DOI 10.14750/ME.2022.021

UNIVERSITY OF MISKOLC

FACULTY OF MECHANICAL ENGINEERING AND INFORMATICS

ARTIFICIAL INTELLIGENCE AND OPTIMIZATION ALGORITHMS FOR DESIGN AND IMPLEMENTATION OF A ROBOTIC PLATFORM

PHD THESES

Prepared by

Hazim Nasir Ghafil

Engineering of mechanics … (BSc),
Engineering of applied mechanics … (MSc)

ISTVÁN SÁLYI DOCTORAL SCHOOL OF MECHANICAL ENGINEERING SCIENCES

TOPIC FIELD OF DESIGN OF MACHINES AND STRUCTURES

TOPIC GROUP OF DESIGN OF MECHATRONICS SYSTEMS

Head of Doctoral School
Prof. Dr. Gabriella Bognár Vadászné
DSc, Full Professor

Head of Topic Group
Prof. Dr. Gabriella Bognár Vadászné
DSc, Full Professor

Scientific Supervisor
Prof. Dr. Károly Jármai
DSc, full professor

Miskolc

2022

DOI 10.14750/ME.2022.021

Contents

DOI 10.14750/ME.2022.021

SUPERVISOR'S RECOMMENDATIONS

Mr. Hazim Nasir Ghafil has worked for five years on his PhD thesis. He has arrived as a Stipendium Hungaricum scholar to the University of Miskolc and was very diligent in doing new research and publishing the findings. In the beginning, we decided to write a book on robotics, and we could do that with Springer Verlag as a monograph. He made a detained literature survey and later theoretical calculations in the first half of his scholarship. He has developed new optimization techniques. In the second half of his scholarship, he developed the physical version of a robot for teaching purposes. First, he made from aluminium elements but later changed to 3D printing. With this new version, he could show the control of the robot with the computer. His research results and achievements show that his theoretical and practical knowledge in this field is internationally accepted.

Date January 5, 2022

Supervisor

Prof. Dr. Károly Jármai

DSc, full professor

LIST OF SYMBOLS AND ABBREVIATIONS

DH	Denavit Hartoerg convention
x	x-axis
y	y-axis
z	z-axis
HTM	Homogenous transformation matrix
T	Overall homogenous transformation matrix
Rot	Rotation
Trans	Translation
θ	Joint angles
S	Sine function
C	Cose function
$\propto$	Twist angle
a	Traslation on x-axis
d	Translation on z-axis
r	Component of the rotation matrix
R	Rotation matrix
H	Homogenous transformation of a link
Fk	Forward kinematics
Ik	Inverse kinematics
Ci	Current configuration

De	Desired configuration
ANN	Artifitial neural network
i	index
n	Total number of items
RRR	Three revolute joints robot
LM	Levenberg-Marquardt
BR	Bayesian Regularization
SCG	Scaled Conjugate Gradient
USB	Universal serial bus
UART	Universal asynchronous receiver-transmitter
I2C	Inter-Integrated Circuit
MCU	Microcontroller unit
Tx	Transmit to a device
Rx	Receive from a device
ADC	Analog degital converter
DAC	Digital analog converter
PWM	Pluse width modulation
D	Angular displacement of a motor
dis	Mapped angular displacement of a motor

1. INTRODUCTION

1.1 Prelude

Nowadays, robots are playing a significant role in all aspects of human life (Daneshmand et al. 2017; Wang et al. 2015; Spolaôr and Benitti 2017) because of the human tendency to fulfil his needs with low cost, high quality and fast production rates, which may be difficult just by workers as well as these robots are preferred especially in some difficult working environments, which should undergo for a serious risk assessment (Gopinath and Johansen 2016; Gopinath et al. 2017; Michalos et al. 2014). The vehicle industry and other automotive engineering are a perfect area being robotized for the above-mentioned reasons. Axiomatically robots' specifications depend on their applications which differ from one purpose to another one like assembly robots (Kramberger et al. 2017; Makris et al. 2017) which are carried out heavy parts, or PCB manipulators which need to carry dynamic loads, of course, both previous examples require precise motion. There are many types of industrial robots, and it is used according to the purpose and desired duty (Spong 2005; Craig 2009). The most common type of robot is serial robot manipulators, which are a series of rigid bodies, called links, are joined together by means of joints (Ghafil et al. 2015), see Figure 1.1. It is economically undesirable to design all the robot manipulators for the same criterion in manufacturing lines of the vehicle industry because the robot's joints and links are subject to different loads in different production lines. It is clear that robot manipulators in assembly lines suffer from more stresses than those in the painting or welding lines. Therefore, there is a need to optimize manipulators links and joints to reach optimum design (Kivelä et al. 2017; Hassan and Abomoharam 2017). Another fact should be considered about using robot manipulators in the industry is that the working area or the configuration space of the manipulators may contain static or dynamic obstacles, which leads us to supply the robots with path or trajectory planning. These paths might just be a predefined set of points (Mathew and Hiremath 2016) in the Cartesian space in case of a static environment or paths continuously change due to the dynamic environment (Tuncer and Yildirim 2012). However, in both cases, these sets of points should transform from a configuration space to a joint space employing inverse kinematics (Iliukhin et al. 2017).

a) b)

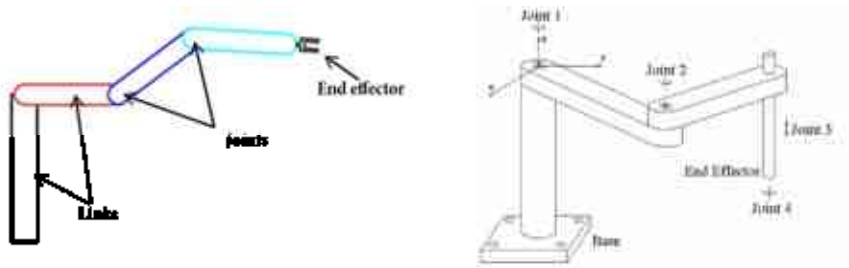

Figure 1.1 (a) 5R manipulator, (b) RRdR manipulator (SCARA robot)

1.2 Statistics

Industrial robots (Ghafil and Jármai 2018b) are spreading rapidly worldwide due to the competition among manufacturers in different countries, and reliable statistics refer to the fast growth of robots in the last few years. World robotics report released by the international federation of robotics IFR (report 2018) has shown how numbers of industrial robots have increased during one year see Figure 1.2 and Figure 1.3. Records 2016 for North American Robot Orders and Shipments (Association 2018), the hottest Applications and Industries orders for robots have increased by 61 percent in the field of assembly and 24 percent in spot welding. Also, the consumer goods and food industry have increased orders for robots by 32 percent. Figure 1.4 shows how robotics units have increased worldwide in the last few years ((IFR) 2018). Many reports (Robots 2018) have shown the worldwide booming of industrial robots.

1.3 Literature review

This section is divided into three sections as optimization algorithms, Kinematics (Ghafil and Jármai 2019b), and artificial intelligence for robot Kinematics.

1.3.1 Optimization algorithms

In this section, well-known optimization algorithms (Ghafil and Jármai 2020c) are reviewed with brief explanations, and we will mention arbitrary chosen twenty algorithms. There are too many metaheuristics that come from different inspiration backgrounds (Nasir Ghafil et al. 2021), and some examples could give the reader an overview of how they were developed. Optimization algorithms were used in most of the scientific fields (Alsamia et al. 2021) with a wide range of applications (Farkas & Jármai 1996, 2003,2008,2013, 2015).

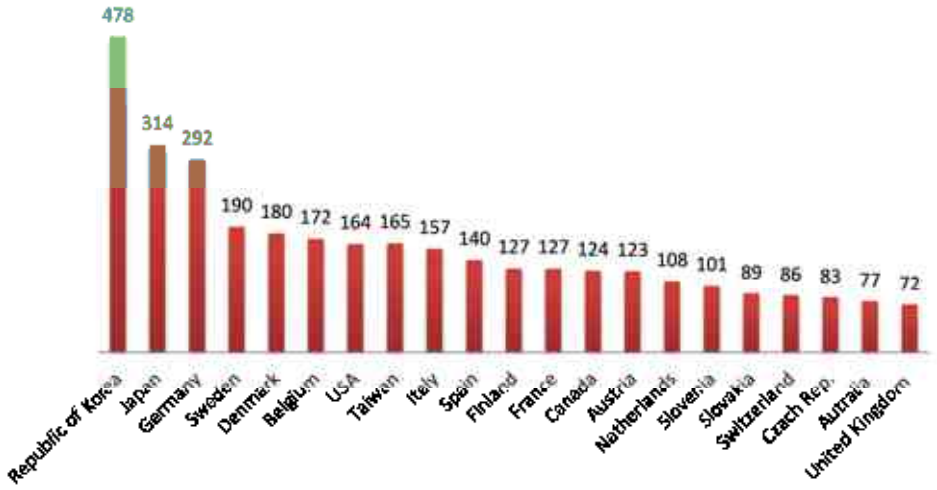

Figure 1.2 Number of multipurpose industrial robots (all types) per 10,000 employees in the manufacturing industry 2014

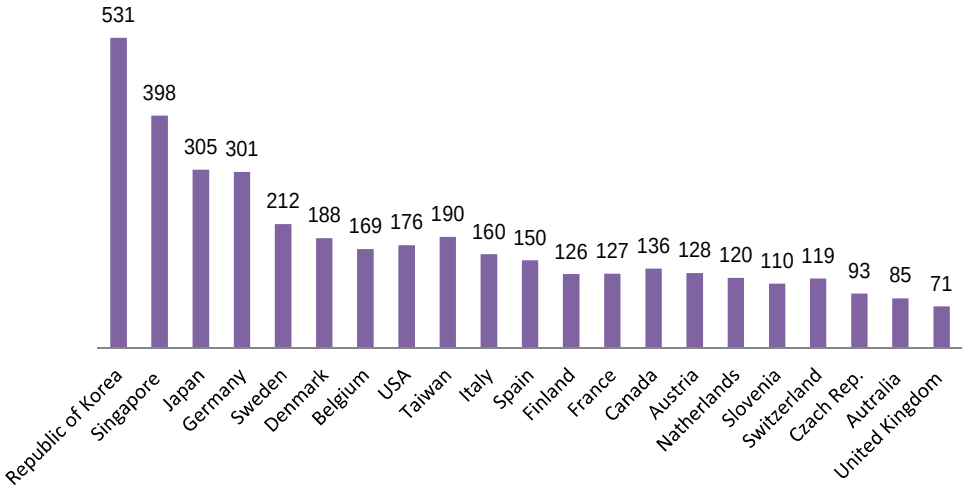

Figure 1.3 Number of multipurpose industrial robots (all types) per 10,000 employees in the manufacturing industry 2015

(Nelder and Mead 1965) have proposed a metaheuristic algorithm which is also referred to as the Amoeba Method. It is a simplex search algorithm commonly used for nonlinear optimization problems. A simple random search algorithm called Random Search (RS) was proposed by (Brooks 1958), where the new solutions are independent of the previous ones. The inspiration of the principle of survival of the fittest, which was proposed by Charles Robert Darwin, has led to the development of the Genetic algorithm (Holland 1975). Inheritance of cultural information among individuals has inspired the Memetic Algorithm (Moscato 1989).

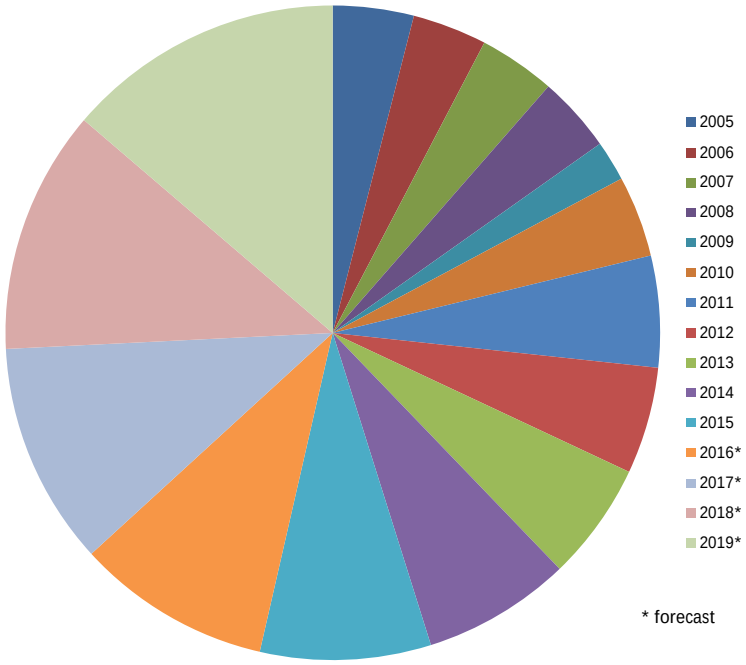

Figure 1.4 Annual supply for industrial robots worldwide

The Simulated Annealing method (Eglese 1990) was described by Kirkpatrick, and it mimics the process of improving steel properties during the annealing process. Ant algorithms, with their first version (Dorigo 1992) and improved version (Dorigo and Gambardella 1997), are combinatorial optimization algorithms that simulate the behaviour of the ants while searching for the shortest path between food and their nest. However, (Socha and Dorigo 2008) have introduced ant algorithm for continuous domains. Cultural Algorithm (CA) (Reynolds 1994) is an evolutionary algorithm that simulates the cultural evolution of human society. The Differential Evolution (DE) algorithm (Storn 1995) was developed as an improvement to the genetic algorithm, which is based on Darwin's Theory of Evolution. Cross-Entropy Method (Rubinstein 1997) is a probabilistic optimization algorithm that was developed based on the closeness between two probability distributions. (Geem et al. 2001) have developed an efficient algorithm called Harmony Search by taking inspiration from jazz musicians. Bacterial Foraging Optimization Algorithm (Liu and Passino 2002) was proposed to solve mathematical problems where it mimics the bacteria activity in nature. (Pham et al. 2006) introduced the concept of the bee algorithm, which as the name, uses the procedure in honey hives to describe a mathematical framework to solve optimization problems. The behaviour of weeds, while they dominate the fields, has motivated the inspiration of invasive weeds optimization IWO (Mehrabian and Lucas 2006). Cuckoo birds lay their eggs in the nests of other birds (host), and this behaviour has motivated the development Cukoo search algorithm (Yang and Deb 2009). Firefly Algorithm (Yang 2009) is a nature-inspired metaheuristic optimization algorithm inspired by the flashing behaviour of fireflies while they communicate with other fireflies. Bat Algorithm (Yang 2010) is a swarm intelligence metaheuristic algorithm which simulates the activities of bats while forging

and communicate with each other. Eberhart and Kennedy (Kennedy 2011) was developed swarm intelligence metaheuristic algorithm mimics the foraging movement of bird and fish swarms. A biologically-inspired algorithm invented by (Yang 2012) to simulate the pollination process of flowers and called flower pollination algorithm.

1.3.2 Robot Kinematics

Many researchers dealt with the analysis for inverse Kinematics of a 5-DOF manipulator considered that it should deal with the problem as an analysis for 6-DOF manipulator losing one degree of freedom (Hayawi 2011), (Al-Azzawi 1996) and deficiency of degrees of freedom makes the control of manipulators with less than 6-DOF very difficult when following a given trajectory (Xu et al. 2005). Some algebraic solutions for inverse Kinematics of a 5-DOF manipulator assume that summation of joint angles 2,3 and 4 (θ_234) is a constant value, so it did not solve all the joint angles. (Abbas 2013) introduced analysis for forward kinematics of 5-DOF and did not deal with inverse Kinematics of the robot. C. S. G Lee, M. Ziegler (Ziegler 1983) solve the problem of choosing a solution among multi solutions generated as a result of left and above arm, right and above arm, left and below arm, right and below arm and wrist flip configurations by assuming an indicator, this way need that joint angles already known.

Many types of research in the field of inverse kinematics acceleration, velocity, or position have used different methods. One of these methods is the optimization algorithms such as hybrid genetic algorithm (Zhang et al. 2008) and particle swan object PSO (Alsarayefi et al. 2019). Many researchers had interested in the field of the inverse kinematics problem position, velocity, and acceleration. Analytical and geometrical approaches were introduced to solve the forward and inverse Kinematics problem for a 6DOF robot manipulator (PUMA) (Bazerghi et al. 1984) (Lee and Ziegler 1984). An iterative method based on the Newton-Raphson method for nonlinear equations solution was proposed to solve the inverse problem and overcome the constraints developed by the closed-form solution encountered in a class of robots; the method is suitable for any robot have revolute or prismatic joints with any DOFs (Goldenberg et al. 1985). A driven formula from a closed-form solution of a redundant robot manipulator using the Lagrangian multiplier method, the proposed method got rid of the repeatability problem that appears in the resolved-motion method (Chang 1987). A modular architecture was developed for the inverse kinematics problem for a robot manipulator. This modular was based on a nonlinear equation solver. The solver was based on the Banach Fixed-point theorem (Tourassis and Neuman 1985). Discussion for the Kinematics problem of a hybrid robot manipulator, which combines the serial chain with parallelism, also discussed the singularity and indeterminacy in the inverse problem (Kumar and Gardner 1990). An efficient approach was presented to solve the inverse Jacobian of the wrist partitioned manipulators in the case of the differential inverse Kinematics problem. The study used PUMA 560 as a case study, and the inverse Jacobian matrix was formed based on the

generated simplified equations (Wu and Young 1990). A proposed closed-loop algorithm to solve the inverse Kinematics without calculating the Jacobian matrix, avoidance of the singularities that may cause numerical instabilities was discussed in (Novakovic and Nemec 1990). Numerical solution for the inverse Kinematics problem for any type of serial robot manipulators was developed using a method based on nonlinear programming technique and forward recursion formulas, singular configurations do not affect the solution generated by this method (Wang and Chen 1991). An iterative algorithm has been introduced on the basis of the global Newton's method to find staircase-like inverse solutions for simple robots (Isobe et al. 1992). The offset modification method presented to solve the inverse Kinematics problem for serial robot manipulators with any DOFs shows that this method is suitable for real-time manipulator control (Grudic and Lawrence 1993). The genetic algorithm is widely used to determine inverse position, velocity, and acceleration (Buckley et al. 1997; Chapelle and Bidaud 2004; Zhang et al. 2004). The combination of genetic algorithm and neural network is also used for inverse positioning of robot manipulators, GA and ANN, complete each other to find the final inverse position (Ghafil 2016; Tomić et al. 2012). Kinematic analysis for a 5DOF robot using a Multi-Layered Feed Forward Neural Network was studied (Kumar and Chand 2015). A heuristic solution was presented for the inverse Kinematics problem, the heuristic consist on combination of the distance between the actual and the desired position of the gripper the best manipulability direction (Pozna et al. 2016). An algorithm for the closed loop inverse Kinematics problem has been presented, the study proposed an iterative method to determine Euler solution then taking the results to achieve the numerical integration by using Crank-Nicolson method to get better convergence for the differential inverse problem (Drexler 2016). This paper deals with the issue of calculating the inverse acceleration problem for serial robot manipulator of any degree of freedom using the evolutionary algorithm harmony search. Peter E. Hart and Nils J. Nillsson (Hart et al. 1968), described how heuristic information form the problem domain can be incorporated into a formal mathematical theory of graph searching and demonstrate an optimality property of a class of search strategies. Dave Ferguson et al (Ferguson et al. 2005) described a family of heuristic-based planning algorithms that have been developed to address various challenges associated with planning in the real world. Each of the algorithms presented has been verified on real systems operating in real domains. Surachai Panich (Panich 2010), Introduced intelligent path planning with A* search algorithm (Mahmood et al. 2019), which use to generate the most efficient path to the goal. this study was an implementation of path planning for intelligent path planning. The implementations are tested and verified with the simulation software. Kavita Krishnaswamy et al. (Krishnaswamy et al. 2011) made an effort to improve path planning performance for a robotic arm, resulting in faster user response in real-time for a robotic arm with multiple degrees of freedom. They proposed that it is possible to achieve rapid response times with an assistive robotic arm by caching frequent arm trajectories and creating a roadmap" of arm movements. By calculating trajectories to possible target goals. Al-Arif et al. (Al-Arif et al. 2012) compared among many algorithms which are, the graph search

algorithm, the breadth-first algorithm, Dijkstra algorithm, and A* algorithm. At the end they conclude that A* algorithm is the best among the previously mentioned algorithms. Alaa Hassan Shabeeb (Shabeeb 2013), in this thesis an algebraic and analytic solutions offered for the inverse Kinematics for Lab-Volt 5150 (Ghafil and Jármai 2019c) it had been assumed that summation of joint angles 2,3 and 4 (θ_234) is a constant value so, this analysis did not offer a complete solution for all the five joint angles of the manipulator introduced. The principle of path planning using the Bezier curves technique, Bezier curves, had been applied in two ways. The first property of Bezier curves was that the curve passes only through the start and the end control points and does not pass through the intermediate control points. It was used to plan free path collision robot path planning by considering the start and the end control points as the initial and the goal position of the robot path, while the control points were considered obstacles. In the second way, the Bezier curve was applied to generate path planning to track control point to curve on surface. In the kinematic synthesis of a multi-linkage robot arm, topology optimization is performed to obtain the best type synthesis (Kong and Gosselin 2007) for a single task or multitask robot manipulator. Next, the minimum required dimensions should be calculated during the dimensional synthesis process(Kivelä et al. 2017). Both of the type and dimensional synthesis can be done in two separated operations or one single operation. (Liu and McPhee 2007) has employed a genetic algorithm to find optimum topology and dimensions of a planar robot. (Pucheta and Cardona 2013) Have employed graph theory-based approach to estimate the type synthesis while the best dimension parameters are calculated by Precision Position Method. On the dimensional level, (Tromme et al. 2015) have dealt with shape optimization of the link of the robot using level set techniques. On the type level, the key point in most works is the graph theory (Kawamoto et al. 2004) where possible topologies are enumerated to get the best type. Instead of depending on designer ingenuity and experience, many works have been done to describe a systematic approach to find the best robot type with the best dimensions for a single or multitasks manipulators. Kinematic synthesis consists of two parts; type synthesis and dimensionality synthesis (Kota and Jarmai 2014). There are many methods had dealt with this issue like heuristic methods and numeric optimisation framework. An evolutionary method based on genetic algorithm and graph theory was proposed by Liu (Liu and McPhee 2007) to find the best mechanism with best dimensions in one single process instead of treat type level and dimensions level separately. An extension to Lie work was done using convertible agents and an evolutionary algorithm (Oliva and Goodman 2010). Many other works have separated the problem into two sub-problems; by enumerating the topological alternatives using graph theory to find the optimum type and using Precision Position Method to find optimum dimensions for the feasible solution.

1.3.3 Artificial intelligence

In this section, we will review the common works that employed artificial intelligence (Ghafil et al. 2019) to solve problems in the robotic field. Multilayer feedforward networks were learned to find the set of joint variables of a three degree of freedom manipulator given end-effector position (Choi and Lawrence 1992). The learning process uses given end-effector positions and their corresponding joint angles. Multiple feedforward networks were found to be more efficient than the single layer to generalize a functional relationship for the inverse Kinematic problem. After training, the network was able to return the joint variables for arbitrary Cartesian positions. ANN has been used with a back propagation algorithm to solve inverse Kinematic for 6R robot manipulator with offset using MATLAB Neural Networks Toolbox (Bingul et al. 2005). The network had 12 inputs representing elements of the homogenous transformation matrix and six outputs representing six joint angles. According to this study, the disadvantage of using ANN for inverse Kinematic is the high number of data sets required for the training process. In other words, the high amount of data set for training leads to a more accurate neural network. Four thousand data sets had been used to learn the network, which is consists of 20 neurons in a single hidden layer. Another work (Duka 2014) employed a three-links planar manipulator used as a test bed for the inverse Kinematic problem by learning a neural network how to generate a function for this issue. Non-uniform and customized topologies were used to learn the IK problem using the concept of network inversion (Tejomurtula and Kak 1999). Kohonen and error backpropagation networks are the most common types of neural networks applied to solve inverse kinematics of robot manipulators. The back-propagation algorithm, which is one of the learning mechanisms, takes a long time for the forward training to find a feasible mapping from Cartesian space to the joint space of the robot. The response of all feed patterns is transported to the output layer. The difference between the desired output and calculated output represents the error that is back propagated to adjust weights iteratively. If the training data set is too large, the time taken for the training process will be tremendous, which is unpractical for real-time training. Also, subnetworks are proposed (Lu and Ito 1995) to find more than one solution for a specific Cartesian position of the end-effector.

1.4 The objective of the study

Based on the survey in section 1.3, we can note that optimization algorithms are open and wide field. They were inspired by natural phenomena, human-made phenomena, etc. Artificial neural networks were used widely to solve inverse Kinematics of robots, but there is no comparison study about which learning algorithm is the best for inverse problems. Consequently, we summarize the objective of the study as follows

1- Developing a generic objective function that can be minimized by an optimization algorithm to find the inverse kinematic solution for any type of robot manipulator with any degree of freedom.

2- Perform a comparison study to find which learning algorithm of neural networks can be the best choice for inverse kinematic problems.

3- Developing a new educational robotic platform with a unique virtual realty environment and using 3D printing technology to design and implement a new robot arm.

