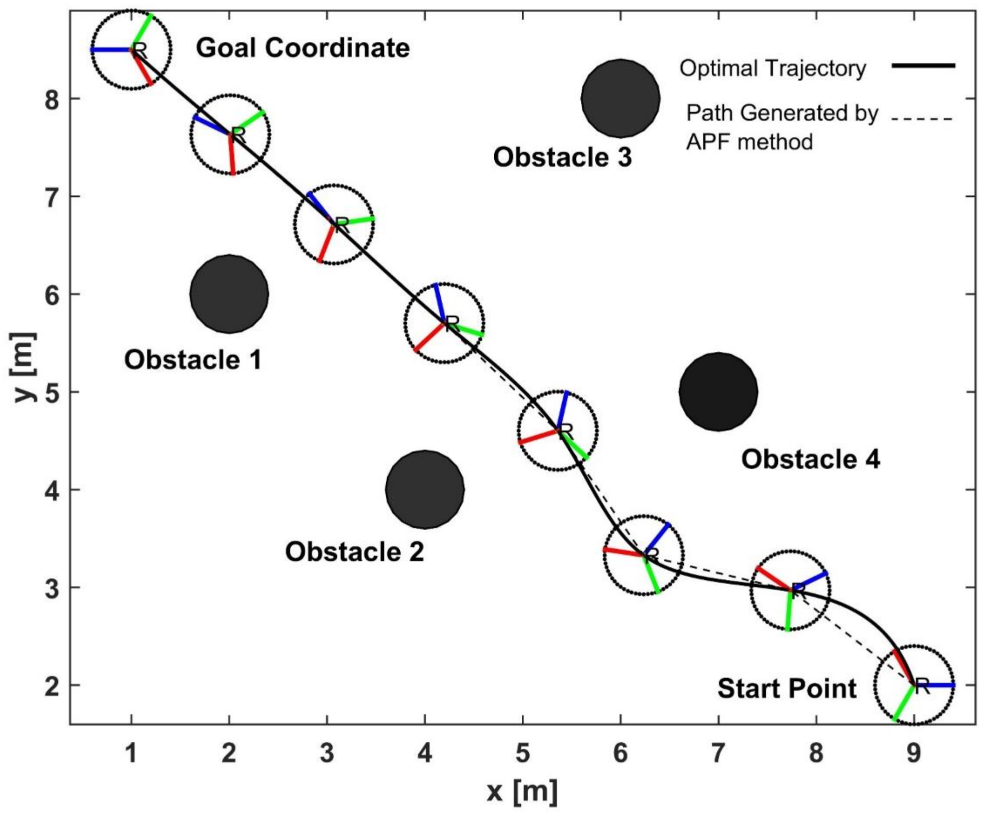


Figure 6.5: Demonstration of collide free optimal trajectory produced between preliminary and termination configurations using the obstacle avoidance proposed method.

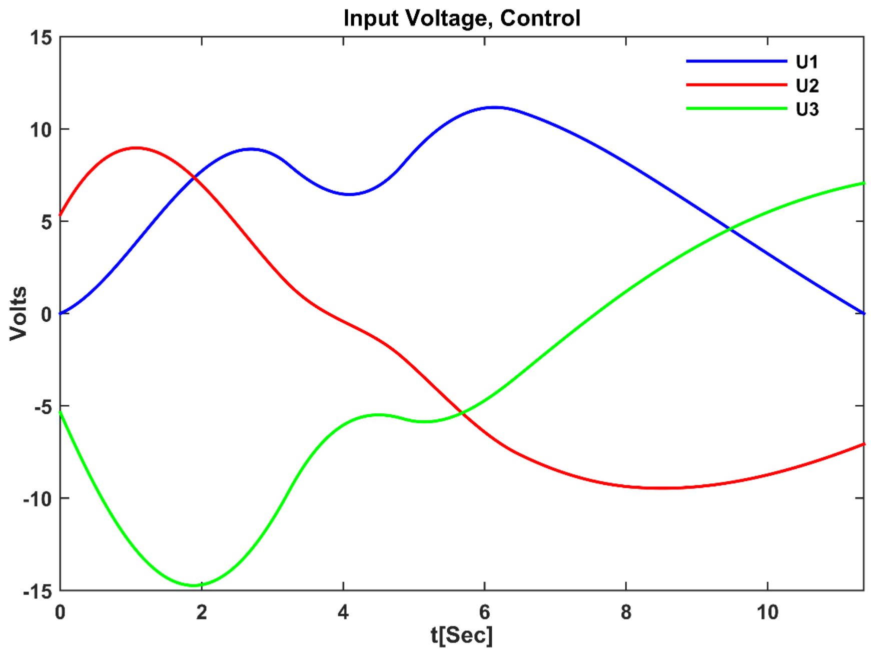


Figure 6.6: Control input voltages of the robot wheels produced by the proposed algorithm to be used by the robot controller.

Figure 6.5 demonstrates how the SAOWMR that utilizes three omnidirectional special wheels effectively transform from the primary position to the final position, passing over all the midpoints while avoiding satisfactorily all the obstacles. Furthermore, Figure 6.6 shows the voltage input of the omnidirectional special wheels that should be followed and utilized by the controller of the mobile robot. The method proposed obviously illustrates the fluency of the motion control, as well as the fulfillment of the mobile robot dynamic system and the restrictions of the path. Moreover, the obstacle avoidance proposed method attained a pretty good outcomes concerning the time execution so that with the range of milliseconds needed for producing the optimum trajectory.

Chapter 7

SENSOR FUSION FOR BALL POSITION ESTIMATION

7.1 Introduction

In robotics field, the sensors are compulsory for the autonomous robot and without them, the robot is, in principle, sightless and deafening. Sensors allow the robot to gather data from the environment in order to perform an autonomous task such as navigation, tracking objects or obstacle avoidance.

This chapter will highlight the Multi Sensor Fusion procedures in the framework of the omnidirectional wheeled mobile robot soccer performing ball tracking and estimating of the ball position in dynamic environment.

In RoboCup environment, an accurate and reliable estimate of the ball position is considered as one of the most essential topics for outstanding team play so that a cooperative and coordination task could be achieved successfully.

Normally, a mobile robot performing tasks in robotic soccer environment can only sense an incomplete part of its surroundings at a specific moment of time. In dynamic environment, sensors as cameras, laser range finder or even ultrasonic sensors are unable to cover the entire area around the robot due to the limited sight of view. As an example, a robot may not observe a ball that is occluded by other robot. Moreover, information formerly extracted concerning currently invisible objects of the environment becomes in general inaccurate and unreliable. Furthermore, a moving objects maneuver in dynamic environment can’t be reliably tracked by a sensors placed on a single robot. To overcome these limits, a single robot may share its sensors information with the other robots in a way that a module of a multi-agent sensor fusion constructed for this purpose. Subsequently, a robot can extend its own view of the environment to a global vision by extracting information from the sensor fusion module. Sharing information among the mobile robots can rise the visibility effective of the environment, permitting for more precise modeling and more proper responses. A more consistent estimation can lead to rise the team play and the overall playing performance by integration a single perception of the mobile robots into a comprehensive perception.

One can differentiate between the global sensor fusion and local sensor fusion. Local sensor fusion typically integrates estimations from several asynchronous or synchronous placed on a single mobile robot such as in [87-90]. The global sensor fusion denotes to integrating the estimation from several robots. Recently, most of the sensor fusion approaches focusing on merging perceptions globally to form one sensing module ensures one consistent estimate.

The domain of the soccer robots is very interesting domain in the research of the sensor fusion approaches. In this domain, the ball tracking and estimating the position of the ball precisely is one of the significant issues. Most of the research published recently regarding this topic considered a global estimates of the sensor fusion. In this sense, a soccer mobile robot can use the global estimation to localize and perform position tracking the of the ball moving in the field even if the robot itself doesn’t see that ball due to occlusion by other robot. The commonly used methods for global sensor fusion regarding the ball tracking are probabilistic technique as Kalman Filters (KF) [91] or Markov Localization [92].

7.2 Related Work

Numerous research works published in the domain of the soccer robot concentrated on the ball position estimation and tracking so that multiple soccer mobile robots collecting data from their surrounding environment and exchanging that information with a global module for multi-agent sensor fusion. Subsequently, a robot soccer can extend its own vision of the environment by obtaining information from the sensor fusion module.

In [93], the authors considered the case of tracking the ball in the RoboCup setup such that the perceptions estimated by the vision sensors are both unreliable and noisy. In this scenario, the vision sensor of the mobile robot is capable of estimating the heading of the ball with respectable precision, but it is unsuccessful to estimate a precise location of the ball in the field particularly if the ball far from the mobile robot. The method proposed to provide tracking and estimation of the ball position depends on the arrangement of the KF and a Markov process. This procedure utilized the Kalman Filter as a sensor fusion algorithm so that each mobile robot shares its perception about the location of the ball to the global sensor fusion module. To generate additional precise and robustness estimation consequences, the paper proposed to utilize the Markov process as a surveillance filter for the KF.

In [94], a method proposed for fusing distributed, uncertain and noisy perceptions gathered from multiple observations concerning tracking and position estimation of a single object. This method validated and tested in RoboCup environment regarding the ball position estimation and tracking. The proposed method based on representing each of the ball perception in a two-dimensional statistical space called observation space. Moreover, each point in that space constructed by two information pieces given in canonical form Gaussian distribution. Based on this representation system, fusing the perceptions coming from sensors located on multiple mobile robots can be done easily by multiplying the points representing those perceptions in the observation space. The research paper suggested to use the Kalman Filter regarding the process of predicting of the ball position.

In [95], The Monte Carlo algorithm [96] is used as a local sensor data fusion to incorporate the estimations of the moving ball from several sensors located in a single robot designed by Mostly Harmless team so that the data from different sensors is merged and integrated into a local world module. Subsequently, a global sensor data fusion module can be constructed by integrating several local world modules together.

In [97], the Milan RoboCup Team considered so that the global sensors data fusion done by utilizing the fuzzy logics concerning. The information estimated by the mobile robots' sensors are characterized representatively and anchored with items from the surrounding environment by the use of an anchoring approach [98]. Subsequently, the perceptions data integrated by the fuzzy logic approach.

In [99], a localization system is proposed based on the odometry calculation and global vision data fusion. The method implemented on the soccer mobile robot performing tasks in RoboCup environment so that the ball position estimating is considered. In this research, a least squares sense utilized for the global sensors data fusion on the basis of the EKF.

7.3 Global Sensor Fusion Methods

In this section, a summary provided concerning the most techniques used for global sensor data fusion starting with simple approaches such as mean techniques to more complicated like Kalman Filter and Monte Carlo method.

7.3.1 Arithmetic Mean

Supposing that a mobile robot maneuvers in dynamic environment. In each time step, the robot contributes its perception about the ball position $\left(\mathrm{b_x},\mathrm{b_y}\right)_{(l)(\mathrm{i})}$ and send this estimation to the central server. The global perception of the ball position $\left(\mathrm{b_x},\mathrm{b_y}\right)_{(g)}$ assuming to be the average of those estimations approaching from each mobile robot.

$$\left(\mathrm{b_x},\mathrm{b_y}\right)_{(g)} = \left(\sum_{\mathrm{i}=1}^{n}\left(\mathrm{b_x}\right)_{(l)(i)},\sum_{\mathrm{i}=1}^{n}\left(\mathrm{b_y}\right)_{(l)(i)}\right)_{(g)} \tag{7.1}$$

The number of the mobile robots sharing their estimations about the ball position is n. Here, every mobile robot' contribution about the position of the ball has equal importance.

7.3.2 Weighted Arithmetic Mean

The perception of a mobile robot about the position of the ball must be weighted concerning the distance among the mobile robot and the ball since the vision sensor is unreliable if the mobile robot is far from the ball. Subsequently, in an adapted version of the Arithmetic mean technique, the estimation of the ball position will be weighted concerning the distance between the ball and the mobile robot besides a time factor which refers to the period of time that the mobile robot has detected the ball:

$$\left(\mathrm{b_x},\mathrm{b_y}\right)_{(g)} = \left(\frac{1}{\sum_{i=1}^{n}\chi_i} \sum_{i=1}^{n}\left(\chi_i\left(\mathrm{b_x}\right)_{(l)(i)}\right), \frac{1}{\sum_{i=1}^{n}\chi_i} \sum_{i=1}^{n}\left(\chi_i\left(\mathrm{b_y}\right)_{(l)(i)}\right) \right)_{(g)} \tag{7.2}$$

With

$$\chi_i = \frac{1}{d_i} conf_i \tag{7.3}$$

In equation (7.3), $conf_i$ is the confidence afforded by the vision and localization system so that some means of importunity assigned to each ball estimation. This confidence factor helps to stabilizes the positional tracking of the ball. The distance between the position estimated of the ball and the position of the mobile robot given as d_i.

7.3.3 Weight Grid

This technique utilizes a grid system to symbolize the estimated ball position on the playing field so that each cell in the grid can take a weight. Supposing that an estimated ball position by a mobile robot performed, a weight function assigns a weight to this estimation based on (7.3). Subsequently, a two dimensional Gaussian distribution is obtained regarding each weighted estimation in such a way that the parameters of the Gaussian specified by the vision system. Furthermore, the results deposited in the corresponding cell in the grid system.

7.3.4 Monte Carlo Localization

Monte Carlo Localization method was proposed in [96] as an enhanced version of the Markov localization method [92]. The key concept behind the Markov Localization method is to assign a probability concerning every potential position in a sample space of an occupancy grid. Bayes’ rule is utilized to re assign the probability distribution in every new sensor reading. The Monte Carlo Localization updated and improved the

Markov Localization method by utilizing a weighted sample as an alternative of the occupancy grid that utilized in Markov Localization.

Regarding the global sensor fusion, the method proposed to define samples that represent the estimated ball positions so that an important factor assigned to each sample. In every iteration of this method, a number of samples selected by chance based on their weights so that a sample assigned a higher weight is more likely to be selected than other with lower weight. Furthermore, some samples are produced near the locally estimated ball position. Subsequently, two addition steps are added: the first step by applying a motion model to the selected samples based on the ball velocity. In the second step, based on the measurements error, the samples with the new local estimated ball position are re-weighted.

7.3.5 Kalman Filter Method

A KF is an optimal estimation process utilizes to approximate a state vector of a system in such a way that a measures subject to noise from different sensors can be combined and utilize. This method designed to find the optimal state observers so that it can predict with the optimality the parameter of interests such as the speed, location, or direction in the existence of the noisy measurements.

The following model expressed in (7.3) characterizes the state space of the dynamic system:

$$\begin{aligned} \mathrm{X}[k] &= \mathrm{A}_1\mathrm{X}[k-1] + \mathrm{A}_2\mathrm{U}[k] + \nu_1[k] \\ \mathrm{Y}[k] &= \mathrm{A}_3\mathrm{X}[k] + \nu_2[k] \end{aligned} \tag{7.4}$$

The vector state of the system in the present sample time k is given in (7.4) as X[k]. The output of the system is Y[k]. Moreover, U[k] is the input of the system in the present sample time k. Furthermore, A_1, A_2 and A_3 are numerical matrices. $\nu_1[k]$and

$\nu_2[k]$ are random variables expected to be drawn from a zero mean Gaussian distribution represent the process noise and the measurements noise respectively.

$$\begin{aligned} \nu_1 &\sim N\left(0, \sigma_{\nu_1}{}^2\right) \\ \nu_2 &\sim N\left(0, \sigma_{\nu_2}{}^2\right) \end{aligned} \tag{7.5}$$

In (7.5), $\sigma_{\nu_1}{}^2$ and $\sigma_{\nu_2}{}^2$are respectively the covariance for the random variables $\nu_1[k]$and $\nu_2[k]$ of the measurements noise and the process noise.

To explain the working principle of the Kalman Filter, consider the following mathematical model of the dynamic system:

$$\begin{aligned} \mathrm{X}[k] &= \mathrm{A}_1\mathrm{X}[k-1] + \mathrm{A}_2\mathrm{U}[k] \\ \mathrm{Y}[k] &= \mathrm{A}_3\mathrm{X}[k] \end{aligned} \tag{7.6}$$

Now, if an input implemented to the system, the system state and the output able to be estimated. However, the mathematical model is only an approximation of the actual system due to the system uncertainties and process noise. The mathematical model of the system in (7.6) can be rewritten to refer to the model prediction of the dynamic system to be as following:

$$\begin{aligned} \hat{\mathrm{X}}[k] &= \mathrm{A}_1\hat{\mathrm{X}}[k-1] + \mathrm{A}_2\mathrm{U}[k] \\ \hat{\mathrm{Y}}[k] &= \mathrm{A}_3\hat{\mathrm{X}}[k] \end{aligned} \tag{7.7}$$

In (7.7), $\widehat{\mathrm{Y}}[k]$ and $\widehat{\mathrm{X}}[k]$ refer respectively to the estimated output and estimated state. On the other hand, if a sensor implemented to measure the real output $\mathrm{Y}[k]$ of the dynamic system given in (7.4). The measured output won't be accurate due to the sensor noise. Kalman Filter calculates the optimal estimation of the state with the minimal variance by multiplying two pieces of information, the prediction and the measurement probability density function together. Computationally, the multiplication of the prediction and the measurement probability density functions given in the following relation:

$$\hat{X}[k] = A_1\hat{X}[k-1] + A_2U[k] + K_k\left(Y[k] - A_3\left(A_1\hat{X}[k-1] + A_2U[k]\right)\right) \tag{7.8}$$

The Kalman Filter gain given in (7.8) as K_k. The Equation (X.8) can be reformulated by the utilization of the priori state estimation X[k]

$$\hat{X}[k] = \hat{X}^{-}[k] + K_k\left(Y[k] - A_3\hat{X}^{-}[k]\right) \tag{7.9}$$

With

$$\hat{X}^{-}[k] = A_1\hat{X}[k-1] + A_2U[k] \tag{7.10}$$

In Equation (7.10), $\hat{X}^{-}[k]$ is the priori state estimation X[k]as it calculated before the current measurement is taken. $\hat{X}[k]$ in Equations (7.9) and (7.10) refers to the posteriori estimation of the state X[k] as it estimated after the current measurement is taken. Kalman Filter is two step algorithm. The first step refers to the prediction process so that the system model is utilized to calculate the priori estimation of the state $\hat{X}^{-}[k]$ and the priori error covariance matrix $P^{-}[k]$. The error covariance matrix P able to be thought as a measure of uncertainty of the estimated state $\hat{X}$. The Kalman Filter prediction equations given as following:

$$\hat{X}^{-}[k] = A_1\hat{X}[k-1] + A_2U[k] \tag{7.11}$$

$$P^{-}[k] = A_1P[k-1]A_1^{T} + Q_1 \tag{7.12}$$

In (7.12), the matrix Q_1 is the covariance matrix of the random variable ν_1 that represent the process noise. In the first iteration of the algorithm, an initial estimates (guess) for $\hat{X}[k\text{-}1]$ and P[k-1] are needed. For the second step of the algorithm refers to the update process. In this step, the algorithm utilizes the priori state estimation $\hat{X}^{-}[k]$ and the priori error covariance matrix $P^{-}[k]$ calculated in the step of the prediction and updates them. The equations that represent the update process are assumed in the subsequent equations:

$$\mathrm{K}[k] = \frac{\mathrm{P}^{-}[k]\mathrm{A}_3{}^{\mathrm{T}}}{\mathrm{A}_3\mathrm{P}^{-}[k]\mathrm{A}_3{}^{\mathrm{T}} + Q_2} \tag{7.13}$$

$$\hat{\mathrm{X}}[k] = \hat{\mathrm{X}}^{-}[k] + \mathrm{K}[k]\left(\mathrm{Y}[k] - \mathrm{A}_3\hat{\mathrm{X}}^{-}[k]\right) \tag{7.14}$$

$$\mathrm{P}[k] = \left(\mathrm{I} - \mathrm{K}[k]\mathrm{A}_3\right)\mathrm{P}^{-}[k] \tag{7.15}$$

In (7.13), the matrix Q_2 is the covariance matrix of the random variable ν_2 that represent the sensor noise. The Kalman gain K is considered such that it reduces the posteriori error covariance P. The Kalman gain K decides how deeply the measurement of the Y[k] and priori state estimation $\widehat{\mathrm{X}}^{-}$[k] contributes to the calculation of the posteriori estimation of the state $\widehat{\mathrm{X}}$[k]. Subsequently, If the measurement noise is small, the measurement is trusted more and contributes to the calculation of the $\widehat{\mathrm{X}}$[k] more than the priori state estimation $\widehat{\mathrm{X}}^{-}$[k] does and the vice versa.

Once the update equations are calculated, the posteriori state estimation $\widehat{\mathrm{X}}[k]$ are utilized to calculate the new priori state estimation $\widehat{\mathrm{X}}^{-}[k]$ and the algorithm repeats itself.

A sensor fusion is one of the most important characteristics of the KF. The KF equation given in the equation (7.9) able to be given on the basis of the *c* number of measurements and *n* number of states so that the matrices' dimension would be change as shown:

$$\hat{X}[k]_{[n\times 1]} = \hat{X}^{-}[k]_{[n\times 1]} + K_{k\ [n\times c]}\left(Y[k]_{[c\times 1]} - A_{3\ [c\times n]}\hat{X}^{-}[k]_{[n\times 1]}\right) \tag{7.16}$$

7.4 Ball Tracking and Position Estimation using Kalman Filter

Although many sensor fusion techniques have been utilized and have given very good results in the literatures of several areas. The Kalman filter method still one of the most efficient and powerful method and compete other more complicated method in the domain of the ball tracking in dynamic environment [100]. This section highlights the problem of the Multi Sensor Fusion utilizing the KF in the context of the mobile robot soccer performing ball position estimation and tracking in RoboCup environment.

