POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Automating Kinematic Resolution of Redundant Planar Manipulators Using Adaptive Neuro-Fuzzy Inference System (ANFIS) Modeling

NEGAR HASSANTABAR
Département de génie mécanique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie mécanique

Novembre 2021

Automating Kinematic Resolution of Redundant Planar Manipulators Using Adaptive Neuro-Fuzzy Inference System (ANFIS) Modeling

DEDICATION

To all who accompanied me on my journey.

ACKNOWLEDGEMENTS

First, I am grateful to my supervisor, Professor Luc Baron, for giving me the opportunity to work under his supervision and for his guidance throughout the way. Then, I would like to express my gratitude to my parents for their endless support and love. I would also like to thank my sister, Bahar, for her encouragements, accompany and always believing in me. Last but not least, my sincere thanks to my friend, Mehdi, who made my journey easier by his precious help and encouragements when I needed them.

RÉSUMÉ

Les manipulateurs robotiques sont utilisés pour différents types d'applications telles que la fabrication, l'assistance médicale, l'entretien dans l'espace et bien d'autres. La nécessité de prendre en compte les limites articulaires dans la planification de trajectoire est complexe car la trajectoire cartésienne requise doit être projetée dans l'espace articulaire du manipulateur. Lorsqu'il y a plus d'articulations que le degré de liberté requis par la tâche, le manipulateur est redondant par rapport à cette tâche. Dans cette situation, le degré de liberté supplémentaire du manipulateur peut être utilisé pour éviter les limites articulaires. Ce mémoire étudie la possibilité de résoudre la cinématique inverse d'un manipulateur redondant à l'aide d'un système intelligent artificiel (ANFIS), tout en considérant les limitations articulaires.

Un algorithme est proposé pour trouver la région réalisable de l'espace articulaire, tout en tenant compte des limitations de chaque articulation, et enregistrer cette région pour toutes les positions d'un point de l'outil du manipulateur dans l'espace de travail. L'algorithme construit un projeteur sur l'espace nul de la matrice Jacobienne afin d'effectuer un auto-mouvement du manipulateur, le long des deux directions de l'auto-mouvement, afin d'atteindre les deux limites de la région réalisable. En utilisant cette technique, la région réalisable fournie par la redondance est cartographiée sur chaque joint individuel pour avoir une meilleure compréhension des limitations provenant des autres joints. Le point médian de la région réalisable de l'articulation la moins stricte est sélectionné comme solution articulaire redondante et utilisé dans les formules cinématique inverse pour calculer la solution des autres articulations. Cet algorithme est utilisé pour trouver les limites de la région réalisable ainsi que la solution de norme minimale pour une grille de positions de l'outil couvrant l'ensemble de l'espace de travail du manipulateur. Ce jeu de données est utilisé comme données d'apprentissage par ANFIS afin de prédire ces valeurs sans avoir à utiliser cet algorithme laborieux, et proposer directement une solution articulaire redondante. Pour une trajectoire cartésienne donnée, la résolution de redondance du manipulateur s'effectue en temps réel en deux étapes. Tout d'abord, les positions finales du manipulateur sont introduites dans ANFIS afin d'obtenir la solution articulaire redondante. Deuxièmement, les formules cinématique inverse sont utilisées pour calculer les autres positions articulaires.

La validation a été menée sur un manipulateur série planaire à trois degrés de liberté pour effectuer une tâche à deux degrés de liberté, le positionnement de son outil. Le manipulateur a un degré de redondance et l'auto-mouvement a été utilisé pour éviter les limites articulaires. La simulation montre l'efficacité de l'algorithme proposé dans le calcul de la région réalisable et de la solution à la norme minimale. De plus, ANFIS a pu prédire directement la solution articulaire redondant, tandis que les formules cinématique inverse ont été utilisées pour calculer avec précision la solution des deux autres articulations.

Outre les techniques proposées, le résultat de cette thèse offre également une meilleure compréhension de l'espace de travail du manipulateur, de la zone réalisable de celui-ci et des limitations articulaires, la redondance et la cinématique inverse.

ABSTRACT

Robotic manipulators are used for different types of applications such as manufacturing, medical assistance, space servicing and many others. The need for considering joint limits in the path planning is complex because the required Cartesian trajectory needs to be mapped into the joint space of the manipulator. When there is more joints than the degree of freedom required by the task, the manipulator is redundant relative to this task. In this situation, the extra degree of freedom of the manipulator can be used to avoid joint limits. This thesis investigates the possibility of solving the inverse kinematic of a redundant manipulator with the help of an artificial intelligent system (ANFIS), while considering the joint limitations.