1.5 Importance of the Study

This section presents a survey among 69 individuals from 14 different countries worldwide. Almost all of them have a high academic degree, master or Ph.D. in their fields, and most of them are teaching staff in their faculties. Most of the participants believe that developing a new educational robotic platform for students is an interesting issue because universities do not give the appropriate intention to teach robots as they believe. The importance of the study can also be seen because the participants are ready to buy such a product if it is available in markets for self-education. People are interested in robots, and they believe that they are the future, and it is worth knowing the minimum about controlling a robot arm for them or their children. This is an excellent indication of a promising market for our educational robotic platform. The robust scientific nature and the commercial intention for the model developed in this thesis are encouraging because they touch great categories of people worldwide. Below are some of the questionnaire responses, Figure 1.5 to Figure 1.14:

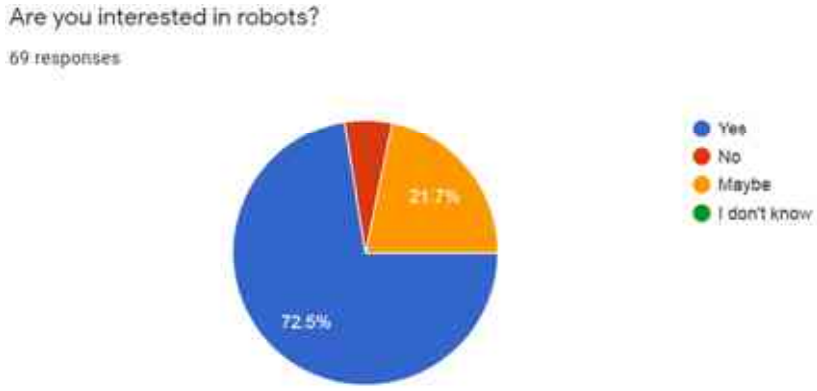

Figure 1.5 Interest of the participants

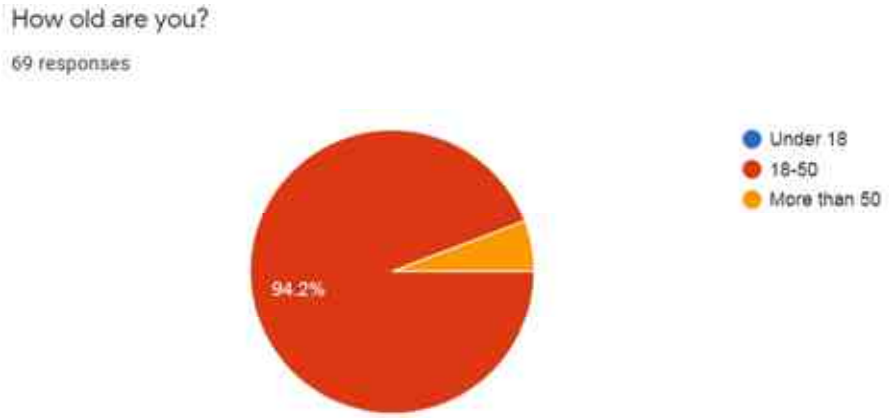

Figure 1.6 Ages of the participants

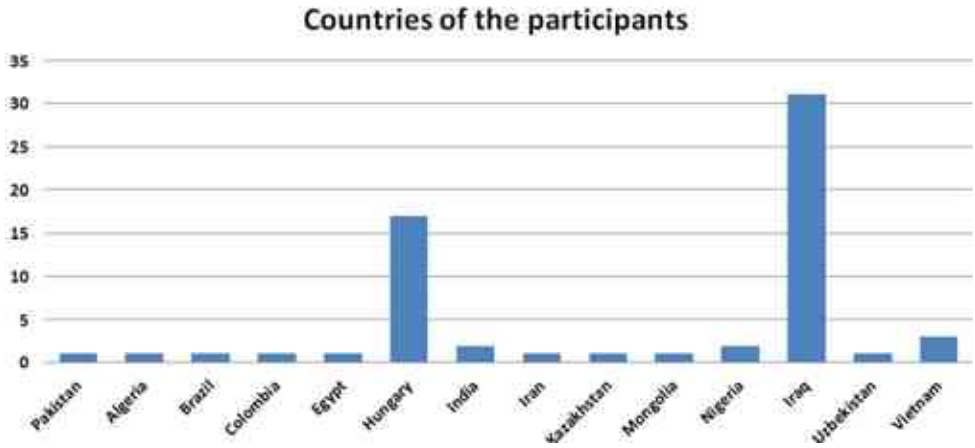

Figure 1.7 Countries of the participants

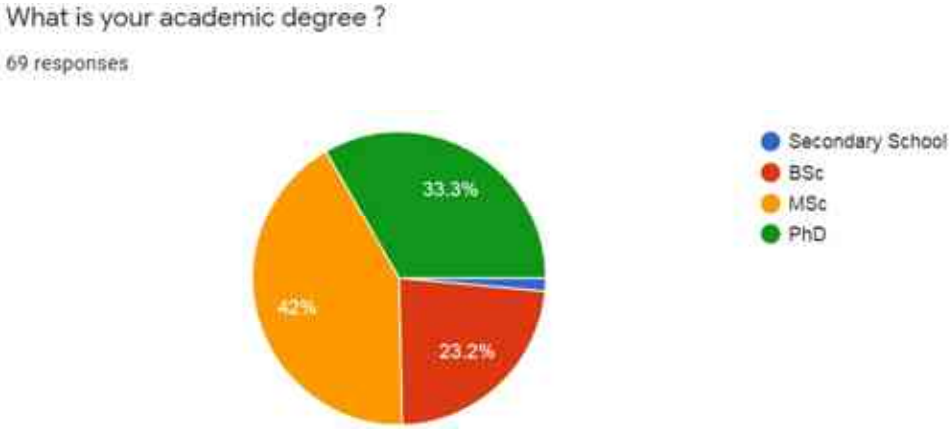

Figure 1.8 Academic degree of the participants

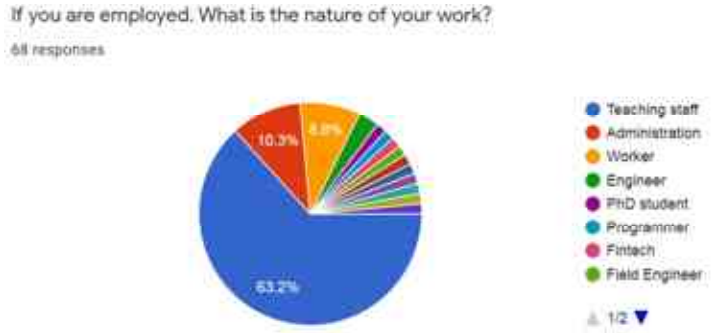

Figure 1.9 work of the participants

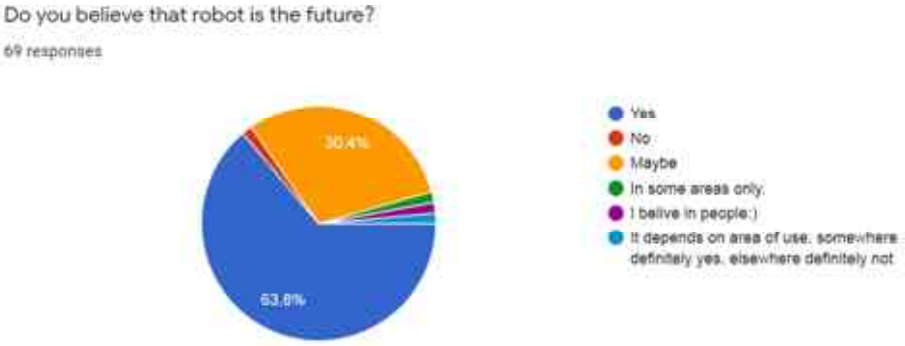

Figure 1.10 Believing of the participants about robots

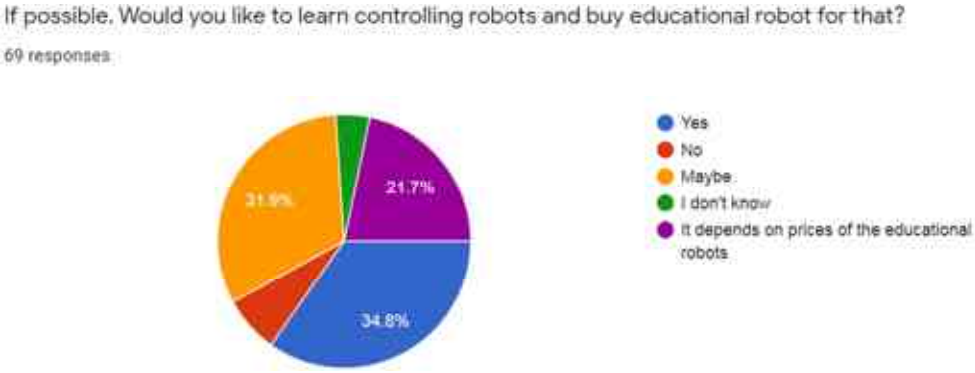

Figure 1.11 Interest of participants to learn robots

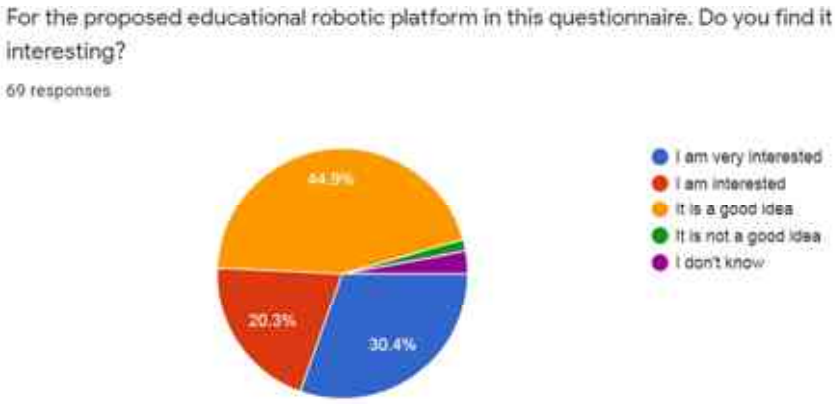

Figure 1.12 Opinion of the participants about the proposed educational robot in this work

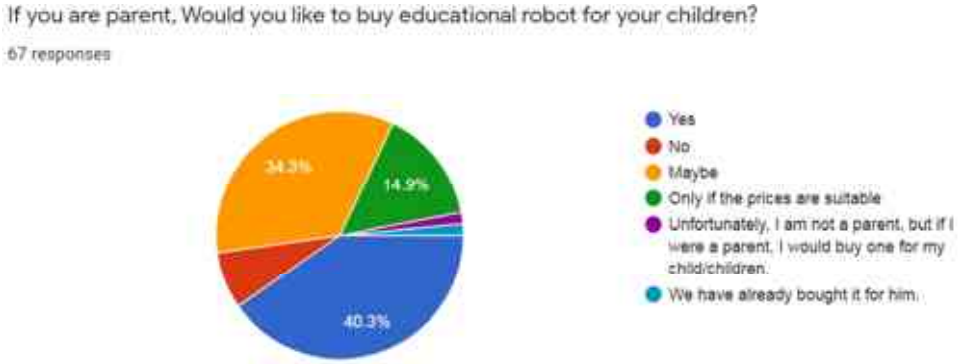

Figure 1.13 Interest of participants to buy educational robots

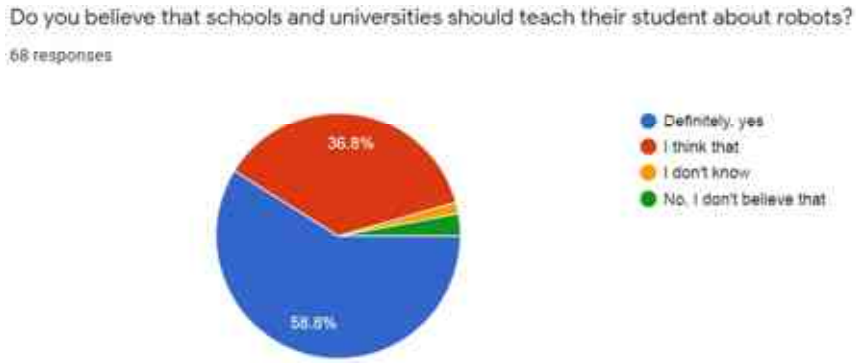

Figure 1.14 Opinion of the participants about learning robots in schools and universities

1.6 Outline of the thesis

Chapter one introduces robots and their importance in the industry and gives an overview of the previous works related to this dissertation. Also, the chapter presents a survey among highly educated students and teaching staff about their opinion about the main topic of the proposed educational robotic platform. Chapter two presents the design and specifications of the 6 DOF robot arm. Also, chapter two presents the mathematical model and set of derived equations that govern the manipulator. Chapter three introduces the inverse kinematic problem and uses optimization algorithms and artificial neural networks to solve it. Chapter four is about path and trajectory planning, where two methods are presented to solve the path planning problem. Chapter five presents the control system of the robotic platform and describes its hardware and software. After that, theses, summary, application possibility, and acknowledgements are presented.

2. GEOMETRIC DESIGN AND SPECIFICATIONS

2.1 Introduction

In this chapter, the mechanical parts of the robot arm are described as well as the specifications of its actuators. The spatial description and configuration spaces are also shown, and by using the Denavit-Hartenberg convention, the forward kinematic equations are derived. The forward equations in this chapter are fundamental because they will be used in chapter three while talking about inverse kinematics.

2.2 Robot geometry and Specifications

The structure of the robot used in this thesis is shown in Figure 2.1; it consists of aluminium parts, each of thickness 2 mm. This commercial robot is 6 degrees of freedom comes with six servo motors. The configuration space is revealed in Figure 2.2 and Figure 2.3; the maximum reach point is about 255 mm from the front edge of the base link of the robot.

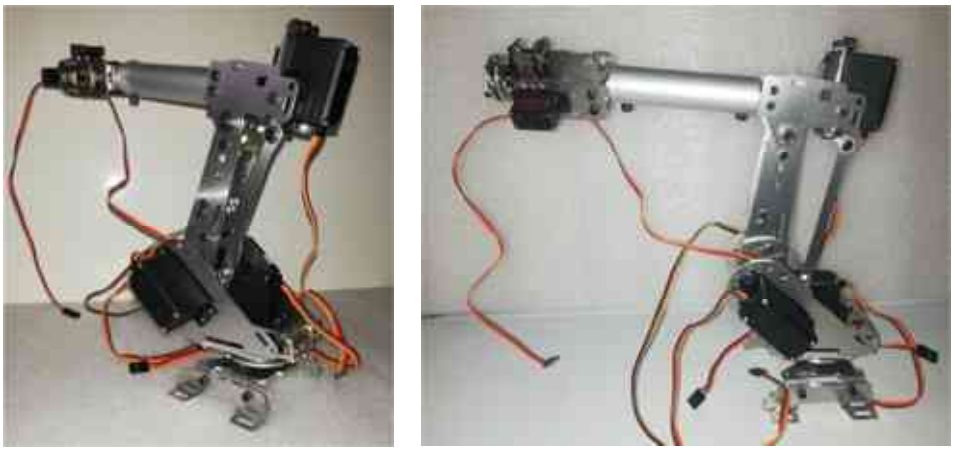

Figure 2.1 Structure of the 6DOF manipulator

The mechanical limitations of this arm are:

1. Small size

2. Joint two cannot rotate more than 57^{o} from the vertical axis in the backside direction in Figure 2.2. More than $57^{o,}$ the arm will fall to the back side because the motor torque cannot support the weight of the arm.

3. Calibration of the robot components is complicated, especially while putting the arm in the zero or rest position. While sending zero signal to the motor and pressing it in its position during the assembly process, the motor shaft will deviate slightly from its zero position even when the zero signal is applied. This is a drawback of using servo motors, and easier if stepper motor or DC motor with an encoder are used.

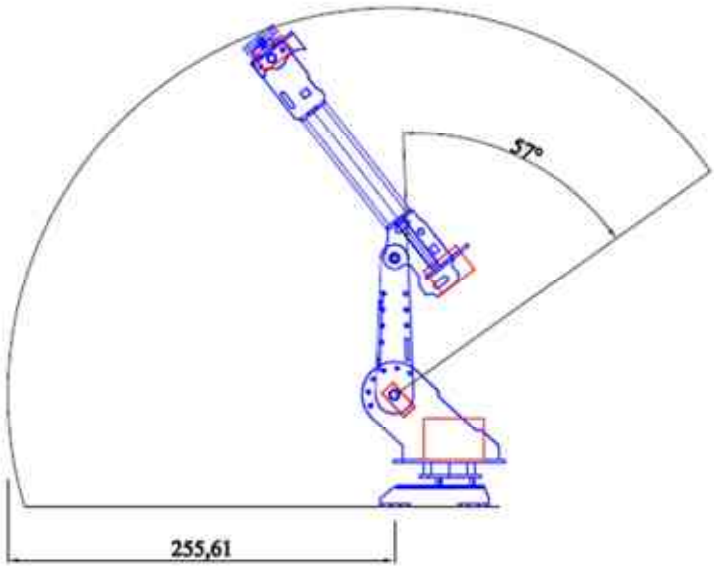

Figure 2.2 Side view of the configuration space

2.3 Denavit-Hartenberg convention

This section must refer to the main equation for all the cases studied during this work. Denavit-Hartenberg convention (Spong et al. 2006) is a procedure to assign coordinate frames rigidly to a manipulator to represent the position and orientation of the end effector of the robot to the inertial frame. Referring to Figure 2.4, Denavit-Hartenberg have two assumptions

- DH1: The axes x_1 is perpendicular to the axis z_0.
- DH2: The axes x_1 intersects the axis z_0.

In this convention, each homogeneous transformation A_i is represented as a product of four basic transformations

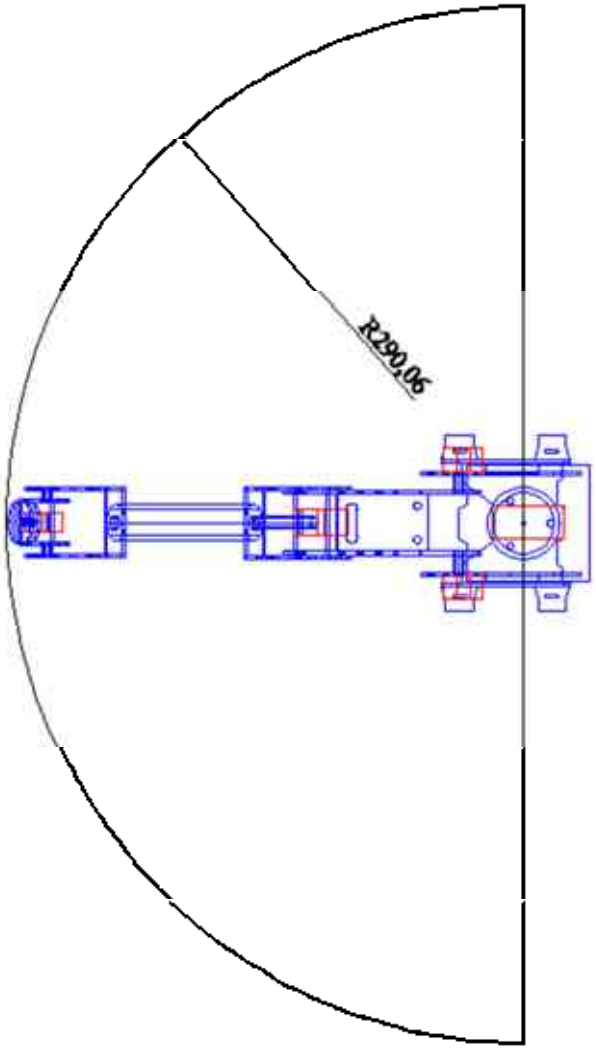

Figure 2.3 Top view of the configuration space

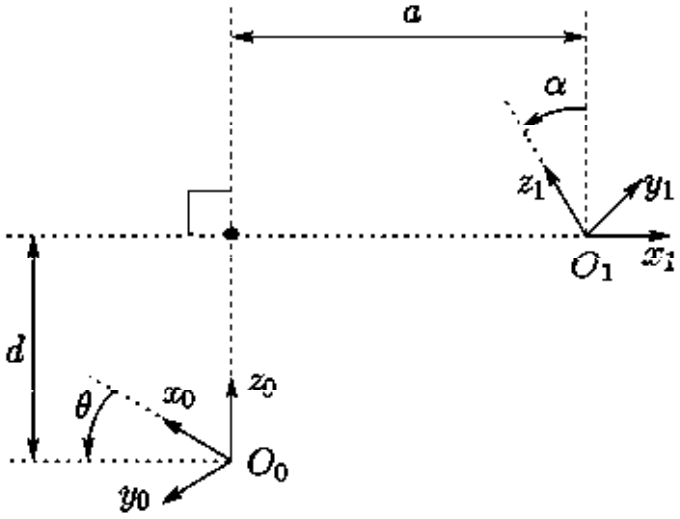

Figure 2.4 Coordinate frames satisfying assumptions DH1 and DH2

$$T_j^i = Rot_{z,\theta_i} * Trans_{z,d_i} * Trans_{x,a_i} * Rot_{x,\alpha_i} \quad (2.1)$$

We will use the notation C_θ for cos θ and S_θ for sin θ

$$T_j^i = \begin{bmatrix} C_{\theta_i} & -S_{\theta_i} & 0 & 0 \\ S_{\theta_i} & C_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C_{\alpha_i} & -S_{\alpha_i} & 0 \\ 0 & S_{\alpha_i} & C_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} C_{\theta_i} & -S_{\theta_i}C_{\alpha_i} & S_{\theta_i}S_{\alpha_i} & a_iC_{\theta_i} \\ S_{\theta_i} & C_{\theta_i}C_{\alpha_i} & -C_{\theta_i}S_{\alpha_i} & a_iS_{\theta_i} \\ 0 & S_{\alpha_i} & C_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.2)$$

Where the four quantities θ_i, a_i, d_i, α_i are parameters associated with the link *i* and joint *i*. Since the matrix A_i is a function of a single variable, it turns out that three of the above four quantities are constant for a given link, during the fourth parameter, θ_i for a revolute joint and d_i for a prismatic joint, is the joint variable.

2.4 Frames assignments and special parameters

Figure 2.5 shows the proposed coordinates frames assignments according to the Denavit-Hartenberg convention. The coordinate frame No. zero is the reference frame, and it has been chosen in the front edge of the base link for the sake of simplifying the measurements. In this proposition, tool fame coincides with frame six because this work does not offer using a specific tool. The goal of the work is to build a robust controller for an educational robotic platform. According to this assignment, the spatial parameters are executed for the links and expressed in Table 2.1. The coordinate frames assignment is general and can be extended or modified due to any change or modification to the physical links. For example, if the maximum reach point is desired to be extended, then a longer link can be used. Consequently, the numerical values in Table 2.1 will be changed to the new length of the new link, but the same frames assignment is still valid.

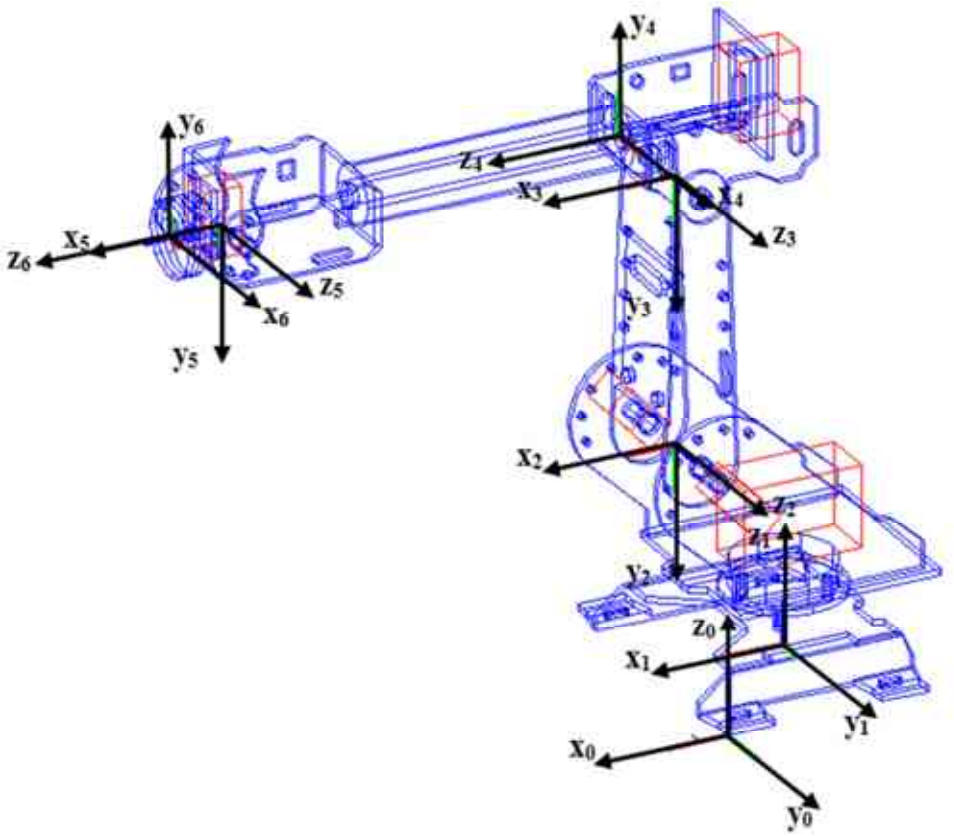

Figure 2.5 Frame coordinates assignments

Table 2.1 spatial parameters of the robot

Link	a_i	a_i	d_i	q_i
0	0	-46.13	0	0
1	-90	36.53	74.19	q_1
2	0	90	0	q_2
3	-90	17.8	0	$q_3 - 90$
4	-90	0	154	q_4
5	90	0	0	q_5
6	0	2	18	q_6

2.5 Forward kinematics

In this section, forward kinematic equations are driven for the robot shown in Figure 2.1 based on the Denavit-Hartenberg convention and spatial parameters in Table 2.1. Figure 2.6 shows a scheme for the forward kinematics where r_{ij} is the elements of the rotation matrix i, j = 1,2,3 and the position vector [x, y, z]^T is represented by the first three elements of the fourth column.