Supposing the case of a ball moving in the field of a flat surface. Assuming that the ball position given as $(x_{\mathrm{b}}, y_{\mathrm{b}})$ in terms of the global coordinate system and assuming that $(v_{x_{\mathrm{b}}}, v_{y_{\mathrm{b}}})$ is the corresponding velocity components of the ball. The mathematical model of the ball given in the following system as a discrete representation of the state space supposing that the ball is rolling without slipping in a straight line trajectory with ignorance of the ball mass:

$$\begin{aligned} x_{\mathrm{b}}[k] &= x_{\mathrm{b}}[k-1] + v_{x_{\mathrm{b}}}[k-1]\Delta t \\ y_{\mathrm{b}}[k] &= y_{\mathrm{b}}[k-1] + v_{y_{\mathrm{b}}}[k-1]\Delta t \\ v_{x_{\mathrm{b}}}[k] &= v_{x_{\mathrm{b}}}[k-1] \\ v_{y_{\mathrm{b}}}[k] &= v_{y_{\mathrm{b}}}[k-1] \end{aligned} \tag{7.17}$$

In (7.17), Δt is the time step. The state of the ball in a sample time k given as following:

$$X[k] = \begin{bmatrix} x_{\mathrm{b}}[k] & y_{\mathrm{b}}[k] & v_{x_{\mathrm{b}}}[k] & v_{y_{\mathrm{b}}}[k] \end{bmatrix}^{\mathrm{T}} \tag{7.18}$$

The state of the ball $X[k]$ in the sample time k can be given in terms of the state of the ball $X[k\text{-}1]$ in the previous sample time k-1 based on (7.17) and (7.18) as following:

$$X[k] = \begin{bmatrix} x_b[k] \\ y_b[k] \\ v_{x_b}[k] \\ v_{y_b}[k] \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_b[k-1] \\ y_b[k-1] \\ v_{x_b}[k-1] \\ v_{y_b}[k-1] \end{bmatrix} = A_1 X[k-1] \tag{7.19}$$

For the precisely tracking of the ball, the global sensor data fusion will be used so that each robot i sends its observation of the ball to the global observation unit. But, the vision sensor generally is competent to estimate the heading of the ball with respectable precision and fails to afford precise range data particularly if the ball is far from the mobile robot. The range and the heading of the ball concerning the mobile at the configuration $(x_{rob\,i}, y_{rob\,i})$ given respectively as follows:

$$r_b = \sqrt{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2} \tag{7.20}$$

$$\phi_b = \tan^{-1}\left(\left(y_b - y_{rob\,i}\right) / \left(x_b - x_{rob\,i}\right)\right) \tag{7.21}$$

The matrix Q_1 of the covariance matrix of the random variable v_1 that represent the sensor noise can't be a constant matrix since the accuracy of the sensor is depending on the distance between the ball and the mobile robot.

$$Q_1 = \begin{bmatrix} r_b \sigma_{r_b}{}^2 & 0 \\ 0 & \sigma_{\phi_b}{}^2 \end{bmatrix} \tag{7.22}$$

Based on the development above, the measurements' functions of the system output Y[k] given as follows:

$$r_b = f_1(X) = \sqrt{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2} \tag{7.23}$$

$$\phi_b = f_2(X) = \tan^{-1}\left(\left(y_b - y_{rob\,i}\right) / \left(x_b - x_{rob\,i}\right)\right) \tag{7.24}$$

Moreover, the 2×4 measurement matrix A_2 can be derived from the Jacobian of the functions given in the (7.23) and (7.24) as follows:

$$A_2 = \begin{bmatrix} \frac{df_1}{dx_b} & \frac{df_1}{dy_b} & \frac{df_1}{dv_{x_b}} & \frac{df_1}{dv_{y_b}} \\ \frac{df_2}{dx_b} & \frac{df_2}{dy_b} & \frac{df_2}{dv_{x_b}} & \frac{df_2}{dv_{y_b}} \end{bmatrix} =$$

$$\begin{bmatrix} \frac{x_b - x_{rob\,i}}{\sqrt{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2}} & \frac{y_b - y_{rob\,i}}{\sqrt{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2}} & 0 & 0 \\ \frac{y_b - y_{rob\,i}}{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2} & \frac{x_b - x_{rob\,i}}{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2} & 0 & 0 \end{bmatrix} \tag{7.25}$$

Subsequently, the mathematic modeling of the ball rolling in the field of surface flat given as following

$$\begin{aligned} X[k] &= A_1 X[k-1] \\ Y[k] &= A_2 X[k] \end{aligned} \tag{7.26}$$

The matrices A_1 and A_2 are given in (7.19) and (7.25) respectively. The Kalman Filter method is two-step process. The first step is the prediction step. The Kalman Filter prediction equations concerning the mathematical model in (7.26) given as follows

$$\hat{X}^-[k] = A_1 \hat{X}[k-1] \tag{7.27}$$

$$P^-[k] = A_1 P[k-1] A_1^T + Q_1 \tag{7.28}$$

Moreover, the equations that represent the update process are given in the following equations:

$$K[k] = \frac{P^-[k] A_2^T}{A_2 P^-[k] A_2^T + Q_2} \tag{7.29}$$

$$\hat{X}[k] = \hat{X}^-[k] + K[k]\left(Y[k] - A_2 \hat{X}^-[k]\right) \tag{7.30}$$

$$P[k] = \left(I - K[k] A_2\right) P^-[k] \tag{7.31}$$

With

$$A_1 = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} \dfrac{x_b - x_{rob\,i}}{\sqrt{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2}} & \dfrac{y_b - y_{rob\,i}}{\sqrt{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2}} & 0 & 0 \\ \dfrac{y_b - y_{rob\,i}}{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2} & \dfrac{x_b - x_{rob\,i}}{\left(x_b - x_{rob\,i}\right)^2 + \left(y_b - y_{rob\,i}\right)^2} & 0 & 0 \end{bmatrix} \qquad (7.32)$$

7.4.1 Experimental Result

Supposing that a ball moving in the field of a flat surface with the ignorance of its mass. This section highlights the experimental result concerning the problem of the Multi Sensor Fusion using the Kalman filter.

Table 7.1 summarizes all the parameters of experimental setup presented concerning ball tracking algorithm using Kalman Filter.

Table 7.1: The Parameters selected for the experimental setup concerning the ball tracking using Kalman Filter

The Parameter description	**The parameter symbol**	**The parameter value**
The standard deviation of the random variable v_1 which represents the process noise	σ_{v_1}	0.01
The standard deviation of the random variable v_2 which represents the measurement noise	σ_{v_2}	0.01
Number of samples	Ns	50
Time step	Δt	0.05 sec
Maneuvering time	t_f	2.5 sec
Ball Actual Initial Location	(x_b, y_b)	(0.4,0.6)
Ball Actual Initial Velocity	(v_{x_b}, v_{y_b})	(-1,-1) m/sec

In the first iteration of the algorithm, an initial estimates (guess) for $\widehat{X}[k\text{-}1]$ and $P[k\text{-}1]$ are needed. In the experimental process, the initial location of the ball will be considered with small uncertainty. An arbitrarily guess with a large uncertainty will be assigned to the ball initial velocity. Table 7.2 expresses the initialization of the KF process.

Table 7.2: The Initialization of the KF process

The Parameter description	**The parameter symbol**	**The parameter value**
Ball guess Initial Location	$\left(\hat{x}_b[k\text{-}1]\,,\hat{y}_b[k\text{-}1]\right)$	$\left(x_b + 0.01 * r_n\,, y_b + 0.01 * r_n\right)$
Ball guess Initial Velocity	$\left(\hat{v}_{x_b}[k\text{-}1], \hat{v}_{y_b}[k\text{-}1]\right)$	$\left(v_{x_b} + 0.5 * r_n\,, v_{y_b} + 0.5 * r_n\right)$
The initial error covariance matrix	$P[k\text{-}1]$	$\begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix}$

In Table 7.2, r_n is a normally distributed pseudorandom number.

7.4.1.1 Case Study 1: Single OWMR Robot Observer

Supposing that a single three-wheeled omnidirectional mobile robot with a vision sensor located on the top of the mobile robot attempting to estimate the range and the heading of the ball given in equations (7.20) and (7.21). The vision sensor generally is competent to estimate the heading of the ball with respectable precision and fails to afford precise range data particularly if the robot is ball far from the mobile robot. It is proposed to use the third-order polynomial to characterize the trajectory of the mobile robot that fulfills the requirement of the given four boundary conditions corresponding to the dynamic system of the OWMR. The third-order polynomial trajectory corresponding to the boundary conditions given in the following equation:

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} t^3 + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} t^2 + \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} t + \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} \tag{7.33}$$

The polynomial coefficients stated based on the boundary conditions of the positions and the velocities concerning the initial and final omnidirectional wheeled mobile robot state given to be as follows:

$$\begin{cases} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \dfrac{1}{t_{\mathrm{f}}^{\,2}} \left(-\dfrac{2}{t_{\mathrm{f}}} \left(\begin{bmatrix} x_{\mathrm{f}} \\ y_{\mathrm{f}} \\ \theta_{\mathrm{f}} \end{bmatrix} - \begin{bmatrix} x_{\mathrm{o}} \\ y_{\mathrm{o}} \\ \theta_{\mathrm{o}} \end{bmatrix} \right) + \begin{bmatrix} \dot{x}_{\mathrm{o}} \\ \dot{y}_{\mathrm{o}} \\ \dot{\theta}_{\mathrm{o}} \end{bmatrix} + \begin{bmatrix} \dot{x}_{\mathrm{f}} \\ \dot{y}_{\mathrm{f}} \\ \dot{\theta}_{\mathrm{f}} \end{bmatrix} \right) \\ \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \dfrac{1}{t_{\mathrm{f}}} \left(\dfrac{3}{t_{\mathrm{f}}} \left(\begin{bmatrix} x_{\mathrm{f}} \\ y_{\mathrm{f}} \\ \theta_{\mathrm{f}} \end{bmatrix} - \begin{bmatrix} x_{\mathrm{o}} \\ y_{\mathrm{o}} \\ \theta_{\mathrm{o}} \end{bmatrix} \right) - 2 \begin{bmatrix} \dot{x}_{\mathrm{o}} \\ \dot{y}_{\mathrm{o}} \\ \dot{\theta}_{\mathrm{o}} \end{bmatrix} - \begin{bmatrix} \dot{x}_{\mathrm{f}} \\ \dot{y}_{\mathrm{f}} \\ \dot{\theta}_{\mathrm{f}} \end{bmatrix} \right) \\ \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} \dot{x}_{\mathrm{o}} \\ \dot{y}_{\mathrm{o}} \\ \dot{\theta}_{\mathrm{o}} \end{bmatrix}, \quad \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} x_{\mathrm{o}} \\ y_{\mathrm{o}} \\ \theta_{\mathrm{o}} \end{bmatrix} \end{cases} \tag{7.34}$$

In Table 7.3, the boundary conditions stated concerning the trajectory of the mobile robot that performing ball tracking given in SI units as follows:

Table 7.3: The boundary conditions of the trajectory concerning the mobile robot performing ball tracking

Preliminary Arrangements	**End Arrangements**
$Z(0) = \begin{bmatrix} 1 \\ 1 \\ \pi/4 \end{bmatrix}, \ \dot{Z}(0) = \begin{bmatrix} 0.1 \\ -0.5 \\ 0.2 \end{bmatrix}$	$Z(t_{\mathrm{f}}) = \begin{bmatrix} 0.5 \\ -1.5 \\ -\pi/2 \end{bmatrix}, \dot{Z}(t_{\mathrm{f}}) = \begin{bmatrix} -0.8 \\ -0.1 \\ 0.4 \end{bmatrix}$

The expectation is that, the observations will quickly constrain the velocities and the covariance matrix. This is not guaranteed, because we have only measurements from the single mobile robot and four variables of the state. The following figure shows the time history and the Kalman Filter estimation of the state vector $[x\ y\ \mathrm{V}x\ \mathrm{V}y]^{\mathrm{T}}$ of the ball concerning only single robot observes the motion of the ball. Figure 7.2 shows how the mobile robot track the ball while it is maneuvering in dynamic environment using Kalman filter for the optimality of the estimation.

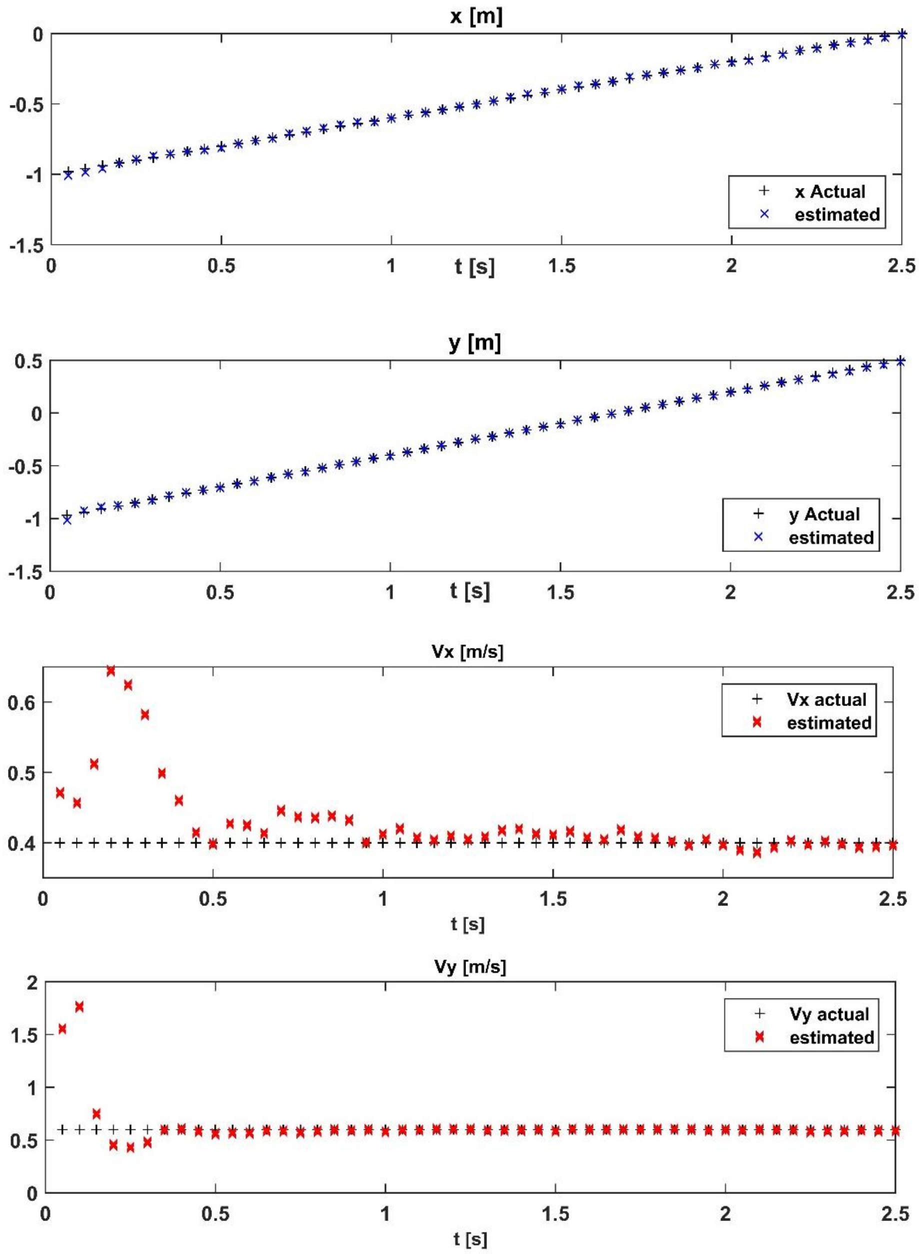


Figure 7.1: The time history and the Kalman Filter estimate of the vector state $[x\ y\ Vx\ Vy]^T$

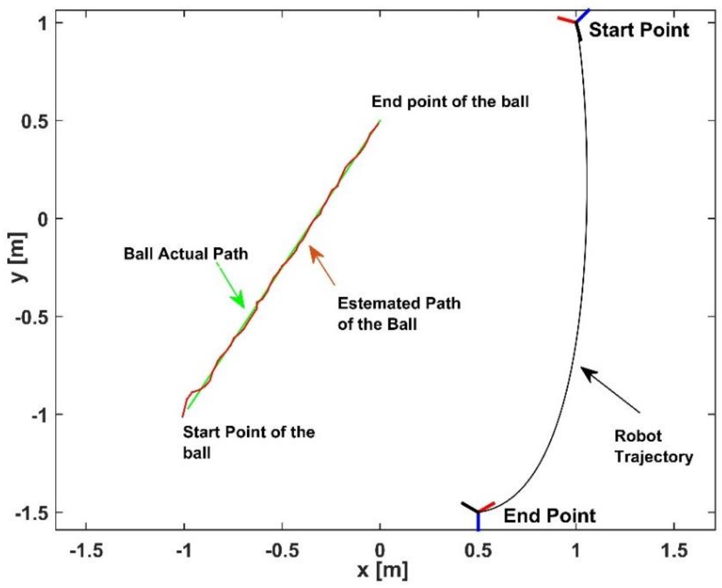


Figure 7.2: Illustration of how the robot track the ball while it is moving in dynamic environment using Kalman Filter

7.4.1.2 Case Study 2: Multiple OWMR Robot Observers

Supposing that three omnidirectional mobile robots with a vision sensors placed on the top of the robots attempting to estimate the range and the heading of the ball so that the aims is to achieve Multi Sensor Fusion using the Kalman filter algorithm for the sensors fusion. In Table 7.4, the boundary conditions stated concerning the trajectories of the mobile robots that performing ball tracking given in SI units as follows:

Table 7.4: The boundary conditions of the trajectory concerning the three mobile robots that performing ball tracking

Movements	Preliminary Arrangements	End Arrangements
Robot 1	$Z(0) = \begin{bmatrix} 1 \\ 1 \\ \pi/4 \end{bmatrix}, \dot{Z}(0) = \begin{bmatrix} 0.1 \\ -0.5 \\ 0.2 \end{bmatrix}$	$Z(t_f) = \begin{bmatrix} 0.5 \\ -1.5 \\ -\pi/2 \end{bmatrix}, \dot{Z}(t_f) = \begin{bmatrix} -0.8 \\ -0.1 \\ 0.4 \end{bmatrix}$
Robot 2	$Z(0) = \begin{bmatrix} 0 \\ 0 \\ -\pi/4 \end{bmatrix}, \dot{Z}(0) = \begin{bmatrix} 0.1 \\ 0.5 \\ -0.2 \end{bmatrix}$	$Z(t_f) = \begin{bmatrix} -0.25 \\ 1.5 \\ \pi/2 \end{bmatrix}, \dot{Z}(t_f) = \begin{bmatrix} -0.8 \\ -0.1 \\ 0.4 \end{bmatrix}$
Robot 3	$Z(0) = \begin{bmatrix} -1.5 \\ 2 \\ \pi/4 \end{bmatrix}, \dot{Z}(0) = \begin{bmatrix} 0.1 \\ -0.5 \\ -0.2 \end{bmatrix}$	$Z(t_f) = \begin{bmatrix} -2 \\ -1 \\ -\pi/6 \end{bmatrix}, \dot{Z}(t_f) = \begin{bmatrix} -0.8 \\ -0.1 \\ 0.4 \end{bmatrix}$

The following figure shows the time history and the Kalman Filter estimation of the state vector $[x\ y\ \mathrm{V}x\ \mathrm{V}y]^{\mathrm{T}}$ of the ball concerning sensor fusion based on Kalman Filter of the three estimations from the sensors placed on the top of the mobile robots maneuvering in dynamic environment. Figure 7.4 shows how the Kalman Filter provide an advantage of the ball tracking by fuse the estimations from three moving mobile robots.

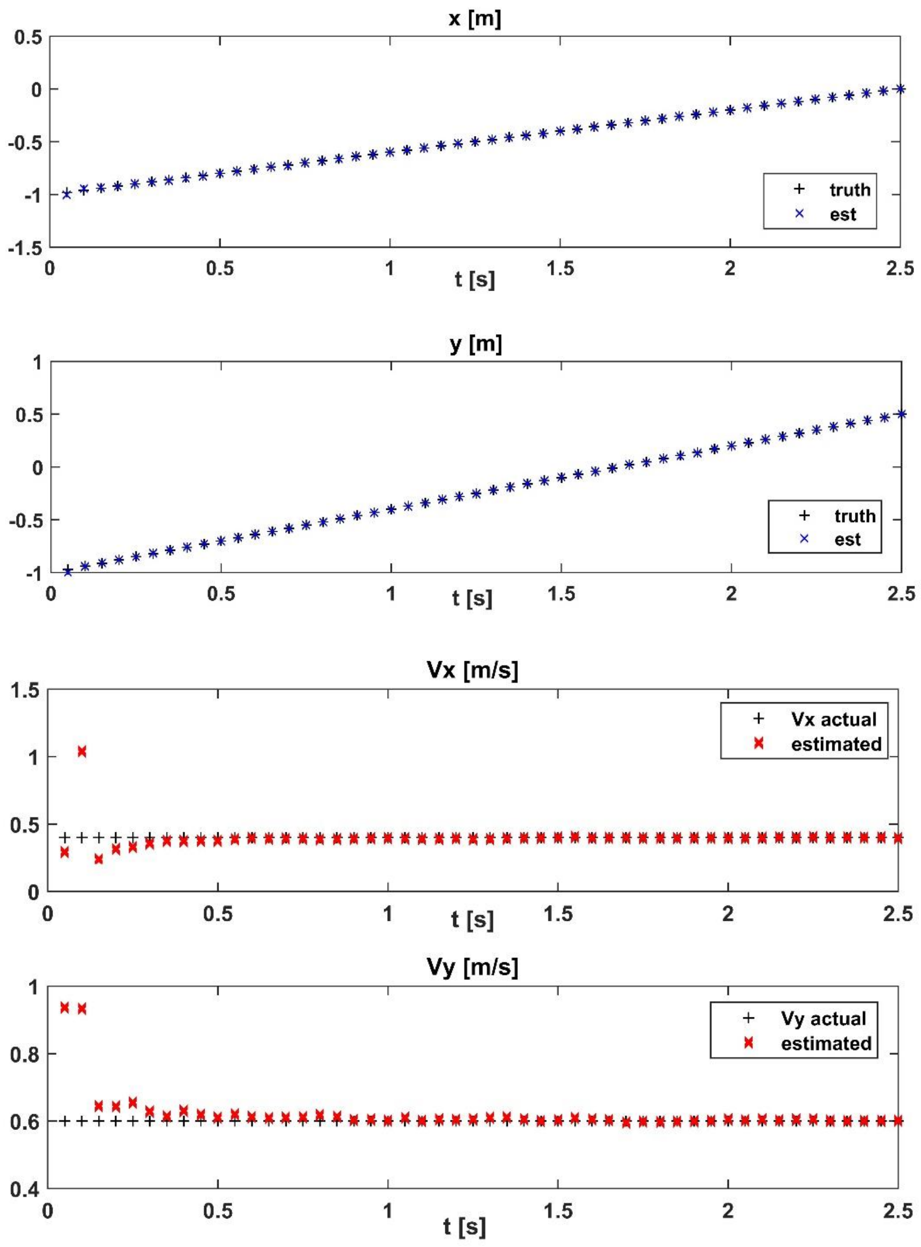


Figure 7.3: The time history and the Kalman Filter estimate of the vector state [x y Vx Vy]$^{\mathrm{T}}$ concerning fuse of three estimation using Kalman filter algorithm

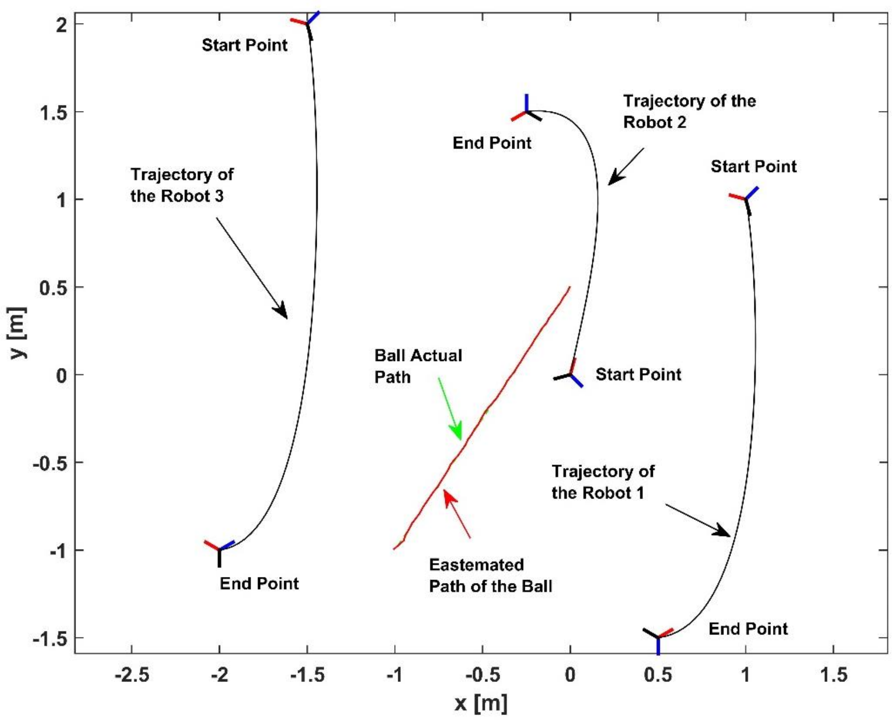


Figure 7.4: Illustration of the ball tracking optimal estimation based on fuse three estimations from three mobile robot using Kalman Filter algorithm.

The comparison of the results in Figures 7.3 and 7.4 with that obtained in 7.1 and 7.2 show clearly how the global data fusion sensors estimation provides a strong based and compete the local based estimate so that a robot can extend its own view of the environment to a global vision by extracting information from the sensor fusion module. Sharing information among the mobile robots can rise the visibility effective of the environment, permitting for more precise modeling and more proper responses. Integration a single perception of the mobile robots into a global perception leads to a more consistent estimation can lead to rise the team play and the overall playing performance. Appendix J provided Matlab code concerning Global Sensor Fusion based on the Kalman Filter Method so that three wheeled omnidirectional mobile robot estimating the motion of the ball in dynamic environment

Chapter 8

MOVING BALL INTERCEPTION AND SHOOTING TO THE GOAL STRATEGY BY THE OMNIDIRECTIONAL MOBILE ROBOT USING SYSTEMATIC APPROACH

8.1 Introduction

In this chapter, one essential problem of the RoboCup competition namely, the moving ball interception relative to the direction of the shoot, is to be discussed. The proposed method focusing on difficult situation that the mobile robot located behind the ball relative to the shooting direction. The proposed method utilized the Artificial Potential Field method to plan the robot trajectory in such a way that several via points generated lead the omnidirectional wheeled mobile robot to get the optimal configuration of shooting the ball to the goal. Subsequently, the ball interception problem is reduced to the placement of the mobile robot with the ball at a specific moment of time so that the robot configuration at this moment of time matches the direction of the shooting the ball to the goal.

8.2 Interception Proposed Algorithm

8.2.1 General Idea

The proposed algorithm takes the information concerning the time history of the estimated prospective positions of the ball in the time window as an input to the proposed algorithm. A KF that examined in the preceding chapter is traditionally utilized for this aim. Moreover, the algorithm proposed that the robot at current time able to localize its configuration correctly.

The proposed algorithm suggests to allocate a time frame work depending on the distance between the current ball position and the current position of the robot at initial time so that the prospectively estimated ball positions will be considered and evaluated.

The main concept of the proposed procedure is to utilize the Artificial Potential Field method so that several via points will be generated so that the mobile robot reaches the optimal configuration of shooting the ball to goal. The optimality is ensured by applying a simple root finding conventional method such as bisection method to select the optimal configuration of the tracked ball in a specific moment of time so that the robot can hit the ball and score a goal.

8.2.2 Detailed Description

Kalman Filter could be utilized to estimate the prospective movement of the ball. Subsequently, the ball position in every time step will be estimated. The algorithm supposed that the robot can localize its configuration correctly. A specific time frame work can be estimated based on the distance between the initial robot configuration and the ball initial position and velocity besides to the specification of the robot and the ball. A simple root finding bisection method could be utilized to localize the optimal position of the ball with its time specification so that the robot can shoot the ball to the goal. In each iteration of the optimization technique a time-specified prospectively ball position will be evaluated so that the following steps will be considered:

Step1: The target robot position computed with respect to the selected prospectively ball position as follows:

$$\begin{bmatrix} x_{trg_i} \\ y_{trg_i} \end{bmatrix} = \begin{bmatrix} x_{ball_i} \\ y_{ball_i} \end{bmatrix} - d \begin{bmatrix} \cos\phi_{shoot_i} \\ \sin\phi_{shoot_i} \end{bmatrix} \tag{8.1}$$

With

$$\phi_{shoot_i} = \tan^{-1}\left(\frac{y_{ball_i} - y_{goal\_center}}{x_{ball_i} - x_{goal\_center}} \right) \tag{8.2}$$

In Equation (8.2), $\left(x_{goall\_center}, y_{goall\_center}\right)$ is the coordinate of the middle of the selected goal. The angle ϕ_{shoot} specifying the desired shooting direction. An instance of interception scenario is schematically shown in Figure 8.1

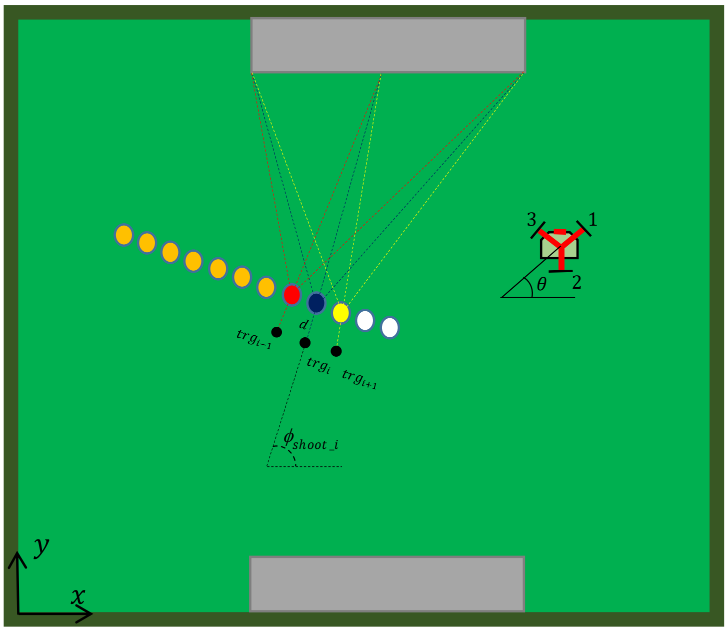


Figure 8.1: A moving Ball interception scenario by the three wheeled mobile robot

Step 2: Artificial Potential Field (APF) method will be used in this step as a motion planner so that numerous via points will be produced along the path to ensure that the mobile robot reach the target of shooting the ball to goal $\left(x_{trg_i}, y_{trg_i}\right)$. The mobile robot's starting point will be assigned a high potential and target robot position of shooting the ball to goal $\left(x_{trg_i}, y_{trg_i}\right)$will be assigned a low potential so that the robot maneuvers from the highest potential to the lowest potential. The selected prospectively ball position $\left(x_{ball_i}, y_{ball_i}\right)$ will be given high potential so that repulsive force acting between the mobile robot and the position of the ball lead the robot to the target position of shooting.

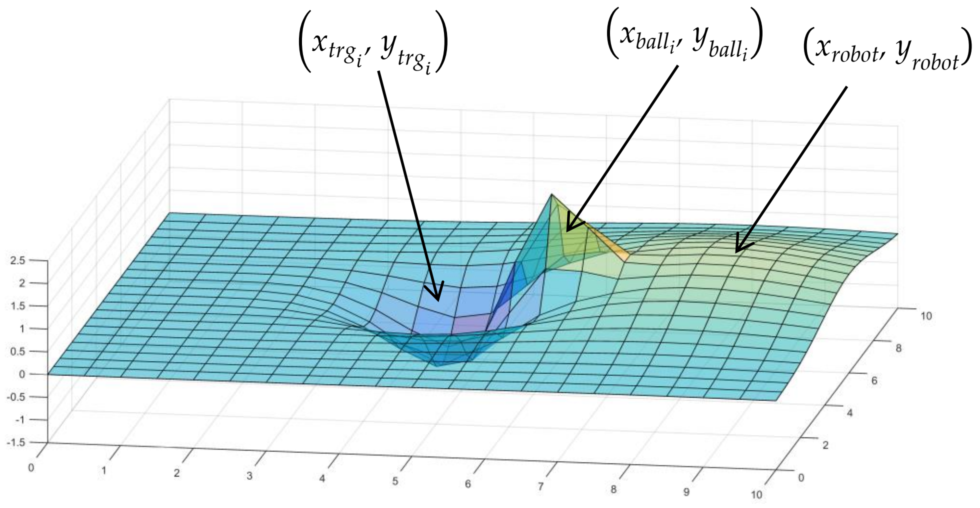


Figure 8.2: Artificial Potential Field as a motion planner for the ball interception

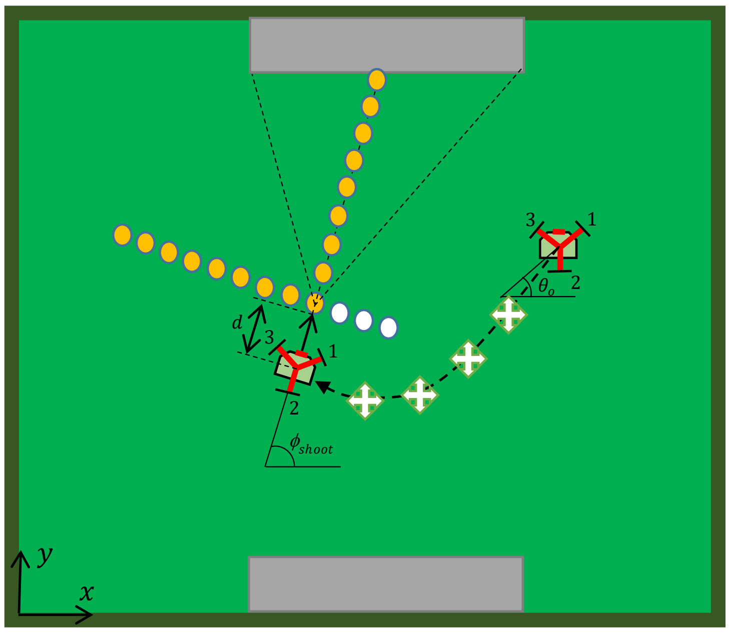


Figure 8.3: A moving Ball interception scenario so that APF method utilized as a motion planner

Figure 8.3 shows two dimensional points generated and constructing the path between the position of the robot and the shoot target position by utilizing the well-known artificial potential field method. The parameters concerning the attractive–repulsive potential chosen on the basis of the dimensions of the mobile robot, the ball and the workplace. Furthermore, a several number of correspondingly organized via points will be selected along the produced path. Corresponding to the modeling of the omnidirectional wheeled mobile robot given in the Chapter 3, the orientation of the mobile robot at the shoot target position based on the shoot to the goal angle of equation (8.2) given in the following relation:

$$\theta_{trg} = \phi_{shoot} - 60^\circ \tag{8.3}$$

As the applied path planning technique handles only x and y components, the orientation of the mobile robot concerning the midpoints can be simply identified utilizing the regular distribution notion. A geometrically approach can be used easily to specify the velocities of the midpoints configuration.

Step 3: It is proposed to use the third-order polynomial to represent the trajectories portions from the initial state to the target of shooting position that fulfills the requirement of the given four boundary conditions corresponding to the dynamic system of the Omni-directional wheeled mobile robot stated in equations (4.1) and (4.2). The third-order polynomial trajectory corresponding to i^{th} portion of the trajectory given in the following equation:

$$\begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix} = \begin{bmatrix} a_{1_i} \\ a_{2_i} \\ a_{3_i} \end{bmatrix} t^3 + \begin{bmatrix} b_{1_i} \\ b_{2_i} \\ b_{3_i} \end{bmatrix} t^2 + \begin{bmatrix} c_{1_i} \\ c_{2_i} \\ c_{3_i} \end{bmatrix} t + \begin{bmatrix} d_{1_i} \\ d_{2_i} \\ d_{3_i} \end{bmatrix} \tag{8.4}$$

The polynomial coefficients stated in equation (8.4) are calculated based on the boundary conditions in equation to be as follows:

$$\begin{cases} \begin{bmatrix} a_{1_i} \\ a_{2_i} \\ a_{3_i} \end{bmatrix} = \dfrac{1}{{t_{\mathrm{f}_i}}^2}\left(-\dfrac{2}{t_{\mathrm{f}_i}}\left(\begin{bmatrix} x_\mathrm{f} \\ y_\mathrm{f} \\ \theta_\mathrm{f} \end{bmatrix} - \begin{bmatrix} x_\mathrm{o} \\ y_\mathrm{o} \\ \theta_\mathrm{o} \end{bmatrix} \right) + \begin{bmatrix} \dot{x}_\mathrm{o} \\ \dot{y}_\mathrm{o} \\ \dot{\theta}_\mathrm{o} \end{bmatrix} + \begin{bmatrix} \dot{x}_\mathrm{f} \\ \dot{y}_\mathrm{f} \\ \dot{\theta}_\mathrm{f} \end{bmatrix} \right) \\ \begin{bmatrix} b_{1_i} \\ b_{2_i} \\ b_{3_i} \end{bmatrix} = \dfrac{1}{t_{\mathrm{f}_i}}\left(\dfrac{3}{t_{\mathrm{f}_i}}\left(\begin{bmatrix} x_\mathrm{f} \\ y_\mathrm{f} \\ \theta_\mathrm{f} \end{bmatrix} - \begin{bmatrix} x_\mathrm{o} \\ y_\mathrm{o} \\ \theta_\mathrm{o} \end{bmatrix} \right) - 2\begin{bmatrix} \dot{x}_\mathrm{o} \\ \dot{y}_\mathrm{o} \\ \dot{\theta}_\mathrm{o} \end{bmatrix} - \begin{bmatrix} \dot{x}_\mathrm{f} \\ \dot{y}_\mathrm{f} \\ \dot{\theta}_\mathrm{f} \end{bmatrix} \right) \\ \begin{bmatrix} c_{1_i} \\ c_{2_i} \\ c_{3_i} \end{bmatrix} = \begin{bmatrix} \dot{x}_\mathrm{o} \\ \dot{y}_\mathrm{o} \\ \dot{\theta}_\mathrm{o} \end{bmatrix}, \quad \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} x_\mathrm{o} \\ y_\mathrm{o} \\ \theta_\mathrm{o} \end{bmatrix} \end{cases} \tag{8.5}$$

As a normal prerequisite for the efficient interception is to minimize the time elapsed till the shoot. The value of the total maneuvering time t_{f_i} of the portion *i* selected so that the acceleration amplitude of the mass center of the robot $\ddot{x}^2 + \ddot{y}^2$ critically reaches the limit $\alpha^{\mathbf{2}}_{\mathrm{max}}$. The total maneuvering time that robot proceeds from the initial state to the target of shooting position with regards to the *n* portions given as follows

$$t_{\mathrm{f1}} = \sum_{i=1}^{n} t_{\mathrm{f}_i} \tag{8.6}$$

Step 4: In this step, the mobile robot should oriented to the ball with the fastest possible speed and shortest possible time so that the acceleration amplitude of the mass center of the robot $\ddot{x}^2 + \ddot{y}^2$ critically reaches the limit $\alpha^{\mathbf{2}}_{\mathrm{max}}$ along the straight line from the target position of the robot to the position of the ball. The maneuvering time that robot proceeds in this step given as t_{f2}.

Step 5: The overall time that the mobile robot proceeds from the initial configuration to the target configuration of the shooting the ball to the goal besides to the computational time that the algorithm take should be calculated as follows

$$t_{\text{f\_robot}} = t_{\text{f1}} + t_{\text{f2}} + t_{comp} \tag{8.7}$$

Step 6: The time required for the ball to proceeds from the initial configuration to the evaluated prospectively ball position $t_{\text{f\_ball}}$ will be compared to the total robot maneuvering time $t_{\text{f\_robot}}$. If the $t_{\text{f\_robot}}$ is greater than $t_{\text{f\_ball}}$, it means the robot can't proceed to the target position of shooting the ball to the goal. The optimization process aims to find the optimal prospectively ball position with the shortest possible time $t_{\text{f\_ball}}$ so that the mobile robot be able to proceed to the target configuration and shoot the ball to the goal. Concerning the case that the optimal solution found, the following relation will be guaranteed

$$t_{\text{f\_robot}} - t_{\text{f\_ball}} \le t_{\text{f\_remain}} \tag{8.8}$$

The difference between the $t_{\text{f\_robot}}$ and $t_{\text{f\_ball}}$ will be distributed on the total maneuvering time of the robot trajectories portions to ensure the alignment in time.

8.3 Experimental Results

Assuming that a ball moving in the field of a flat surface with the ignorance of its mass. Besides, the three wheeled OMR located behind the ball relative to the shooting direction. The purpose of this experimentation is to utilize the method proposed in section 8.2 so that the robot can proceed with the optimality and shoot the ball to the goal. The following table summarizes the parameters of the ball interception method.

Table 8.1: The Parameters selected for the experimental setup concerning the ball interception method

The Parameter description	**The parameter symbol**	**The parameter value**
Time step	Δt	0.05 sec
Number of samples	Ns	140
Robot Initial position	(x_r, y_r, θ_r)	(6,8,0)
Robot Initial velocity	$(\dot{x}_r, \dot{y}_r, \dot{\theta}_r)$	(0,0,0)
Ball Initial Location	(x_b, y_b)	(2,8)
Ball Initial Velocity	(v_{x_b}, v_{y_b})	(0.7,-0.4) m/sec
Selected Goal Coordinate	$([x_{g1} \quad x_{g2}], [y_{g1} \quad y_{g2}])$	([3 7],[10 10])
The distance of the alignment the robot at target position of shooting with the ball	L	1 m

The APF technique will be used as a motion planner concerning the SAOWMR with 3 omnidirectional wheels. The following table demonstrates the parameter of the APF method used to generate the path from the robot's initial configuration to the target position of shooting the ball to the goal.

Table 8.2: APF Parameters utilized to produce the robot path from the robot initial configuration to the target position of shooting

Number of the artificial particles	60
The radius of the circle constructing the artificial points	0.09
The depth of the attractant σ_T	2
The depth of the repellant σ_o	3
Width of the attractant μ_T	0.5
Width of the repellant μ_o	2

The following figure illustrates the path generated using APF method from the mobile robot initial configuration to the target configuration of the shoot

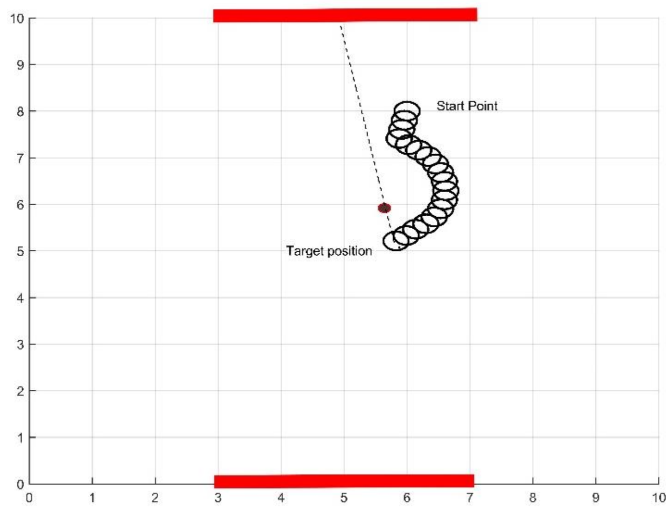


Figure 8.4: The mobile robot path generated using APF method from the mobile robot initial configuration to the target of the shoot

The experimental process shows that the proposed method efficiently succeeded to achieve moving ball interception so that the mobile robot can shoot the ball to the goal. The following figure shows how the three wheeled mobile robot proceed to the interception point and shoot the ball to the goal

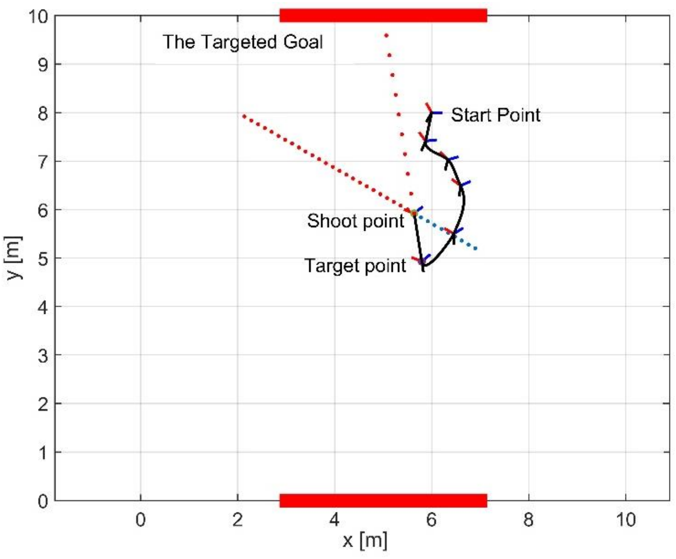


Figure 8.5: moving ball interception using the method proposed

The method proposed obviously illustrates the fulfillment of the robot dynamic system and the constraints of the path, besides the fluency of the movement. The Matlab code utilized to validate the ball interception proposed method so that the mobile robot located behind the ball in terms to the shooting direction can shoot the ball to the goal provided in Appendix K.

Chapter 9

CONCLUSIONS AND FUTURE WORK

This thesis presents a complete and compact and mathematical model for the SAOWMR with *n* omnidirectional wheels including kinematics and dynamics aspects. This general mathematical model offers an occasion to perform comparisons, research and experiments on these OWMRs that have two, three, four, six, or even more omnidirectional wheels without the necessity to switch models or develop a new model.

Moreover, a novel computationally effective method in the trajectory planning optimization (TPO) is developed in this thesis concerning the compact mathematical model of the SAOWMR with *n* omnidirectional wheels. The developed procedure based on approximating the control input vector applied to the terminals of the DC motors relevant to the robot’s omnidirectional wheels so that the robot able to maneuver from an initial to a final configuration of boundary conditions with the optimality of total required energy and maneuvering time while a system of constraints remains maintained. The experimental results regarding the optimal trajectory generation proposed method shows a real-time and accuracy performance that the produced trajectory tends to be without any terminal error.

Furthermore, a new method proposed in this thesis based on the Artificial Potential Field (APF) method to solve the obstacle avoidance optimal trajectory planning problem concerning the SAOWMR with *n* omnidirectional wheels. The proposed method obviously illustrates the fulfillment of the robot dynamic system and the constraints of the path, as well as the fluency of the motion control.

A global Multi Sensor Fusion achieved in this thesis based on the Kalman filter in the context of the mobile robot soccer performing ball position estimation and tracking in dynamic environment. Additionally, a new proposed method regarding the moving ball interception relative to the direction of the shoot so that the omnidirectional wheeled mobile robot can maneuver and estimate the optimal configuration of shooting the ball to the goal based on the Artificial Potential Field method.

Future work of this thesis will be focusing on the completion the research by testing and optimizing a real prototype utilizing vision-based system to achieve and develop navigational processes such as dynamic obstacle avoidance, optimal trajectory generation and ball interception.

REFERENCES

[1] Mariappan, M., Sing, J. C., Wee, C. C., Khoo, B., & Wong, W. K. (2014, December). Simultaneous rotation and translation movement for four omnidirectional wheels holonomic mobile robot. In *2014 IEEE International Symposium on Robotics and Manufacturing Automation (ROMA)* (pp. 69-73). IEEE.

[2] Kale, S., & Shriramwar, S.S. (2009). FPGA-Based Controller for a Mobile Robot. *arXiv*, 0908.221.

[3] Endo, G., & Hirose, S. (2000). Study on Roller-Walker—System Integration and Basic Experiments—. *Journal of the Robotics Society of Japan*, *18*(2), 270-277.

[4] Qian, J., Zi, B., Wang, D., Ma, Y., & Zhang, D. (2017). The design and development of an omni-directional mobile robot oriented to an intelligent manufacturing system. *Sensors*, *17*(9), 2073.

[5] Guo, S., Diao, Q., & Xi, F. (2017). Vision based navigation for Omni-directional mobile industrial robot. *Procedia Computer Science*, *105*, 20-26.

[6] Guo, S., Fang, T. T., Song, T., Xi, F. F., & Wei, B. G. (2018). Tracking and localization for omni-directional mobile industrial robot using reflectors. *Advances in Manufacturing*, *6*(1), 118-125.

[7] Tian, Y., Zhang, S., Liu, J., Chen, F., Li, L., & Xia, B. (2017). Research on a new omnidirectional mobile platform with heavy loading and flexible motion. *Advances in Mechanical Engineering*, *9*(9), 1687814017726683.