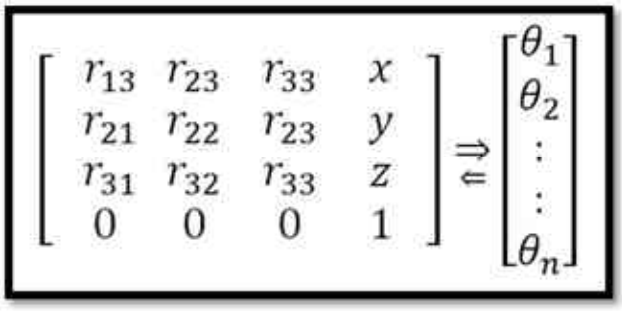

Figure 2.6 Robot kinematics

Each physical link is represented by a homogenous transformation matrix, and these matrices are 4×4 in size. The homogenous transformation matrix has the following form:

$$\text{Homogenous transformation matrix} =: \begin{bmatrix} & & & x \\ & (r)3\times3 & & y \\ & & & z \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where the 3×3 matrix represents the orientation matrix which is the orientation of the end-effector with respect to the base frame. The 3×1 vector $[x \quad y \quad z]^T$ is the position vector which is the position of the end-effector in Cartesian space relative to the base frame of the manipulator.

The only variables in the forward kinematic equations are the joint variables or joint angles. In other words, forward kinematics outputs are the position and orientation of the tool tip in the Cartesian space relative to the base given joints angles of the robot arm. Equation (2.3) to (2.9) express the homogenous transformation matrices of the links.

$$H_0 = \begin{bmatrix} 1 & 0 & 0 & a_0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$H_1 = \begin{bmatrix} C_{q1} & 0 & -S_{q1} & a_1 C_{q1} \\ S_{q1} & 0 & C_{q1} & a_1 S_{q1} \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$H_2 = \begin{bmatrix} C_{q2} & -S_{q2} & 0 & a_2 C_{q2} \\ S_{q2} & C_{q2} & 0 & a_2 S_{q2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

$$H_3 = \begin{bmatrix} C_{q3-90} & 0 & -S_{q3-90} & a_3 C_{q3-90} \\ S_{q3-90} & 0 & C_{q3-90} & a_3 S_{q3-90} \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

$$H_4 = \begin{bmatrix} C_{q4} & 0 & -S_{q4} & 0 \\ S_{q4} & 0 & C_{q4} & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

$$H_5 = \begin{bmatrix} C_{q5} & 0 & S_{q5} & 0 \\ S_{q5} & 0 & -C_{q5} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

$$H_6 = \begin{bmatrix} C_{q6} & -S_{q6} & 0 & a_6 C_{q6} \\ S_{q6} & C_{q6} & 0 & a_6 S_{q6} \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

where θ is the actuator angle while the attached number is the specific actuator and these angles are numbered from 1 to 6. The summation of two or more angles are represented by joining the

numbers of the corresponding angles for example θ_{23} is the summation of joint variables 2 and 3.

C and *S* are shortcuts for trigonometric functions *cos* and *sin*, respectively, and the spatial parameters are as follows:

$a_0 = -46.13\,mm$

$a_1 = 36.53\,mm$

$d_1 = 74.19\,mm$

$d_3 = 17.8\,mm$

$d_4 = 154\,mm$

$a_6 = 2\,mm$

$d_6 = 18\,mm$

The forward kinematics can be reached by multiplying equations (2.3) to (2.9) to obtain the overall transformation matrix, which holds the position and orientation of the robot hand.

$$T = H_0\,H_1\,H_2\,H_3\,H_4\,H_5\,H_6 \tag{2.10}$$

The 16 elements the *T* matrix are derived and described in details in equations (2.11) to (2.26), and it is clear from these equations that T matrix is a function only for joints variables of the actuators.

$$T_{11} = S_{q6}\left(C_{q4}S_{q1} - S_{q4}C_{q1}C_{q23}\right) + C_{q6}\left[C_{q5}\left(S_{q1}S_{q4} + C_{q4}C_{q1}C_{q23}\right) + S_{q5}C_{q1}S_{q23}\right] \tag{2.11}$$

$$T_{21} = -S_{q6}\left(C_{q1}C_{q4} + S_{q4}S_{q1}C_{q23}\right) - C_{q6}\left[C_{q5}\left(C_{q1}S_{q4} - C_{q4}S_{q1}C_{q23}\right) - S_{q5}S_{q1}S_{q23}\right] \tag{2.12}$$

$$T_{31} = C_{q6}\left(S_{q5}C_{q23} - C_{q4}C_{q5}S_{q23}\right) + S_{q4}S_{q6}S_{q23} \tag{2.13}$$

$$T_{41} = 0 \tag{2.14}$$

$$T_{12} = C_{q6}\left(C_{q4}S_{q1} - S_{q4}C_{q1}C_{q23}\right) - S_{q6}\left[C_{q5}\left(S_{q1}S_{q4} + C_{q4}C_{q1}C_{q23}\right) + S_{q5}C_{q1}S_{q23}\right] \tag{2.15}$$

$$T_{22} = S_{q6}\left[C_{q5}\left(C_{q1}S_{q4} - C_{q4}S_{q1}C_{q23}\right) - S_{q5}S_{q1}S_{q23}\right] - C_{q6}\left(C_{q1}C_{q4} + S_{q4}S_{q1}C_{q23}\right) \tag{2.16}$$

$$T_{32} = \mathrm{C}_{q6}\mathrm{S}_{q4}\mathrm{S}_{q23} - \mathrm{S}_{q6}\left(\mathrm{S}_{q5}\mathrm{C}_{q23} - \mathrm{C}_{q4}\mathrm{C}_{q5}\mathrm{S}_{q23}\right) \tag{2.17}$$

$$T_{42} = 0 \tag{2.18}$$

$$T_{13} = \mathrm{S}_{q5}\left(\mathrm{S}_{q1}\mathrm{S}_{q4} + \mathrm{C}_{q4}\mathrm{C}_{q1}\mathrm{C}_{q23}\right) - \mathrm{C}_{q5}\mathrm{C}_{q1}\mathrm{S}_{q23} \tag{2.19}$$

$$T_{23} = -\mathrm{S}_{q5}\left(\mathrm{C}_{q1}\mathrm{S}_{q4} - \mathrm{C}_{q4}\mathrm{S}_{q1}\mathrm{C}_{q23}\right) - \mathrm{C}_{q5}\mathrm{S}_{q1}\mathrm{S}_{q23} \tag{2.20}$$

$$T_{33} = -\mathrm{C}_{q5}\mathrm{C}_{q23} - \mathrm{C}_{q4}\mathrm{S}_{q5}\mathrm{S}_{q23} \tag{2.21}$$

$$T_{43} = 0 \tag{2.22}$$

$$\begin{aligned} T_{41} = x = {} & a_0 - d_4 C_{q1} S_{q23} + a_1 C_{q1} + d_6\left[S_{q5}\left(S_{q1}S_{q4} + C_{q4}C_{q1}C_{q23}\right) - C_{q5}C_{q1}S_{q23}\right] + \\ & a_6 S_{q6}\left(C_{q4}S_{q1} - S_{q4}C_{q1}C_{q23}\right) + a_6 C_{q6}\left[C_{q5}\left(S_{q1}S_{q4} + C_{q1}C_{q4}C_{q23}\right) + S_{q5}C_{q1}S_{q23}\right] + \\ & a_3 C_1 C_2 C_3 - a_3 C_1 S_2 S_3 + a_2 C_{q1} C_{q2} \end{aligned} \tag{2.23}$$

$$\begin{aligned} T_{42} = y = {} & a_1 S_{q1} - d_4 S_{q1} S_{q23} - d_6\left[S_{q5}\left(C_{q1}S_{q4} - C_{q4}S_{q1}C_{q23}\right) + C_{q5}S_{q1}C_{q23}\right] + a_2 C_{q2} S_{q1} - \\ & a_6 S_{q6}\left(C_{q1}C_{q4} + S_{q4}S_{q1}C_{q23}\right) - a_6 C_{q6}\left[C_{q5}\left(C_{q1}S_{q4} - C_{q4}S_{q1}C_{q23}\right) - S_{q5}S_{q1}S_{q23}\right] + a_3 S_{q1} C_{q23} \end{aligned} \tag{2.24}$$

$$\begin{aligned} T_{43} = z = {} & d_1 - d_4 C_{q23} - a_2 S_{q2} - d_6\left(C_{q5}C_{q23} + C_{q4}S_{q5}S_{q23}\right) - a_3 S_{q23} + \\ & a_6\left[C_{q6}\left(S_{q5}C_{q23} - C_{q4}C_{q5}S_{q23}\right) + S_{q4}S_{q6}S_{q23}\right] \end{aligned} \tag{2.25}$$

$$T_{44} = 1 \tag{2.26}$$

2.6 Motors specifications

Figure 2.7 reveals MG 996 digital servo motor which is used to actuate joints 1, 2, and 3 of the robot arm. It is a high torque with metal gearing and dual bearing producing 10 kg stalling torque though its tiny size.

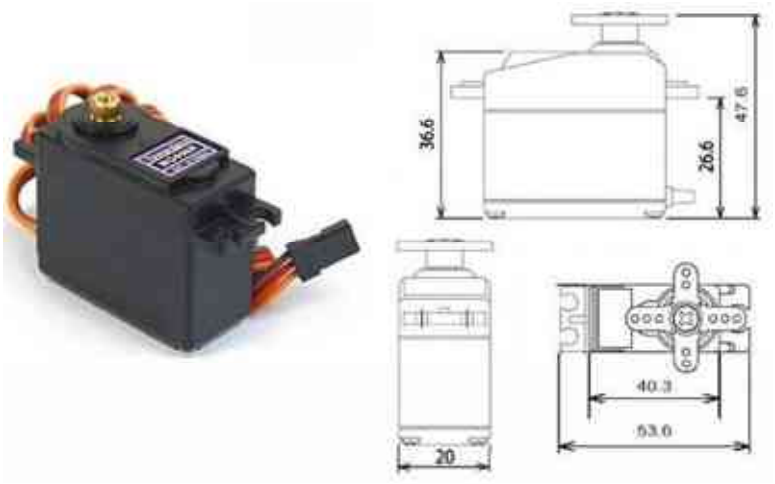

Figure 2.7 MG 996 servo motor

2.7 3D printed version

Figure 2.8 shows a 3D printed version of the robot shown in Figure 2.1, the filament used is PETG, and it has good mechanical properties as well as shiny finishing.

Figure 2.8 3D printed 6 DOF robot arm using PETG filament

The 3D printing option has many advantages on the aluminium version; the manufacturing process for 3D printing is simpler than the aluminium one. All we need is a 3D printer that can be used everywhere. The 3D printing is silent and does not have massive equipment like laser cutter as in the case of the aluminium arm. The mechanical properties of the PETG sheet of 4 mm are similar to the properties of the aluminium sheet of 2 mm thickness. However, even with the difference in thickness, the PETG arm is lighter in weight compared to the aluminium one.

Anyway, studying the mechanical properties and considering stress-strain analysis is out of the scope of this work, and it is suggested as future work for other researchers. Figure 2.9 shows the printing process of link 4 of the robot manipulator using the Ender series printer.

Figure 2.9 Printing operation for link 4 for the robot arm

3. INVERSE KINEMATICS

3.1 Introduction

Inverse kinematic IK solution for a specific configuration of the end-effector of a robot is a crucial function in the control system. Optimization algorithms like particle swarm optimization and artificial bee colony (Ghafil and Jármai 2018a) can guarantee an optimal solution for the problem because of the nature of the objective function, which is a differentiable one. However, we cannot use optimization algorithms for real-time applications because they consume a moderated time. Artificial neural networks ANN are perfect for real-time application even if their solutions are approximated and not as accurate as optimization algorithms. We shall manage between accuracy and immediate approximated solutions, especially for robots equipped with sensors and need fast solutions to respond to sudden action. This is a reasonable justification to consider learning neural networks how to solve the inverse kinematic problem of a robot. In this Chapter, the effect of using the Levenberg-Marquardt learning algorithm, Bayesian regularization learning algorithm, and scaled conjugate gradient learning algorithm is studied while learning one input layer, one hidden layer, and one output layer how to find the inverse position of a RRR robot arm. The topology of the neural network has been discussed in detail as well as the advantage and disadvantages of the abovementioned learning algorithms.

3.2 Optimization algorithms for IK

This is the problem of finding the joint variables from the given position and orientation of the end-effector. While forward kinematics detects the position and orientation of the end effector from the given set of joint variables, inverse kinematics is the inverse operation, but it is somewhat complicated.

$$\begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix} \Rightarrow \Leftarrow \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

There are many approaches mentioned in other books to deal with this problem, like closed-form and a geometric one. These methods are robot-dependent and differ from one manipulator to another. One can find the solution for the 6DOF manipulator but hardly or cannot find a theoretical solution for another 5DOF. In this section, we will focus on the solution from the viewpoint of neural networks to develop a functional relationship for any robot manipulator. While forward equations are a straightforward process, we will rely on these equations to establish the set of training data for the inverse problem. From the viewpoint of optimization algorithms (Ghafil and Jármai 2020b), consider Figure 3.1, for a specific robot configuration, the current position vector of the end-effector can be represented by the distance from the base of the end-effector of the manipulator while the desired position vector represents the task point. Obviously, if the difference between these two vectors is zero, then the tooltip will be in the proper position at the task point, and this is the objective function f of the inverse problem

$$f = \|Ci - De\| \tag{3.2}$$

where *Ci* denotes the instantaneous position vector, and *De* is the desired position vector. In other words, equation (3.4) is the function that has to be minimized as much as possible, and it is just the distance between the end-effector and task point.

$$f = \sqrt{(x_{Ci} - x_t)^2 + (y_{Ci} - y_t)^2 + (z_{Ci} - z_t)^2} + (R_{Ci} - R_t) \tag{3.3}$$

Where *t* refers to the task point coordinates which is given for inverse Kinematic problem, R_{ci} is the rotation matrix of the iterative solution, which is provided by the optimization algorithm, and R_t is the desired or given rotation matrix of the end-effector.

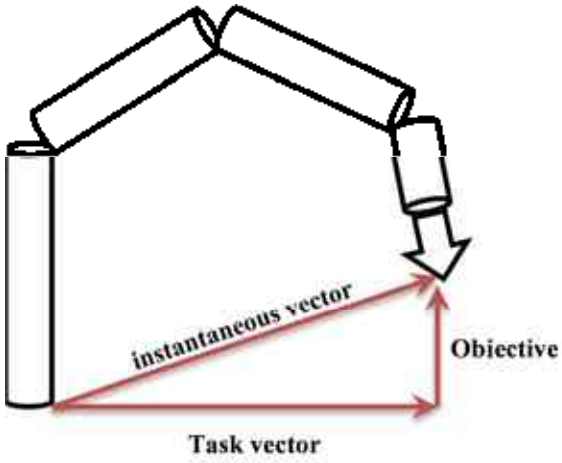

Figure 3.1 Representation of the objective function for inverse Kinematic problem

If the first term of equation (3.3) has been used alone as an objective function, we may get the end-effector in the task point but with many choices of orientations. Figure 3.2 reveals an example of two orientation options for the same task point, and manipulators with a high degree of freedom have even an infinite number of orientations. However, it is worth considering that the developed objective function in this section can solve the inverse kinematics of any type of serial robot arm with any degree of freedom.

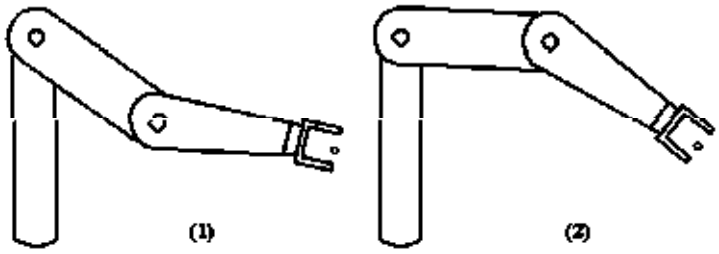

Figure 3.2 Different orientations for the same position

Once again, the inverse Kinematic solution obtained from optimization algorithms needs time measured by minutes (at least one or half a minute). This is inconvenient if we need a quick response in part of a second. The inverse kinematic problem here is a minimization problem where equation 3.3 should be driven to zero. The best performance was found by using particle swarm optimization. However, it is not global since PSO cannot return a solution with a cost function value less than 59 even with a high number of iteration and populations. In other words, PSO, like other metaheuristics, falls into local minima and cannot escape from that to the global point in the search space. However, the provided solution by PSO is reliable and acceptable, but the big difference will be on orientation value. For each independent run, PSO will return a similar position vector and totally different configuration or, say, different orientation, see Figure 3.3. We can accept falling in local minima and choose the proper configuration for a point in space by setting constraints during the iterative process of the PSO.

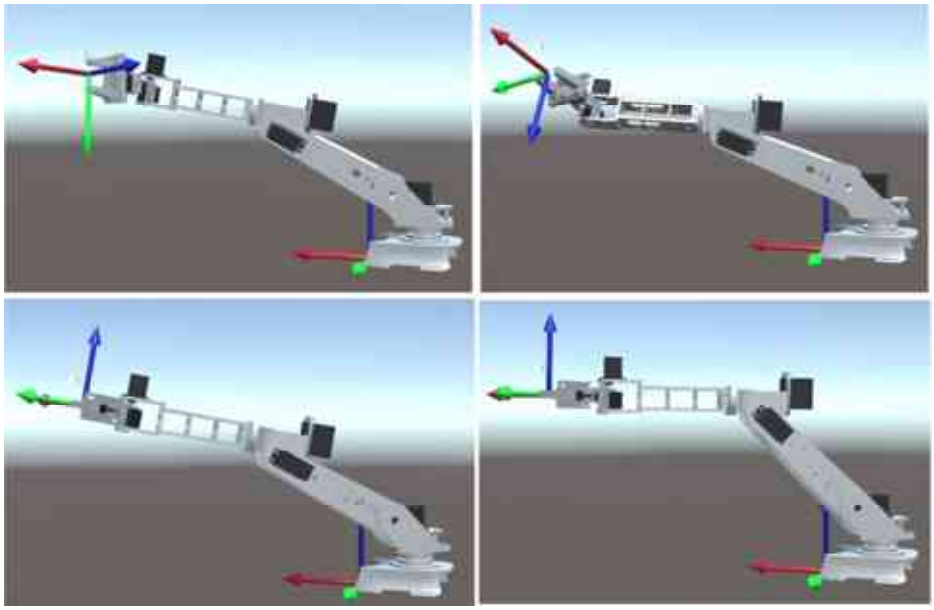

Figure 3.3 Different configurations for each independent run for the PSO algorithm

3.3 Artificial neural networks

In this thesis, an artificial neural network will be discussed and presented as another option to solve the inverse kinematics for the robot arm. Artificial neural network provides a nonlinear mapping between inputs and outputs. It is suitable for the robot controller to find a set of joint angles for a corresponding end-effector position and orientation.

3.3.1 Activation function

The goal of using activation functions is to enable the neural network to deal with essential data and ignore irrelevant information. Figure 3.4 illustrates the mechanism work of the activation function, which is similar to the functionality of the nerve cell body where it segregates the data according to its importance.

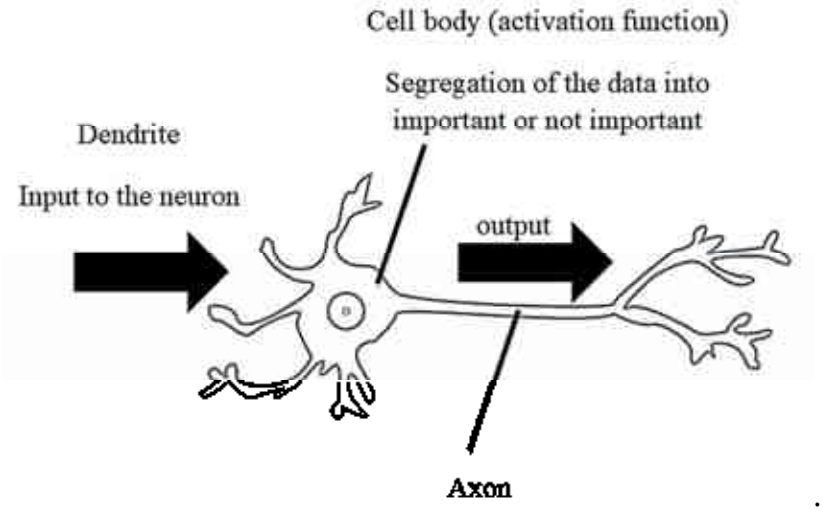

Figure 3.4 Analogy scheme of an activation function/nerve cell body

This mechanism helps the artificial neural network to analyze the inputs according to their importance and pay less attention to the data of the lower utility. Thus, the responsibility of the activation function is to activate the neuron or not according to a set of inputs and provide the output only if it is activated. Specifically, the activation function, also called a transfer function, fires outputs depending on the nature of the inputs, the same as what is happening in our minds. There are many types of activation functions; each one is appropriate for a specific application. To find the inverse kinematic solution, which is a regression problem, the *tansig* was used as an activation function in this study. Figure 3.5 shows the mathematical representation of the tansig activation function.

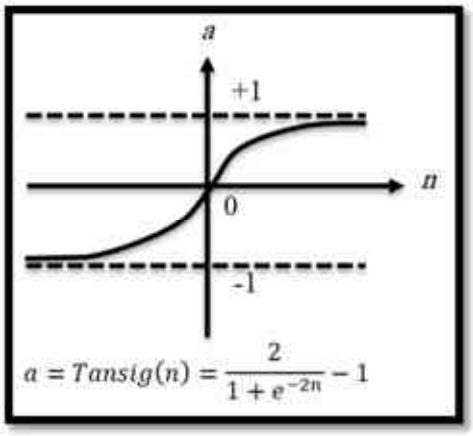

Figure 3.5 Tansig activation function

3.3.2 Artificial neural networks for IK

From forward Kinematic equations, a data set of 2000 input/output training data is generated to learn the proposed topology of neural networks how to solve the inverse position problem of the

robot using information out of the training data. Figure 3.6. reveals the topology of the artificial neural network used in this study, where it has 12 inputs, one hidden layer, and three outputs. The input layer is the elements of the homogenous transformation matrix that relate the end-effector to the base frame, and they are explained in equation (3.9). In this chapter, we have used one hidden layer with a number of neurons n=10 and n=100. The output layer is consists of three neurons that represent the three joint angles of the robot arm.

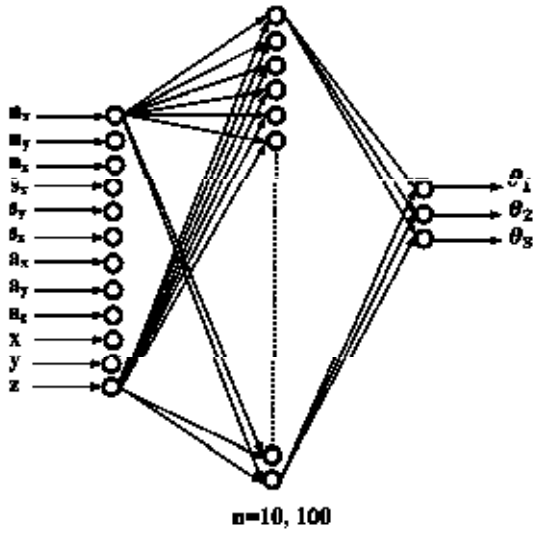

Figure 3. 6 ANN architecture for RRR robot manipulator

3.3.3 Results and discussion

Three learning algorithms are available for MATLAB neural network toolbox, and we have employed them and compared the obtained results in cases of n=10 and n=100. The proposed neural network has been learned using three learning algorithms; Levenberg-Marquardt algorithm LM, Bayesian Regularization algorithm BR, and Scaled Conjugate Gradient algorithm SCG, where each of these algorithms has its own advantage and disadvantage that should be explained in this study. Figures 3.7 to 3.12 show the performance of Levenberg-Marquardt, Bayesian Regularization, and Scaled Conjugate Gradient, respectively, on the learning process in the case of the number of neurons equal to 10. The Levenberg-Marquardt algorithm needs less time than Bayesian Regularization and more time than Scaled Conjugate Gradient, but it needs more memory. Also, LM is more accurate than SCG and less accurate than BR. By repeating the same comparison in the case of n=100, see Figures 3.13 to 3.18, it is easy to configure the considerable reduction in mean square error during the test compared with the case when n=10.

A data set of 2000 items is used for the learning process where 70% of the items are used for training while 30% have been used for the test.