[8] Tasaki, R., Kitazaki, M., Miura, J., & Terashima, K. (2015, May). Prototype design of medical round supporting robot "Terapio". In *2015 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 829-834). IEEE.

[9] Moreno, J., Clotet, E., Lupiañez, R., Tresanchez, M., Martínez, D., Pallejà, T., ... & Palacín, J. (2016). Design, implementation and validation of the three-wheel holonomic motion system of the assistant personal robot (APR). *Sensors*, *16*(10), 1658.

[10] Tzou, J. H., & Chiang, F. C. (2009, December). The development of the mobile robot for taking care of elderly people. In *2009 Fourth International Conference on Innovative Computing, Information and Control (ICICIC)* (pp. 540-543). IEEE.

[11] Hu, J., Edsinger, A., Lim, Y. J., Donaldson, N., Solano, M., Solochek, A., & Marchessault, R. (2011, May). An advanced medical robotic system augmenting healthcare capabilities-robotic nursing assistant. In *2011 IEEE international conference on robotics and automation* (pp. 6264-6269). IEEE.

[12] Udengaard, M., & Iagnemma, K. (2008, May). Design of an omnidirectional mobile robot for rough terrain. In *2008 IEEE International Conference on Robotics and Automation* (pp. 1666-1671). IEEE.

[13] Ishigami, G., Iagnemma, K., Overholt, J., & Hudas, G. (2015). Design, development, and mobility evaluation of an omnidirectional mobile robot for rough terrain. *Journal of Field Robotics*, *32*(6), 880-896.

[14] Ishigami, G., Pineda, E., Overholt, J., Hudas, G., & Iagnemma, K. (2014). Design, development, and mobility test of an omnidirectional mobile robot for rough Terrain. In *Field and Service Robotics* (pp. 599-611). Springer, Berlin, Heidelberg.

[15] Tavakoli, M., & Viegas, C. (2014). Analysis and application of dual-row omnidirectional wheels for climbing robots. *Mechatronics*, *24*(5), 436-448.

[16] Samani, H. A., Abdollahi, A., Ostadi, H., & Rad, S. Z. (2004). Design and development of a comprehensive omni directional soccer player robot. *International Journal of Advanced Robotic Systems*, *1*(3), 20.

[17] Rahman, F. A., Hartanto, R. T., Chairistian, R. I., Abdurrahman, Y., Suryani, O. F., Astrid, M. V., ... & Ardiyanto, I. (2017). Motion Planning in Dynamic Environment for Middle Size League using Theta* and Polynomial Trajectory Generator. In *5th Indonesian Symposium on Robotic Systems and Control* (pp. 113-117).

[18] Mariappan, E. D. M., Ramu, V., & Ganesan, T. (2011). Fuzzy Logic Based Navigation Safety System for a Remote Controlled Orthopaedic Robot (OTOROB). *Research and Development (IJRRD)*, *1*(1), 21-41.

[19] Yu, H., Spenko, M., & Dubowsky, S. (2004). Omni-directional mobility using active split offset castors. *J. Mech. Des.*, *126*(5), 822-829.

[20] Grabowiecki, J. Vehicle Wheel. U.S. Patent No. 1,303,535, June 1919.

[21] Ilon, B. E. (1975). *U.S. Patent No. 3,876,255*. Washington, DC: U.S. Patent and Trademark Office.

[22] Podnar, G. W. (1985). The uranus mobile robot. *Autonomous Mobile Robots: Ann. Rep*.

[23] Muir, P. F., & Neuman, C. P. (1987). Kinematic modeling of wheeled mobile robots. *Journal of robotic systems*, *4*(2), 281-340.

[24] Koestler, A., & Bräunl, T. (2004, December). Mobile robot simulation with realistic error models. In *International Conference on Autonomous Robots and Agents, ICARA* (Vol. 1, pp. 46-51).

[25] Song, J. B., & Byun, K. S. (2004). Design and Control of a Four-Wheeled Omnidirectional Mobile Robot with Steerable Omnidirectional Wheels. *Journal of Robotic Systems*, *21*(4), 193-208.

[26] Salih, J. E. M., Rizon, M., Yaacob, S., Adom, A. H., & Mamat, M. R. (2006). Designing Omni-Directional Mobile Robot with Mecanum Wheel [J. In *American Journal of Applied Science s*.

[27] Doroftei, I., Grosu, V., & Spinu, V. (2007). *Omnidirectional mobile robot-design and implementation*. INTECH Open Access Publisher.

[28] Dickerson, S. L., & Lapin, B. D. (1991, March). Control of an omni-directional robotic vehicle with Mecanum wheels. In *NTC'91-National Telesystems Conference Proceedings* (pp. 323-328). IEEE.

[29] Nagatani, K., Tachibana, S., Sofne, M., & Tanaka, Y. (2000, October). Improvement of odometry for omnidirectional vehicle using optical flow information. In *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113)* (Vol. 1, pp. 468-473). IEEE.

[30] Pin, F. G., & Killough, S. M. (1994). A new family of omnidirectional and holonomic wheeled platforms for mobile robots. *IEEE transactions on robotics and automation*, *10*(4), 480-489.

[31] Byun, K. S., & Song, J. B. (2003). Design and construction of continuous alternate wheels for an omnidirectional mobile robot. *Journal of Robotic Systems*, *20*(9), 569-579.

[32] West, M., & Asada, H. (1995, May). Design and control of ball wheel omnidirectional vehicles. In *Proceedings of 1995 IEEE international conference on robotics and automation* (Vol. 2, pp. 1931-1938). IEEE.

[33] Li, Y., Dai, S., Zhao, L., Yan, X., & Shi, Y. (2019). Topological design methods for mecanum wheel configurations of an omnidirectional mobile robot. *Symmetry*, *11*(10), 1268.

[34] Gfrerrer, A. (2008). Geometry and kinematics of the Mecanum wheel. *Computer Aided Geometric Design*, *25*(9), 784-791.

[35] Gao, P., Peng, J., Yu, W., Li, S., & Qin, X. (2017). Design and motion analysis of a mecanum three-round omni-directional mobile platform. *J. Northwest. Polytech. Univ*, *35*(5), 857-862.

[36] Lafaye, J., Gouaillier, D., & Wieber, P. B. (2014, November). Linear model predictive control of the locomotion of Pepper, a humanoid robot with omnidirectional wheels. In *2014 IEEE-RAS International Conference on Humanoid Robots* (pp. 336-341). IEEE.

[37] Song, J. B., & Byun, K. S. (2009). Steering control algorithm for efficient drive of a mobile robot with steerable omni-directional wheels. *Journal of mechanical science and technology*, *23*(10), 2747.

[38] Muir, P. F., & Neuman, C. P. (1990). Kinematic modeling for feedback control of an omnidirectional wheeled mobile robot. In *Autonomous robot vehicles* (pp. 25-31). Springer, New York, NY.

[39] Leow, Y. P., Low, K. H., & Loh, W. K. (2002, December). Kinematic modelling and analysis of mobile robots with omni-directional wheels. In *7th International*

Conference on Control, Automation, Robotics and Vision, 2002. ICARCV 2002. (Vol. 2, pp. 820-825). IEEE.

[40] Loh, W. K., Low, K. H., & Leow, Y. P. (2003, September). Mechatronics design and kinematic modelling of a singularityless omni-directional wheeled mobile robot. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)* (Vol. 3, pp. 3237-3242). IEEE.

[41] Watanabe, K. (1998, April). Control of an omnidirectional mobile robot. In *1998 Second International Conference. Knowledge-Based Intelligent Electronic Systems. Proceedings KES'98 (Cat. No. 98EX111)* (Vol. 1, pp. 51-60). IEEE.

[42] Filipe, J., Ferrier, J. L., & Cetto, J. A. (Eds.). (2008). *Informatics in Control, Automation and Robotics: Selected Papers from the International Conference on Informatics in Control, Automation and Robotics 2007* (Vol. 24). Springer Science & Business Media.

[43] Kalm, T. (2004). ar-Nagy, R. D’Andrea, and P. Ganguly,“Near-optimal dynamics trajectory generation and control of an omnidirectional vehicle,”. *Robotics and Autonomous Systems*, *46*(1), 47-64.

[44] Purwin, O., & D’Andrea, R. (2006). Trajectory generation and control for four wheeled omnidirectional vehicles. *Robotics and Autonomous Systems, 54*(1), 13-22.

[45] Kalmár-Nagy, T., Ganguly, P., & D'Andrea, R. (2002, May). Real-time trajectory generation for omnidirectional vehicles. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)* (Vol. 1, pp. 286-291). IEEE.

[46] Alwan, H. M. (2020). Dynamic Analysis Modeling of a Holonomic Wheeled Mobile Robot with Mecanum Wheels Using Virtual Work Method. *Journal Mechanical Engineering Research and Development*, *43*(6), 373-380.

[47] Hou, L., Zhang, L., & Kim, J. (2019). Energy modeling and power measurement for mobile robots. *Energies*, *12*(1), 27.

[48] Alwan, H. M., Volkov, A. N., & Shbani, A. (2021, February). Solution of Inverse and Forward Kinematics Problems for Mobile Robot with Six Mecanum Wheels. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1094, No. 1, p. 012071). IOP Publishing.

[49] Rijalusalam, D. U., & Iswanto, I. (2021). Implementation Kinematics Modeling and Odometry of Four Omni Wheel Mobile Robot on The Trajectory Planning and Motion Control Based Microcontroller. *Journal of Robotics and Control (JRC)*, *2*(5), 448-455.

[50] Karras, G. C., & Fourlas, G. K. (2020). Model predictive fault tolerant control for omni-directional mobile robots. *Journal of Intelligent & Robotic Systems*, *97*(3), 635-655.

[51] Cuevas, F., Castillo, O., & Cortés-Antonio, P. (2020). Omnidirectional Four Wheel Mobile Robot Control with a Type-2 Fuzzy Logic Behavior-Based Strategy.

[52] Yang, H., Wang, S., Zuo, Z., & Li, P. (2020). Trajectory tracking for a wheeled mobile robot with an omnidirectional wheel on uneven ground. *IET Control Theory & Applications*, *14*(7), 921-929.

[53] Serrano-Pérez, O., Villarreal-Cervantes, M. G., González-Robles, J. C., & Rodríguez-Molina, A. (2019). Meta-heuristic algorithms for the control tuning of omnidirectional mobile robots. *Engineering Optimization*.

[54] Abiyev, R. H., Günsel, I. S., Akkaya, N., Aytac, E., Çağman, A., & Abizada, S. (2017). Fuzzy control of omnidirectional robot. *Procedia Computer Science*, *120*, 608-616.

[55] Abiyev, R. H., Akkaya, N., & Gunsel, I. (2018). Control of omnidirectional robot using Z-number-based fuzzy system. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*(1), 238-252.

[56] Hacene, N., & Mendil, B. (2019). Motion analysis and control of three-wheeled omnidirectional mobile robot. *Journal of Control, Automation and Electrical Systems*, *30*(2), 194-213.

[57] Alshorman, A. M., Alshorman, O., Irfan, M., Glowacz, A., Muhammad, F., & Caesarendra, W. (2020). Fuzzy-based fault-tolerant control for omnidirectional mobile robot. *Machines*, *8*(3), 55.

[58] Wang, C., Liu, X., Yang, X., Hu, F., Jiang, A., & Yang, C. (2018). Trajectory tracking of an omni-directional wheeled mobile robot using a model predictive control strategy. *Applied Sciences*, *8*(2), 231.

[59] Thi, K. D. H., Nguyen, M. C., Vo, H. T., Nguyen, D. D., & Bui, A. D. (2019, January). Trajectory tracking control for four-wheeled omnidirectional mobile robot using Backstepping technique aggregated with sliding mode control. In *2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)* (pp. 131-134). IEEE.

[60] Xie, Y., Zhang, X., Meng, W., Xie, S., Jiang, L., Meng, J., & Wang, S. (2020, July). Coupled sliding mode control of an omnidirectional mobile robot with variable modes. In *2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)* (pp. 1792-1797). IEEE.

[61] Dong, F., Jin, D., Zhao, X., & Han, J. (2021). Adaptive Robust Constraint Following Control for Omnidirectional Mobile Robot: An Indirect Approach. *IEEE Access*, *9*, 8877-8887.

[62] Sahoo, S. R., & Chiddarwar, S. S. (2020). Flatness-based control scheme for hardware-in-the-loop simulations of omnidirectional mobile robot. *Simulation*, *96*(2), 169-183.

[63] Amudhan, A. N., Sakthivel, P., Sudheer, A. P., & Kumar, T. S. (2019, July). Design of controllers for omnidirectional robot based on the ystem identification technique for trajectory tracking. In *Journal of Physics: Conference Series* (Vol. 1240, No. 1, p. 012146). IOP Publishing.

[64] Kalmár-Nagy, T. (2016). Real-time trajectory generation for omni-directional vehicles by constrained dynamic inversion. *Mechatronics*, *35,* 44-53.

[65] Wu, M., Dai, S. L., & Yang, C. (2020). Mixed reality enhanced user interactive path planning for omnidirectional mobile robot. *Applied Sciences*, *10*(3), 1135.

[66] Saenz, A., Santibañez, V., Bugarin, E., Dzul, A., Ríos, H., & Villalobos-Chin, J. (2021). Velocity Control of an Omnidirectional Wheeled Mobile Robot Using Computed Voltage Control with Visual Feedback: Experimental Results. *International Journal of Control, Automation and Systems*, *19*(2), 1089-1102.

[67] Ou, J., & Wang, M. (2019, July). Path planning for omnidirectional wheeled mobile robot by improved ant colony optimization. In *2019 Chinese Control Conference (CCC)* (pp. 2668-2673). IEEE.

[68] Choi, J. W., Curry, R. E., & Elkaim, G. H. (2009, June). Obstacle avoiding real-time trajectory generation and control of omnidirectional vehicles. In *2009 American Control Conference* (pp. 5510-5515). IEEE.

[69] Azim Mohseni, N., & Fakharian, A. (2017). Direct Optimal Motion Planning for Omni-directional Mobile Robots under Limitation on Velocity and Acceleration. *Journal of Optimization in Industrial Engineering*, *10*(22), 93-101.

[70] Williams, R. L., & Wu, J. (2010). Dynamic obstacle avoidance for an omnidirectional mobile robot. *Journal of Robotics*, *2010*.

[71] Liu, X., Chen, H., Wang, C., Hu, F., & Yang, X. (2018, August). MPC control and path planning of Omni-directional mobile robot with potential field method. In *International Conference on Intelligent Robotics and Applications* (pp. 170-181). Springer, Cham.

[72] Kim, C., Suh, J., & Han, J. H. (2020). Development of a hybrid path planning algorithm and a bio-inspired control for an omni-wheel mobile robot. *Sensors*, *20*(15), 4258.

[73] Azizi, M. R., Rastegarpanah, A., & Stolkin, R. (2021). Motion Planning and Control of an Omnidirectional Mobile Robot in Dynamic Environments. *Robotics*, *10*(1), 48.

[74] Cuevas, F., Castillo, O., & Cortes-Antonio, P. (2019, June). Towards an adaptive control strategy based on type-2 fuzzy logic for autonomous mobile robots. In *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (pp. 1-6). IEEE.

[75] Hacene, N., & Mendil, B. (2019). Fuzzy behavior-based control of three wheeled omnidirectional mobile robot. *International Journal of Automation and Computing*, *16*(2), 163-185.

[76] Ajeil, F. H., Ibraheem, I. K., Azar, A. T., & Humaidi, A. J. (2020). Autonomous navigation and obstacle avoidance of an omnidirectional mobile robot using swarm optimization and sensors deployment. *International Journal of Advanced Robotic Systems*, *17*(3), 1729881420929498.

[77] Zhang, Y., Zhang, C. H., & Shao, X. (2021). User preference-aware navigation for mobile robot in domestic via defined virtual area. *Journal of Network and Computer Applications*, *173*, 102885.

[78] Ran, T., Yuan, L., & Zhang, J. B. (2021). Scene perception based visual navigation of mobile robot in indoor environment. *ISA transactions*, *109*, 389-400.

[79] Gamal, O., Cai, X., & Roth, H. (2020, October). Learning from Fuzzy System Demonstration: Autonomous Navigation of Mobile Robot in Static Indoor Environment using Multimodal Deep Learning. In *2020 24th International Conference on System Theory, Control and Computing (ICSTCC)* (pp. 218-225). IEEE.

[80] Al Khatib, E. I., Jaradat, M. A. K., & Abdel-Hafez, M. F. (2020). Low-cost reduced navigation system for mobile robot in indoor/outdoor environments. *IEEE Access*, *8*, 25014-25026.

[81] Iqbal, J., Xu, R., Sun, S., & Li, C. (2020). Simulation of an autonomous mobile robot for LiDAR-based in-field phenotyping and Navigation. *Robotics*, *9*(2), 46.

[82] Bai, X., Dong, L., Ge, L., Xu, H., Zhang, J., & Yan, J. (2020). Robust localization of mobile robot in industrial environments with non-line-of-sight situation. *IEEE Access*, *8*, 22537-22545.

[83] Kim, E. K., & Kim, S. S. (2020). Generation of Feature Map for Improving Localization of Mobile Robot based on Stereo Camera. *The Journal of Korea Institute of Information, Electronics, and Communication Technology*, *13*(1), 58-63.

[84] Li, H., Mao, Y., You, W., Ye, B., & Zhou, X. (2020, October). A neural network approach to indoor mobile robot localization. In *2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)* (pp. 66-69). IEEE.

[85] Greenberg, J. N., & Tan, X. (2020). Dynamic optical localization of a mobile robot using Kalman filtering-based position prediction. *IEEE/ASME Transactions on Mechatronics*, *25*(5), 2483-2492.

[86] Almasri, E., & Uyguroğlu, M. K. (2021). Trajectory Planning Optimization for the Autonomous Three-Wheeled Omnidirectional Mobile Robot. *2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, 1–6. https://doi.org/10.1109/HORA52670.2021.9461321

[87] Gutmann, J. S., Hatzack, W., Herrmann, I., Nebel, B., Rittinger, F., Topor, A., & Weigel, T. (1999). Reliable Self-Localization, Multirobot Sensor Integration, Accurate Path-Planning and Basic Soccer Skills: Playing an E ective Game of Robotic Soccer.

[88] Larsen, T. D., Hansen, K. L., Andersen, N. A., & Ravn, O. (1999, August). Design of Kalman filters for mobile robots; evaluation of the kinematic and odometric approach. In *Proceedings of the 1999 IEEE international conference on control applications (Cat. No. 99CH36328)* (Vol. 2, pp. 1021-1026). IEEE.

[89] Moreno, L., Armingol, J. M., De La Escalera, A., & Salichs, M. A. (1999, October). Global integration of ultrasonic sensors information in mobile robot localization. In *Proceedings of the Ninth International Conference on Advanced Robotics* (pp. 283-288). Tokyo, Japan.

[90] Sasiadek, J. Z., & Hartana, P. (2000, July). Sensor data fusion using Kalman filter. In *Proceedings of the Third International Conference on Information Fusion* (Vol. 2, pp. WED5-19). IEEE.

[91] Kalman, R. E. (1960). A new approach to linear filtering and prediction problems.

[92] Fox, D., Burgard, W., & Thrun, S. (1999). Markov localization for mobile robots in dynamic environments. *Journal of artificial intelligence research*, *11*, 391-427.

[93] Dietl, M., Gutmann, J. S., & Nebel, B. (2001, October). Cooperative sensing in dynamic environments. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)* (Vol. 3, pp. 1706-1713). IEEE.

[94] Stroupe, A. W., Martin, M. C., & Balch, T. (2001, May). Distributed sensor fusion for object position estimation by multi-robot systems. In *Proceedings 2001 ICRA. IEEE international conference on robotics and automation (Cat. No. 01CH37164)* (Vol. 2, pp. 1092-1098). IEEE.

[95] Steinbauer, G., Faschinger, M., Fraser, G., Mühlenfeld, A., Richter, S., Wöber, G., & Wolf, J. (2003). Mostly harmless team description. In *Proc. RoboCup Symposium*.

[96] Dellaert, F., Fox, D., Burgard, W., & Thrun, S. (1999, May). Monte carlo localization for mobile robots. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)* (Vol. 2, pp. 1322-1328). IEEE.

[97] Bonarini, A., Furlan, A., Malago, L., Marzorati, D., Matteucci, M., Migliore, D., ... & Sorrenti, D. (2009). Milan RoboCup team 2009. *Proc of the RoboCup*.

[98] Bonarini, A., Matteucci, M., & Restelli, M. (2001, November). Anchoring: do we need new solutions to an old problem or do we have old solutions for a new problem. In *Proceedings of the AAAI Fall Symposium on Anchoring Symbols to*

Sensor Data in Single and Multiple Robot Systems (pp. 79-86). AAAI Press Menlo Park, CA.

[99] Gonçalves, J., Lima, J., & Costa, P. (2008). Real-time tracking of an omnidirectional robot: an extended kalman filter approach. In *International Conference on Informatics in Control, Automation and Robotics*.

[100] Ferrein, A., Hermanns, L., & Lakemeyer, G. (2005, July). Comparing sensor fusion techniques for ball position estimation. In *Robot Soccer World Cup* (pp. 154-165). Springer, Berlin, Heidelberg.

APPENDICES

Appendix A: Dynamical Equations of the DC brushed motor

The power efficiency of converting 100% of electrical power to mechanical power through the rotating movement given in the following equality:

$$Iu = \tau\omega \tag{A.1}$$

u is the voltage applied on the terminal input of the motor, I is the current input, τ is the torque generated and ω the angular velocity of the motor shaft. Due to the power losses generated by the long coil of wire in the armature, the equality in (A.1) will be reformulated so that the resistor R and the inductor L of the armature will be considered

$$Iu = \tau\omega + I^2R + \frac{d}{dt}(\frac{1}{2}LI^2) \tag{A.2}$$

Subsequently,

$$Iu = \tau\omega + I^2R + LI\frac{dI}{dt} \tag{A.3}$$

By dividing the both sides of the equality (A.3) by I we obtain

$$u = \frac{\tau}{I}\omega + IR + L\frac{dI}{dt} \tag{A.4}$$

The ratio $\frac{\tau}{I}$ in equation (A.4) is called torque constant k_τ. Torque constant relates the current going through the DC motor and the torque associated with that current.

$$u = k_\tau\omega + IR + L\frac{dI}{dt} \tag{A.5}$$

The unit of the torque constant is (N.m/A) or (V.s/Rad). If we consider a case where the motor is at steady state. The equation (A.6) rearranged to be in the following form

$$\tau = \frac{k_\tau}{R}u - \frac{{k_\tau}^2}{R}\omega \tag{A.6}$$

With the no-slip condition, the force produced by a DC motor driven wheel is derived from the equation (A.6) if we consider the scalar relation between the angular velocity ω_i (rad/s) and translational velocity v_i (m/s) of the mass center of the ith omnidirectional wheel that can be given as following

$$v_i = r\omega_i \tag{A.7}$$

In equation (A.7), r is the radius of the omnidirectional wheel. The force F_i (Newton) generated by the ith omnidirectional wheel DC motor is given in the following relation in terms of the torque generated

$$F_i = \frac{\tau_i}{r} \tag{A.8}$$

Based on the equation (A.7), (A.8) and (A.9), The force F_i generated by the ith omnidirectional wheel DC motor written in compact form as follows:

$$F_i = \frac{k_\tau}{Rr}u_i - \frac{{k_\tau}^2}{Rr^2}v_i \tag{A.9}$$

Equation (A.9) can be rewritten in terms of the parameters $\alpha = \frac{k_\tau}{Rr}$ (N/V) and $\beta = \frac{{k_\tau}^2}{Rr^2}$ (N.s/m) as in the following

$$F_i = \alpha u_i - \beta v_i \tag{A.10}$$

The power p_i (Watt) produced by the ith omnidirectional wheel DC motor is given in the following relation on the basis of the torque generated and the angular velocity of the motor shaft

$$p_i = \tau_i \, w_i \tag{A.11}$$

By substituting the equation (A.6) into (A.11), The power p_i generated by the ith omnidirectional wheel DC motor will be given in (A.12) in terms of the parameters α and β

$$p_i = \frac{r}{k_\tau}\left(\alpha u_i^2 - \beta v_i \, u_i\right) \tag{A.12}$$

Appendix B: Details about the derive of equation (4.16)

Supposing that the SAOWMR with n omnidirectional wheels follows a straight line starting from an initial condition at $Z(t = 0)$ and completion with a final condition at $Z(t = t_\mathrm{f})$, the trajectory of the robot can be stated as in the following equation:

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} \dfrac{x_\mathrm{f} - x_\mathrm{o}}{t_\mathrm{f}} \\ \dfrac{y_\mathrm{f} - y_\mathrm{o}}{t_\mathrm{f}} \\ \dfrac{\theta_\mathrm{f} - \theta_\mathrm{o}}{t_\mathrm{f}} \end{bmatrix} t + \begin{bmatrix} x_\mathrm{o} \\ y_\mathrm{o} \\ \theta_\mathrm{o} \end{bmatrix} \quad t \in \begin{bmatrix} 0 & t_\mathrm{f} \end{bmatrix} \tag{B.1}$$

Based on the dynamic equation of motion of the SAOWMR utilizes n-omnidirectional wheels stated in equation (3.23) and detailed in the third chapter of this thesis, the control vector U so that the robot can follows a straight line can be found by substituting equation (B.1) in the dynamic model of the robot as follows:

$$\mathrm{U} = \frac{3\beta}{n\alpha t_\mathrm{f}} \begin{bmatrix} -(x_\mathrm{f} - x_\mathrm{o})\sin\left(\dfrac{\theta_\mathrm{f} - \theta_\mathrm{o}}{t_\mathrm{f}} t + \theta_\mathrm{o}\right) + (y_\mathrm{f} - y_\mathrm{o})\cos\left(\dfrac{\theta_\mathrm{f} - \theta_\mathrm{o}}{t_\mathrm{f}} t + \theta_\mathrm{o}\right) + L(\theta_\mathrm{f} - \theta_\mathrm{o}) \\ \vdots \\ -(x_\mathrm{f} - x_\mathrm{o})\sin\left((i-1)\varphi + \dfrac{\theta_\mathrm{f} \text{-} \theta_\mathrm{o}}{t_\mathrm{f}} t + \theta_\mathrm{o}\right) + (y_\mathrm{f} - y_\mathrm{o})\cos\left((i-1)\varphi + \dfrac{\theta_\mathrm{f} - \theta_\mathrm{o}}{t_\mathrm{f}} t + \theta_\mathrm{o}\right) + L(\theta_\mathrm{f} - \theta_\mathrm{o}) \\ \vdots \\ -(x_\mathrm{f} - x_\mathrm{o})\sin\left((n-1)\varphi + \dfrac{\theta_\mathrm{f} - \theta_\mathrm{o}}{t_\mathrm{f}} t + \theta_\mathrm{o}\right) + (y_\mathrm{f} - y_\mathrm{o})\cos\left((n-1)\varphi + \dfrac{\theta_\mathrm{f} - \theta_\mathrm{o}}{t_\mathrm{f}} t + \theta_\mathrm{o}\right) + L(\theta_\mathrm{f} - \theta_\mathrm{o}) \end{bmatrix} \tag{B.2}$$

The maximum input voltage u_o applied to the terminals input of the DC motors is considered by utilizing the following equation:

$$u_\mathrm{o} = \frac{3\beta}{n\alpha t_\mathrm{f}} \left(\sqrt{(x_\mathrm{f} - x_\mathrm{o})^2 + (y_\mathrm{f} - y_\mathrm{o})^2} + L(\theta_\mathrm{f} - \theta_\mathrm{o}) \right) \tag{B.3}$$

Subsequently, the total steering time t_f so that the maximum input voltage u_o getting the maximum limit u_max able to be calculated based on the equation (B.3) as follows:

$$t_{f_o} = \frac{3\beta}{n\alpha u_{\max}}\left(\sqrt{\left(x_f - x_o\right)^2 + \left(y_f - y_o\right)^2} + \left(\theta_f - \theta_o\right)L\right) \tag{B.4}$$

Appendix C: Matlab function code “*acceleration_bound*” to find the critical value of the total maneuvering time so that the maximum acceleration amplitude reaching the maximum constraint limit

```
Function tf_L=acceleration_bound(zo1,zo2,zo3,zf1,zf2,zf3,vo1,vo2,vo3,vf1,vf2,vf3)
% [zo1,zo2,zo3] initial configuration for robot position
% [vo1,vo2,vo3] initial configuration for robot acceleration
% [zf1,zf2,zf3] final configuration for robot position
% [vf1,vf2,vf3] final configuration for robot acceleration
alfa_m=2; %maximum acceleration amplitude
syms ft1 tf2 % total time
syms t; % time
lam_4=(alfa_m^2)/4;
lam_2=-(((2*vo1+vf1)^2)+ ((2*vo2+vf2)^2));
lam_1=6*((zf1-zo1)*((2*vo1)+vf1)+ (zf2-zo2)*((2*vo2)+vf2));
lam_0=-9*(((zf1-zo1)^2)+((zf2-zo2)^2));
a1=(1/(t^2))*(((-2/t)*(zf1-zo1))+vo1+vf1);
a2=(1/(t^2))*(((-2/t)*(zf2-zo2))+vo2+vf2);
b1=(1/(t))*(((3/t)*(zf1-zo1))+(-2*vo1)+(-vf1));
b2=(1/(t))*(((3/t)*(zf2-zo2))+(-2*vo2)+(-vf2));
delta1=36*((a1^2)+(a2^2));
delta2=24*((a1*b1)+(a2*b2));
delta3=4*((b1^2)+(b2^2));
s1=double(solve((lam_4*(ft1^4))+(lam_2*(ft1^2))+(lam_1*(ft1^1))+(lam_0)==0));
s2=double(solve((delta1*(t^2))+(delta2*(t))+delta3-(alfa_m^2)==0));
s1=s1(real(s1)>0&imag(s1)==0);
s2=s2(real(s2)>0&imag(s2)==0);
tf_L=max(s1,s2);
end
```

Appendix D: Simulated Annealing Algorithm Function Matlab code

```
function [x0,f0]=simulated_anl(f,x0,l,u,Mmax,TolFun)
if nargin<6
   TolFun=1e-2;  %tolerance
   if nargin<5
       Mmax=50;
   end
end
x=x0; % starting point
fx=feval(f,x);f0=fx;

for m=0:Mmax

    T=m/Mmax;
    mu=10^(T*100);

   for k=0:50

        dx=mu_inv(2*rand(size(x))-1,mu).*(u-l);
        x1=x+dx;

        x1=(x1 < l).*l+(l <= x1).*(x1 <= u).*x1+(u < x1).*u;

        fx1=feval(f,x1);
        df=fx1-fx;

        if (df < 0 || rand < exp(-T*df/(abs(fx)+eps)/TolFun))==1
            x=x1;fx=fx1;
        end

        if fx1 < f0 ==1
        x0=x1;f0=fx1;
        end
    end
end
end
function x=mu_inv(y,mu)
%This function is used to generate new point according to lower and upper
%and a random factor proportional to current point.
x=(((1+mu).^abs(y)-1)/mu).*sign(y);
end
```

Appendix E: Matlab function code “*bisectionMethod*”

```
function c = bisectionMethod(f,a,b,error)
c=(a+b)/2;
i=0;
while abs(f(c))>error
    i=i+1;
    if f(c)*f(a)>0
        a=c;
    else
        b=c;
    end
    c=(a+b)/2;
end
```

Appendix F: Matlab function code “*maximum_point_control_input*” to find the critical value of the total maneuvering time so that the maximum input voltage applied on the input terminals of the wheels’ Dc motor getting the maximum limit

```
function maximum_p =maximum_point_control_input(tf)

%% experiment 1
zo=[-1;0;pi/4];
zf=[0.5;-1.5;-pi/2];
vo=[0.1;-0.5;0.2];
vf=[-0.8;-0.1;0.4];
% %% experiment 2
% zo=[-2.5;1.7;-pi/2];
% zf=[-1.1;0;-pi/6];
% vo=[-0.6;0.5;-0.6];
% vf=[-0.1;0.8;0.2];

zo1=zo(1);
zo2=zo(2);
zo3=zo(3);
zf1=zf(1);
zf2=zf(2);
zf3=zf(3);
vo1=vo(1);
vo2=vo(2);
vo3=vo(3);
vf1=vf(1);
vf2=vf(2);
vf3=vf(3);

L=0.09;             % Robot Platform's radius
Umax=14.8;          % limitation on the Dc motor power source
alfa=10;            % DC motor's characteristic constant
beta=146;           % tDC motor's characteristic constant

x0=(beta/(alfa*Umax))*(sqrt(((zf1-zo1)^2)+((zf2-zo2)^2))+((zf3-zo3)*L)); % starting
point
max_point= @(x)min(min(-1*abs((657*vo3)/500-(vf3+2*vo3-(3*zf3-
3*zo3)/tf)/(216*tf)+x^2*((1971*(vf3+vo3-(2*zf3- 2*zo3)/tf))/(500*tf^2) + (219*cos(zo3
+ vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf)*(vf2 + vo2 - (2*zf2 - 2*zo2)/tf))/(5*tf^2) - (219*sin(zo3 + vo3*x +
(x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf)*(vf1 + vo1 - (2*zf1 - 2*zo1)/tf))/(5*tf^2)) - x*((657*(vf3 + 2*vo3 -
(3*zf3 - 3*zo3)/tf))/(250*tf) - (vf3 + vo3 - (2*zf3 - 2*zo3)/tf)/(72*tf^2) +
(2*cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3
- (3*zf3 - 3*zo3)/tf))/tf)*((219*(vf2 + 2*vo2 - (3*zf2 - 3*zo2)/tf))/(5*tf) -
(381*(vf2 + vo2 - (2*zf2 - 2*zo2)/tf))/(250*tf^2)))/3 - (2*sin(zo3 + vo3*x +
(x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf)*((219*(vf1 + 2*vo1 - (3*zf1 - 3*zo1)/tf))/(5*tf) - (381*(vf1 + vo1 -
(2*zf1 - 2*zo1)/tf))/(250*tf^2)))/3) + (2*cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3
- 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)*((219*vo2)/10 -
(127*(vf2 + 2*vo2 - (3*zf2 - 3*zo2)/tf))/(250*tf)))/3 - (2*sin(zo3 + vo3*x +
(x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf)*((219*vo1)/10 - (127*(vf1 + 2*vo1 - (3*zf1 -
3*zo1)/tf))/(250*tf)))/3),-1*abs((657*vo3)/500 - (vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf)/(216*tf) - x*((657*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/(250*tf) +
((219*(vf1 + 2*vo1 - (3*zf1 - 3*zo1)/tf))/(5*tf) - (381*(vf1 + vo1 - (2*zf1 -
2*zo1)/tf))/(250*tf^2))*(sin(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 -
2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 - (3^(1/2)*cos(zo3
+ vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf))/3) - ((219*(vf2 + 2*vo2 - (3*zf2 - 3*zo2)/tf))/(5*tf) - (381*(vf2 +
vo2 - (2*zf2 - 2*zo2)/tf))/(250*tf^2))*(cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 -
2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 + (3^(1/2)*sin(zo3
+ vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf))/3) - (vf3 + vo3 - (2*zf3 - 2*zo3)/tf)/(72*tf^2)) + ((219*vo1)/10 -
(127*(vf1 + 2*vo1 - (3*zf1 - 3*zo1)/tf))/(250*tf))*(sin(zo3 + vo3*x + (x^3*(vf3 + vo3
- (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 -
(3^(1/2)*cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 +
2*vo3 - (3*zf3 - 3*zo3)/tf))/tf))/3) - ((219*vo2)/10 - (127*(vf2 + 2*vo2 - (3*zf2 -
3*zo2)/tf))/(250*tf))*(cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2
- (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 + (3^(1/2)*sin(zo3 + vo3*x +
(x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf))/3) + x^2*((1971*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/(500*tf^2) +
(657*(sin(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 +
2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 - (3^(1/2)*cos(zo3 + vo3*x + (x^3*(vf3 + vo3 -
(2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf))/3)*(vf1 +
vo1 - (2*zf1 - 2*zo1)/tf))/(10*tf^2) - (657*(cos(zo3 + vo3*x + (x^3*(vf3 + vo3 -
(2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 +
(3^(1/2)*sin(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 +
2*vo3 - (3*zf3 - 3*zo3)/tf))/tf))/3)*(vf2 + vo2 - (2*zf2 - 2*zo2)/tf))/(10*tf^2)))),-
1*abs((657*vo3)/500 - (vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf)/(216*tf) - x*((657*(vf3 +
2*vo3 - (3*zf3 - 3*zo3)/tf))/(250*tf) + ((219*(vf1 + 2*vo1 - (3*zf1 -
3*zo1)/tf))/(5*tf) - (381*(vf1 + vo1 - (2*zf1 - 2*zo1)/tf))/(250*tf^2))*(sin(zo3 +
vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf)/3 + (3^(1/2)*cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 -
2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf))/3) - ((219*(vf2 +
2*vo2 - (3*zf2 - 3*zo2)/tf))/(5*tf) - (381*(vf2 + vo2 - (2*zf2 -
2*zo2)/tf))/(250*tf^2))*(cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 -
2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 - (3^(1/2)*sin(zo3
+ vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf))/3) - (vf3 + vo3 - (2*zf3 - 2*zo3)/tf)/(72*tf^2)) + ((219*vo1)/10 -
(127*(vf1 + 2*vo1 - (3*zf1 - 3*zo1)/tf))/(250*tf))*(sin(zo3 + vo3*x + (x^3*(vf3 + vo3
- (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 +
(3^(1/2)*cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 +
2*vo3 - (3*zf3 - 3*zo3)/tf))/tf))/3) - ((219*vo2)/10 - (127*(vf2 + 2*vo2 - (3*zf2 -
3*zo2)/tf))/(250*tf))*(cos(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2
- (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 - (3^(1/2)*sin(zo3 + vo3*x +
(x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 -
3*zo3)/tf))/tf))/3) + x^2*((1971*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/(500*tf^2) +
(657*(sin(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 +
2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 + (3^(1/2)*cos(zo3 + vo3*x + (x^3*(vf3 + vo3 -
(2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf))/3)*(vf1 +
vo1 - (2*zf1 - 2*zo1)/tf))/(10*tf^2) - (657*(cos(zo3 + vo3*x + (x^3*(vf3 + vo3 -
(2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 + 2*vo3 - (3*zf3 - 3*zo3)/tf))/tf)/3 -
(3^(1/2)*sin(zo3 + vo3*x + (x^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tf))/tf^2 - (x^2*(vf3 +
2*vo3 - (3*zf3 - 3*zo3)/tf))/tf))/3)*(vf2 + vo2 - (2*zf2 - 2*zo2)/tf))/(10*tf^2))));
s=simulated_anl(max_point,x0,0,tf,50,10^-3);
maximum_p=abs(max_point(s))-Umax;
end
```

Appendix G: The proposed method to solve the optimal control problem provided in Chapter 4

```
clc
clear all
close all

%% parameters
m=2.54;             % mass of the Vehicle
I=6.25*(10^-3);     % mass-moment of inertia of the vehicle
amax=2;             % limitation on the acceleration amplitude
L=0.09;             % Robot Platform's radius
Umax=14.8;          % limitation on the Dc motor power source
alfa=10;            % DC motor's characteristic constant
beta=146;           % tDC motor's characteristic constant
h=0.01;         %% step size

%% experiment 1
zo=[-1;0;pi/4];
zf=[0.5;-1.5;-pi/2];
vo=[0.1;-0.5;0.2];
vf=[-0.8;-0.1;0.4];

%% experiment 2
% zo=[-2.5;1.7;-pi/2];
% zf=[-1.1;0;-pi/6];
% vo=[-0.6;0.5;-0.6];
% vf=[-0.1;0.8;0.2];

zo1=zo(1);
zo2=zo(2);
zo3=zo(3);
zf1=zf(1);
zf2=zf(2);
zf3=zf(3);
vo1=vo(1);
vo2=vo(2);
vo3=vo(3);
vf1=vf(1);
vf2=vf(2);
vf3=vf(3);

tf_a=acceleration_bound(zo1,zo2,zf1,zf2,vo1,vo2,vf1,vf2); %acceleration constraint
tf_u=bisectionMethod(@maximum_point_control_input,2,5,0.01); %saturation constraint
tff=max(tf_a,tf_u);  %total maneuvering time;

M=[(m/alfa) 0 0;0 (m/alfa) 0;0 0 (I/(L*alfa))];
Minv=inv(M);
A=[((3*beta)/(2*alfa)) 0 0;0 ((3*beta)/(2*alfa)) 0;0 0 ((3*beta*L)/(alfa)) ];
C=4*eye(3);
K=2*eye(3);

x=sym('x','positive'); %time
Zref1=zo1 + vo1*x + (x.^3*(vf1 + vo1 - (2*zf1 - 2*zo1)/tff))/tff.^2 - (x.^2*(vf1 +
2*vo1 - (3*zf1 - 3*zo1)/tff))/tff;
```

```
Zref2=zo2 + vo2*x + (x.^3*(vf2 + vo2 - (2*zf2 - 2*zo2)/tff))/tff.^2 - (x.^2*(vf2 +
2*vo2 - (3*zf2 - 3*zo2)/tff))/tff;
Zref3=zo3 + vo3*x + (x.^3*(vf3 + vo3 - (2*zf3 - 2*zo3)/tff))/tff.^2 - (x.^2*(vf3 +
2*vo3 - (3*zf3 - 3*zo3)/tff))/tff;
Zref=[Zref1;Zref2;Zref3];

dZref1=diff(Zref1);
ddZref1=diff(diff(Zref1));
dZref2=diff(Zref2);
ddZref2=diff(diff(Zref2));
dZref3=diff(Zref3);
ddZref3=diff(diff(Zref3));

g=(M*(diff(diff(Zref))))+(A*diff(Zref));
W=[-sin(Zref3) cos(Zref3) L;-sin(Zref3+(2*(pi/3))) cos(Zref3+(2*(pi/3))) L;-
sin(Zref3+(4*(pi/3))) cos(Zref3+(4*(pi/3))) L]*[dZref1;dZref2;dZref3];

%% Finding the Control Input
P=[-sin(Zref3) -sin(Zref3+((2*pi)/3)) -sin(Zref3+((4*pi)/3)) ;cos(Zref3)
cos(Zref3+((2*pi)/3)) cos(Zref3+((4*pi)/3));1 1 1];
Pinv=[-(2*sin(Zref3))/3     (2*cos(Zref3))/3      1/3
sin(Zref3)/3-(3^(1/2)*cos(Zref3))/3  -cos(Zref3)/3-(3^(1/2)*sin(Zref3))/3  1/3
sin(Zref3)/3+(3^(1/2)*cos(Zref3))/3  (3^(1/2)*sin(Zref3))/3-cos(Zref3)/3   1/3];

u=simplify(Pinv*g);
F=(alfa*u)-(beta*W);
Current=(0.02/0.293)*F;
IP=[u(1)*Current(1);u(2)*Current(2);u(3)*Current(3)];

u1=subs(u(1),x,0:h:tff);
u1=double(subs(u1,tf,tff));
[maximum1 ,index_max1]=max(u1);
[minimum1 ,index_min1]=min(u1);
maximum_value1=max(abs(minimum1),maximum1);
fitnessfunction1=maximum_value1-14.8;
control1=plot(0:h:tff,u1,'b');
hold on;

u2=subs(u(2),x,0:h:tff);
u2=double(subs(u2,tf,tff));
[maximum2 ,index_max2]=max(u2);
[minimum2 ,index_min2]=min(u2);
maximum_value2=max(abs(minimum2),maximum2);
fitnessfunction2=maximum_value2-14.8;
control2=plot(0:h:tff,u2,'r');
hold on;

u3=subs(u(3),x,0:h:tff);
u3=double(subs(u3,tf,tff));
[maximum3 ,index_max3]=max(u3);
[minimum3 ,index_min3]=min(u3);
maximum_value3=max(abs(minimum3),maximum3);
fitnessfunction3=maximum_value3-14.8;
control3=plot(0:h:tff,u3,'g');
xlim([0 tff])
ylim([-15 15])
title('Control')
ylabel('control[volts]')
xlabel('t[Sec]')
```

```
control1(1).LineWidth = 2;
control2(1).LineWidth = 2;
control3(1).LineWidth = 2;
legend('U1','U2','U3')

maximum=max(maximum_value1,max(maximum_value2,maximum_value3));
fitnessfunction=maximum-14.8;

%% trajectory x_y_t
Zref1=subs(Zref1,x,0:h:tff);
Zref1=double(subs(Zref1,tf,tff));
Zref2=subs(Zref2,x,0:h:tff);
Zref2=double(subs(Zref2,tf,tff));
Zref3=subs(Zref3,x,0:h:tff);
Zref3=double(subs(Zref3,tf,tff));
figure;
zrefx=plot(0:h:tff,Zref1,'b');
xlim([0 tff])

hold on
zrefy=plot(0:h:tff,Zref2,'r');
xlim([0 tff])
hold on
zrefteta=plot(0:h:tff,Zref3,'k');
xlim([0 tff])
title('Reference Trajectory')
ylabel('displacement')
xlabel('t[Sec]')
zrefx(1).LineWidth = 2;
zrefy(1).LineWidth = 2;
zrefteta(1).LineWidth = 2;
legend('X','Y','\theta')
%% X against Y
figure;
axis equal
xagainsty= plot(Zref1,Zref2,'k--');
txt1 = '   Start Point';
text(zo1,zo2,txt1)
txt11 = '   End Point';
text(zf1,zf2,txt11)
xagainsty(1).LineWidth = 0.5;

hold on;

plot([zo1 L*cos(zo3)+zo1],[zo2 L*sin(zo3)+zo2],'b')
hold on;
plot([zo1 L*cos((zo3)+(2*(pi/3)))+zo1],[zo2 L*sin((zo3)+(2*(pi/3)))+zo2],'r')
hold on;
plot([zo1 L*cos((zo3)+(4*(pi/3)))+zo1],[zo2 L*sin((zo3)+(4*(pi/3)))+zo2],'k')
axis equal

hold on;
plot([zf1 L*cos(zf3)+zf1],[zf2 L*sin(zf3)+zf2],'--b')
hold on;
plot([zf1 L*cos((zf3)+(2*(pi/3)))+zf1],[zf2 L*sin((zf3)+(2*(pi/3)))+zf2],'--r')
hold on;
plot([zf1 L*cos((zf3)+(4*(pi/3)))+zf1],[zf2 L*sin((zf3)+(4*(pi/3)))+zf2],'--k')

%% velocity
```

```
VX=subs(dZref1,x,0:h:tff);
VX=double(subs(VX,tf,tff));
VY=subs(dZref2,x,0:h:tff);
VY=double(subs(VY,tf,tff));
TETAPRIME=subs(dZref3,x,0:h:tff);
TETAPRIME=subs(TETAPRIME,tf,tff);

figure;
velocityx=plot(0:h:tff,VX,'b');
xlim([0 tff])
ylim([-1 1.5])
hold on
velocityy=plot(0:h:tff,VY,'r');
hold on
tetaprim=plot(0:h:tff,TETAPRIME,'k');
title('Linear and Angular Velocities')
ylabel('velocity')
xlabel('t[Sec]')
velocityx(1).LineWidth = 2;
velocityy(1).LineWidth = 2;
tetaprim(1).LineWidth = 2;
legend('VX[m/s]','VY[m/s]','angular velocity[rad/sec]')

%% ACCELERATION
AX=subs(ddZref1,x,0:h:tff);
AX=double(subs(AX,tf,tff));
AY=subs(ddZref2,x,0:h:tff);
AY=double(subs(AY,tf,tff));
AA=((AX.^2)+(AY.^2));

figure;
accelerationx=plot(0:h:tff,AX,'b');
xlim([0 tff])
hold on
accelerationy=plot(0:h:tff,AY,'r');
xlim([0 tff])
sqrtacceleration=plot(0:h:tff,AA,'k');
xlim([0 tff])
ylim([-2 5])
title('Accelerations')
ylabel('Acceleration[m/S^2]')
xlabel('t[Sec]')
accelerationx(1).LineWidth =2;
accelerationy(1).LineWidth = 2;
sqrtacceleration(1).LineWidth = 2;
legend('Ax','Ay','Ax^2 + Ay^2')

%% wheels velocity
w1=subs(W(1),x,0:h:tff);
w1=double(subs(w1,tf,tff));
w2=subs(W(2),x,0:h:tff);
w2=double(subs(w2,tf,tff));
w3=subs(W(3),x,0:h:tff);
w3=double(subs(w3,tf,tff));

figure;
velocityw1=plot(0:h:tff,w1,'b-.');
hold on
velocityw2=plot(0:h:tff,w2,'r-.');
```

```
hold on
velocityw3=plot(0:h:tff,w3,'k-.');
xlim([0 tff])
title('Wheels Velocity ')
ylabel('velocity value [m/s]')
xlabel('t[Sec]')
velocityw1(1).LineWidth = 1.5;
velocityw2(1).LineWidth = 1.5;
velocityw3(1).LineWidth = 1.5;
legend('Wheel 1 velocity','Wheel 2 velocity','Wheel 3 velocity')

%% wheels force
f1=subs(F(1),x,0:h:tff);
f1=double(subs(f1,tf,tff));
f2=subs(F(2),x,0:h:tff);
f2=double(subs(f2,tf,tff));
f3=subs(F(3),x,0:h:tff);
f3=double(subs(f3,tf,tff));

figure;
force1=plot(0:h:tff,f1,'b-.');
hold on
forcr2=plot(0:h:tff,f2,'r-.');
hold on
force3=plot(0:h:tff,f3,'k-.');
xlim([0 tff])
title('Force ')
ylabel('Force [Newton]')
xlabel('t[Sec]')
force1(1).LineWidth = 1.5;
force2(1).LineWidth = 1.5;
force3(1).LineWidth = 1.5;
legend('Wheel 1 FORCE','Wheel 2 FORCE','Wheel 3 FORCE')

%% Ip input power
ip1=subs(IP(1),x,0:h:tff);
ip1=double(subs(ip1,tf,tff));
ip2=subs(IP(2),x,0:h:tff);
ip2=double(subs(ip2,tf,tff));
ip3=subs(IP(3),x,0:h:tff);
ip3=double(subs(ip3,tf,tff));

power11=zeros(1,length(Zref3));
power22=zeros(1,length(Zref3));
power33=zeros(1,length(Zref3));

for i=1:1:length(Zref3)
     if ip1(i)>0
 power11(i)=ip1(i);
     else
 power11(i)=-ip1(i);
     end
end
for i=1:1:length(Zref3)
     if ip2(i)>0
 power22(i)=ip2(i);
     else
 power22(i)=-ip2(i);
     end
```

```
end
for i=1:1:length(Zref3)
     if ip3(i)>0
 power33(i)=ip3(i);
     else
 power33(i)=-ip3(i);
     end
 end


figure;
input_power1=plot(0:h:tff,ip1,'b-.');
hold on
input_power2=plot(0:h:tff,ip2,'r-.');
hold on
input_power3=plot(0:h:tff,ip3,'k-.');
xlim([0 tff])
title('Force ')
ylabel('Force [Newton]')
xlabel('t[Sec]')
input_power1(1).LineWidth = 1.5;
input_power2(1).LineWidth = 1.5;
input_power3(1).LineWidth = 1.5;
legend('input power','input power','input power')

figure;
powerx11=plot(0:h:tff,power11,'-.');
xlim([0 tff])
hold on
powerx22=plot(0:h:tff,power22,':');
xlim([0 tff])
hold on
powerx33=plot(0:h:tff,power33,'--');
xlim([0 tff])

title('Input Power ')
ylabel('Input Power [VA]')
xlabel('t[Sec]')
powerx11(1).LineWidth = 1.5;
powerx22(1).LineWidth = 1.5;
powerx33(1).LineWidth = 1.5;

legend('Power 1','Power 2','Power 3')

total_energy=double(trapz(0:h:tff,power11)+trapz(0:h:tff,power22)+trapz(0:h:tff,power
33));

%% Ploting the Error

e1 = zeros(1,length(0:h:tff));
e2 = zeros(1,length(0:h:tff));
e3 = zeros(1,length(0:h:tff));
figure;
errorploty1=plot(0:h:tff,e1,'b--');
xlim([0 tff])
ylim([-0.05 0.45])
hold on
errorploty2=plot(0:h:tff,e2,'r--');
hold on
```

```
errorploty3=plot(0:h:tff,e3,'k--');
title('ERROR')
ylabel('error')
xlabel('t[Sec]')
errorploty1(1).LineWidth = 2;
errorploty2(1).LineWidth = 2;
errorploty3(1).LineWidth = 2;
legend('error x','error y','error \theta')
```