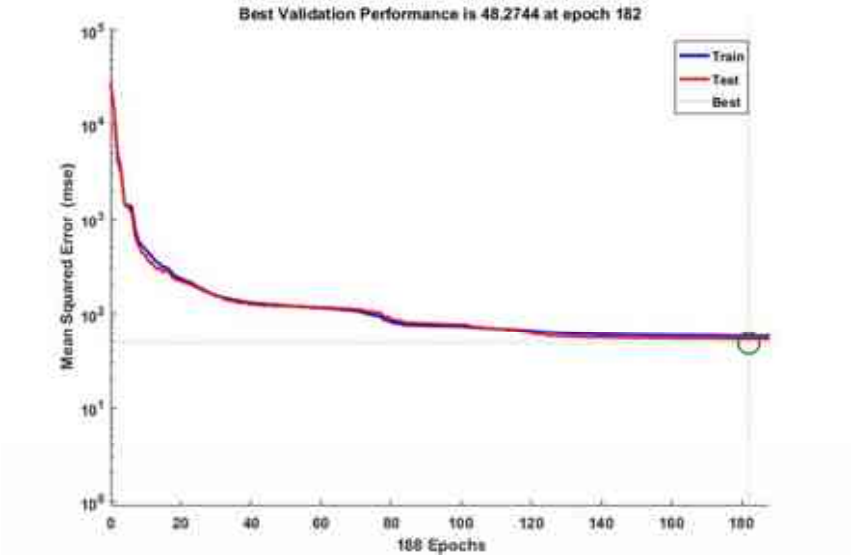

Figure 3.7 Mean square error for Levenberg-Marquardt algorithm with n=10

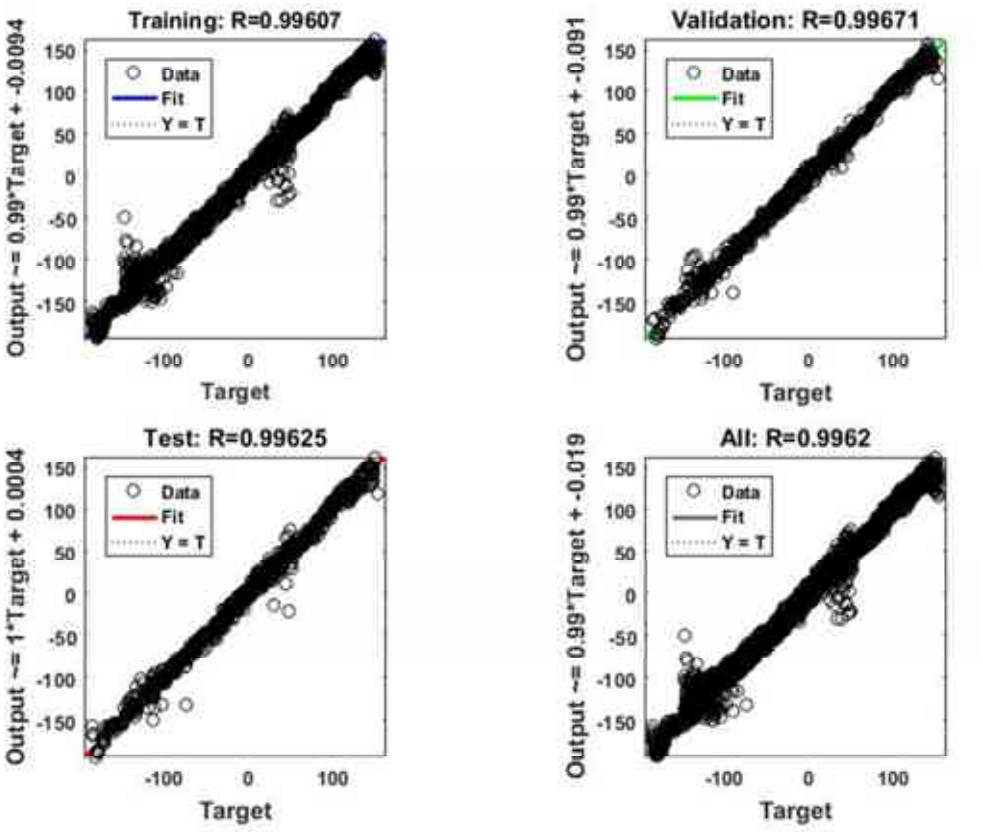

Figure 3.8 Regression of Levenberg-Marquardt algorithm with n=10

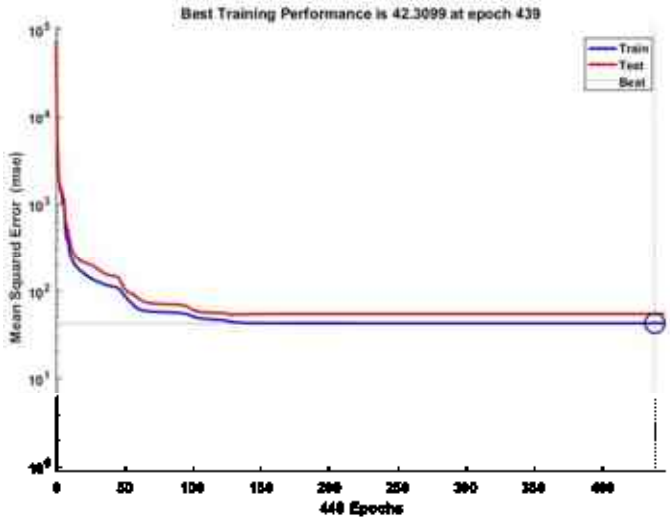

Figure 3.9 Mean square error for Bayesian Regularization algorithm with n=10

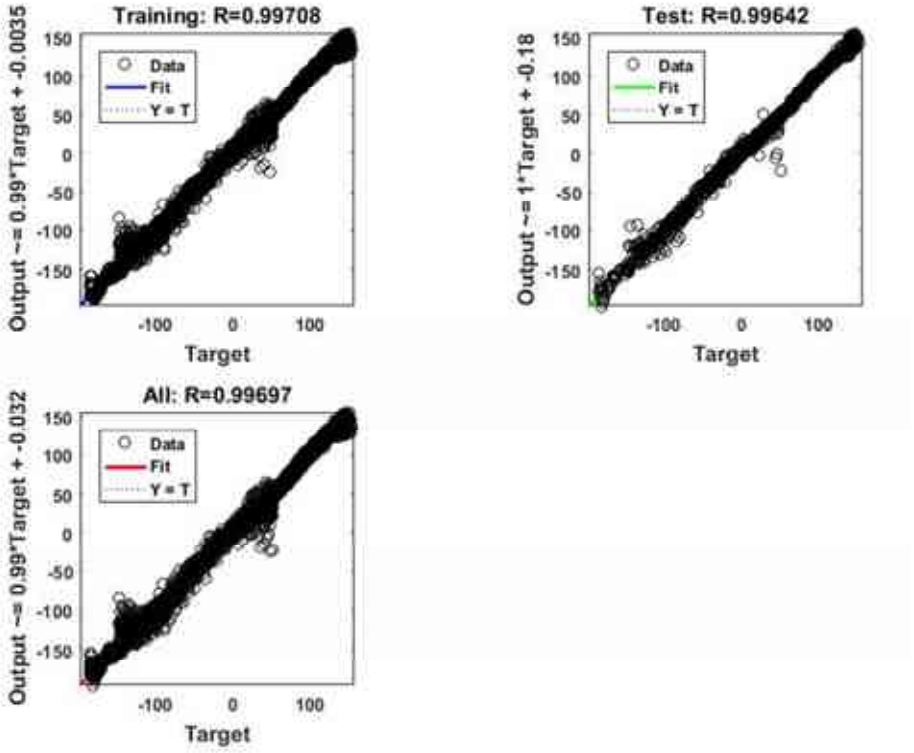

Figure 3.10 Regression of Bayesian Regularization algorithm with n=10

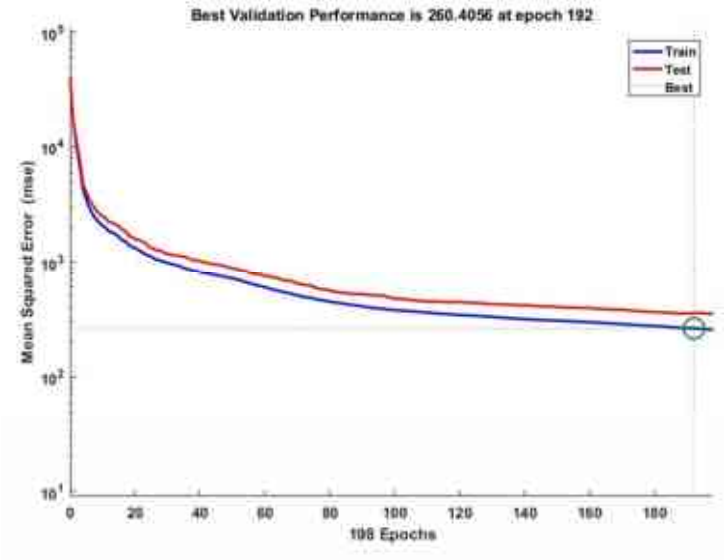

Figure 3.11 Mean square error for Scaled Conjugate Gradient algorithm with n=10

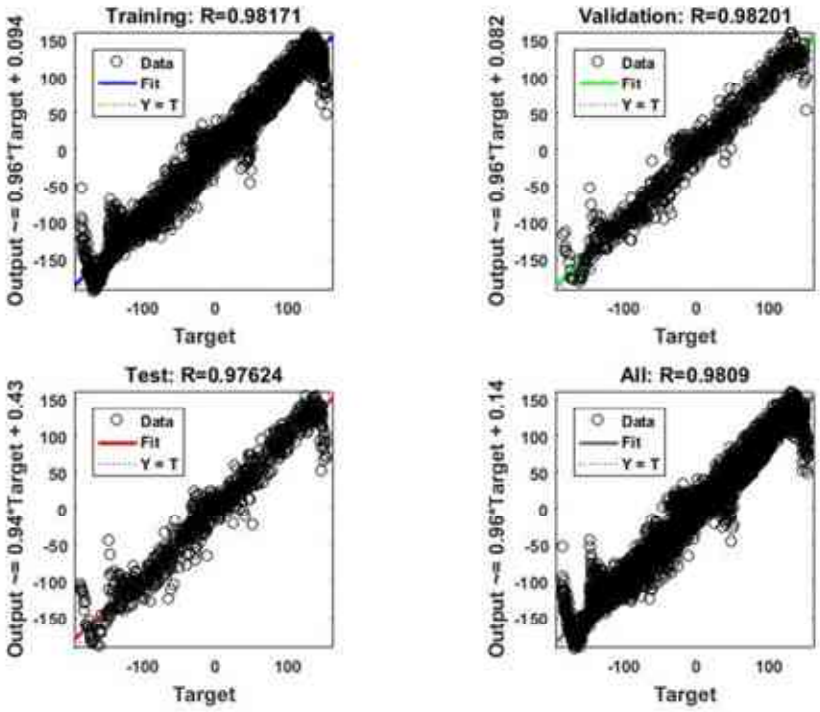

Figure 3.12 Regression of the Scaled Conjugate Gradient algorithm with n=10

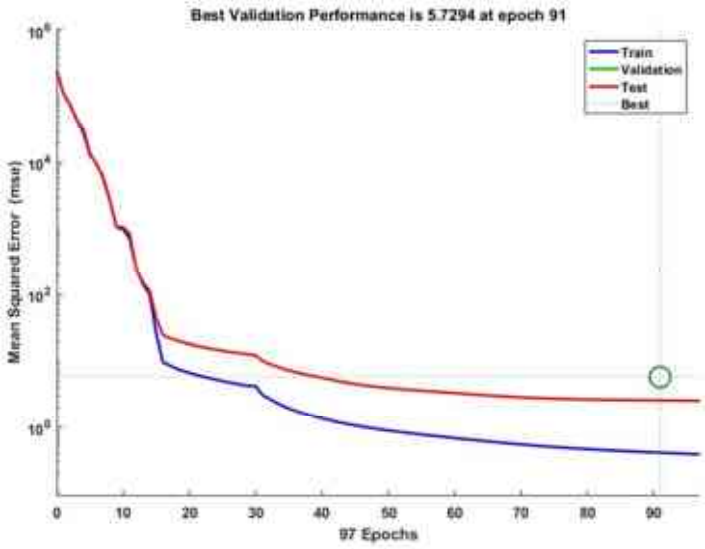

Figure 3.13 Mean square error for Levenberg-Marquardt algorithm with n=100

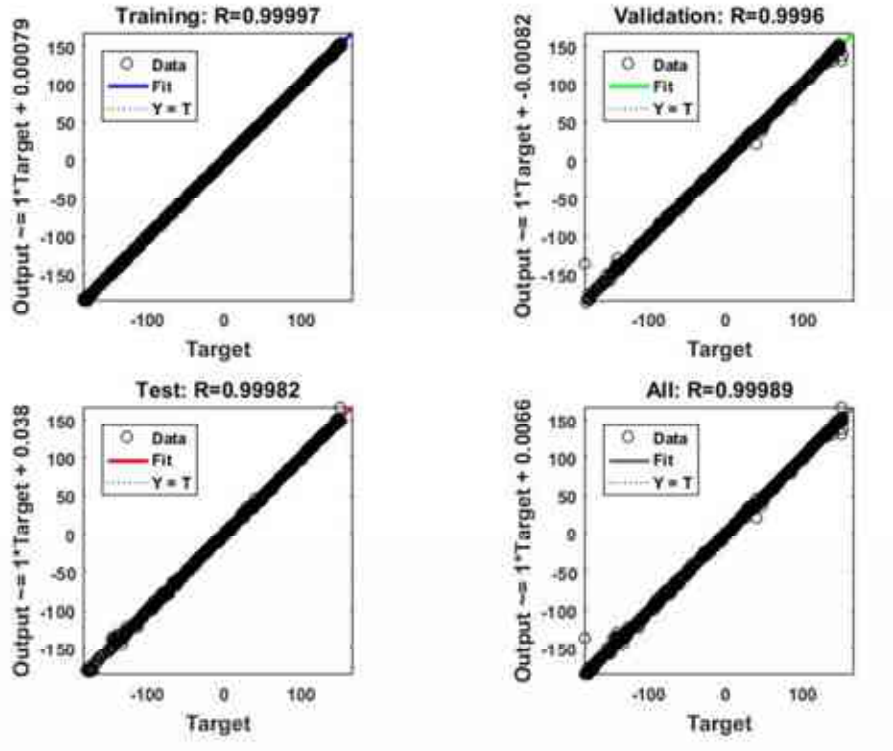

Figure 3.14 Regression of the Levenberg-Marquardt algorithm with n=100

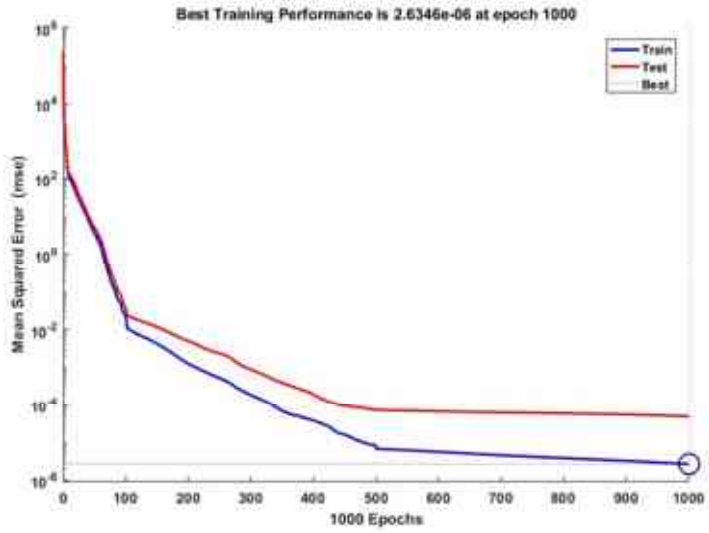

Figure 3.15 Mean square error for Bayesian Regularization algorithm with n=100

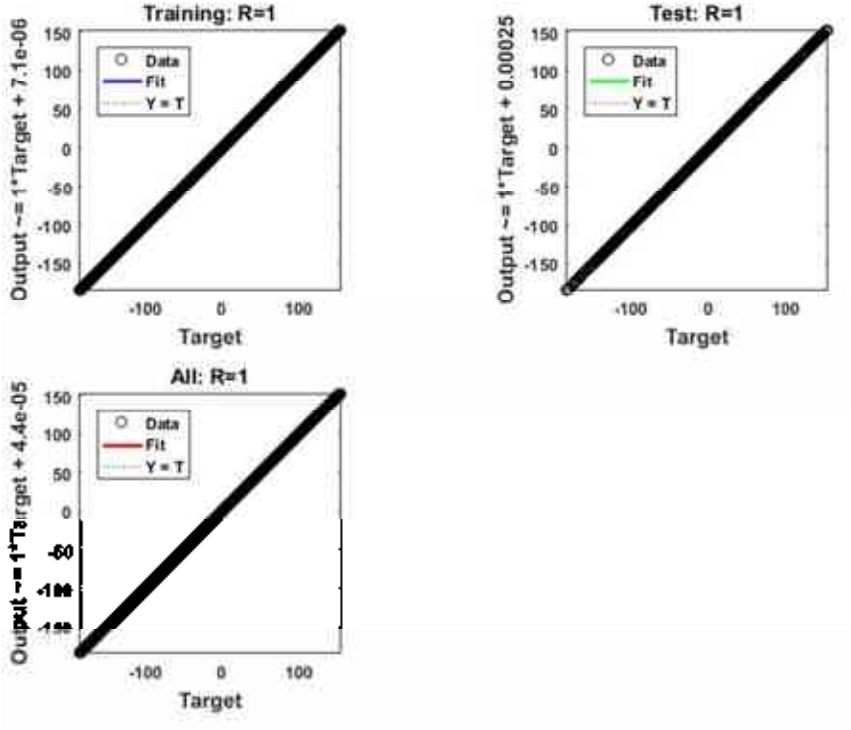

Figure 3.16 Regression of the Bayesian Regularization algorithm with n=100

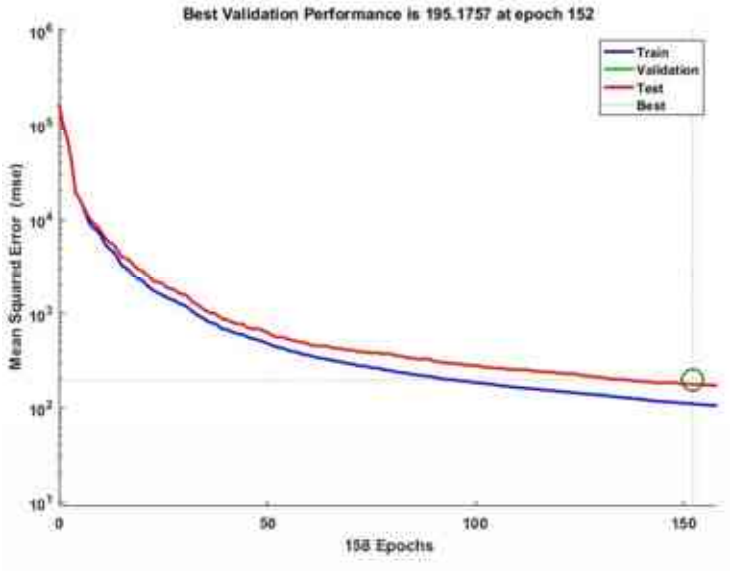

Figure 3.17 Mean square error for Scaled Conjugate Gradient algorithm with n=100

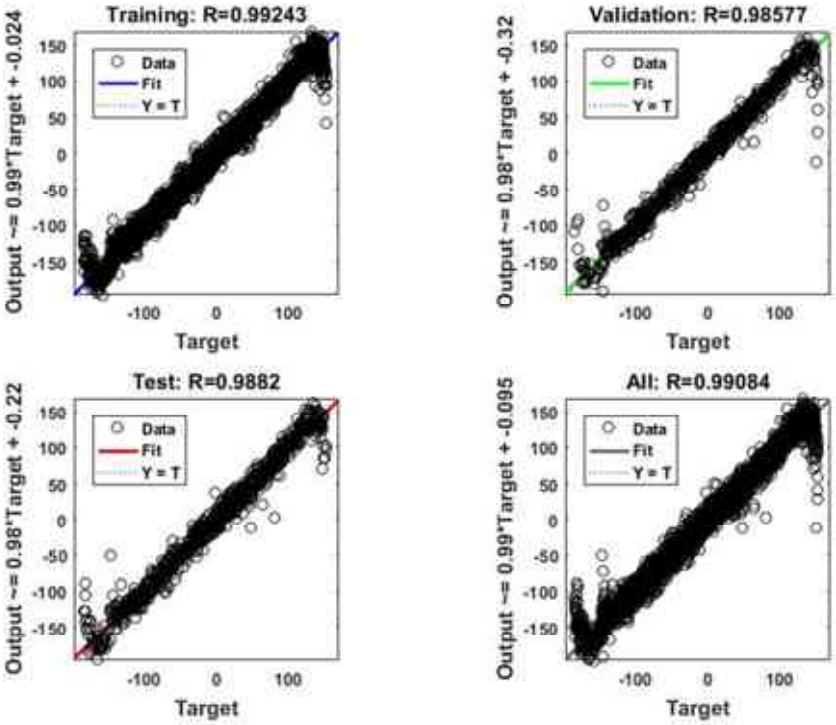

Figure 3.18 Regression of the Scaled Conjugate Gradient algorithm with n=100

Axiomatically, an increasing number of neurons in the hidden layer can lead to the best results with less error. However, here we have to evaluate the performance of the three learning algorithms. Table 3.1 reveals the elapsed time to learn the network how to solve inverse Kinematic of the robot where BR needs more time than LM while SCG has taken a very short time. Back to Figure 3.13 to Figure 3.18, increasing the number of neurons to 100 has led to a

dramatic drop in mean square error MSE in the case of LM and BR learning algorithms. However, the SCG algorithm still cannot learn the network solving inverse Kinematic. Figure 3.17 and Figure 3.18 show the weak performance of the SCG algorithm on this kind of nonlinear problem. Table 3.2 demonstrates MSE induced using the three mentioned learning algorithms on the proposed problem.

Table 3.1 Time consuming of Levenberg-Marquardt, Bayesian Regularization, and Scaled Conjugate Gradient learning algorithms

learning algorithm	time (sec)	
	n=10	n=100
Levenberg-Marquardt	4	171
Bayesian Regularization	8	1840
Scaled Conjugate Gradient	0.01	1

Table 3.2 Mean square error of using Levenberg-Marquardt, Bayesian Regularization, and Scaled Conjugate Gradient learning algorithms

learning algorithm	Mean square error MSE	
	n=10	n=100
Levenberg-Marquardt	48.2744	5.7294
Bayesian Regularization	42.3099	2.6346E-06
Scaled Conjugate Gradient	260.4056	195.1757

3.4 Conclusion

A neural network is a powerful tool for nonlinear problems like the inverse kinematic problem. The neural network represents a function between inputs and outputs represented by distributed weights on connections among neurons. In this chapter, we have studied the effect of using three different learning algorithms, namely, Levenberg-Marquardt algorithm, Bayesian Regularization algorithm, and Scaled Conjugate Gradient algorithm, to learn a neural network how to solve inverse Kinematic of a three revolute joints robot manipulator. The topology of the network consists of one input layer with twelve neurons, one hidden layer with the number of neurons n=10, 100, and one output layer with three neurons. The twelve neurons in the input layer represent the given element of the homogenous transformation matrix, while the three neurons in the output layer represent the desired joint angles of the robot. During the study, we have found that the Scaled Conjugate Gradient learning algorithm was unable to learn the network the

proposed problem even with a high number of neurons in the hidden layer. Bayesian Regularization learning algorithm returned the best results but with elapsed time greater than what is required for the Levenberg-Marquardt algorithm.

4. PATH AND TRAJECTORY PLANNING

4.1 Introduction

In this chapter, we will conduct path and trajectory planning of the end effector of a selected robot manipulator. Optimization algorithms were used for this problem. A comparative study was done among these metaheuristics, namely invasive weeds optimization, harmony search, Dynamic differential annealed optimization, flower pollination (Ghafil and Jármai 2019a), particle swarm optimization, and firefly algorithm. The spline curves method is employed for the path planning as well as using the A* heuristic algorithm.

4.2 Path planning

Over the last decades, robot manipulators have received much of the robot community's attention due to their wide applications in industry for many tasks. The most important function in robot motion is to plan this motion to reach a point in the shortest path, consuming less time and, consequently, consuming less power for its actuators. The main advantage of planning is that it enables a robot to achieve complex goals when moving from one configuration to another in a cluttered environment. Path planning is just a geometric operation to describe the motion of a robot. Path planning for articulated robotic manipulators is usually more challenging than for mobile robots because of the high degrees of freedom.

4.2.1 A-star algorithm

A* algorithm was introduced as a path generator that takes start position, goal position, and obstacles and returns a series of points that define the optimal path traversed between start and goal position in the configuration subspace. In this study, path planning occurs at the Cartesian-Coordinate level then transforms it to the joint-Coordinate level via the associated inverse Kinematics. The inverse kinematics of a robot requires the specification of the end effector's orientation. In this study the orientation is assumed to be constant and gained from the orientation of the path generated itself. Path planning 2D space greatly saves time because

planning in 3D space is very expensive in consumes time. However, in some cases, under simple 3D situations, it can be mapped 3D problem area to 2D in order to use the traditional A* algorithm. The A* (pronounced “Ay star”) search algorithm deals with the searching area as a grid collection and generates a path between two given points. On this path, the whole robot is moved in the case of mobile robots, or a part of the robot will follow the path in the case of stationary robot manipulators. As shown in Figure 4.1, each node is chosen according to the relation [24]:

$$f(n) = h(n) + g(n). \tag{4.1}$$

Start node moves to the next node, which has a minimum cost function *f(n)* where *h(n)* is the distance between the n node, which at the start of the algorithm be the start node and any of its successors and *g(n)* is the distance between this any successor and the target

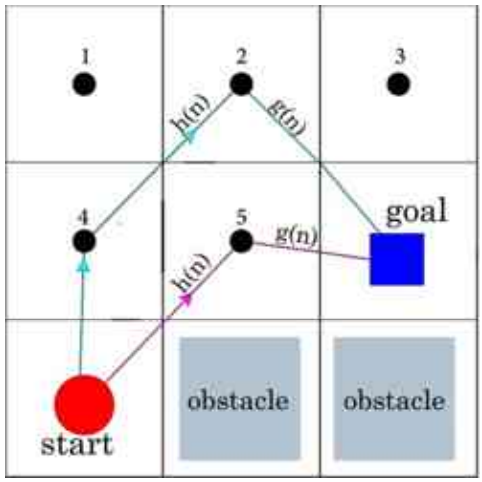

Figure 4.1 grid used by A* algorithm to reach the goal

To force the path to take a specific way, this can be done by throwing obstacles, and due to avoidance of these obstacles, the path takes the desired path in its map, as shown in Figure 4.2.

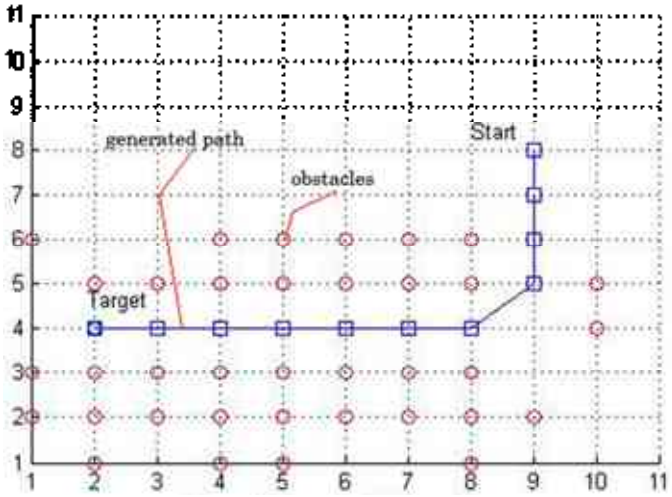

Figure 4.2 MATLAB plot window, forcing a path to take a specific way

4.2.2 Experiment on A*

Figure 4.3 presents a model of an environment full of obstacles designed by the AutoCAD package. The coordinates of these obstacles are defined in the A* algorithm and run to generate a path between the start point, which is its coordinates (120,50) and the goal point, which is its coordinates (111,262). Figure 4.4 reveals the path generated by the A-star algorithm.

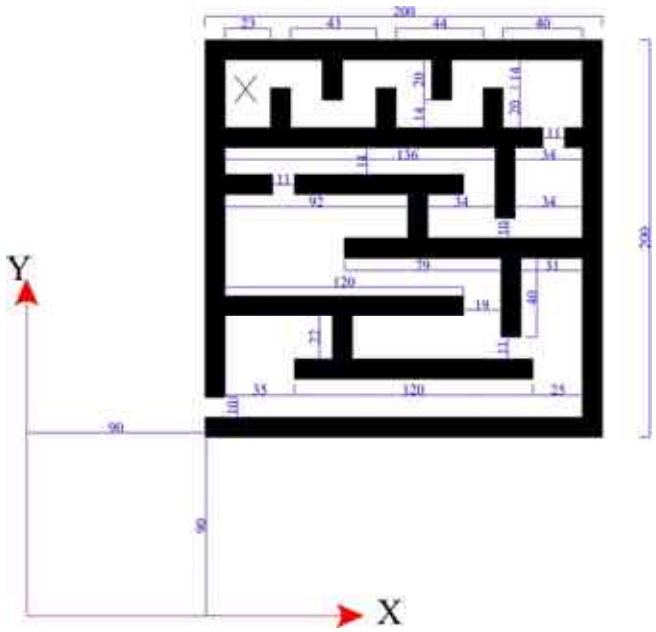

Figure 4.3 Layout of a static environment full of obstacles

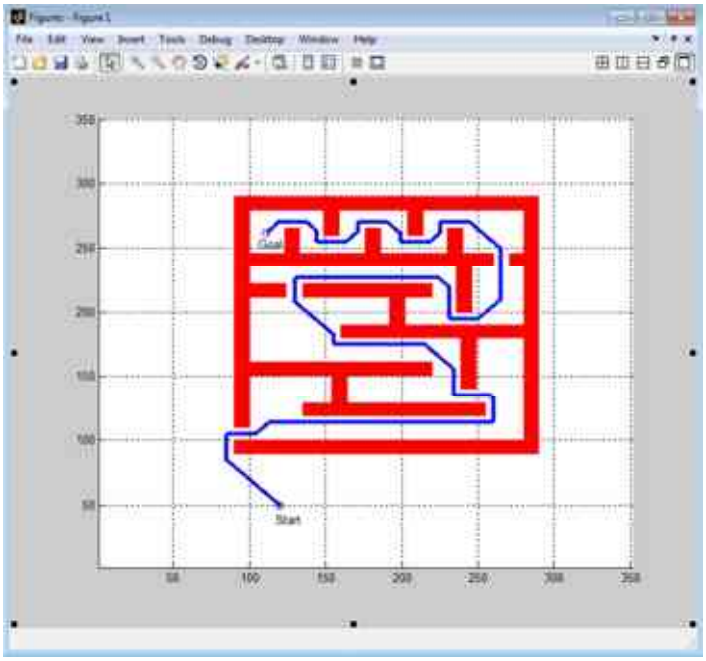

Figure 4.4 optimum path which avoids all the obstacles in the environment

4.2.3 Metaheuristics for Path planning

Metaheuristic algorithms were also applied to practical engineering problems such as path planning (GHAFIL 2013). Path planning is a fundamental optimization problem in robotics, and it has received significant consideration over the last decades. The best usage of path planning can be seen in industrial robots (Gasparetto and Zanotto 2010) for the automotive industry and self-driving vehicles (Li and Shao 2015). Path planning is the problem of finding the shortest path between two points in a free-of-collision environment or with a collision environment. Definitely, the shortest path between two points on a curve in a free of obstacle workspace is just a straight line, while the problem gets complicated when the workspace contains obstacles. Consider the workspace shown in Figure 4.5; there is a constrained environment with brown circular obstacles. The workspace is limited such that $x \in [-10,30]$ and $y \in [-10,30]$ mm, and the constraints are as shown in Table 4.1. In this example, a cubic spline curve was used to generate the desired path with smooth nature. To find the optimum path by an optimisation algorithm one can follow the following procedure:

1- Define three variables that represent the control points on the cubic spline, and each variable is a structure of two sub-variables that represent the x-axis and y-axis for the control point, so in total, we have six variables, i.e., three x-axes and three y-axes for the three points.

2- Set random variables for the six optimization variables according to side constraints.

3- A cubic spline is fitted onto the control points as well as start and goal points, using the specified (x,y) coordinates. Basically, the construction of the cubic spline curve take the following form

$$s_i = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \tag{4.2}$$

by fixing the coefficients $a_i, b_i, c_i,$ and d_i, the entire curve is fixed by using boundary conditions for the Not-a-knot spline where the five points, start, goal, and three control points represent these boundary conditions. In this study, we have used the MATLAB toolbox to find the curve given points.

4- Use the obtained cubic spline curve to generate, for example, 100 equally spaced points along the curve. MATLAB uses Not-a-knot spline (Behforooz 1995) represented by function *splin* (Mathworks 2019), which is used in code implementation for this study. It is possible to use the 4th polynomial instead of the cubic spline curve. However, we have used the cubic spline curve in this study for simplicity because MATLAB calculates it automatically.

5- Evaluate the distance between each successive point on the curve and sum them to calculate the total length of the curve. The following is the objective function of the path planning problem that should be minimized

$$O = \sum_{i=1}^{n} \sqrt{(x_{i-1} - x_i)^2 + (y_{i-1} - y_i)^2} + penalty \tag{4.3}$$

Where O is the objective (length of the curve), i=1,..., n. the static penalty was used for path planning problem where it should have a high value when the path contacts any obstacle in the workspace; otherwise, it has zero value.

6- If the spline curve collides with an obstacle, set a penalty to a high value, for example, 1000; if not, set it to zero.

7- Repeat the process each time the optimizer updates solution and stop optimization if the limit number of iterations is reached.

Table 4.2 shows the competition among PSO, FPA, DDAO (Ghafil and Jármai 2020a), FF, HS, and IWO to find the minimum distance between start and goal points explained in Figure 4.5.

For this study, ten independent runs were used with 10000 FE and the same workspace illustrated in Figure 4.5. DDAO has found a path shorter than what FPA, HS, and IWO have found, while PSO returned the shortest path among all algorithms.

Table 4.1 Definition of circular obstacles in the 2-dimensional path programming problem

No.	x-axis	y-axis	dimensions
1	6	6	r = 3
2	20	11	r = 2
3	23	6	3*3
4	9	15	7*7
5	7	24	2*2
6	16	23	3*3
7	24	23	6*2

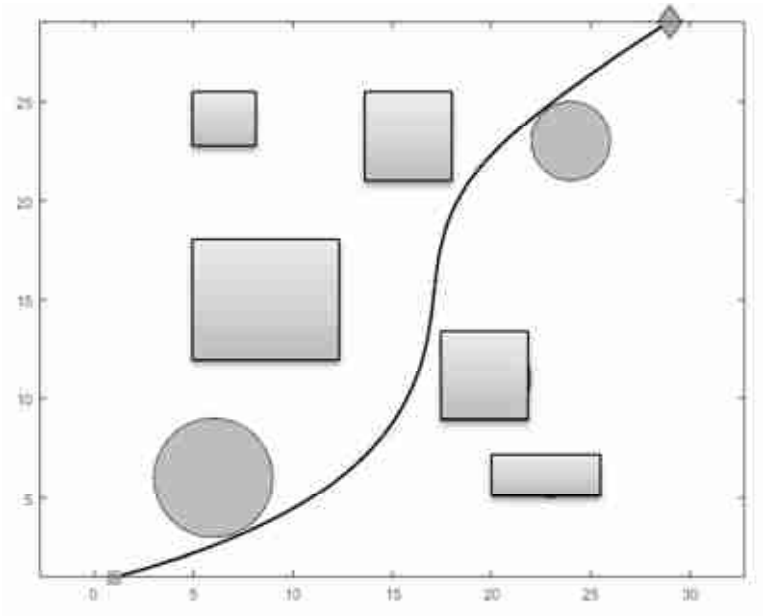

Figure 4.5 The path connecting start and goal points

Table 4.2 Comparison of different optimization algorithms on the path planning problem using a Spline curve

	PSO	FPA	DDAO	FF	HS	IWO
Best	**40.6636**	43.2796	41.6742	40.8410	43.0543	43.2263
STD	1.1275	4.1464	1.4902	1.2253	**0.3036**	2.9872

4.3 Trajectory planning

Knowing the dynamic equations of the robot arm is very necessary for the mechanical design of the arm. Some of the forming parameters in these equations are the angular position, velocity, and acceleration of the actuators. Using the Euler-Lagrange equations, one can estimate the stress-strain distribution along with the robot physical links. This is necessary for the fatigue design of the arm or estimating the failure limits. Employing the Euler-Lagrange equations, it could be easy to do vibration analysis, which is very important in position tracking and control.

In this work, trajectory planning (Ghafil and Jármai 2018c) was done using flower pollination algorithm FPA (Yang 2012) and (Alyasseri et al. 2018) to calculate angular velocity and acceleration trajectories needed for dynamical analysis for each robot configuration. FPA is used for global optimization by many researchers in different topics. The flower pollination algorithm was used for the path planning process to calculate optimum positions of the intermediate points, which will be used in the trajectory planning process. The same algorithm is used to optimize the position of the robot tip for the inverse kinematic, which is necessary to transfer all points, start, end, and intermediate points of the Cartesian space to joint space where the trajectory occurs (Spong et al. 2006). Link 1 and link 2 of the robot are considered for the trajectory planning in this work, with the z-axis in the direction of the reader.

4.3.1 via points optimization

It is assumed there are obstacles in the configuration space of the robot that constrain the motion of the robot arm while moving from start to goal point. Figure 4.6 illustrates the path which is planned in the constrained environment, where this path has been planned, using three points cubic spline curve to smooth the corners. These three points, as mentioned above, will be later the via points for the trajectory planning. FPA is used for the optimization process by handling iterative three points, which are responsible for forming the cubic spline curve. FPA main parameters are explained in

Table 4.3, while workspace parameters which are handled to the algorithm are; minimum x-value =0, minimum y-value =0, maximum x-value = 1620, maximum y-value = 1630 with four circular obstacles shown in Figure 4.6. The cubic spline curve, which is governed by the three points, consists of 100 sub-points and starts with (1470,33) and ends with (174,1350), and the fitness function is taken to be the total distances between each of these points.

Table 4.3 Setting of FPA internal parameters

parameter	Value
Number of iterations	100
Number of sub-iterations	10
EF	1000
Population size	25
Maximum temperature	50
Cooling rate	0.9995

$$\text{fitness} = \sum \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} + \text{penalty} \quad (4.4)$$

where i = 1,2,…99 and penalty taken to be 1000 whenever the curve touch an obstacle

Each variable that should be optimized is a structure of 2 points representing the x- and y-axis. It is clear from Figure 4.6 that the FPA algorithm gives good results on path planning. Figure 4.7 shows the cost reduction of the fitness function during iterations. The solution is stable at around 1000 iterations. The three circles on the path in Figure 4.6 represent the final positions of the governing points of the curve (via points).

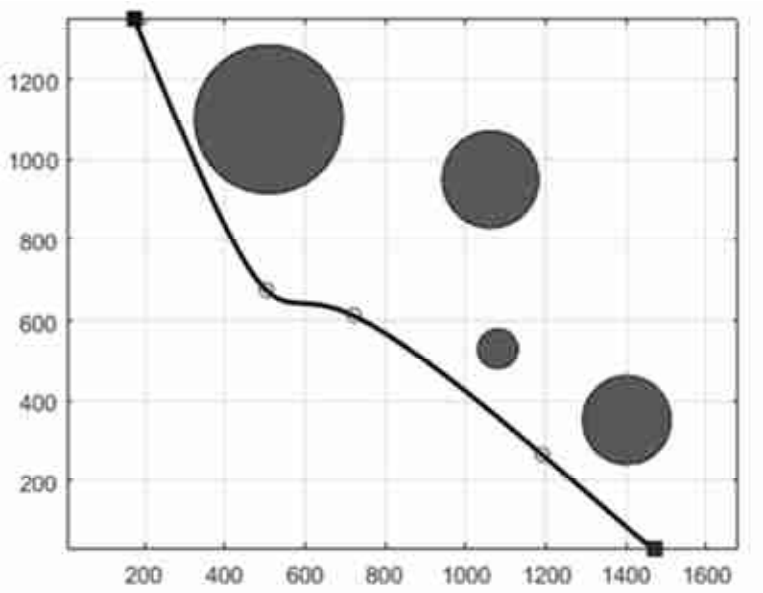

Figure 4.6 Via points on the path

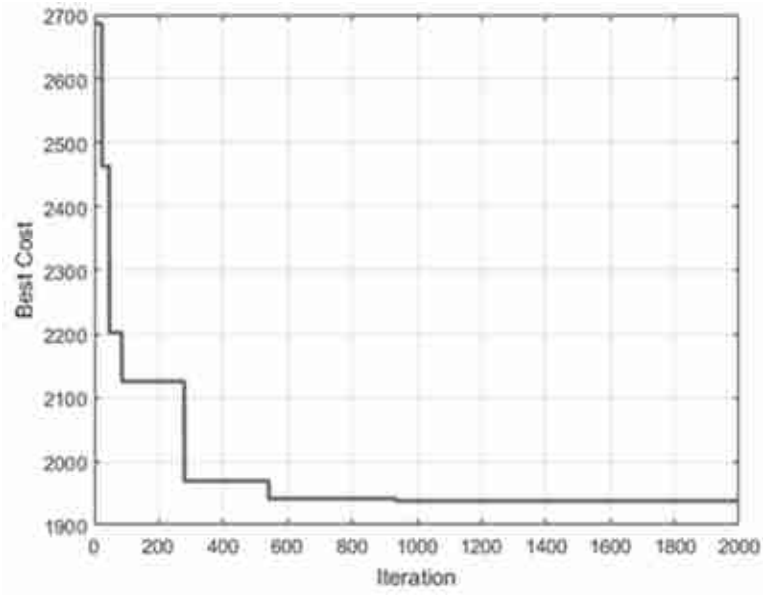

Figure 4.7 Best cost during the optimization process

4.3.2 Mapping to the joint space

It is necessary to map all points in the working space, start, goal, and via points to the joint space by using inverse Kinematics for the sake of the trajectory planning. FPA also has been used to optimize the forward kinematic equations of the two links planar robot with the same parameters mentioned in

Table 4.3. The fitness function of the inverse problem is the distance between the base and the end-effector frame of the robot.

4.3.3 polynomial trajectories

In this section, a polynomial equation of 8^{th} order has been used to plane trajectory between start and goal points during three via points. The equation takes the following form:

$$q(t) = a_1 + a_2t + a_3t^2 + a_4t^3 + a_5t^4 + a_6t^5 + a_7t^6 + a_8t^7 + a_9t^8 \quad (4.5)$$

Where t; is the time variable and a_1 ….a_9 are constants

This equation can be solved by using the following boundary conditions, which represent the inverse form of the Cartesian points shown in Table 4.4, while Figure 4.8 represent the corresponding pose for each point:

For joint variable 1:

$q(0) = 12.7,\ \dot{q}(0) = 0,\ \ddot{q}(0) = 0, q(1) = 107.5\ ,\ \dot{q}(1) = 0,\ \ddot{q}(1) = 0,\ q(0.25) = 48.2\ ,$ $q(0.5) = 91.2\ , q(0.75) = 109$

For joint variable 2:

$q(0) = 22.82\ ,\ \dot{q}(0) = 0,\ \ddot{q}(0) = 0, q(1) = -49.7,\ \dot{q}(1) = 0,\ \ddot{q}(1) = 0,\ q(0.25) = -71.1,$ $q(0.5) = -101.7, q(0.75) = -111.7$

Solving this problem in MATLAB is a standard work, and the result is:

$$q_1(t) = 12.7 + 8383.4t^3 - 41147t^4 + 88190t^6 - 98866t^7 + 56593t^8 - 13058t^9 \quad (4.6)$$

$$q_2(t) = 22.82 - 33651t^3 + 2.0385 * 10^5t^4 - 5.1 * 10^5t^6 + 6.41 * 10^5t^7 - 4 * 10^5t^8 + 98851t^9 \quad (4.7)$$

Deriving equations (4.6) and (4.7) lead to the velocity and acceleration profile for each joint.

Table 4.4 Start and goal and via points in Cartesian and joint space

Point	x-axis mm	y-axis mm	Corresponding joint angles degree	
			q_1	q_2
1	1191	266.9	48.2	-71.1
2	721.4	612.6	91.2	-101.7
3	504.5	674.1	109	-111.7
Start point	1470	33	12.7	22.82
Goal point	174	1350	107.5	-49.7

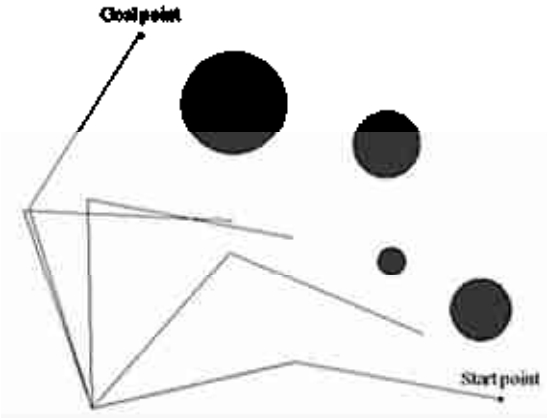

Figure 4.8 Inverse kinematic for each point

Figure 4.9 to Figure 4.14 illustrate the angular position, velocity, and acceleration for the two links during simulation time [0,1] seconds. It is clear from the Figures that the acceleration planned using the 9th order polynomial equation is a continuous function, and that is desired to reduce vibrations during run time.

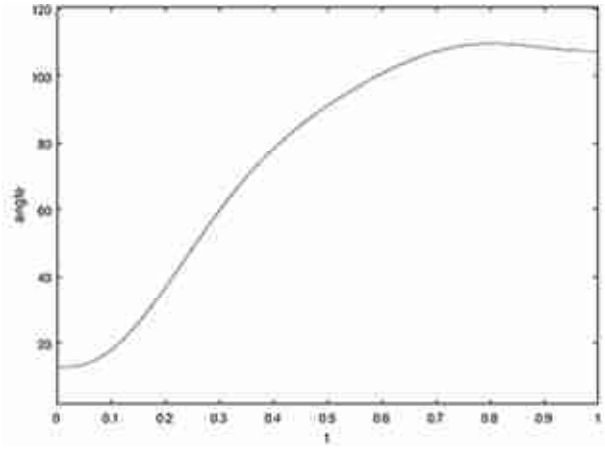

Figure 4.9 Angular position of joint 1

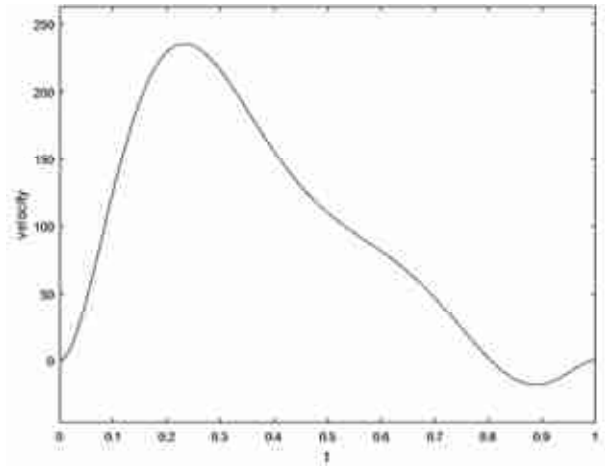

Figure 4.10 Angular velocity of joint 1

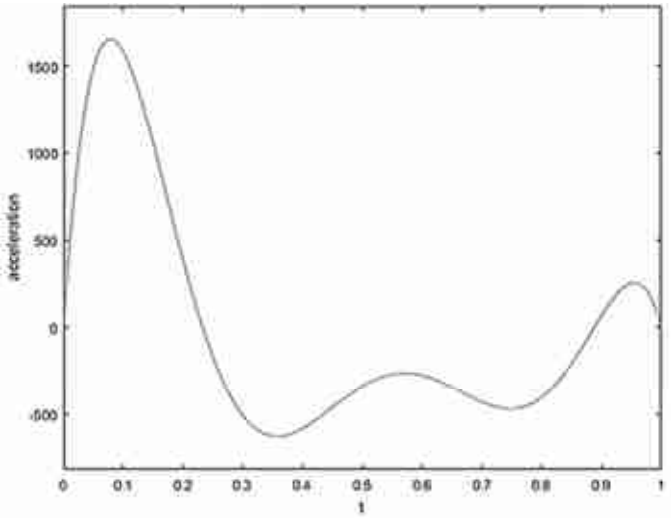

Figure 4.11 Angular acceleration of joint 1

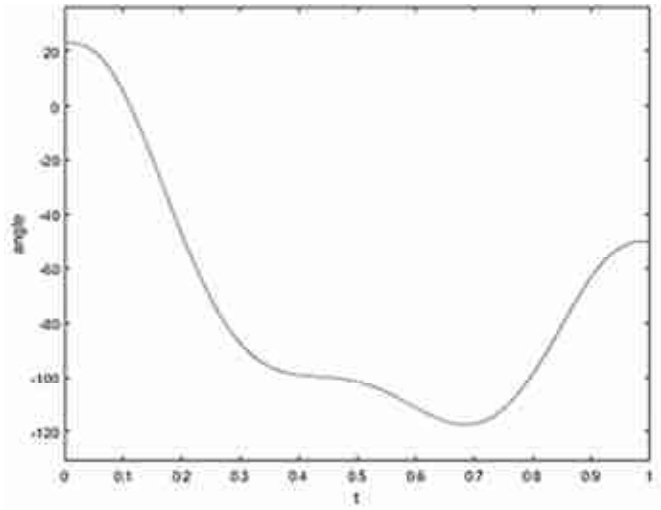

Figure 4.12 Angular position of link 2

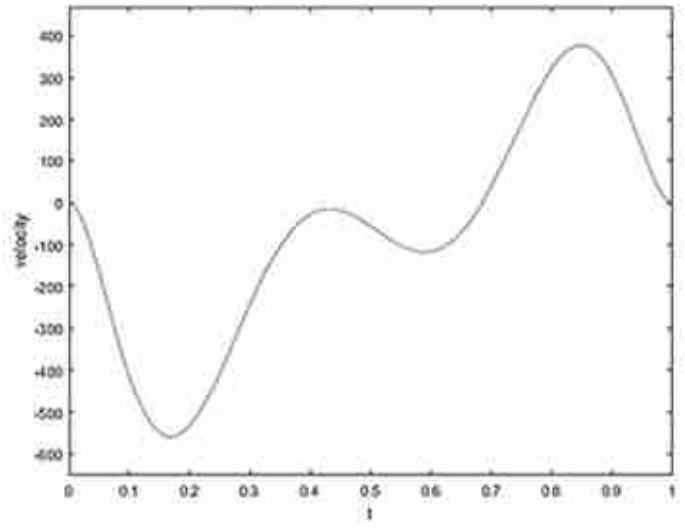

Figure 4.13 Angular velocity of link 2

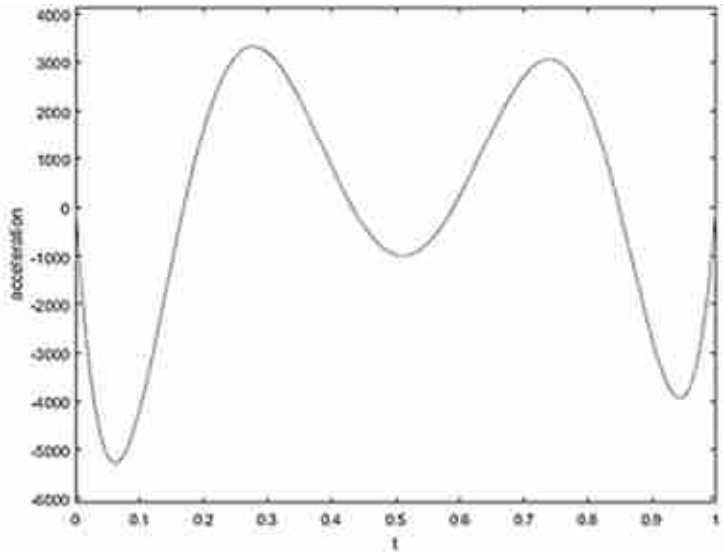

Figure 4.14 Angular acceleration of link 2

4.4 Conclusion

Path planning and trajectory planning are nonlinear problems that can not be solved easily using classical mathematical operations. Metaheuristic algorithms were used in this chapter to find the optimum position for the control points on a cubic spline to form the shortest path between two given points. The performance of the algorithms was compared with the well-known optimization algorithm, and it was found that PSO has overcome all of its competitive algorithms.