Appendix H: Motion planning concerning the numerical test in chapter 6 by utilizing the Artificial Potential Field method.

```
clc
clear
close all

%......................Artificial Potential Field........................

Robot_Coordinate     = [9 2]      ;% Rabot Coordinate = [x-axis y-axis]
Obstacle1_Coordinate = [1.5 6]    ;% Obstacle1 Coordinate = [x-axis y-axis]
Obstacle2_Coordinate = [4 4]      ;% Obstacle2 Coordinate = [x-axis y-axis]
Obstacle3_Coordinate = [6 8]      ;% Obstacle2 Coordinate = [x-axis y-axis]
Obstacle4_Coordinate = [7 5]      ;% Obstacle2 Coordinate = [x-axis y-axis]

Goal_Coordinate      = [1 8.5]    ;% Target Coordinate = [x-axis y-axis]
Sensor_Range         =1           ;% Sensor Range being used
Step_Size    = 0.4*Sensor_Range   ;% Radius of Circle for Artificial Points


Obstacle             = [1 1.9]    ;% Obstacle = [Aplha Mew][0.5 0.5]
Goal                 = [1 0.1]    ;% Target   = [Alpha Mew][4 70]

NPTS                 = 60                  ;% Number of Artificial Points %60
StepDegree           = 360/NPTS            ;% Step Degree in Degree for Artificial
Points
Confirm_Message      = 'Solution Exists'   ;% Message Displayed if a Good Artificial
Point Exists
Error_Message        = 'No Solution Exists';% Message Displayed if no Good Artificial
Point Exists
Bacteria_x           = Robot_Coordinate(1) ;% Artificial Best Point x
Bacteria_y           = Robot_Coordinate(2) ;% Artificial Best Point y
%................................Plot...................................

hold on
axis([0 10 0 10])

%...........................Create a Target..............................

backx  = Goal_Coordinate(1) - 0.15;backy1 = Goal_Coordinate(2) + 0.15;
frontx = Goal_Coordinate(1) + 0.15;fronty1 = Goal_Coordinate(2) + 0.15;
middlex = Goal_Coordinate(1);middley1 = Goal_Coordinate(2)+ 0.15;
middley2 = Goal_Coordinate(2)- 0.15;
tri2 = [backx middlex frontx ;backy1 middley1 fronty1 ];
plot(tri2(1,:), tri2(2,:));
tri3 = [middlex middlex ;middley1 middley2];
plot(tri3(1,:), tri3(2,:));

txt1 = '   Start Point';
text(Robot_Coordinate(1),Robot_Coordinate(2),txt1)
txt11 = '   End Point';
text(Goal_Coordinate(1),Goal_Coordinate(2),txt11)

............Create a purple transparent circular Obstacle...............
```

```
xc = Obstacle1_Coordinate(1);
yc = Obstacle1_Coordinate(2);
r = 0.4;
x = r*sin(-pi:0.2*pi:pi) + xc;
y = r*cos(-pi:0.2*pi:pi) + yc;
c = [0.1 0.1 0.1];
fill(x, y, c, 'FaceAlpha', 0.5)

xc = Obstacle2_Coordinate(1);
yc = Obstacle2_Coordinate(2);
r = 0.4;
x = r*sin(-pi:0.2*pi:pi) + xc;
y = r*cos(-pi:0.2*pi:pi) + yc;
c = [0.1 0.1 0.1];
fill(x, y, c, 'FaceAlpha', 0.5)

xc = Obstacle3_Coordinate(1);
yc = Obstacle3_Coordinate(2);
r = 0.4;
x = r*sin(-pi:0.2*pi:pi) + xc;
y = r*cos(-pi:0.2*pi:pi) + yc;
c = [0.1 0.1 0.1];
fill(x, y, c, 'FaceAlpha', 0.5)

xc = Obstacle4_Coordinate(1);
yc = Obstacle4_Coordinate(2);
r = 0.4;
x = r*sin(-pi:0.2*pi:pi) + xc;
y = r*cos(-pi:0.2*pi:pi) + yc;
c = [0.1 0.1 0.1];
fill(x, y, c, 'FaceAlpha', 0.5)

text(2,6,'Obstacle 1')
text(4,4,'Obstacle 2')
text(6,8,'Obstacle 3')
text(7,5,'Obstacle 4')

counter=0;
DTG = 1;
while(DTG > 0.2)
    counter=counter+1;
    xxx(counter)=Robot_Coordinate(1)
    yyy(counter)=Robot_Coordinate(2)
    % .............................Create a Robot...........................

    backx  = Robot_Coordinate(1) - 0.15;backy1 = Robot_Coordinate(2) + 0.15;
    backy2 = Robot_Coordinate(2) - 0.15;frontx = Robot_Coordinate(1) + 0.075;
    fronty1 = Robot_Coordinate(2) + 0.15;fronty2 = Robot_Coordinate(2);
    tri1 = [backx backx frontx frontx backx frontx;backy2 backy1 fronty1 fronty2
fronty2 backy2];
    %.........................Potential Calculations.......................

    J_ObstT = Obstacle(1)*exp(-Obstacle(2)*((Robot_Coordinate(1)-
Obstacle1_Coordinate(1))^2+(Robot_Coordinate(2)-Obstacle1_Coordinate(2))^2));
    J_ObstT = J_ObstT + Obstacle(1)*exp(-Obstacle(2)*((Robot_Coordinate(1)-
Obstacle2_Coordinate(1))^2+(Robot_Coordinate(2)-Obstacle2_Coordinate(2))^2));
    J_ObstT = J_ObstT + Obstacle(1)*exp(-Obstacle(2)*((Robot_Coordinate(1)-
Obstacle3_Coordinate(1))^2+(Robot_Coordinate(2)-Obstacle3_Coordinate(2))^2));
```

```
    J_ObstT = J_ObstT + Obstacle(1)*exp(-Obstacle(2)*((Robot_Coordinate(1)-
Obstacle4_Coordinate(1))^2+(Robot_Coordinate(2)-Obstacle4_Coordinate(2))^2));
    J_GoalT = -Goal(1)*exp(-Goal(2)*((Robot_Coordinate(1)-
Goal_Coordinate(1))^2+(Robot_Coordinate(2)-Goal_Coordinate(2))^2));
    JT = J_ObstT + J_GoalT;
    %...........................Distance to Goal...........................
    DTG = sqrt((Robot_Coordinate(1)-Goal_Coordinate(1))^2+(Robot_Coordinate(2)-
Goal_Coordinate(2))^2);

    %...........................Artificial Points..........................

    Theta = zeros(1,NPTS);
    Theta(1) = StepDegree;
    for i=1:NPTS-1
        Theta(i+1) = Theta(i) + StepDegree;
    end
    Bacteria_x = zeros(1,NPTS);
    Bacteria_y = zeros(1,NPTS);
    for i=1:NPTS
        Bacteria_x(i) = Robot_Coordinate(1)+(Step_Size*cos(pi*Theta(i)/180));
        Bacteria_y(i) = Robot_Coordinate(2)+(Step_Size*sin(pi*Theta(i)/180));
    end
    plot(Bacteria_x,Bacteria_y,'k.')
    pause(0.1);
    %........Calculating Cost Function and Distance of Artificial Point....

    [m,n] = size(Bacteria_x);
    J_ObstT_Bacteria = zeros(1,n);
    J_GoalT_Bacteria = zeros(1,n);
    JT_Bacteria = zeros(1,n);
    DTG_Bacteria = zeros(1,n);
    for i=1:n
        J_ObstT_Bacteria(i) = Obstacle(1)*exp(-Obstacle(2)*((Bacteria_x(i)-
Obstacle1_Coordinate(1))^2+(Bacteria_y(i)-Obstacle1_Coordinate(2))^2));
        J_ObstT_Bacteria(i) = J_ObstT_Bacteria(i) + Obstacle(1)*exp(-
Obstacle(2)*((Bacteria_x(i)-Obstacle2_Coordinate(1))^2+(Bacteria_y(i)-
Obstacle2_Coordinate(2))^2));
        J_ObstT_Bacteria(i) = J_ObstT_Bacteria(i) + Obstacle(1)*exp(-
Obstacle(2)*((Bacteria_x(i)-Obstacle3_Coordinate(1))^2+(Bacteria_y(i)-
Obstacle3_Coordinate(2))^2));
        J_ObstT_Bacteria(i) = J_ObstT_Bacteria(i) + Obstacle(1)*exp(-
Obstacle(2)*((Bacteria_x(i)-Obstacle4_Coordinate(1))^2+(Bacteria_y(i)-
Obstacle4_Coordinate(2))^2));
        J_GoalT_Bacteria(i) = -Goal(1)*exp(-Goal(2)*((Bacteria_x(i)-
Goal_Coordinate(1))^2+(Bacteria_y(i)-Goal_Coordinate(2))^2));
        JT_Bacteria(i) = J_ObstT_Bacteria(i) + J_GoalT_Bacteria(i);
        DTG_Bacteria(i) = sqrt((Bacteria_x(i)-Goal_Coordinate(1))^2+(Bacteria_y(i)-
Goal_Coordinate(2))^2);
    end
    J_GoalT_Bacteria;
    %...................Error in Cost Function & Distance.................
    err_J   = zeros(1,n);
    err_DTG = zeros(1,n);
    Fitness = zeros(1,n);
    for i=1:n
        err_J(i)   = JT_Bacteria(i)  - JT;
        err_DTG(i) = DTG_Bacteria(i) - DTG;
        Fitness(i) = -err_DTG(i);
    end
    Fitness;
    %.........................Best Point Selection.......................
    Check = 0;
    for t=1:n
        [~,k]= max(Fitness);
        if err_J(k) < 0
            if err_DTG(k)<0
                Check = Check + 1;
                Robot_Coordinate(1) = Bacteria_x(k);
                Robot_Coordinate(2) = Bacteria_y(k);
                DTG = sqrt((Robot_Coordinate(1)-
Goal_Coordinate(1))^2+(Robot_Coordinate(2)-Goal_Coordinate(2))^2);
            else
                Fitness(k) = 0;
            end
        else
            Fitness(k) = 0;
        end
    end
    Check;
    if Check == 0
        disp(Error_Message);
    else
        disp(Confirm_Message);
    end
    All_Data = zeros(NPTS,3);
    All_Data(:,1) = Robot_Coordinate(1);
    All_Data(:,2) = Robot_Coordinate(2);
    All_Data(:,3) = Theta;
    All_Data(:,4) = J_ObstT_Bacteria;
    All_Data(:,5) = J_GoalT_Bacteria;
    All_Data(:,6) = JT_Bacteria;
    All_Data(:,7) = err_J;
    All_Data(:,8) = err_DTG;
    All_Data;
    %...............................Plot...............................

    %..........................Create a Target.........................
    backx  = Goal_Coordinate(1) - 0.15;backy1 = Goal_Coordinate(2) + 0.15;
    frontx = Goal_Coordinate(1) + 0.15;fronty1 = Goal_Coordinate(2) + 0.15;
    middlex = Goal_Coordinate(1);middley1 = Goal_Coordinate(2)+ 0.15;
    middley2 = Goal_Coordinate(2)- 0.15;
    tri2 = [backx middlex frontx ;backy1 middley1 fronty1 ];
    hold on
    tri3 = [middlex middlex ;middley1 middley2];

end
Variables = {'Rx' 'Ry' 'Theta' 'J_Obst_B' 'J_GoalT' 'JT' 'err_J' 'err_DTG'};
filename = 'E:\PathPlanning_Yasim\RPO First Iteration\RPO_Data.xlsx';
```

Appendix I: Matlab code aims to generate the free collision optimal trajectory by utilizing the method proposed in chapter 6

```
%% Obstacle Avoidance using potential field and spline interpolation
clc
clear all
h=0.001; % step size
tf=11.3610;

n=1;

X=[9.0000  7.7355  6.2358  5.3581  4.1987  3.0097  2.0107 1];
Y=[2.0000  2.9704  3.3291  4.6021  5.7023  6.7729  7.7235 8.5];

x=X;
y=Y;
teta=[0 pi];
teta=teta(1):((teta(2))-(teta(1)))/(n*(length(x)-1)):teta(2);

T=0:tf/(n*(length(x)-1)):tf;

%% spline interpolations
cs1 = spline(T,X);
cs2 = spline(T,Y);
cs3 = spline(T,teta);

%% Robot parameters
m=2.54;            % mass of the Vehicle
I=6.25*(10^-3);    %mass-moment of inertia of the vehicle

amax=2;            % limitation on the acceleration amplitude
L=0.09;            % Robot Platform's radius
Umax=14.8;         % limitation on the Dc motor power source
alfa=10;           % DC motor's characteristic constant
beta=146;          % tDC motor's characteristic constant

M=[(m/alfa) 0 0;0 (m/alfa) 0;0 0 (I/(L*alfa))];
Minv=inv(M);
A=[((3*beta)/(2*alfa)) 0 0;0 ((3*beta)/(2*alfa)) 0;0 0 ((3*beta*L)/(alfa)) ];
C=4*eye(3);
K=2*eye(3);

%% zref3
tt=zeros(1,(n*(length(x)-1)));
for i=2:1:(n*(length(x)-1))
    tt(i)=tt(i-1)+(tf/(n*(length(x)-1)));
end
tt=[tt tf];
t=sym('t','positive');
Zref1 = sym('Zref1', [1 (n*(length(x)-1))]);
Zref2 = sym('Zref2', [1 (n*(length(x)-1))]);
Zref3 = sym('Zref3', [1 (n*(length(x)-1))]);
dZref1 = sym('dZref1', [1 (n*(length(x)-1))]);
dZref2 = sym('dZref2', [1 (n*(length(x)-1))]);
dZref3 = sym('dZref3', [1 (n*(length(x)-1))]);
```

```
ddZref1 = sym('ddZref1', [1 (n*(length(x)-1))]);
ddZref2 = sym('ddZref2', [1 (n*(length(x)-1))]);
ddZref3 = sym('ddZref3', [1 (n*(length(x)-1))]);
Zref = sym('Zref3', [3 (n*(length(x)-1))]);
g = sym('g', [3 (n*(length(x)-1))]);
W = sym('W', [3 (n*(length(x)-1))]);
Pinv1 = sym('Pinv1', [3 (n*(length(x)-1))]);
Pinv2 = sym('Pinv2', [3 (n*(length(x)-1))]);
Pinv3 = sym('Pinv3', [3 (n*(length(x)-1))]);
u1 = sym('u1', [1 (n*(length(x)-1))]);
u2 = sym('u2', [1 (n*(length(x)-1))]);
u3 = sym('u3', [1 (n*(length(x)-1))]);
for i=1:1:(n*(length(x)-1))
Zref1(i)=((t^3)*cs1.coefs(i,1))+((t^2)*cs1.coefs(i,2))+(cs1.coefs(i,3)*(t))+(cs1.coef
s(i,4));
dZref1(i)=diff(Zref1(i));
% ddZref1(i)=diff(dZref1(i));
Zref2(i)=((t^3)*cs2.coefs(i,1))+((t^2)*cs2.coefs(i,2))+(cs2.coefs(i,3)*(t))+(cs2.coef
s(i,4));
dZref2(i)=diff(Zref2(i));
% ddZref2(i)=diff(dZref2(i));
Zref3(i)=((t^3)*cs3.coefs(i,1))+((t^2)*cs3.coefs(i,2))+(cs3.coefs(i,3)*(t))+(cs3.coef
s(i,4));
dZref3(i)=diff(Zref3(i));
% ddZref3(i)=diff(dZref3(i));
Zref(i)=[Zref1(i);Zref2(i);Zref3(i)];
g(i)=(M*(diff(diff(Zref(i)))))+(A*diff(Zref(i)));
W(i)=[-sin(Zref3(i)) cos(Zref3(i)) L;-sin(Zref3(i)+(2*(pi/3)))
cos(Zref3(i)+(2*(pi/3))) L;-sin(Zref3(i)+(4*(pi/3))) cos(Zref3(i)+(4*(pi/3)))
L]*[dZref1(i);dZref2(i);dZref3(i)];
Pinv1(i)=[-(2*sin(Zref3(i)))/3;                (2*cos(Zref3(i)))/3;
1/3];
Pinv2(i)=[sin(Zref3(i))/3-(3^(1/2)*cos(Zref3(i)))/3;          -cos(Zref3(i))/3-
(3^(1/2)*sin(Zref3(i)))/3;     1/3];
Pinv3(i)=[sin(Zref3(i))/3+(3^(1/2)*cos(Zref3(i)))/3;
(3^(1/2)*sin(Zref3(i)))/3-cos(Zref3(i))/3;        1/3];
u1(i)=simplify(Pinv1(i)'*g(i));
u2(i)=simplify(Pinv2(i)'*g(i));
u3(i)=simplify(Pinv3(i)'*g(i));
end
Zref11=zeros((n*(length(x)-1)),length(0:h:(tf/(n*(length(x)-1)))));
Zref22=zeros((n*(length(x)-1)),length(0:h:(tf/(n*(length(x)-1)))));
Zref33=zeros((n*(length(x)-1)),length(0:h:(tf/(n*(length(x)-1)))));

u11=zeros((n*(length(x)-1)),length(0:h:(tf/(n*(length(x)-1)))));
u22=zeros((n*(length(x)-1)),length(0:h:(tf/(n*(length(x)-1)))));
u33=zeros((n*(length(x)-1)),length(0:h:(tf/(n*(length(x)-1)))));

for i=1:1:(n*(length(x)-1))
Zref11(i,:)=double(subs(Zref1(i),t,0:h:(tf/(n*(length(x)-1)))));
Zref22(i,:)=double(subs(Zref2(i),t,0:h:(tf/(n*(length(x)-1)))));
Zref33(i,:)=double(subs(Zref3(i),t,0:h:(tf/(n*(length(x)-1)))));
end

for i=1:1:(n*(length(x)-1))
u11(i,:)=double(subs(u1(i),t,0:h:(tf/(n*(length(x)-1)))));
u22(i,:)=double(subs(u2(i),t,0:h:(tf/(n*(length(x)-1)))));
u33(i,:)=double(subs(u3(i),t,0:h:(tf/(n*(length(x)-1)))));
end
```

```
u111=zeros(1,(n*(length(x)-1))*((((tf/(n*(length(x)-1)))/h)+1)));
u222=zeros(1,(n*(length(x)-1))*((((tf/(n*(length(x)-1)))/h)+1)));
u333=zeros(1,(n*(length(x)-1))*((((tf/(n*(length(x)-1)))/h)+1)));

Zref111=zeros(1,(n*(length(x)-1))*((((tf/(n*(length(x)-1)))/h)+1)));
Zref222=zeros(1,(n*(length(x)-1))*((((tf/(n*(length(x)-1)))/h)+1)));
Zref333=zeros(1,(n*(length(x)-1))*((((tf/(n*(length(x)-1)))/h)+1)));

Zref111(1,1:((((tf/(n*(length(x)-1)))/h))+1))=Zref11(1,:);
Zref222(1,1:((((tf/(n*(length(x)-1)))/h))+1))=Zref22(1,:);
Zref333(1,1:((((tf/(n*(length(x)-1)))/h))+1))=Zref33(1,:);

u111(1,1:((((tf/(n*(length(x)-1)))/h))+1))=u11(1,:);
u222(1,1:((((tf/(n*(length(x)-1)))/h))+1))=u22(1,:);
u333(1,1:((((tf/(n*(length(x)-1)))/h))+1))=u33(1,:);

for i=2:(n*(length(x)-1))
Zref111(1,(((i-1)*(((tf/(n*(length(x)-1)))/h)+1)))+1:(((i-1)*(((tf/(n*(length(x)-
1)))/h)+1)))+((((tf/(n*(length(x)-1)))/h)+1)))=Zref11(i,:);
Zref222(1,(((i-1)*(((tf/(n*(length(x)-1)))/h)+1)))+1:(((i-1)*(((tf/(n*(length(x)-
1)))/h)+1))+(((tf/(n*(length(x)-1)))/h)+1)))=Zref22(i,:);
Zref333(1,(((i-1)*(((tf/(n*(length(x)-1)))/h)+1)))+1:(((i-1)*(((tf/(n*(length(x)-
1)))/h)+1)))+((((tf/(n*(length(x)-1)))/h)+1)))=Zref33(i,:);
end

for i=2:(n*(length(x)-1))
u111(1,(((i-1)*(((tf/(n*(length(x)-1)))/h)+1)))+1:(((i-1)*(((tf/(n*(length(x)-
1)))/h)+1)))+((((tf/(n*(length(x)-1)))/h)+1)))=u11(i,:);
u222(1,(((i-1)*(((tf/(n*(length(x)-1)))/h)+1)))+1:(((i-1)*(((tf/(n*(length(x)-
1)))/h)+1))+(((tf/(n*(length(x)-1)))/h)+1)))=u22(i,:);
u333(1,(((i-1)*(((tf/(n*(length(x)-1)))/h)+1)))+1:(((i-1)*(((tf/(n*(length(x)-
1)))/h)+1)))+((((tf/(n*(length(x)-1)))/h)+1)))=u33(i,:);
end


figure;
zrefx=plot(0:tf/(length(Zref111)-1):tf,Zref111,'b-.');
xlim([0 tf])
hold on
zrefy=plot(0:tf/(length(Zref111)-1):tf,Zref222,'r:');
hold on
zrefteta=plot(0:tf/(length(Zref111)-1):tf,Zref333,'g--');
title('Reference Trajectory')
ylabel('displacement')
xlabel('t[Sec]')
zrefx(1).LineWidth = 1.5;
zrefy(1).LineWidth = 1.5;
zrefteta(1).LineWidth = 1.5;
legend('X','Y','\theta')
%% X against Y
figure;
axis equal
xagainsty= plot(Zref111,Zref222,'k');
txt1 = '   Start Point';
text(x(1),y(1),txt1)
txt11 = '   End Point';
text(x(length(x)),y(length(y)),txt11)
xagainsty(1).LineWidth = 2;
```

```
figure;
Pu1=plot(0:tf/(length(u111)-1):tf,u111,'b');
xlim([0 tf])
hold on
Pu2=plot(0:tf/(length(u222)-1):tf,u222,'r');
hold on
Pu3=plot(0:tf/(length(u333)-1):tf,u333,'g');

title('Input Voltage, Control')
ylabel('Volts')
xlabel('t[Sec]')
Pu1(1).LineWidth = 1.5;
Pu2(1).LineWidth = 1.5;
Pu3(1).LineWidth = 1.5;
legend('U1','U2','U3')
```