5. ROBOTIC PLATFORM: HOST AND DEVICES

5.1 Introduction

In this chapter, full descriptions of the technical issues of the robotic platform will be discussed. The platform mainly consists of two devices connected together by serial communication. The first device is the PC which is the host, while the second device is the target microcontroller which is the STM32 MCU family. The host holds the intelligent system which processes data, visualizes it using a rich graphics engine, and sends the processed data to the target device. Artificial neural networks and optimization algorithms are used to process the data before sending the fine results to the MCU. The XNA game development engine is used to visualize the data and show it on a graphical user interface. The proposed communication protocol between the host and device is chosen to be universal asynchronous receiver transmitter UART. On the other hand, The device, which is, STM32F407G-DISC1 is chosen to control six servo motors that represent the actuators of the 6DOF robot manipulator used in this thesis. In other words, MCU is the in-between device that receives processed data from the host and distributes them to the appropriate actuators. The target device translates the process data given from the host to pulse width modulation (PWM) signals before sending them to the motors. The control units in servos only understand square pluses to rotate their shafts to the desired angles.

5.2 Serial communications

There are different protocols for serial data communications like USB, Ethernet, I2C, RS-485, CAN, RS-232, etc., that differ in specifications and requirements. The proper protocol is chosen according to the application and the application requirements. Table 5.1 reveals some serial communications protocols with their specifications and typical usage. The maximum distance that can be achieved using SPI is 10 feet or less than that, depending on the speed of data on the

bus. The speed on the bus is the speed of the serial clock, and this speed is related to a specific application and power consumption; the higher speed, the higher consumption of power.

Table 5.1 Common serial communication protocols

Protocol	Type	Max distance (ft)	Mx speed (bps)	Typical usage
USB 3.0	Dual simplex serial	9(typical) up to 49 (with 5 hubs)	5G	Mass storage video
USB 2.0	half-duplex serial	16(98 with 5 hubs)	1.5M, 12M, 480M	Keyboard, mouse, drive speaker, printer, camera
Ethernet	serial	1600	10G	Network communicatio ns
I2C	synchronous serial	18	3.4 M in high speed mode	Micocontroller communicatio ns
RS-232	asynchronous serial	50-100	20k	Modem, mouse, instrument
RS-485	asynchronous serial	4000	10M	Data acquisitions and control system
SPI	synchronous serial	10	fPCLK/2	Interfacing peripherals for high-speed requirements like SD card or camera

The speed of the SPI is half of its clock; for example, if the SPI gets 80 MHz of the peripheral clock, then the maximum speed on its bus will be 40 MHz. This protocol works for very high speed but for short distances. To work on longer distances, we have to look for other protocols like I2C, which is commonly used in embedded systems applications. The speed of the I2C is very low compared with SPI, but it works perfectly for longer distances than what SPI achieves. Regardless that I2C is more complex than SPI, designers should select the proper protocol

according to the desired application. Typical applications for SPI are gathering data from high-frequency sensors, communication with the display, communication with flash, etc. For distances more than 18 feet, I2C is no longer being used, and another option is to employ RS-485, which can provide communications over 4000 ft. For relatively small instruments on a single PCB that can have displays, flash, camera, or sensors, then SPI and I2C can be used. On the other hand, for building automation like factories and homes, I2C and SPI cannot be used, and other protocols are preferred like RS-485, CAN, or Ethernet can be used.

5.2.1 UART communication

UART refers to the universal asynchronous receiver transmitter, while USART refers to the universal synchronous, asynchronous receiver transmitter. Each one is just hardware that converts parallel data into serial data. UART supports asynchronous mode, while USART supports both synchronous and asynchronous modes. All modern microcontrollers come with USART to be used in both synchronous and asynchronous modes. Unlike RS-232, SPI, Ethernet, USB, etc., there is no dedicated port for USART communication. USART and UART are commonly employed in conjugation with protocols like SPI, RS-434, USB, etc. The clock is sent on the stream without start and stop bits in the synchronous transmissions. The start and stop bits can be seen in protocols like I2C and SPI and avoiding using them in USART improves the data transmission efficiency by stopping sending extra bits. The hardware of the USART contains the following components:

1. Baudrate generator: It generates the required baud rate for communication.

2. TX and RX shift registers: they are used for transmitting and receiving control blocks, respectively, so they are just Transmit/Receive buffers.

3. First-in, First-out (FIFO) buffer memory: a FIFO is a very efficient feature but very expensive at the same time; for instance, STM32F4xx based microcontroller does not have any FIFO.

For asynchronous mode, the clock will not be sent along with the block of data. Instead, we use synchronization bits like start and will be used along with data. In this thesis, we have used the asynchronous mode of the USART hardware to communicate with the target device, which controls servo motors.

5.2.1.1 UART pins

UART bi-directional (BIDI) communication requires a minimum of two pins, TX, and RX when there is no hardware flow control. UART receives data over RX and transmits data over TX, as shown in Figure 5.1.

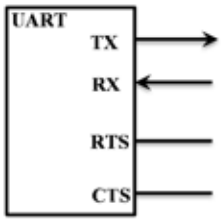

Figure 5.1 UART pins

TX line is held high at the idle state of the line when there are no data to transmit. To receive data on the RX line, the UART peripheral continuously samples the RX line to detect the start bit of the block of data. CTS and RTS are used when a designer wants to use hardware flow control. CTS stands for Clear To Send. When hardware flow control is used, the data transmission on the TX line occurs only if the CTS pin is low. CTS pin should be pulled too low by another device on the stream to enable the data transmissions. RTS stands for Request To Send; the UART on the device uses this line to inform the other device that it needs data. RTS and CTS pins are connected between two devices in a manner like what is illustrated in Figure 5.2. RTS of a device is connected to the CTS of another device if it wants data.

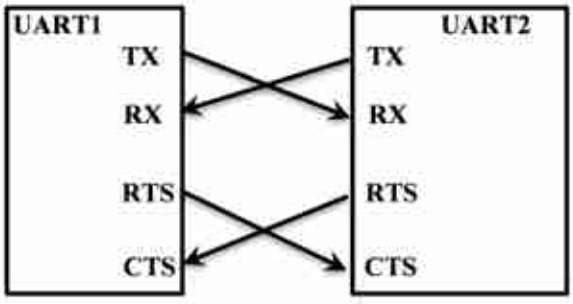

Figure 5.2 UART pins connection between two devices

5.2.1.2 UART frame formats

Frame refers to the whole data packet which is sent or received during data communication. The format of the frame differs from one communication to another. First, UART begins with a start bit that is consistently low for the one-bit duration, then follows data bits, which are 5 to 9 bits, as shown in Figure 5.3 and Figure 5.4. This number can be decided using configuration registers

in the UART peripheral. Parity bit follows the data bits, and it is an optional bit that can be chosen even parity or odd parity mechanism. The frame ends with a stop bit which is always at a high state and can be configured as 1, 1.5, or 2-bit duration.

Figure 5. 3 9-bits word length

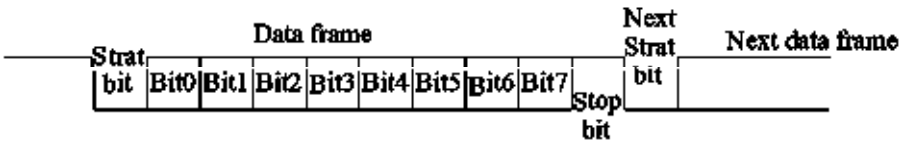

Figure 5.4 8-bits word length

5.2.1.3 UART baud rate

The importance of the baud rate is how fast the data is transmitted on a serial bus; it is expressed in units of bits per second (bps). The inverse of the baud rate is the time duration for a single bit to transmit. This time expresses how long the transmitter holds a serial line high or low. Baud rates can have any value, but the only requirement is that both transmitting and receiving devices should use the same rate. The most common baud rate is 9600, while other standard values are 2400, 4800, 19200, 38400, 57600, 115200. The higher a baud rate, the faster the data is sent or received, but there are some limits to how fast data can be transferred, like the peripheral clock frequency itself.

5.2.1.4 UART synchronization bits

The synchronization bits are two or three special bits transferred with each data frame. They are the start bit, and the stop bits and these bits mark the starting and ending of a packet. Always, there is only one start bit, but the number of stop bits is configurable in the STM32 family to either 1, 1.5, or 2. Stop bit, typically, is 1 for most applications but for high baud rate applications, using two stop bits may be helpful to prevent some issues. The start bit is indicated by a transition of the line state from high to low, while the stop bit transmits to the idle state from low to high.

5.2.1.5 UART parity bits

The simplest form of error detection is using parity bit, which is the number of ones in the packet representing the binary form of a number. For example, the number 55 has this binary form 00110111, so the number of ones is 5, which is an odd number. For this example, when considering odd parity, then the parity bit should be 0, while when considering even parity, the parity bit should be 0. While sending the processed data from the host to the device, the frames may corrupt during transmission.

5.2.1.6 UART block diagram

Figure 5.5 shows the functional diagram of the USART in STM32f4xx micro-controller with details noted by dashed lines, and this Figure is essential to understand what are the necessary required settings. The USART hardware in this thesis is used in asynchronous mode; however, the hardware block in Figure 5.5 is general for both synchronous and asynchronous data transfer. In detail 1, the communication pins are shown, which are TX and RX for transmitting and receiving, respectively; RTS and CTS for hardware flow control. In detail 2, a serial clock will be used when the hardware is in synchronous mode. UART is a full-duplex in communication, and that means that it can transmit and receive data simultaneously where the required two registers for that are shown in detail 3. One is Transmit Data Register, and another one is Receive Data Register, and each one has its own shift register. In detail 4, there are control registers with various control bits to control the transmission block and the receiver block. Detail 5 expresses the baud rate generation block, which generates the desired baud rate for the data flow. The mantissa and fraction fields of the USART_BRR register must be configured accurately to generate the appropriate baud rate.

5.2.1.7 UART peripheral clock

STM32F40xxx family comes with many USART hardware units as well as many timers on the same MCU. This is a powerful feature that enables working on multi-tasks operations on the same device. In this work, USART2 is used to communicate with computer IDE, and this component is associated with the processor through the APB1 bus. That means that the maximum theoretical clock speed that can be achieved on UASRT2 is 71 MHz because the maximum speed on the APB1 bus is 142 MHz, as shown in Figure 5.6. The RX and TX pins on STM32F407G-DISC1 are shown in Figure 5.7.

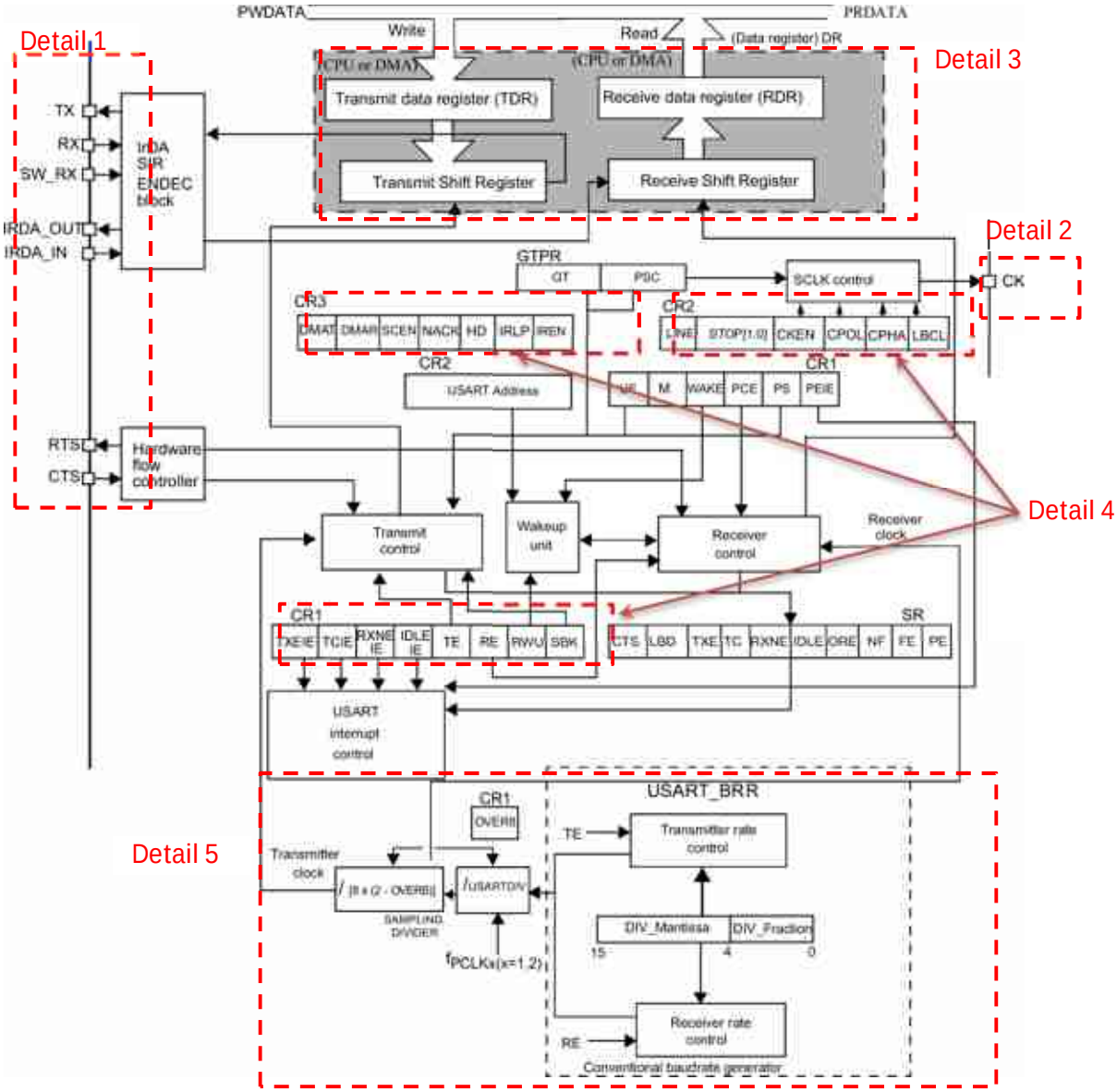

Figure 5. 5 USART hardware

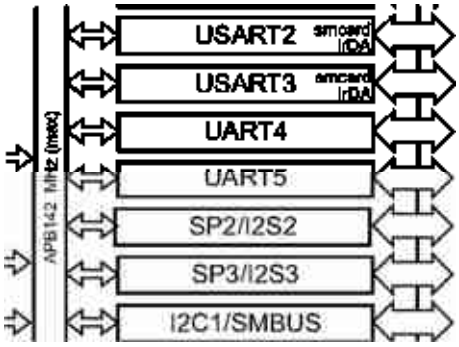

Figure 5. 6 Section from STM32F40xxx block diagram

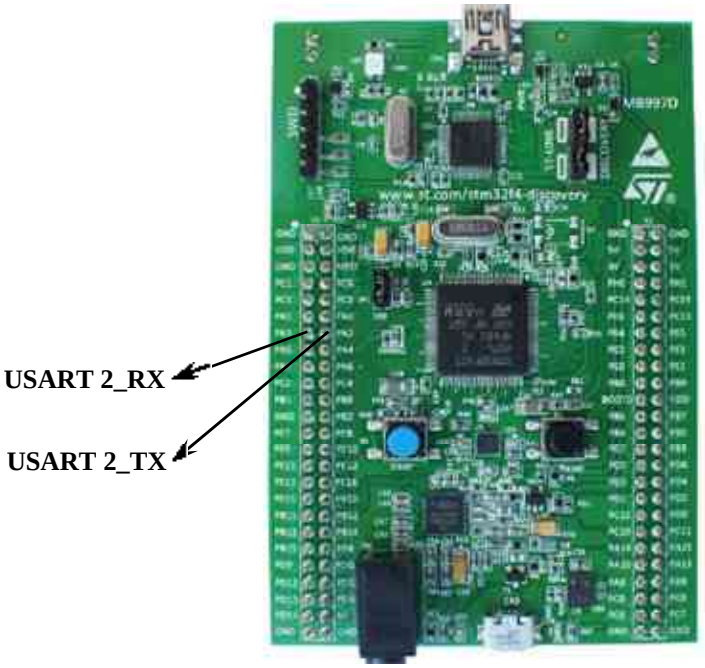

Figure 5.7 USART 2 communication pins on STM32F407G-DISC1

5.2.1.8 UART Transmitter

Back to Figure 5.5, parallel data are converted to serial data sequences by the transmit shift register, which is the heart of the transmitter of the USART hardware. The data comes to the shift register from the transmit data register (TDR), which in turn takes its data by the software. The data will not be loaded to the shift register unless the stop bit has been transmitted from the previous packet.

5.2.1.9 UART Receiver

The heart of the receiver is the receive shift register, where serial data is converted to parallel data. After detection of the stop bit on the RX pin, the received data bits in the shift register will be transferred to the received data register (RDR). The received data register will flush its data to the desired memory location while reading the register by the software. In other words, RDR will be empty after reading while TDR will be empty after acknowledging receiving word from the previous transmission.

5.3 Timers

In this section, timers in STM32 family microcontrollers will be described in detail because the proposed controller of the robot extensively uses this option. The timer is nothing but a

peripheral in the STM32 microcontroller as well as other types of microcontrollers. The timer hardware in the microcontroller can be used for different time-based applications like

1. Generating time-based orders or functionalities such as delay generation.

2. Measuring input signal frequency is like measuring the time period of a waveform.

3. Generating different types of waveforms.

4. Measuring the width of a pulse.

5. Producing pulse width modulation (PWM) signals, and this is very important in our case while actuating servomotors.

6. Triggering external devices like ADC or DAC

Basically, timer peripheral counts from 0 to some pre-programmed value (up counting) or counts from a pre-programmed value to zero (down counting). In other words, due to a specific configuration, timers can up count or down count, and this is the very basic job of the timer. For best understanding, consider Figure 5.8, which shows the functionality of the timer. For each clock pulse, the timer counts until it reaches a pre-programmed value, let say 5 (period), and when the timer reaches this value, it goes down to zero and counts again to the period and so on. The period can be set to the configuration register of the timer, and it is worth mentioning that when the timer goes down to zero after reaching the period, it releases an event that can interrupt the processor of the microcontroller unit. This event is called the update event, and it releases for both up counting and down counting when reaching the reference value of the counting process. The time gap between timing elements (1,2, ….period) is called the time period, and it depends upon the clock frequency of the timer peripheral. The clock frequency of the timer can be varied, and for high frequencies, the time period will be smaller than what is expected at lower frequencies. In other words, the counting speed of the timer depends on the counting clock frequency, which is pre-scaled from the clock frequency of the timer.

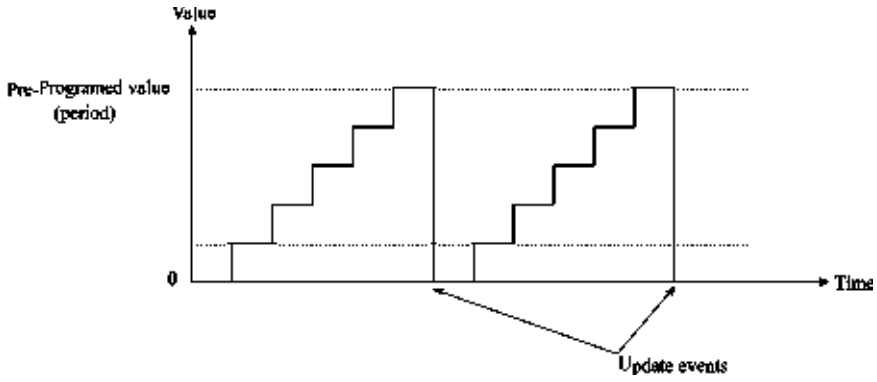

Figure 5.8 Principle of the timer peripheral

5.4 Types of timers

Timers can be classified into three groups:

1. Basic Timers, which are available in almost all STM32 MCUs.

2. General-Purpose Timer is also available in most STM32 MCUs.

3. Advanced Timers, which are available in some STM32 MCUs.

These three types are different in features where advanced timers have more features compared to general-purpose timers and basic timers.

5.5 Time base unit

The basic timers are a simple counting engine, and they are mostly used for time-base generation, and they do not have any input/output channels. The time base unit includes

1. 16 bit up counter

2. Counter register (TIMx-CNT), where x refers to the number of the timer peripheral in MCU, and this register holds the period value.

3. Prescaler register (TIMx-PSC), which controls the speed of counting.

4. Auto reload register (TIMx-ARR), which holds the reload value and makes the timer repeat counting again after reaching the period value.

5.6 Clock configuration of the target device

ARM CORTEX processor comes with a build-in internal clock to operate different peripherals on the microcontroller. The value of the internal clock is 16 MHz which is appropriate for a wide range of applications. In this thesis, an external clock source is used to deliver a 90 MHz frequency to provide fast communication via USART peripheral. Crystal ceramic resonator is used as an external clock source, and choosing the proper frequency value is an optimization problem. The higher the clock speed, the higher the power consumption. In this work, 90 MHz is fast enough for serial communications, and the optimized clock frequency is out of the scope of this proposed platform. Using proper scalars, the clock speed on buses APB1 and APB2 was set to 45 MHz. Timer 1 is connected to the bus APB1, while timer two and three peripherals are connected to bus APB2 (see Figure 5.9).

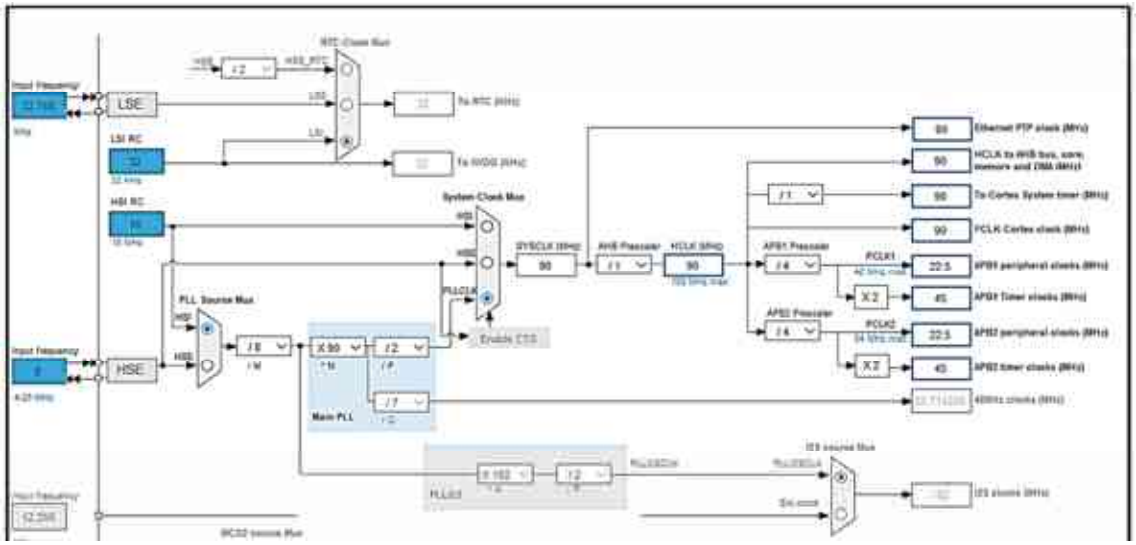

Figure 5.9 Clock configuration of the STM32 device

Consequently, in order to provide 20 ms (50 Hz), the prescale value on buses APB1 and APB2 should be 900 000. According to the clock settings, 25 pulse width corresponds to 1.0 ms, 75 should drive the width of the duty cycle to 1.5 ms, while 125 corresponds to 2.0 ms.

5.7 Pulse width modulation

The shaft of the servo motor rotates relatively to the width of the pulse, which is received to the controller of the motor. The servo motors which are used in the proposed design are working on 50 Hz, and this frequency corresponds to 20 ms width. To control six servos, only three timers are employed with two channels for each timer. In total, six timer channels are responsible for

feeding six servos with control signals. Figure 5.10 shows the logic analysis while testing the performance of the six channels of the three timers; TIM1, TIM2, and TIM3.

Figure 5.10 Pulse width modulation of different sizes

The specifications of the servos indicate that the width of the duty cycle corresponds to a specific angular displacement for the motors shaft over the 20 ms (50Hz). The duty cycle of width 1.5 ms corresponds to 0^o angle, 1.0 ms corresponds to -90^o, and 2 ms corresponds to $+90^o$. The high state of the duty cycle is 4.8 V to 7.2 V, as shown in Figure 5.11.

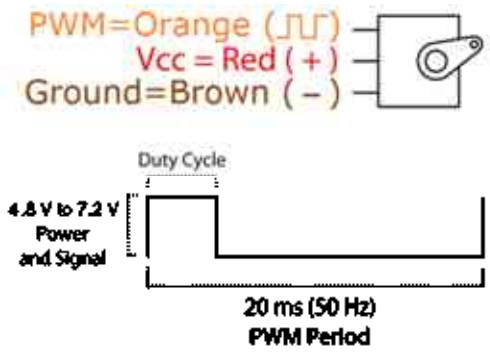

Figure 5.11 Pin connection and electric specifications of the servos

5.8 Firmware

The firmware was burned to the flash of the MCU using STM32CubeIDE through ST-Link connected to the STM32F407G. The firmware is listening to the commands coming from the host machine and setting orders to the timers to deliver proper PWM pulses accordingly. The device and host are connected together by means of UART protocols which are embedded on the

USB cable. In other words, the firmware can be considered as the in-between item between the host machine and the timer-peripherals.

5.9 The system

Figure 5.12 reveals the system of the educational robotic platform which is proposed by this study. The data is transferred in unidirectional mode from the host to the device, where the firmware is settled. The six servos are fed by an external power supply which is AC to DC adapter. The output voltage is 5.0 V with a maximum 30 A current. The voltage supply of the device is chosen to be fed from the USB port of the host. Separating the power supply of the servo motors and target device is necessary because the motors draw high currents and can cause problems to the device board, making it reset or shut down during runtime.

Figure 5. 12 System of the educational robotic platform

5.10 Noise on the system

Jitter is a common problem in servo motors, and some types of motors are very sensitive to noise on the control wire. They are connecting a set of motors on a control board results in a noisy environment. The problem worsens when motors are on load and take more electric current. One solution, which is adopted in this platform, is to employ and connect a capacitor 1000-7400 uF. A capacitor can prevent sudden pulses of current withdrawal from the power supply and damping noise to the minimum. The jitter problem makes the operation of the robot useless since accurate positioning is impossible in this case. Another problem is that jitter loosens the mechanical connections of links and applies fatigue loads on the physical parts. Consequently,

the noise on the control board reduces the lifetime of the joints and links as well as positioning problems.

5.11 3D modelling

Real-life measurements have been taken to the mechanical parts and links of the robot manipulator. Then a 3D model was built for the robot using the CAD package to generate the cad file, which is necessary for the virtual environment of the robot. X-files and FBX files are very common file extensions to be employed in the XNA game engine, which is used in this study.

5.12 XNA game engine

With the advent of high-resolution graphics, high-speed computing, and user interaction devices, virtual reality (VR) has emerged as a major new technology in recent years. An important new concept introduced by many VR systems is immersion, which refers to the feeling of complete immersion in a three-dimensional computer-generated environment by means of user-centered perspective achieved through tracking the user. This is a huge step forward compared to classical modelling and CAD/CAM packages; VR technology is currently used in a broad range of applications, known as flight simulators, walkthroughs, video games, and medicine (virtual surgery).

In this work, a 3D virtual reality application was developed to simulate the real environment that contains the real robot manipulator. The language used is C#.Net 2010 XNA 4.0 Framework. PC specifications are Processor: Intel(R) Core (TM) i5 CPU, Memory: 4GB RAM, and DirectX Version: DirectX 11. XNA was developed for game and graphics programming. It was developed by Microsoft and was started a few years ago. At the GDC (annually biggest Game Developers Conference) in 2004, Microsoft announced XNA for the first time. XNA is not just a framework like DirectX; it also contains a lot of tools and even a custom integrated development environment (IDE) derived from Visual Studio to make the game programmer's life easier. Because no tools or bits were released until 2006, DirectX developers only noticed the "XNA" logo in the upper-right corner of the DirectX SDK documentation from 2004 to 2006. A special XNA Game Studio Professional version targeted at professional game studios was released in the summer of 2007.

5.13 Graphical user interface of the system

Figure 5.13 and Figure 5.14 show the graphical user interface of the software, which is developed to control the robot arm. The 3D model in the virtual environment has the same dimensions in the real world. Both virtual and real robots are moved in synchronization with each other. The angular displacement of the joints drives the robot to move in its virtual world, and these movements are transferred to real-life using the USART communication protocol.

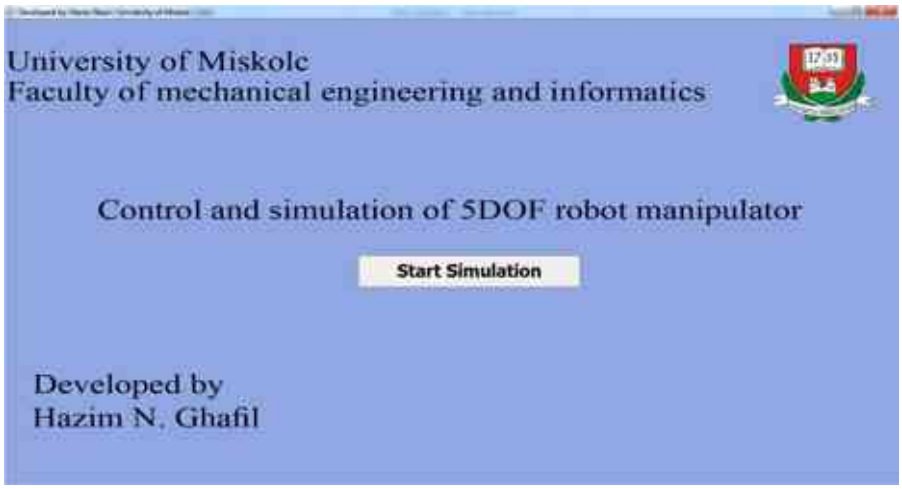

Figure 5.13 System of the educational robotic platform

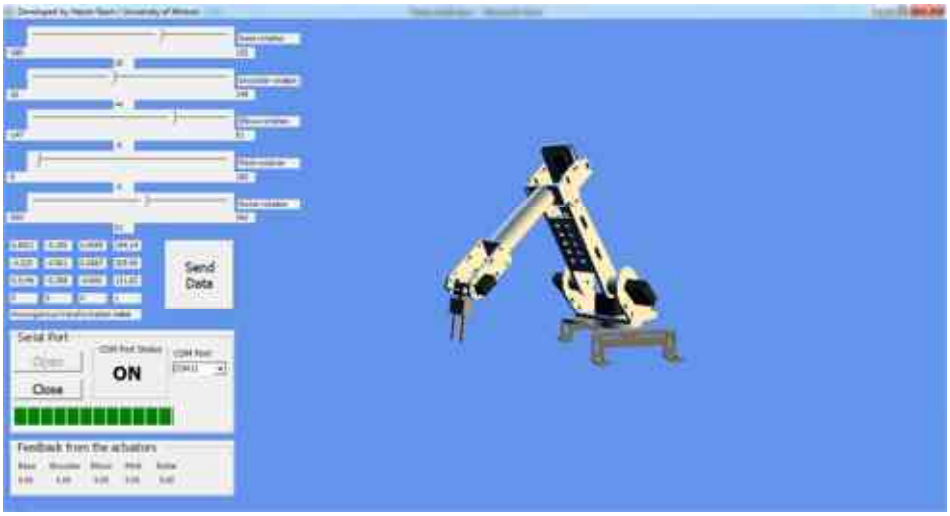

Figure 5.14 System of the educational robotic platform

The software consists of 6 slide bars, each one related to a specific joint variable, and both virtual and real robots share these variables. The software generates the homogenous transformation matrix, which holds the Cartesian position and orientation of the end-effector. The user sets the serial port specifications to handle agreed settings between the host and the

device. Since the wire connections are not so long and the working environment is not so much noisy, parity bits are used as a simple technique for error detection.

5.14 Improved host application

The XNA framework has poor utilities and is hard to use, and even in the last years, Microsoft has dropped this framework from its new releases of .NET frameworks. According to this, the host application for the robotic platform had been revised, and another better host application was built. Unity 3D and C# .NET 2019 were used for the developed new version. Figure 5.15 reveals the improved host application that has to control the physical robot on the real-world side, where it explains the forward kinematic section. The application uses the mechanism described in section 2.5 to return the position and orientation of the end-effector relative to the base of the robot.

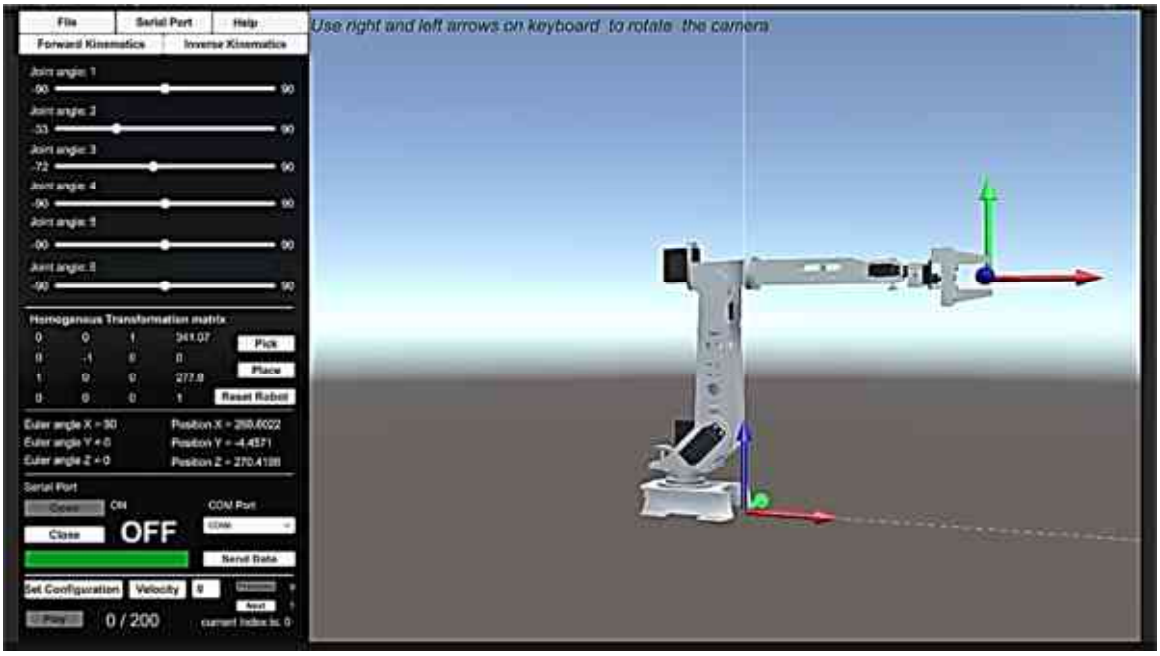

Figure 5.15 Host application written using Unity 3D and C# .NET

Everything in this application was built from scratch without any tools only by using C# libraries and Unity 3D game engine. It is worth saying that Unity 3D is originally founded for video games developments. Thus, our application is a kind of game environment connected to the real world via a serial bus to a microcontroller that controls a group of servo motors. In other words, the proposed robotic platform is a virtual reality application synchronizing the virtual environment on a computer with real-world equipment.

Figure 5.16 shows the inverse kinematic section in the application where the inputs are the position and orientation of the end-effector. At the same time, the outputs are a set of joint angular displacements. Particle swarm optimization takes the inputs and, after several trials, returns the outputs, and the mechanism is described in section 3.2. The application can record different motions and save them and reload them again in a different session. Figure 5.17 illustrates the improved educational robotic platform with a 3D printed arm and Unity 3D-based host application.

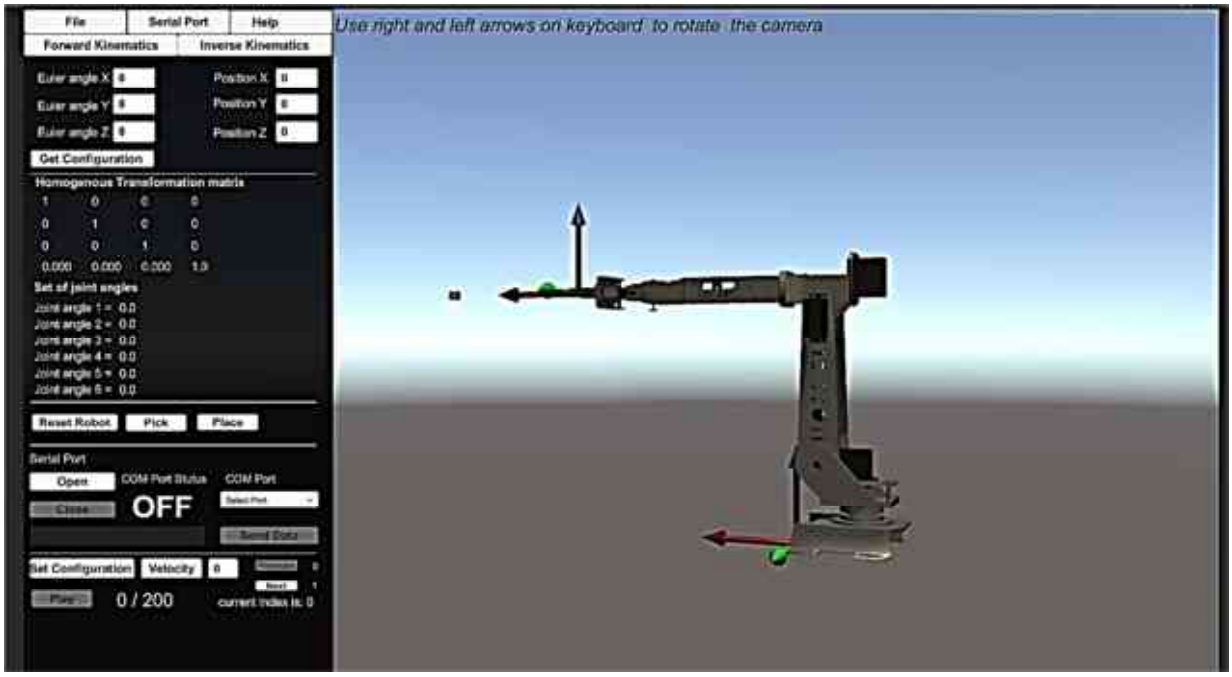

Figure 5.16 Inverse kinematic section

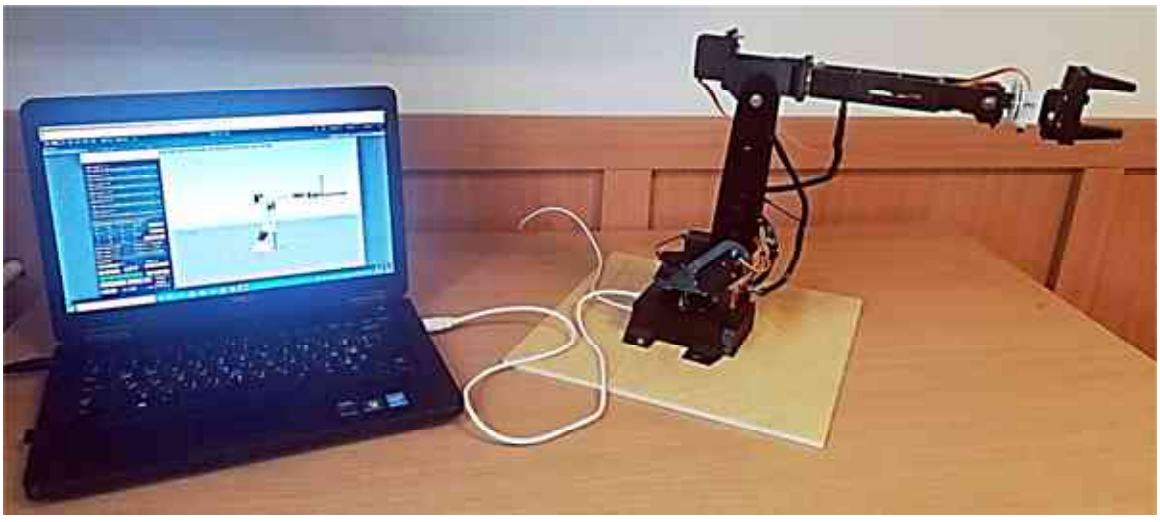

Figure 5.17 Improved educational robotic platform

5.15 Angular displacements to PWMs

The slide bars in the GUI have ranges [-90,90], representing the total range of the servo motors, which is [0,180]. In other meaning, servo motors move physically, between 0 and 180, but for mathematical and user-friendship purposes, we considered the range [-90,90] as an input by the final user. The abovementioned mapping occurs on the host side before sending to the target device, as shown in Figure 5.18.

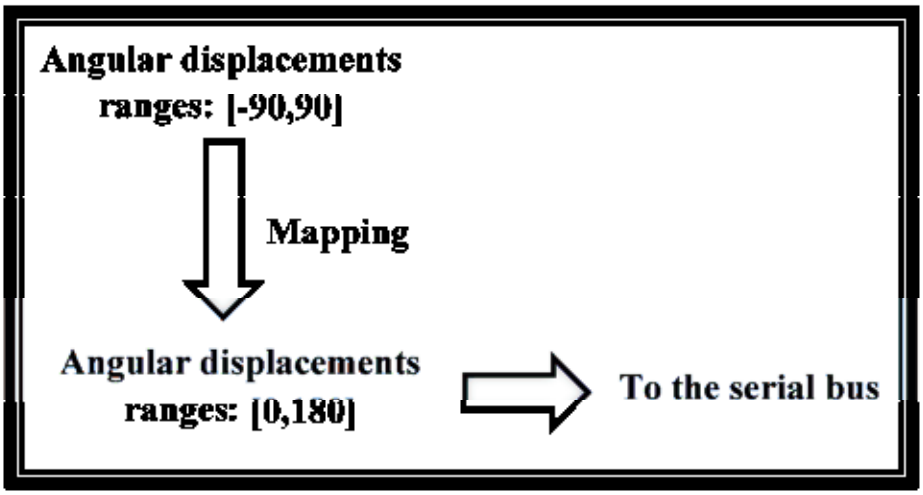

Figure 5.18 Mapping mathematical to actual ranges of the servos

On the other side, the target device, which is STM32 microcontroller, is simultaneously controlling six servo motors by sending frequent pluses on their control line. According to the clock configuration of the microcontroller, the minimum servo position occurs at 1 ms, which is equivalent to 500 on the timer capture/compare register. The maximum servo position for this type of motor occurs by setting the duty cycle to 2 ms, which corresponds to 2400 on the capture/compare register of the timers. In other words, the range of motion of the motors on the target device is [500, 2400].

Thus, a novel equation is developed in this study to map the range of motion from [0^o,180^o] to [500,2400] as follows:

$$D = 500 + dis\frac{160}{15} \tag{5.1}$$

Where D is the angular displacement of the motor on the range [500,2400] while dis is the mapped displacement on [0,180], the error percentage is very low using equation (5.1). For example, 180^o can be transformed to 2420 while it should be 2400.

6. THESES

The findings of this work can be described as follows:

1.A- Building a robust controller for the robot manipulator to control its movements and configurations. The unique controller is consists of two distinguished subsystems, one on the host and another on the target device. The first sub-system, which lies on the host machine, takes user commands as inputs and gives digital values for the joint angles of the virtual model as outputs. The second subsystem, which lies on the microcontroller device, takes the digital values of the joint angles of the virtual model as inputs and gives electrical signals (PWM's) as outputs to the corresponding servo motors. Serial communications via USART play the role of a bridge between the two systems. In this work, the advantages of the ARM-based microcontroller STM32F407 discovery are used by setting six different timers channels; each one corresponds to a servo motor.

1.B- Creating and Developing a virtual reality application using a game engine and C#.NET framework to handle user commands through the graphical user interface. Game engines are widely used to develop video games, but in this work, it has been employed to build a simulation environment for the robot. This application takes the user commands as inputs and sends the outputs to the microcontroller over a bridge of serial communications. The advantage of this application is that it offers a visualized interface to the user to see and decide where to move the robot within a constrained workspace with obstacles. This application is perfect for remote-controlling in a predefined workspace that can be dangerous or poisonous. Users can only deal with a virtual work area that is similar to the real one and control the manipulator remotely.

2. Developing a new artificial neural network to solve the inverse kinematic problem of the robot. ANN consists of a single layer with 100 neurons is developed to take the Cartesian position of the end-effector of the robot as input and return the corresponding joint angles as outputs. This technique gives the controller the ability to give the robot a rapid response with an acceptable error in positioning the end-effector.

3. Developing generic objective function that can be used to solve the inverse kinematic problem using optimization algorithms. There is a variety of met heuristics with different specifications

and abilities to solve optimization problems. In this work, a general-purpose objective function is developed to be optimized by any metaheuristic or heuristic algorithm. The optimization problem is minimizing the error in positioning the end-effector by setting appropriate or optimal joint angles for the manipulator. The developed objective function is differentiable, and the global value is known as zero. Thus, we have found that classical optimizers like hill-climbing are better than other met heuristics algorithms to find optimal joint angles related to a known end-effector position. The inverse kinematic problem in robotics is very complicated and tedious; some robots with 7 DOF or more impossible to find inverse solutions for them using traditional and analytical methods. By developing this objective function, the solution for the inverse kinematic problem becomes very easy and straightforward, even for those robots with a high degree of freedom. However, the optimization method is more time-consuming and accurate than using ANN. Thus, an optimization method was added to the controller in case of slow motion, and accurate movements were required.

7. SUMMARY

Critical synchronization among a set of mechanical, electrical, electronic, and digital components was achieved to design and implement a robotic platform. A six degree of freedom robot arm made of an aluminium sheet of thickness 2 mm was introduced in this work. The robot manipulator is actuated by six digital servo motors; each one has the capability to rotate in the range of [-90^{o},$+90^{o}$]. The target device, which is the hardware of the robot controller, consists of an STM32F407-DISK ARM-based microcontroller. The controller gets its inputs and orders from the host machine via serial communications using the USART protocol. The host, which is a computer, transmits orders to the robot controller from a 3D virtual reality environment mimicking robot workspace in real life. All the above-mentioned hardware and software components are working together to synchronize the movements of a 3D virtual robot in the host machine with the movement of the real-world robot manipulator. Kinematics is included in the controller to move the robot precisely to the desired position and orientation. Forward kinematics is performed straightforwardly by transiting a set of six joint angles of the virtual model to the corresponding actuators on the real robot. Also, inverse kinematics was included in the controller, which is more complicated than forward kinematics. The set of joint angles is generated from the end-effector of the robot in the Cartesian space using two techniques. The first technique is artificial neural networks, while the other one is optimization algorithms. A single layer of 100 neurons was trained to find a set of joint angles from a corresponding Cartesian end-effector position. The option of using ANN for the inverse kinematics is developed to have a quick robot response but with an error percentage more than what is found by optimization algorithms. Solving the inverse kinematics problem using optimization algorithms is more precise than what is expected from ANN but with a longer time duration. Thus, the ANN option is used to have a quick response for the robot in emergency cases, and the optimization algorithms option is used for normal slow motion.

8. ACKNOWLEDGEMENT

I would like to thank my family for their patience during my study at the University of Miskolc; I could not do anything without their support and encouragement over the years. Also, I would like to thank Dr. Károly Jármai for supervising my works and giving me instructions and advice. Special thanks to the University of Miskolc and Tempus Public Foundation, which provide me with the opportunity to complete my Ph.D. through the Stipendium Hungaricum scholarship. Also, many thanks to the University of Kufa in Iraq for their extra financial supports. The research was supported by the Hungarian National Research, Development and Innovation Office under the project number K 134358.

To my wife and children ... THANK YOU VERY MUCH!

9. APPLICATION POSSIBILITY

The proposed work has the following possible applications

1. In section 1.5, a survey was performed, and most of the participants recommended developing a new educational robotic platform that can be targeted to students of different ages. Thus, the proposed platform is targeted mainly to students for education purposes

2. Some robot manipulators are required to operate in toxic or radiant environments that can even be dangerous to be tracked by a human. Thus the proposed platform with its virtual reality application is perfect for controlling robots in a well-predefined workspace.

3. With more focus on rigidity and using high-strength materials for manufacturing the arm and choosing proper motors specifications, the proposed platform can be used for industrial applications. Robots have wide applications in the industry. Therefore, the possible applications of the platform are very large accordingly.