Appendix J: Matlab code concerning Global Sensor Fusion based on the Kalman Filter Method so that three wheeled omnidirectional mobile robot estimating the motion of the ball in dynamic environment

```
clc
clear all
%% ball initial configuration
u0 = 0.4;
w0 = 0.6;
r0 = -1;
z0 = -1;
L=0.09;   % Robot Platform's radius

dt = 0.05; % step size
tf=2.5;  % total maneuvering time
N = tf/dt; % number od samples
t = (1:(N))*dt;  %time

%% Robot1
zo_1=[1;1;pi/4];
zf_1=[0.5;-1.5;-pi/2];
vo_1=[0.1;-0.5;0.2];
vf_1=[-0.8;-0.1;0.4];

zo1_1=zo_1(1);
zo2_1=zo_1(2);
zo3_1=zo_1(3);
zf1_1=zf_1(1);
zf2_1=zf_1(2);
zf3_1=zf_1(3);
vo1_1=vo_1(1);
vo2_1=vo_1(2);
vo3_1=vo_1(3);
vf1_1=vf_1(1);
vf2_1=vf_1(2);
vf3_1=vf_1(3);
x = (1:(N))*dt;
Zref1_1=zo1_1 + vo1_1*x + (x.^3*(vf1_1 + vo1_1 - (2*zf1_1 - 2*zo1_1)/tf))/tf.^2 -
(x.^2*(vf1_1 + 2*vo1_1 - (3*zf1_1 - 3*zo1_1)/tf))/tf;
Zref2_1=zo2_1 + vo2_1*x + (x.^3*(vf2_1 + vo2_1 - (2*zf2_1 - 2*zo2_1)/tf))/tf.^2 -
(x.^2*(vf2_1 + 2*vo2_1 - (3*zf2_1 - 3*zo2_1)/tf))/tf;
Zref3_1=zo3_1 + vo3_1*x + (x.^3*(vf3_1 + vo3_1 - (2*zf3_1 - 2*zo3_1)/tf))/tf.^2 -
(x.^2*(vf3_1 + 2*vo3_1 - (3*zf3_1 - 3*zo3_1)/tf))/tf;
Zref_1=[Zref1_1;Zref2_1;Zref3_1]
xrobot_1=Zref1_1
yrobot_1=Zref2_1
%% Robot 2
zo_2=[0;0;-pi/4];
zf_2=[-0.25;1.5;pi/2];
vo_2=[0.1;0.5;-0.2];
```

```
vf_2=[-0.8;-0.1;0.4];

zo1_2=zo_2(1);
zo2_2=zo_2(2);
zo3_2=zo_2(3);
zf1_2=zf_2(1);
zf2_2=zf_2(2);
zf3_2=zf_2(3);
vo1_2=vo_2(1);
vo2_2=vo_2(2);
vo3_2=vo_2(3);
vf1_2=vf_2(1);
vf2_2=vf_2(2);
vf3_2=vf_2(3);
   x = (1:(N))*dt;
Zref1_2=zo1_2 + vo1_2*x + (x.^3*(vf1_2 + vo1_2 - (2*zf1_2 - 2*zo1_2)/tf))/tf.^2 -
(x.^2*(vf1_2 + 2*vo1_2 - (3*zf1_2 - 3*zo1_2)/tf))/tf;
Zref2_2=zo2_2 + vo2_2*x + (x.^3*(vf2_2 + vo2_2 - (2*zf2_2 - 2*zo2_2)/tf))/tf.^2 -
(x.^2*(vf2_2 + 2*vo2_2 - (3*zf2_2 - 3*zo2_2)/tf))/tf;
Zref3_2=zo3_2 + vo3_2*x + (x.^3*(vf3_2 + vo3_2 - (2*zf3_2 - 2*zo3_2)/tf))/tf.^2 -
(x.^2*(vf3_2 + 2*vo3_2 - (3*zf3_2 - 3*zo3_2)/tf))/tf;
Zref_2=[Zref1_2;Zref2_2;Zref3_2]

xrobot_2=Zref1_2
yrobot_2=Zref2_2
%% Robot 3
zo_3=[-1.5;2;pi/4];
zf_3=[-2;-1;pi/6];
vo_3=[0.1;-0.5;-0.2];
vf_3=[-0.8;-0.1;0.4];

zo1_3=zo_3(1);
zo2_3=zo_3(2);
zo3_3=zo_3(3);
zf1_3=zf_3(1);
zf2_3=zf_3(2);
zf3_3=zf_3(3);
vo1_3=vo_3(1);
vo2_3=vo_3(2);
vo3_3=vo_3(3);
vf1_3=vf_3(1);
vf2_3=vf_3(2);
vf3_3=vf_3(3);

Zref1_3=zo1_3 + vo1_3*x + (x.^3*(vf1_3 + vo1_3 - (2*zf1_3 - 2*zo1_3)/tf))/tf.^2 -
(x.^2*(vf1_3 + 2*vo1_3 - (3*zf1_3 - 3*zo1_3)/tf))/tf;
Zref2_3=zo2_3 + vo2_3*x + (x.^3*(vf2_3 + vo2_3 - (2*zf2_3 - 2*zo2_3)/tf))/tf.^2 -
(x.^2*(vf2_3 + 2*vo2_3 - (3*zf2_3 - 3*zo2_3)/tf))/tf;
Zref3_3=zo3_3 + vo3_3*x + (x.^3*(vf3_3 + vo3_3 - (2*zf3_3 - 2*zo3_3)/tf))/tf.^2 -
(x.^2*(vf3_3 + 2*vo3_3 - (3*zf3_3 - 3*zo3_3)/tf))/tf;
Zref_3=[Zref1_3;Zref2_3;Zref3_3]
sigo = 0.005;
xrobot_3=Zref1_3
yrobot_3=Zref2_3

%% Kalman Filter Algorithm
rtrue = r0 + u0*t
ztrue = z0+  w0*t
ytrue = [
 atan((ztrue-yrobot_1)./(rtrue-xrobot_1))
 sqrt( ((rtrue-xrobot_1).^2)+((ztrue-yrobot_1).^2))
 atan((ztrue-yrobot_2)./(rtrue-xrobot_2))
 sqrt( ((rtrue-xrobot_2).^2)+((ztrue-yrobot_2).^2))
 atan((ztrue-yrobot_3)./(rtrue-xrobot_3))
 sqrt( ((rtrue-xrobot_3).^2)+((ztrue-yrobot_3).^2))
 ];

% Make observation vector
yo = [atan((ztrue-yrobot_1)./(rtrue-xrobot_1)) + sigo*randn(1,N)
      sqrt( ((rtrue-xrobot_1).^2)+((ztrue-yrobot_1).^2)  )  + sigo*randn(1,N)
      atan((ztrue-yrobot_2)./(rtrue-xrobot_2)) + sigo*randn(1,N)
      sqrt( ((rtrue-xrobot_2).^2)+((ztrue-yrobot_2).^2)  )  + sigo*randn(1,N)
      atan((ztrue-yrobot_3)./(rtrue-xrobot_3)) + sigo*randn(1,N)
      sqrt( ((rtrue-xrobot_3).^2)+((ztrue-yrobot_3).^2)  )  + sigo*randn(1,N)
      ];
% Initialize quantities to be stored at every timestep
xhat = nan(4,N);  % State vector
sigr = nan(1,N);
sigz = nan(1,N);
sigu = nan(1,N);
sigw = nan(1,N);
% Initialize estimated state and covariance matrix
uguess = u0+0.2*randn(1);
vguess = w0+0.2*randn(1);
xhat(:,1) = [r0+sigo*randn(1) z0+sigo*randn(1) uguess vguess];

rvar0 = 0.01;
zvar0 = 0.01;
uvar0 = 10^2;
wvar0 = 10^2;
Chat = diag([rvar0 zvar0 uvar0 wvar0]);

% Set up time-independent matrices used in Kalman filter
Co = sigo^2;  % Covariance 'matrix' [1x1] of observations
F = [1 0 dt 0; 0 1 0 dt; 0 0 1 0; 0 0 0 1];  % Update matrix
% Time-step using Kalman filter
for n = 2:N

  % Predict
  rhat = xhat(1,n-1);
  zhat = xhat(2,n-1);
  uhat = xhat(3,n-1);
  what = xhat(4,n-1);
  xp = [rhat + uhat*dt; zhat + what*dt ; ...
        uhat; what]; % f(xhat)
  Cp = (F*Chat*F')+diag([Co Co Co Co]);  % 2x2

  % Update
  rp = xp(1);
  zp = xp(2);

  hp = [atan(((zp-yrobot_1(n))./(rp-xrobot_1(n))))
        sqrt(((rp-xrobot_1(n)).^2)+((zp-yrobot_1(n)).^2))
        atan(((zp-yrobot_2(n))./(rp-xrobot_2(n))))
        sqrt(((rp-xrobot_2(n)).^2)+((zp-yrobot_2(n)).^2))
        atan(((zp-yrobot_3(n))./(rp-xrobot_3(n))))
        sqrt(((rp-xrobot_3(n)).^2)+((zp-yrobot_3(n)).^2))
        ];
  H = [-(zp-yrobot_1(n))./(((rp-xrobot_1(n)).^2)+((zp-yrobot_1(n)).^2))
(rp-xrobot_1(n))./(((rp-xrobot_1(n)).^2)+((zp-yrobot_1(n)).^2))                 0 0
        (rp-xrobot_1(n))./(sqrt( ((rp-xrobot_1(n)).^2)+((zp-yrobot_1(n)).^2)))
(zp-yrobot_1(n))./(sqrt( ((rp-xrobot_1(n)).^2)+((zp-yrobot_1(n)).^2)))          0 0
        -(zp-yrobot_2(n))./(((rp-xrobot_2(n)).^2)+((zp-yrobot_2(n)).^2))
(rp-xrobot_2(n))./(((rp-xrobot_2(n)).^2)+((zp-yrobot_2(n)).^2))                 0 0
        (rp-xrobot_2(n))./(sqrt( ((rp-xrobot_2(n)).^2)+((zp-yrobot_2(n)).^2)))
(zp-yrobot_2(n))./(sqrt( ((rp-xrobot_2(n)).^2)+((zp-yrobot_2(n)).^2)))          0 0
        -(zp-yrobot_3(n))./(((rp-xrobot_3(n)).^2)+((zp-yrobot_3(n)).^2))
(rp-xrobot_3(n))./(((rp-xrobot_3(n)).^2)+((zp-yrobot_3(n)).^2))                 0 0
         (rp-xrobot_3(n))./(sqrt( ((rp-xrobot_3(n)).^2)+((zp-yrobot_3(n)).^2)))
(zp-yrobot_3(n))./(sqrt( ((rp-xrobot_3(n)).^2)+((zp-yrobot_3(n)).^2)))          0 0
        ];

  K = Cp*H'/(H*Cp*H' + diag([Co*sqrt( ((rp-xrobot_1(n)).^2)+((zp-yrobot_1(n)).^2)) Co
Co*sqrt( ((rp-xrobot_2(n)).^2)+((zp-yrobot_2(n)).^2))   Co  Co*sqrt( ((rp-
xrobot_3(n)).^2)+((zp-yrobot_3(n)).^2))   Co]   ) ) % Use plain division since
inverting 1x1 matrix
  Chat = (eye(4)-K*H)*Cp
  xhat(:,n) = xp + K*(yo(:,n) - hp);
  sigr(:,n) = sqrt(Chat(1,1));  % std of rhat uncertainty at step n
  sigz(:,n) = sqrt(Chat(2,2));  % std of zhat uncertainty at step n
  sigu(:,n) = sqrt(Chat(3,3));  % std of zhat uncertainty at step n
  sigw(:,n) = sqrt(Chat(4,4));  % std of zhat uncertainty at step n
end

% Redefine zhat and rhat to include all times.
rhat = xhat(1,:);
zhat = xhat(2,:);

% Plot exact and estimated r, including 2-std uncertainty bounds.
figure(1)
clf
subplot(2,1,1)
plot(t,rtrue,'k+',...
     t,rhat,'bx')
xlabel('t [s]')
legend('truth','est','+2\sigma','-2\sigma','Location','NorthEast')
title('x [m]')

% Plot exact and estimated z, including 2-std uncertainty bounds.
subplot(2,1,2)
plot(t,ztrue,'k+',...
     t,zhat,'bx')
xlabel('t [s]')
legend('truth','est','+2\sigma','-2\sigma','Location','South')
title('y [m]')

%% X against Y
figure;
axis equal

plot([zo1_1 L*cos(zo3_1)+zo1_1],[zo2_1 L*sin(zo3_1)+zo2_1],'b')
hold on;
plot([zo1_1 L*cos((zo3_1)+(2*(pi/3)))+zo1_1],[zo2_1
L*sin((zo3_1)+(2*(pi/3)))+zo2_1],'r')
hold on;
plot([zo1_1 L*cos((zo3_1)+(4*(pi/3)))+zo1_1],[zo2_1
L*sin((zo3_1)+(4*(pi/3)))+zo2_1],'k')
```

```
axis equal
hold on;

plot([zf1_1 L*cos(zf3_1)+zf1_1],[zf2_1 L*sin(zf3_1)+zf2_1],'--b')
hold on;
plot([zf1_1 L*cos((zf3_1)+(2*(pi/3)))+zf1_1],[zf2_1
L*sin((zf3_1)+(2*(pi/3)))+zf2_1],'--r')
hold on;
plot([zf1_1 L*cos((zf3_1)+(4*(pi/3)))+zf1_1],[zf2_1
L*sin((zf3_1)+(4*(pi/3)))+zf2_1],'--k')

hold on;

plot([zo1_2 L*cos(zo3_2)+zo1_2],[zo2_2 L*sin(zo3_2)+zo2_2],'b')
hold on;
plot([zo1_2 L*cos((zo3_2)+(2*(pi/3)))+zo1_2],[zo2_2
L*sin((zo3_2)+(2*(pi/3)))+zo2_2],'r')
hold on;
plot([zo1_2 L*cos((zo3_2)+(4*(pi/3)))+zo1_2],[zo2_2
L*sin((zo3_2)+(4*(pi/3)))+zo2_2],'k')
axis equal
hold on;

plot([zf1_2 L*cos(zf3_2)+zf1_2],[zf2_2 L*sin(zf3_2)+zf2_2],'--b')
hold on;
plot([zf1_2 L*cos((zf3_2)+(2*(pi/3)))+zf1_2],[zf2_2
L*sin((zf3_2)+(2*(pi/3)))+zf2_2],'--r')
hold on;
plot([zf1_2 L*cos((zf3_2)+(4*(pi/3)))+zf1_2],[zf2_2
L*sin((zf3_2)+(4*(pi/3)))+zf2_2],'--k')

hold on;

plot([zo1_3 L*cos(zo3_3)+zo1_3],[zo2_3 L*sin(zo3_3)+zo2_3],'b')
hold on;
plot([zo1_3 L*cos((zo3_3)+(2*(pi/3)))+zo1_3],[zo2_3
L*sin((zo3_3)+(2*(pi/3)))+zo2_3],'r')
hold on;
plot([zo1_3 L*cos((zo3_3)+(4*(pi/3)))+zo1_3],[zo2_3
L*sin((zo3_3)+(4*(pi/3)))+zo2_3],'k')
axis equal
hold on;

plot([zf1_3 L*cos(zf3_3)+zf1_3],[zf2_3 L*sin(zf3_3)+zf2_3],'--b')
hold on;
plot([zf1_3 L*cos((zf3_3)+(2*(pi/3)))+zf1_3],[zf2_3
L*sin((zf3_3)+(2*(pi/3)))+zf2_3],'--r')
hold on;
plot([zf1_3 L*cos((zf3_3)+(4*(pi/3)))+zf1_3],[zf2_3
L*sin((zf3_3)+(4*(pi/3)))+zf2_3],'--k')

plot(rtrue,ztrue,'--g')

txt1 = '   Start Point';
text(zo1_1,zo2_1,txt1)
txt11 = '   End Point';
text(zf1_1,zf2_1,txt11)

txt2 = '   Start Point';
```

```
text(zo1_2,zo2_2,txt1)
txt22 = '   End Point';
text(zf1_2,zf2_2,txt11)

txt3 = '   Start Point';
text(zo1_3,zo2_3,txt1)
txt33 = '   End Point';
text(zf1_3,zf2_3,txt11)

for i=2:1:N
    xagainsty_robot_1= plot([Zref1_1(i-1) Zref1_1(i)],[Zref2_1(i-1) Zref2_1(i)],'k');
    xagainsty_robot_2= plot([Zref1_2(i-1) Zref1_2(i)],[Zref2_2(i-1) Zref2_2(i)],'k');
    xagainsty_robot_3= plot([Zref1_3(i-1) Zref1_3(i)],[Zref2_3(i-1) Zref2_3(i)],'k');
    xagainsty_ball_estimation=plot([rhat(i-1) rhat(i)],[zhat(i-1) zhat(i)],'r');

    xagainsty_ball_estimation(1).LineWidth = 1;

    pause (0.1)
end

figure
uhat = xhat(3,:);
utrue = u0*ones(1,N);
subplot(2,1,1)
plot(t,utrue,'k+',...
     t,uhat,'bx')
xlabel('t [s]')
legend('Vx actual','estimated','Location','NorthEast')
title('Vx [m/s]')

what = xhat(4,:);
wtrue = w0*ones(1,N) ;
subplot(2,1,2)
plot(t,wtrue,'k+',...
     t,what,'bx')
xlabel('t [s]')
legend('Vy actual','estimated','Location','NorthEast')
title('Vy [m/s]')
```

Appendix K: Matlab code for moving ball interception considering the case that the mobile robot located behind the ball with respect to the shooting direction

```
clc
clear all
% via points given by the motion planner
xxx=[6.0000  5.8753   6.3343    6.5950   6.1376    5.485     5.64];
yyy=[8.0000  7.4131   7.0279    6.4932   5.4658    4.9321    5.92];

% time specifications and velocity done by geometric approach
ttt1=0:0.6792/(length(xxx)-1):0.6792;
ttt=[ttt1 0.6792];
Vxxxi=[0 -0.1610 0.5812 0.4924 -0.4731 0];
Vyyyi=[0 -0.7577 -0.4878 -1.0098 -1.0626 0];
Vttti=[0  0.1461 0.1433  0.2138  0.1171  0];

Vxxxf=[-0.1610  0.5812  0.4924 -0.4731  0 0];
Vyyyf=[-0.7577 -0.4878 -1.0098 -1.0626  0 0];
Vtttf=[ 0.1461  0.1433  0.2138  0.1171  0 0];

tfff=[0.7746 0.7897 0.5295 0.9669 0.4153  0.3763];
total_time=0.7746+0.7897+0.5295+0.9669+0.4153+0.3763;

% ball initial configuration
u0 = 0.7;
w0 = -0.4;
r0 = 2;
z0 = 8;
L=0.2;
sigo = 0.01; % sandard deviation
dt = 0.2; %time step
tf=7;   % final maneuvering time
N = tf/dt; % Number of samples
t = (1:(N))*dt; %time

rtrue = r0 + u0*t % ball motion on x axis
ztrue = z0+  w0*t % ball motion on y axis
%% Robot motion
dtt=0.0001; % step size
xrobot=zeros(1,38523);
yrobot=zeros(1,38523);

zo_1=[6;8;0];
zf_1=[5.8753;7.4131;0.1132];
vo_1=[0;0;0];
vf_1=[-0.1610;-0.7577;0.1754];
tt1f_1=0.7746
dttt=0.0001

zo1_1=zo_1(1);
zo2_1=zo_1(2);
zo3_1=zo_1(3);
```

```
zf1_1=zf_1(1);
zf2_1=zf_1(2);
zf3_1=zf_1(3);
vo1_1=vo_1(1);
vo2_1=vo_1(2);
vo3_1=vo_1(3);
vf1_1=vf_1(1);
vf2_1=vf_1(2);
vf3_1=vf_1(3);
N=tt1f_1/dttt;
x = (1:(N))*dttt;
Zref1_1=zo1_1 + vo1_1*x + (x.^3*(vf1_1 + vo1_1 - (2*zf1_1 -
2*zo1_1)/tt1f_1))/tt1f_1.^2 - (x.^2*(vf1_1 + 2*vo1_1 - (3*zf1_1 -
3*zo1_1)/tt1f_1))/tt1f_1;
Zref2_1=zo2_1 + vo2_1*x + (x.^3*(vf2_1 + vo2_1 - (2*zf2_1 -
2*zo2_1)/tt1f_1))/tt1f_1.^2 - (x.^2*(vf2_1 + 2*vo2_1 - (3*zf2_1 -
3*zo2_1)/tt1f_1))/tt1f_1;
Zref3_1=zo3_1 + vo3_1*x + (x.^3*(vf3_1 + vo3_1 - (2*zf3_1 -
2*zo3_1)/tt1f_1))/tt1f_1.^2 - (x.^2*(vf3_1 + 2*vo3_1 - (3*zf3_1 -
3*zo3_1)/tt1f_1))/tt1f_1;

xrobot(1,1:length(Zref1_1))=Zref1_1;
yrobot(1,1:length(Zref1_1))=Zref2_1;

zo_2=[5.8753;7.4131;0.1132];
zf_2=[6.3343;7.0279;0.2717];

vo_2=[-0.1610;-0.7577;0.1754];
vf_2=[0.5812;-0.4878;0.1720];
tt1f_2=0.7897
dttt=0.0001

zo1_2=zo_2(1);
zo2_2=zo_2(2);
zo3_2=zo_2(3);
zf1_2=zf_2(1);
zf2_2=zf_2(2);
zf3_2=zf_2(3);
vo1_2=vo_2(1);
vo2_2=vo_2(2);
vo3_2=vo_2(3);
vf1_2=vf_2(1);
vf2_2=vf_2(2);
vf3_2=vf_2(3);
N=tt1f_2/dttt;
x = (1:(N))*dttt;
Zref1_2=zo1_2 + vo1_2*x + (x.^3*(vf1_2 + vo1_2 - (2*zf1_2 -
2*zo1_2)/tt1f_2))/tt1f_2.^2 - (x.^2*(vf1_2 + 2*vo1_2 - (3*zf1_2 -
3*zo1_2)/tt1f_2))/tt1f_2;
Zref2_2=zo2_2 + vo2_2*x + (x.^3*(vf2_2 + vo2_2 - (2*zf2_2 -
2*zo2_2)/tt1f_2))/tt1f_2.^2 - (x.^2*(vf2_2 + 2*vo2_2 - (3*zf2_2 -
3*zo2_2)/tt1f_2))/tt1f_2;
Zref3_2=zo3_2 + vo3_2*x + (x.^3*(vf3_2 + vo3_2 - (2*zf3_2 -
2*zo3_2)/tt1f_2))/tt1f_2.^2 - (x.^2*(vf3_2 + 2*vo3_2 - (3*zf3_2 -
3*zo3_2)/tt1f_2))/tt1f_2;

xrobot(1,length(Zref1_1)+1:length(Zref1_1)+length(Zref1_2))=Zref1_2;
yrobot(1,length(Zref1_1)+1:length(Zref1_1)+length(Zref1_2))=Zref2_2;
```

```
zo_3=[6.3343;7.0279;0.2717];
zf_3=[6.5950;6.4932;0.4075];
vo_3=[0.5812;-0.4878;0.1720];
vf_3=[0.4924;-1.0098;0.2565];
tt1f_3=0.5295
dttt=0.0001

zo1_3=zo_3(1);
zo2_3=zo_3(2);
zo3_3=zo_3(3);
zf1_3=zf_3(1);
zf2_3=zf_3(2);
zf3_3=zf_3(3);
vo1_3=vo_3(1);
vo2_3=vo_3(2);
vo3_3=vo_3(3);
vf1_3=vf_3(1);
vf2_3=vf_3(2);
vf3_3=vf_3(3);
N=tt1f_3/dttt;
x = (1:(N))*dttt;
Zref1_3=zo1_3 + vo1_3*x + (x.^3*(vf1_3 + vo1_3 - (2*zf1_3 -
2*zo1_3)/tt1f_3))/tt1f_3.^2 - (x.^2*(vf1_3 + 2*vo1_3 - (3*zf1_3 -
3*zo1_3)/tt1f_3))/tt1f_3;
Zref2_3=zo2_3 + vo2_3*x + (x.^3*(vf2_3 + vo2_3 - (2*zf2_3 -
2*zo2_3)/tt1f_3))/tt1f_3.^2 - (x.^2*(vf2_3 + 2*vo2_3 - (3*zf2_3 -
3*zo2_3)/tt1f_3))/tt1f_3;
Zref3_3=zo3_3 + vo3_3*x + (x.^3*(vf3_3 + vo3_3 - (2*zf3_3 -
2*zo3_3)/tt1f_3))/tt1f_3.^2 - (x.^2*(vf3_3 + 2*vo3_3 - (3*zf3_3 -
3*zo3_3)/tt1f_3))/tt1f_3;

xrobot(1,length(Zref1_1)+length(Zref1_2)+1:length(Zref1_1)+length(Zref1_2)+length(Zre
f1_3))=Zref1_3;
yrobot(1,length(Zref1_1)+length(Zref1_2)+1:length(Zref1_1)+length(Zref1_2)+length(Zre
f1_3))=Zref2_3;

zo_4=[6.5950;6.4932;0.4075];
zf_4=[6.4562;5.5032;0.5432];

vo_4=[0.4924;-1.0098;0.2565];
vf_4=[-0.4731;-1.0626;0.1405];
tt1f_4=0.9669;
dttt=0.0001;

zo1_4=zo_4(1);
zo2_4=zo_4(2);
zo3_4=zo_4(3);
zf1_4=zf_4(1);
zf2_4=zf_4(2);
zf3_4=zf_4(3);
vo1_4=vo_4(1);
vo2_4=vo_4(2);
vo3_4=vo_4(3);
vf1_4=vf_4(1);
vf2_4=vf_4(2);
vf3_4=vf_4(3);
N=tt1f_4/dttt;
x = (1:(N))*dttt;
```

```
Zref1_4=zo1_4 + vo1_4*x + (x.^3*(vf1_4 + vo1_4 - (2*zf1_4 -
2*zo1_4)/tt1f_4))/tt1f_4.^2 - (x.^2*(vf1_4 + 2*vo1_4 - (3*zf1_4 -
3*zo1_4)/tt1f_4))/tt1f_4;
Zref2_4=zo2_4 + vo2_4*x + (x.^3*(vf2_4 + vo2_4 - (2*zf2_4 -
2*zo2_4)/tt1f_4))/tt1f_4.^2 - (x.^2*(vf2_4 + 2*vo2_4 - (3*zf2_4 -
3*zo2_4)/tt1f_4))/tt1f_4;
Zref3_4=zo3_4 + vo3_4*x + (x.^3*(vf3_4 + vo3_4 - (2*zf3_4 -
2*zo3_4)/tt1f_4))/tt1f_4.^2 - (x.^2*(vf3_4 + 2*vo3_4 - (3*zf3_4 -
3*zo3_4)/tt1f_4))/tt1f_4;

xrobot(1,length(Zref1_1)+length(Zref1_2)+length(Zref1_3)+1:length(Zref1_1)+length(Zre
f1_2)+length(Zref1_3)+length(Zref1_4))=Zref1_4;
yrobot(1,length(Zref1_1)+length(Zref1_2)+length(Zref1_3)+1:length(Zref1_1)+length(Zre
f1_2)+length(Zref1_3)+length(Zref1_4))=Zref2_4;

zo_5=[6.4562;5.5032;0.5432];
zf_5=[5.795;4.9321;0.6792];

vo_5=[-0.4731;-1.0626;0.1405];
vf_5=[-0.3730;2.3776;0];

tt1f_5=0.4153;
dttt=0.0001;

zo1_5=zo_5(1);
zo2_5=zo_5(2);
zo3_5=zo_5(3);
zf1_5=zf_5(1);
zf2_5=zf_5(2);
zf3_5=zf_5(3);
vo1_5=vo_5(1);
vo2_5=vo_5(2);
vo3_5=vo_5(3);
vf1_5=vf_5(1);
vf2_5=vf_5(2);
vf3_5=vf_5(3);
N=tt1f_5/dttt;
x = (1:(N))*dttt;
Zref1_5=zo1_5 + vo1_5*x + (x.^3*(vf1_5 + vo1_5 - (2*zf1_5 -
2*zo1_5)/tt1f_5))/tt1f_5.^2 - (x.^2*(vf1_5 + 2*vo1_5 - (3*zf1_5 -
3*zo1_5)/tt1f_5))/tt1f_5;
Zref2_5=zo2_5 + vo2_5*x + (x.^3*(vf2_5 + vo2_5 - (2*zf2_5 -
2*zo2_5)/tt1f_5))/tt1f_5.^2 - (x.^2*(vf2_5 + 2*vo2_5 - (3*zf2_5 -
3*zo2_5)/tt1f_5))/tt1f_5;
Zref3_5=zo3_5 + vo3_5*x + (x.^3*(vf3_5 + vo3_5 - (2*zf3_5 -
2*zo3_5)/tt1f_5))/tt1f_5.^2 - (x.^2*(vf3_5 + 2*vo3_5 - (3*zf3_5 -
3*zo3_5)/tt1f_5))/tt1f_5;

xrobot(1,length(Zref1_1)+length(Zref1_2)+length(Zref1_3)+length(Zref1_4)+1:length(Zre
f1_1)+length(Zref1_2)+length(Zref1_3)+length(Zref1_4)+length(Zref1_5))=Zref1_5;
yrobot(1,length(Zref1_1)+length(Zref1_2)+length(Zref1_3)+length(Zref1_4)+1:length(Zre
f1_1)+length(Zref1_2)+length(Zref1_3)+length(Zref1_4)+length(Zref1_5))=Zref2_5;

zo_6=[5.795;4.9321;0.6792];
zf_6=[5.64;5.92;0.6792];

vo_6=[-0.3730;2.3776;0];
vf_6=[-0.3730;2.3776;0];
```

```
tt1f_6=0.3763;
dttt=0.0001;

zo1_6=zo_6(1);
zo2_6=zo_6(2);
zo3_6=zo_6(3);
zf1_6=zf_6(1);
zf2_6=zf_6(2);
zf3_6=zf_6(3);
vo1_6=vo_6(1);
vo2_6=vo_6(2);
vo3_6=vo_6(3);
vf1_6=vf_6(1);
vf2_6=vf_6(2);
vf3_6=vf_6(3);
N=tt1f_6/dttt;
x = (1:(N))*dttt;
Zref1_6=zo1_6 + vo1_6*x + (x.^3*(vf1_6 + vo1_6 - (2*zf1_6 -
2*zo1_6)/tt1f_6))/tt1f_6.^2 - (x.^2*(vf1_6 + 2*vo1_6 - (3*zf1_6 -
3*zo1_6)/tt1f_6))/tt1f_6;
Zref2_6=zo2_6 + vo2_6*x + (x.^3*(vf2_6 + vo2_6 - (2*zf2_6 -
2*zo2_6)/tt1f_6))/tt1f_6.^2 - (x.^2*(vf2_6 + 2*vo2_6 - (3*zf2_6 -
3*zo2_6)/tt1f_6))/tt1f_6;
Zref3_6=zo3_6 + vo3_6*x + (x.^3*(vf3_6 + vo3_6 - (2*zf3_6 -
2*zo3_6)/tt1f_6))/tt1f_6.^2 - (x.^2*(vf3_6 + 2*vo3_6 - (3*zf3_6 -
3*zo3_6)/tt1f_6))/tt1f_6;

xrobot(1,length(Zref1_1)+length(Zref1_2)+length(Zref1_3)+length(Zref1_4)+length(Zref1
_5)+1:length(Zref1_1)+length(Zref1_2)+length(Zref1_3)+length(Zref1_4)+length(Zref1_5)
+length(Zref1_6))=Zref1_6;
yrobot(1,length(Zref1_1)+length(Zref1_2)+length(Zref1_3)+length(Zref1_4)+length(Zref1
_5)+1:length(Zref1_1)+length(Zref1_2)+length(Zref1_3)+length(Zref1_4)+length(Zref1_5)
+length(Zref1_6))=Zref2_6;

%% X against Y
figure;

xlabel('x [m]')
ylabel('y [m]')
plot([3 7], [10 10],'r');
plot([3 7], [0 0],'r');
plot([zo1_1 L*cos(zo3_1)+zo1_1],[zo2_1 L*sin(zo3_1)+zo2_1],'b')
hold on;
plot([zo1_1 L*cos((zo3_1)+(2*(pi/3)))+zo1_1],[zo2_1
L*sin((zo3_1)+(2*(pi/3)))+zo2_1],'r')
hold on;
plot([zo1_1 L*cos((zo3_1)+(4*(pi/3)))+zo1_1],[zo2_1
L*sin((zo3_1)+(4*(pi/3)))+zo2_1],'k')
axis equal

hold on;
plot([zf1_1 L*cos(zf3_1)+zf1_1],[zf2_1 L*sin(zf3_1)+zf2_1],'--b')
hold on;
plot([zf1_1 L*cos((zf3_1)+(2*(pi/3)))+zf1_1],[zf2_1
L*sin((zf3_1)+(2*(pi/3)))+zf2_1],'--r')
hold on;
plot([zf1_1 L*cos((zf3_1)+(4*(pi/3)))+zf1_1],[zf2_1
L*sin((zf3_1)+(4*(pi/3)))+zf2_1],'--k')
```

```
hold on;
axis equal
plot(rtrue,ztrue,'.', 'MarkerSize', 10)
hold on;
plot(5.795,4.9321,'.', 'MarkerSize', 20)
hold on;
plot(5.64,5.92,'.', 'MarkerSize', 20)

txt1 = '   Start Point';
text(zo1_1,zo2_1,txt1)
txt11 = '   Target point';
text(5.795,4.9321,txt11)
txt111 = '   shoot point';
text(5.64,5.92,txt111)

xlim([0 10])
ylim([0 10])
axis equal

plot([zo1_2 L*cos(zo3_2)+zo1_2],[zo2_2 L*sin(zo3_2)+zo2_2],'b')
hold on;
plot([zo1_2 L*cos((zo3_2)+(2*(pi/3)))+zo1_2],[zo2_2
L*sin((zo3_2)+(2*(pi/3)))+zo2_2],'r')
hold on;
plot([zo1_2 L*cos((zo3_2)+(4*(pi/3)))+zo1_2],[zo2_2
L*sin((zo3_2)+(4*(pi/3)))+zo2_2],'k')
axis equal

hold on;
plot([zf1_2 L*cos(zf3_2)+zf1_2],[zf2_2 L*sin(zf3_2)+zf2_2],'--b')
hold on;
plot([zf1_2 L*cos((zf3_2)+(2*(pi/3)))+zf1_2],[zf2_2
L*sin((zf3_2)+(2*(pi/3)))+zf2_2],'--r')
hold on;
plot([zf1_2 L*cos((zf3_2)+(4*(pi/3)))+zf1_2],[zf2_2
L*sin((zf3_2)+(4*(pi/3)))+zf2_2],'--k')

plot([zo1_3 L*cos(zo3_3)+zo1_3],[zo2_3 L*sin(zo3_3)+zo2_3],'b')
hold on;
plot([zo1_3 L*cos((zo3_3)+(2*(pi/3)))+zo1_3],[zo2_3
L*sin((zo3_3)+(2*(pi/3)))+zo2_3],'r')
hold on;
plot([zo1_3 L*cos((zo3_3)+(4*(pi/3)))+zo1_3],[zo2_3
L*sin((zo3_3)+(4*(pi/3)))+zo2_3],'k')
axis equal

hold on;
plot([zf1_3 L*cos(zf3_3)+zf1_3],[zf2_3 L*sin(zf3_3)+zf2_3],'--b')
hold on;
plot([zf1_3 L*cos((zf3_3)+(2*(pi/3)))+zf1_3],[zf2_3
L*sin((zf3_3)+(2*(pi/3)))+zf2_3],'--r')
hold on;
plot([zf1_3 L*cos((zf3_3)+(4*(pi/3)))+zf1_3],[zf2_3
L*sin((zf3_3)+(4*(pi/3)))+zf2_3],'--k')

plot([zo1_4 L*cos(zo3_4)+zo1_4],[zo2_4 L*sin(zo3_4)+zo2_4],'b')
hold on;
plot([zo1_4 L*cos((zo3_4)+(2*(pi/3)))+zo1_4],[zo2_4
L*sin((zo3_4)+(2*(pi/3)))+zo2_4],'r')
```

```
hold on;
plot([zo1_4 L*cos((zo3_4)+(4*(pi/3)))+zo1_4],[zo2_4
L*sin((zo3_4)+(4*(pi/3)))+zo2_4],'k')
axis equal

hold on;
plot([zf1_4 L*cos(zf3_4)+zf1_4],[zf2_4 L*sin(zf3_4)+zf2_4],'--b')
hold on;
plot([zf1_4 L*cos((zf3_4)+(2*(pi/3)))+zf1_4],[zf2_4
L*sin((zf3_4)+(2*(pi/3)))+zf2_4],'--r')
hold on;
plot([zf1_4 L*cos((zf3_4)+(4*(pi/3)))+zf1_4],[zf2_4
L*sin((zf3_4)+(4*(pi/3)))+zf2_4],'--k')

plot([zo1_5 L*cos(zo3_5)+zo1_5],[zo2_5 L*sin(zo3_5)+zo2_5],'b')
hold on;
plot([zo1_5 L*cos((zo3_5)+(2*(pi/3)))+zo1_5],[zo2_5
L*sin((zo3_5)+(2*(pi/3)))+zo2_5],'r')
hold on;
plot([zo1_5 L*cos((zo3_5)+(4*(pi/3)))+zo1_5],[zo2_5
L*sin((zo3_5)+(4*(pi/3)))+zo2_5],'k')
axis equal

hold on;
plot([zf1_5 L*cos(zf3_5)+zf1_5],[zf2_5 L*sin(zf3_5)+zf2_5],'--b')
hold on;
plot([zf1_5 L*cos((zf3_5)+(2*(pi/3)))+zf1_5],[zf2_5
L*sin((zf3_5)+(2*(pi/3)))+zf2_5],'--r')
hold on;
plot([zf1_5 L*cos((zf3_5)+(4*(pi/3)))+zf1_5],[zf2_5
L*sin((zf3_5)+(4*(pi/3)))+zf2_5],'--k')

plot([zo1_6 L*cos(zo3_6)+zo1_6],[zo2_6 L*sin(zo3_6)+zo2_6],'b')
hold on;
plot([zo1_6 L*cos((zo3_6)+(2*(pi/3)))+zo1_6],[zo2_6
L*sin((zo3_6)+(2*(pi/3)))+zo2_6],'r')
hold on;
plot([zo1_6 L*cos((zo3_6)+(4*(pi/3)))+zo1_6],[zo2_6
L*sin((zo3_6)+(4*(pi/3)))+zo2_6],'k')
axis equal

hold on;
plot([zf1_6 L*cos(zf3_6)+zf1_6],[zf2_6 L*sin(zf3_6)+zf2_6],'--b')
hold on;
plot([zf1_6 L*cos((zf3_6)+(2*(pi/3)))+zf1_6],[zf2_6
L*sin((zf3_6)+(2*(pi/3)))+zf2_6],'--r')
hold on;
plot([zf1_6 L*cos((zf3_6)+(4*(pi/3)))+zf1_6],[zf2_6
L*sin((zf3_6)+(4*(pi/3)))+zf2_6],'--k')

hold on;
axis equal

plot(5.795,4.9321,'.', 'MarkerSize', 20)
hold on;
plot(5.64,5.92,'.', 'MarkerSize', 20)

%% simulation
sda=plot([3 7], [10 10],'r');
```

```
sda(1).LineWidth = 10;

sdda=plot([3 7], [0 0],'r');
sdda(1).LineWidth = 10;

plot(rtrue(1),ztrue(1),'.' ,'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(2),ztrue(2),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(3),ztrue(3),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(4),ztrue(4),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
robot_track1=plot(xrobot(1:7746),yrobot(1:7746),'k')
robot_track1(1).LineWidth = 2;

plot(rtrue(5),ztrue(5),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(6),ztrue(6),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(7),ztrue(7),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(8),ztrue(8),'.', 'MarkerSize', 10,'color','r')
pause(0.2)

robot_track2=plot(xrobot(7746+1:7746+7897),yrobot(7746+1:7746+7897),'k')
robot_track2(1).LineWidth = 2;

plot(rtrue(9),ztrue(9),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(10),ztrue(10),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(11),ztrue(11),'.', 'MarkerSize', 10,'color','r')
pause(0.2)

robot_track3=plot(xrobot(7746+7897+1:7746+7897+5295),yrobot(7746+7897+1:7746+7897+529
5),'k')
robot_track3(1).LineWidth = 2;

plot(rtrue(12),ztrue(12),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(13),ztrue(13),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(14),ztrue(14),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(15),ztrue(15),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(16),ztrue(16),'.', 'MarkerSize', 10,'color','r')
pause(0.2)

robot_track4=plot(xrobot(7746+7897+5295+1:7746+7897+5295+9669),yrobot(7746+7897+5295+
1:7746+7897+5295+9669),'k')
robot_track4(1).LineWidth = 2;

plot(rtrue(17),ztrue(17),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(18),ztrue(18),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(19),ztrue(19),'.', 'MarkerSize', 10,'color','r')
```

```
pause(0.2)
plot(rtrue(20),ztrue(20),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(21),ztrue(21),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(22),ztrue(22),'.', 'MarkerSize', 10,'color','r')
pause(0.2)

robot_track5=plot(xrobot(7746+7897+5295+9669+1:7746+7897+5295+9669+4153),yrobot(7746+
7897+5295+9669+1:7746+7897+5295+9669+4153),'k')
robot_track5(1).LineWidth = 2;

plot(rtrue(23),ztrue(23),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(24),ztrue(24),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(25),ztrue(25),'.', 'MarkerSize', 10,'color','r')
pause(0.2)
plot(rtrue(26),ztrue(26),'.', 'MarkerSize', 10,'color','r')

robot_track6=plot(xrobot(7746+7897+5295+9669+4153+1:7746+7897+5295+9669+4153+3760),yr
obot(7746+7897+5295+9669+4153+1:7746+7897+5295+9669+4153+3760),'k')
robot_track6(1).LineWidth = 2;

xc=zeros(1,30);
yc=zeros(1,30);
xc(1)=5.64;
yc(1)=5.92;
for i=2:30
    xc(i)=xc(i-1)-0.064;
    yc(i)=yc(i-1)+0.408;
end
for ii=1:30
plot(xc(ii),yc(ii),'.', 'MarkerSize', 10,'color','r')
pause(0.05)
end
```