REFERENCES

(IFR) IfftIFoR (2018) Avialable online:. https://ifr.org/ifr-press-releases/news/world-robotics-report-2016. Accessed March 31, 2019

Abbas TF (2013) Forward Kinematics Modeling of 5 DOF Stationary Articulated Robots. Engineering and Technology Journal 31 (3 Part (A) Engineering):500-512

Al-Arif SMR, Ferdous AI, Nijami SH (2012) Comparative study of different path planning algorithms: a water based rescue system. International Journal of Computer Applications 39

Al-Azzawi DZ (1996) Trajectory generation and tracking control of an arc welding industrial robot. M. Sc. thesis, Nahrain University,

Alsamia S, Ibrahim DS, Ghafil HN (2021) Optimization of drilling performance using various metaheuristics. Pollack Periodica

Alsarayefi S, Gafil HN, Jálics K (2019) Optimisation of the fibre size for a fiber glass epoxy composite. Design of Machines and Structures 9 (1):5-12

Association RI (2018) available online:. https://www.robotics.org/content-detail.cfm/Industrial-Robotics-News/2016-Breaks-Records-for-North-American-Robot-Orders-and-Shipments/content_id/6378. Accessed March 31, 2019

Bazerghi A, Goldenberg AA, Apkarian J (1984) An exact kinematic model of PUMA 600 manipulator. IEEE transactions on systems, man, and cybernetics (3):483-487

Behforooz GH (1995) A comparison of theE (3) and not-a-knot cubic splines. Applied Mathematics and Computation 72 (2-3):219-223

Bingul Z, Ertunc H, Oysu C (2005) Applying neural network to inverse kinematic problem for 6R robot manipulator with offset wrist. In: Adaptive and Natural Computing Algorithms. Springer, pp 112-115

Brooks SH (1958) A discussion of random methods for seeking maxima. Operations research 6 (2):244-251

Buckley KA, Hopkins SH, Turton BC (1997) Solution of inverse kinematics problems of a highly kinematically redundant manipulator using genetic algorithms.

Chang P (1987) A closed-form solution for inverse kinematics of robot manipulators with redundancy. IEEE Journal on Robotics and Automation 3 (5):393-403

Chapelle F, Bidaud P (2004) Closed form solutions for inverse kinematics approximation of general 6R manipulators. Mechanism and machine theory 39 (3):323-338

Choi BB, Lawrence C (1992) Inverse kinematics problem in robotics using neural networks.

Craig JJ (2009) Introduction to robotics: mechanics and control, 3/E. Pearson Education India,

Daneshmand M, Bilici O, Bolotnikova A, Anbarjafari G (2017) Medical robots with potential applications in participatory and opportunistic remote sensing: A review. Robotics and Autonomous Systems 95:160-180

Dorigo M (1992) Optimization, learning and natural algorithms. PhD Thesis, Politecnico di Milano

Dorigo M, Gambardella LM (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on evolutionary computation 1 (1):53-66

Drexler DA Solution of the closed-loop inverse kinematics algorithm using the Crank-Nicolson method. In: 2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI), 2016. IEEE, pp 351-356

Duka A-V (2014) Neural network based inverse kinematics solution for trajectory tracking of a robotic arm. Procedia Technology 12:20-27

Eglese R (1990) Simulated annealing: a tool for operational research. European journal of operational research 46 (3):271-281

Farkas, József, and Károly Jármai. Analysis and optimum design of metal structures, Balkema, 1997.

Farkas, József, and Károly Jármai. Economic design of metal structures, Millpress, 2003.

References

Farkas, József, and Károly Jármai. Design and optimization of metal structures. Horwood, 2008.

Farkas, J., Jármai, K. and Snyman, J.A., 2010. Global minimum cost design of a welded square stiffened plate supported at four corners. Structural and Multidisciplinary Optimization, 40(1-6), p.477.

Farkas, József, and Károly Jármai. Optimum design of steel structures. Berlin: Springer Verlag, 2013.

Jármai, Károly, and József Farkas. "Minimum cost design of a rectangular box column composed from cellular plates with welded T-stiffeners." Design, Fabrication and Economy of Metal Structures. Springer, Berlin, Heidelberg, 2013. 29-36.

Farkas, József, and Károly Jármai. Innovative design of metal structures. Gazdász Kiadó és Nyomda, 2015. (in Hungarian)

Ferguson D, Likhachev M, Stentz A A guide to heuristic-based path planning. In: Proceedings of the international workshop on planning under uncertainty for autonomous systems, international conference on automated planning and scheduling (ICAPS), 2005. pp 9-18

Gasparetto A, Zanotto V (2010) Optimal trajectory planning for industrial robots. Advances in Engineering Software 41 (4):548-556

Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. simulation 76 (2):60-68

Ghafil H, Jármai K (2018a) Comparative study of particle swarm optimization and artificial bee colony algorithms.

Ghafil H, Jármai K Optimum dynamic analysis of a robot arm using flower pollination algorithm. In: Advances and Trends in Engineering Sciences and Technologies III: Proceedings of the 3rd International Conference on Engineering Sciences and Technologies (ESaT 2018), September 12-14, 2018, High Tatras Mountains, Tatranské Matliare, Slovak Republic, 2019a. CRC Press, p 81

GHAFIL HN (2013) OPTIMUM PATH PLANNING AND PERFORMANCE ANALYSIS OF A ROBOT MANIPULATOR. Al Nahrain University

Ghafil HN (2016) Inverse acceleration solution for robot manipulators using harmony search algorithm. International Journal of Computer Applications 144 (6):1-7

Ghafil HN, Jármai K Research and Application of Industrial Robot Manipulators in Vehicle and Automotive Engineering, a Survey. In: Vehicle and Automotive Engineering, 2018b. Springer, pp 611-623

Ghafil HN, Jármai K (2018c) Trajectory planning of a robot arm using flower pollination algorithm.

Ghafil HN, Jármai K (2019b) Kinematic-based structural optimization of robots. Pollack Periodica 14 (3):213-222

Ghafil HN, Jármai K On performance of ED7500 and lab-volt 5150 5DOF manipulators: calibration and analysis. In: MultiScience—XXXIII. MicroCAD international multidisciplinary scientific conference University of Miskolc, 2019c. pp 23-24

Ghafil HN, Jármai K (2020a) Dynamic differential annealed optimization: New metaheuristic optimization algorithm for engineering applications. Applied Soft Computing 93:106392

Ghafil HN, Jármai K Optimization Algorithms for Inverse Kinematics of Robots with MATLAB Source Code. In: Vehicle and Automotive Engineering, 2020b. Springer, pp 468-477

Ghafil HN, Jármai K (2020c) Optimization for Robot Modelling with MATLAB. Springer,

Ghafil HN, László K, Jármai K Investigating three learning algorithms of a neural networks during inverse kinematics of robots. In: Solutions for Sustainable Development: Proceedings of the 1st International Conference on Engineering Solutions for Sustainable Development (ICESSD 2019), October 3-4, 2019, Miskolc, Hungary, 2019. CRC Press, p 33

Ghafil HN, Mohammed AH, Hadi NH (2015) A virtual reality environment for 5-DOF robot manipulator based on XNA framework. International Journal of Computer Applications 113 (3)

Goldenberg A, Benhabib B, Fenton R (1985) A complete generalized solution to the inverse kinematics of robots. IEEE Journal on Robotics and Automation 1 (1):14-20

Gopinath V, Johansen K (2016) Risk assessment process for collaborative assembly–a job safety analysis approach. Procedia CIRP 44:199-203

References

Gopinath V, Ore F, Johansen K (2017) Safe assembly cell layout through risk assessment–an application with hand guided industrial robot. Procedia CIRP 63:430-435

Grudic GZ, Lawrence PD (1993) Iterative inverse kinematics with manipulator configuration control. IEEE transactions on robotics and Automation 9 (4):476-483

Hart PE, Nilsson NJ, Raphael B (1968) A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics 4 (2):100-107

Hassan A, Abomoharam M (2017) Modeling and design optimization of a robot gripper mechanism. Robotics and Computer-Integrated Manufacturing 46:94-103

Hayawi MJ (2011) Analytical inverse kinematics algorithm of a 5-DOF robot arm. Journal of College of Education for Pure Science 1 (4):92-104

Holland J (1975) Adaption in Nature and Artificial Systems, Michigan. The University of Michigan Press,

Iliukhin V, Mitkovskii K, Bizyanova D, Akopyan A (2017) The modeling of inverse kinematics for 5 DOF manipulator. Procedia Engineering 176:498-505

Isobe T, Nagasaka K, Yamamoto S (1992) A new approach to kinematic control of simple manipulators. IEEE transactions on systems, man, and cybernetics 22 (5):1116-1124

Kawamoto A, Bendsøe MP, Sigmund O (2004) Planar articulated mechanism design by graph theoretical enumeration. Structural and Multidisciplinary Optimization 27 (4):295-299

Kennedy J (2011) Particle swarm optimization. In: Encyclopedia of machine learning. Springer, pp 760-766

Kivelä T, Mattila J, Puura J (2017) A generic method to optimize a redundant serial robotic manipulator's structure. Automation in Construction 81:172-179

Kong X, Gosselin C (2007) Type Synthesis of Single-Loop Kinematic Chains. Type Synthesis of Parallel Mechanisms:43-53

Kota L, Jarmai K (2014) Efficient algorithms for optimization of objects and systems. Pollack Periodica 9 (1):121-132

Kramberger A, Gams A, Nemec B, Chrysostomou D, Madsen O, Ude A (2017) Generalization of orientation trajectories and force-torque profiles for robotic assembly. Robotics and Autonomous Systems 98:333-346

Krishnaswamy K, Sleeman J, Oates T Real-time path planning for a robotic arm. In: Proceedings of the 4th International Conference on Pervasive Technologies Related to Assistive Environments, 2011. ACM, p 11

Kumar RR, Chand P Inverse kinematics solution for trajectory tracking using artificial neural networks for SCORBOT ER-4u. In: 2015 6th International Conference on Automation, Robotics and Applications (ICARA), 2015. IEEE, pp 364-369

Kumar V, Gardner JF (1990) Kinematics of redundantly actuated closed chains. IEEE Transactions on Robotics and automation 6 (2):269-274

Lee C, Ziegler M (1984) Geometric approach in solving inverse kinematics of PUMA robots. IEEE Transactions on Aerospace and Electronic Systems (6):695-706

Li B, Shao Z (2015) Simultaneous dynamic optimization: A trajectory planning method for nonholonomic car-like robots. Advances in Engineering Software 87:30-42

Liu Y, McPhee J (2007) Automated kinematic synthesis of planar mechanisms with revolute joints. Mechanics Based Design of Structures and Machines 35 (4):405-445

Liu Y, Passino K (2002) Biomimicry of social foraging bacteria for distributed optimization: models, principles, and emergent behaviors. Journal of optimization theory and applications 115 (3):603-628

Lu B-L, Ito K Regularization of inverse kinematics for redundant manipulators using neural network inversions. In: Proceedings of ICNN'95-International Conference on Neural Networks, 1995. IEEE, pp 2726-2731

Mahmood BAD, Ghafil H, Jármai K (2019) Optimizing Heuristic Graph Formation with Application in Kinematic Synthesis of a Robot Arm with Revolute Joints. Journal of Engineering and Applied Sciences 14 (6):1976-1948

References

Makris S, Tsarouchi P, Matthaiakis A-S, Athanasatos A, Chatzigeorgiou X, Stefos M, Giavridis K, Aivaliotis S (2017) Dual arm robot in cooperation with humans for flexible assembly. CIRP Annals 66 (1):13-16

Mathew R, Hiremath SS (2016) Trajectory tracking and control of differential drive robot for predefined regular geometrical path. Procedia Technology 25:1273-1280

Mathworks (2019) Documentation https://ch.mathworks.com/help/matlab/ref/spline.html. Accessed April,16,2019

Mehrabian AR, Lucas C (2006) A novel numerical optimization algorithm inspired from weed colonization. Ecological informatics 1 (4):355-366

Michalos G, Makris S, Spiliotopoulos J, Misios I, Tsarouchi P, Chryssolouris G (2014) ROBO-PARTNER: Seamless human-robot cooperation for intelligent, flexible and safe operations in the assembly factories of the future. Procedia CIRP 23:71-76

Moscato P (1989) On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Caltech concurrent computation program, C3P Report 826:1989

Nasir Ghafil H, Alsamia S, Jármai K (2021) Fertilization optimization algorithm on CEC2015 and large scale problems. Pollack Periodica

Nelder JA, Mead R (1965) A simplex method for function minimization. The computer journal 7 (4):308-313

Novakovic ZR, Nemec B (1990) A solution of the inverse kinematics problem using the sliding mode. IEEE Transactions on Robotics and Automation 6 (2):247-252

Oliva JC, Goodman ED (2010) Simultaneous Type and Dimensional Synthesis of Planar 1DOF Mechanisms Using Evolutionary Search and Convertible Agents (DETC2009-86722). Journal of Mechanisms and Robotics 2 (3):031001

Panich S (2010) The shortest path with intelligent algorithm. Journal of Mathematics and Statistics 6 (3):276-278

Pham DT, Ghanbarzadeh A, Koç E, Otri S, Rahim S, Zaidi M (2006) The bees algorithm—a novel tool for complex optimisation problems. In: Intelligent Production Machines and Systems. Elsevier, pp 454-459

Pozna CR, Horváth E, Hollósi J The inverse kinematics problem, a heuristical approach. In: 2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI), 2016. IEEE, pp 299-304

Pucheta MA, Cardona A (2013) Topological and dimensional synthesis of planar linkages for multiple kinematic tasks. Multibody System Dynamics 29 (2):189-211

report Wr (2018) Avialable online:. https://ifr.org/downloads/press/02_2016/2016FEB_PI__IFR_Roboterdichte_nach_Regionen_QS1.pdf, last accessed 2018/02/29. Accessed March 31, 2019

Reynolds RG An introduction to cultural algorithms. In: Proceedings of the third annual conference on evolutionary programming, 1994. World Scientific, pp 131-139

Robots ESWRI (2018) Avialable online: .

Rubinstein RY (1997) Optimization of computer simulation models with rare events. European Journal of Operational Research 99 (1):89-112

Shabeeb AH (2013) Path Planning of Robot Manipulator Using Bezier Technique. M. Sc, thesis, Department of Production Engineering and Metallurgy …,

Socha K, Dorigo M (2008) Ant colony optimization for continuous domains. European journal of operational research 185 (3):1155-1173

Spolaôr N, Benitti FBV (2017) Robotics applications grounded in learning theories on tertiary education: A systematic review. Computers & Education 112:97-107

Spong MW (2005) Seth. Hutchinson, and M. Vidyasagar. Robot Modeling and Control

Spong MW, Hutchinson S, Vidyasagar M (2006) Robot modeling and control.

Storn R (1995) Differrential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, International Computer Science Institute 11

References

Tejomurtula S, Kak S (1999) Inverse kinematics in robotics using neural networks. Information Sciences 116 (2-4):147-164

Tomić M, Miloradović B, Janković M Connectionist-genetic based algorithm for positioning industrial manipulator. In: 11th Symposium on Neural Network Applications in Electrical Engineering, 2012. IEEE, pp 59-64

Tourassis VD, Neuman CP (1985) Inverse dynamics applications of discrete robot models. IEEE transactions on systems, man, and cybernetics (6):798-803

Tromme E, Tortorelli D, Brüls O, Duysinx P (2015) Structural optimization of multibody system components described using level set techniques. Structural and Multidisciplinary Optimization 52 (5):959-971

Tuncer A, Yildirim M (2012) Dynamic path planning of mobile robots with improved genetic algorithm. Computers & Electrical Engineering 38 (6):1564-1572

Wang H, Chen W, Lai Y, He T (2015) Trajectory planning of tokamak flexible in-vessel inspection robot. Fusion Engineering and Design 98:1678-1682

Wang L-C, Chen C-C (1991) A combined optimization method for solving the inverse kinematics problems of mechanical manipulators. IEEE Transactions on Robotics and Automation 7 (4):489-499

Wu C-H, Young K-Y (1990) An efficient solution of a differential inverse kinematics problem for wrist-partitioned robots. IEEE Transactions on Robotics and Automation 6 (1):117-123

Xu D, Calderon CAA, Gan JQ, Hu H, Tan M (2005) An analysis of the inverse kinematics for a 5-DOF manipulator. International Journal of Automation and Computing 2 (2):114-124

Yang X-S Firefly algorithms for multimodal optimization. In: International symposium on stochastic algorithms, 2009. Springer, pp 169-178

Yang X-S (2010) A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization (NICSO 2010). Springer, pp 65-74

Yang X-S Flower pollination algorithm for global optimization. In: International conference on unconventional computing and natural computation, 2012. Springer, pp 240-249

Yang X-S, Deb S Cuckoo search via Lévy flights. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), 2009. IEEE, pp 210-214

Zhang Y-G, Huang Y-M, Lin Y-Z, Cheng X, Gao F An approach for robot inverse velocity solution using genetic algorithm. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX826), 2004. IEEE, pp 2944-2948

Zhang Y-G, Huang Y-M, Xie L-M Robot inverse acceleration solution based on hybrid genetic algorithm. In: 2008 International Conference on Machine Learning and Cybernetics, 2008. IEEE, pp 2099-2103

Ziegler M (1983) A geometric approach in solving the inverse kinematics of PUMA robot. Department of electrical and computer engineering-The university of Michigan

LIST OF PUBLICATIONS RELATED TO THE TOPIC OF THE RESEARCH FIELD

1. Hazim, Nasir Ghafil ; Jármai, Károly Optimáló algoritmusok robotok inverz kinematikájához MATLAB forráskóddal GÉP 72 : 1-2 pp. 71-74. , 4 p. (2021)
2. Hazim, Nasir Ghafil ; Karoly, Jarmai Optimization Algorithms for Inverse Kinematics of Robots with MATLAB Source Code LECTURE NOTES IN MECHANICAL ENGINEERING 22 pp. 468-477. , 10 p. (2021)
3. Ghafil, Hazim Nasir ; Jármai, Károly Dynamic differential annealed optimization: A new metaheuristic optimization algorithm for engineering applications APPLIED SOFT COMPUTING 93 Paper: 106392 (2020)
4. Hazim, Nasir Ghafil ; Jármai, Károly IDEGHÁLÓK ROBOTOK INVERZ KINEMATIKAI VIZSGÁLATÁHOZ: ÁTTEKINTÉS ÉS ALKALMAZÁS GÉP 71 : 2 pp. 5-10. , 6 p. (2020)
5. Hazim, Nasir Ghafil ; Károly, Jármai Optimization for Robot Modeling with MATLAB Basel, Switzerland: Springer Nature Switzerland AG (2020) , 220 p. DOI ISBN: 9783030404093 ISBN: 3030404099 ISBN: 9783030404109
6. Hazim, Nasir GHAFIL ; S., ALSAMIA ; Károly, JÁRMAI Novel Levy flight stochastic optimization algorithm for various engineering problems In: Iványi, Péter Abstract book for the 16th MIKLÓS IVÁNYI INTERNATIONAL PHD & DLA SYMPOSIUM (2020) p. 124, 1 p.
7. Alsarayefi, Saad ; Jálics, Károly ; Ghafil, Hazim Nasir Optimization of the fiber size for a fiber glass epoxy composite DESIGN OF MACHINES AND STRUCTURES 9 : 1 pp. 5-12. , 8 p. (2019) REAL Complete document
8. Ghafil, Hazim Nasir ; Jármai, Károly: Kinematic-based structural optimization of robots POLLACK PERIODICA: AN INTERNATIONAL JOURNAL FOR ENGINEERING AND INFORMATION SCIENCES 14 : 3 pp. 213-222. , 10 p. (2019)
9. Hazim, Nasir ; Jármai, Károly A részecske csoport és a mesterséges méhcsalád módszerek összehasonlító vizsgálata GÉP 70 : 2 pp. 3-8. , 6 p. (2019).
10. Hazim, Nasir Ghafil ; Kovács, László ; Károly, Jármai: Investigating three learning algorithms of a neural networks during inverse kinematics of robots In: Szita Tóthné, Klára; Jármai, Károly; Voith, Katalin (ed.) Solutions for Sustainable Development: Proceedings of the 1st International Conference on Engineering Solutions for Sustainable Development, (ICESSD 2019) London, United Kingdom / England: CRC Press (2019) 394 p. pp. 33-40. , 8 p.
11. Hazim, Nasir Ghafil ; Károly, Jármai: A novel hybrid algorithm with application in the kinematic-based structural optimization of robots In: Proceedings of the 13th World Congress on Structural and Multidisciplinary Optimization (WCSMO 13) International

Society for Structural and Multidisciplinary Optimization (ISSMO) , (2019) pp. 206-206. , 1 p.

12. Hazim, Nasir Ghafil ; Károly, Jármai: On Performance of ED7500 and Lab-Volt 5150 5DOF Manipulator: Calibration and Analysis In: Kékesi, Tamás (ed.) MultiScience - XXXIII. microCAD International Multidisciplinary Scientific Conference University of Miskolc, Hungary: University of Miskolc (2019) pp. 1-9. Paper: D2-4, 9 p.
13. HN, Ghafil ; K., Jármai: Optimum dynamic analysis of a robot arm using flower pollination algorithm In: Peter, Platko; Mohamad, Al Ali (eds.) Advances and Trends in Engineering Sciences and Technologies III. London, UK / England: CRC Press - Taylor and Francis Group , (2019) pp. 81-88. , 8 p.
14. Mahmood Badr Ali, Doaa ; Nasir Ghafil, Hazim ; Károly, Jármai: Optimizing Heuristic Graph Formation with Application in Kinematic Synthesis of a Robot Arm with Revolute Joints JOURNAL OF ENGINEERING AND APPLIED SCIENCES 14 : 6 pp. 1976-1948. Paper: 14 (2019)
15. Ghafil, Hazim ; Jármai, Károly: Optimal stress-strain analysis of a robot arm through finite-element and simulated annealing algorithm In: Fülöp, Attila; Iványi, Péter (eds.) Abstract book for the 14th Miklós Iványi International PhD & DLA Symposium: Architectural, Engineering and Information Sciences Pécs, Hungary: Pollack Press , (2018) p. 70 Paper: 26
16. Ghafil, Hazim ; Jármai, Károly: Kinematic based structural optimization of robots In: Fülöp, Attila; Iványi, Péter (eds.) Abstract book for the 14th Miklós Iványi International PhD & DLA Symposium: Architectural, Engineering and Information Sciences Pécs, Hungary: Pollack Press , (2018) p. 69 Paper: 25
17. Ghafil, Hazim ; Shaymaa, Mahmood Badr: MATLAB FOR ENGINEERS- BASICS AND APPLICATIONS WORLD JOURNAL OF ENGINEERING RESEARCH AND TECHNOLOGY 4: 1 p. 1 (2018).
18. Ghafil, HN ; Jármai, K: Research and application of industrial robot manipulators in vehicle and automotive engineering, a survey, LECTURE NOTES IN MECHANICAL ENGINEERING, pp. 611-623. Springer International Publishing AG, ISBN 978-3-319-75677-6 (eBook) https://doi.org/10.1007/978-3-319-75677-6.
19. Ghafil, HN; Jármai, K: Comparative study of particle swarm optimization and artificial bee colony algorithms In: Kékesi, Tamás (ed.) Multiscience XXXII. MicroCAD International Multidisciplinary Scientific Conference University of Miskolc, Hungary: University of Miskolc (2018) Paper: D1_Hazim_Nasir_Ghafil.pdf, 6 p.
20. Hazim, Nasir Ghafil ; Jármai, Károly: Ipari robot és manipulátorok kutatása és alkalmazás jármű-és autóipari mérnöki területeken, áttekintés GÉP 69 : 2 pp. 36-43. , 8 p. (2018).
21. H N, Ghafil; K, Jármai: Trajectory planning of a robot arm using flower pollination algorithm In: Peter, PLATKO; Mohamad, AL ALI; Štefan, KUŠNÍR (Eds.) 3rd International Conference on Engineering Sciences and Technologies: ESAT 2018

Košice, Slovakia: Technical University of Kosice, (2018) Paper: 023 - Ghafil, Jarmai.pdf, 4 p.

22. Ghafil, Hazim Nasir Inverse Acceleration Solution for Robot Manipulators using Harmony Search Algorithm INTERNATIONAL JOURNAL OF COMPUTER APPLICATIONS 6: 114 pp. 1-7. , 7 p. (2016).
23. Nasir Ghafil, Hazim: Vibration Analysis of a Generator Anti-Vibration Rubber Mounts Journal of Computer Science & Computational Mathematics [electronic only] 6: 4 pp. 105-111. , 7 p. (2016).
24. HN Ghafil, AH Mohammed, NH Hadi: A Virtual Reality Environment for 5-DOF Robot Manipulator based on XNA Framework INTERNATIONAL JOURNAL OF COMPUTER APPLICATIONS 113 : 3 pp. 33-37. 5 p. (2015).
25. HN Ghafil, AH Mohammed, NH Hadi: "Optimum Path Planning and Performance Analysis of a Robot Manipulator.". Thesis (2013), Alnahrain university Baghdad- Iraq.
26. Nasir Ghafil H, Alsamia S, Jármai K (2021) Fertilization optimization algorithm on CEC2015 and large scale problems. POLLACK PERIODICA. DOI: https://doi.org/10.1556/606.2021.00343
27. Alsamia S, Ibrahim DS, Ghafil HN (2021) Optimization of drilling performance using various metaheuristics. POLLACK PERIODICA, DOI: https://doi.org/10.1556/606.2021.00307
28. Hazim Albedran et al., “Interpolated spline method for a thermal distribution of a pipe with a turbulent heat flow”, MULTIDISCIPLINARY SCIENCES Vol. 11 No. 5 (2021). URL: https://ojs.uni-miskolc.hu/index.php/multi/article/view/883/659.
29. Ali Habeeb Askar, Hazim Albedran, et al, “A new method to predict temperature distribution on a tube at constant heat flux”. MULTIDISCIPLINARY SCIENCES Vol. 11 No. 5 (2021). URL: https://ojs.uni-iskolc.hu/index.php/multi/article/view/884/660