An algorithm is proposed to find the feasible region of the joint space, while considering the limitations of every joints, and record this region for any end-point positions of the manipulator of the workspace. The algorithm builds a projector on the null-space of the Jacobian matrix in order to perform a self-motion of the manipulator, along both directions of the self-motion, in order to reach the two limits of the feasible region on any joint. Using this technique, the feasible region provided by the redundancy is mapped on each individual joint to have a better understanding of the limitations coming from the other joints. The middle-point of the feasible region of the least relaxed joint is selected as the redundant joint solution and used in the inverse kinematic formula to calculate the solution of the other joints. This algorithm is used to find the limits of feasible region together with the minimum-norm solution for a grid of end-point positions covering the whole workspace of the manipulator. This data-set is used as training data by an ANFIS in order to predict these values without having to use this time consuming algorithm, and propose directly a redundant joint solution. For a given Cartesian trajectory, the redundancy resolution of the manipulator is performed in real-time is two steps. First, the end-point positions of the manipulator is feed into the ANFIS in order to obtain the redundant joint solution. Second, the inverse kinematic formula are used to compute the other joint positions.

Validation was conducted on a planar three degrees of freedom serial manipulator while performing a two degrees of freedom task, the positioning of its end-point. The manipulator has one degree of redundancy and the self-motion was used to avoid joint limits. The simulation show the effectiveness of the proposed algorithm in computing the feasible region and the minimum-norm solution. Moreover, ANFIS was able to directly predict the redundant joint solution, while the inverse kinematic formula was used to compute accurately the solution of the other two joints.

Aside from the proposed techniques, the result of this thesis also offers a better understanding of the workspace of the robotic manipulator, feasible region of the manipulator and joint limitations, redundancy and inverse kinematic.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

DOF	Degree Of Freedom
ANFIS	Adaptive Network-based Fuzzy Inference System
IK	Inverse Kinematics
EE	End-Effector
AI	Artificial Intelligence
GPM	Gradient Projection Method
$\mathbf{p}$	Cartesian position of the point on EE
$\dot{\mathbf{p}}$	Velocity of the point on EE
$\boldsymbol{\theta}$	Joint position
$\dot{\boldsymbol{\theta}}$	Joint velocity
$\Delta\mathbf{p}$	Displacement
$\Delta\boldsymbol{\theta}$	Joint displacement

CHAPITRE 1 INTRODUCTION

1.1 Elements of the Problem and Problem Definition

According to ISO 8373, Manipulating industrial robots-Vocabulary (1994):

Manipulator: "A machine, the mechanism of which usually consists of a series of segments, jointed or sliding relative to one another, for the purpose of grasping and/or moving objects (pieces or tools) usually in several degrees of freedom. It may be controlled by an operator, a programmable electronic controller, or any logic system (for example cam device, wired, etc.)" [1]". The manipulators are designed to help humans with repetitive tasks or the tasks which are dangerous or difficult to handle for humans. Figure 1.1 shows an example of an industrial manipulator.

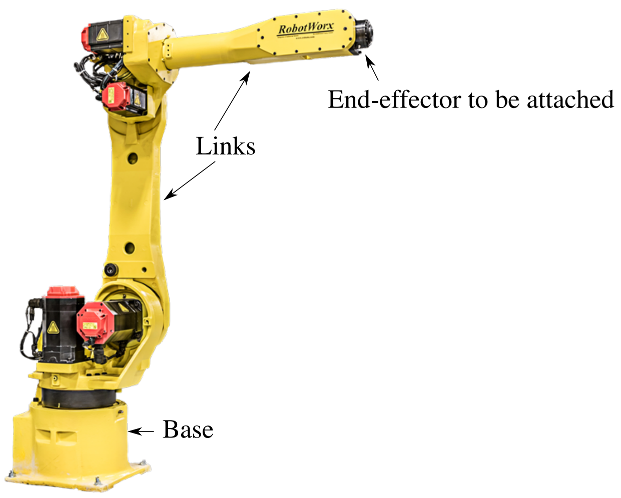


Figure 1.1 Serial 6-DOF industrial manipulator [2]

In order that a manipulator performs a particular task, the end-effector should take the specified positions and the joint parameters must satisfy the kinematic rules. However, there might exist more than one possibility for the manipulator if it is redundant. Thus, taking advantage of the redundancy we can select the joint path while the manipulator is performing a task. However, despite of the plenty of choices we can make if the robot is redundant, there are some limitations such as singularities and joint limitations.

This research aims to answer the following main research question:

— When planning a joint motion trajectory of a serial redundant manipulator, is it possible to use an artificial intelligence (AI) model to help us in finding an exact inverse kinematic solution by taking advantage of the redundancy resolution while considering the robot joint limitation?

1.1.1 Research Scope

This research focuses on serial manipulators and will propose algorithms to solve the kinematic redundancy of manipulators with one degree of redundancy and find the space of possible joint positions in which we can select a joint path without exceeding the joint limits. For validation purposes a planar manipulator has been chosen for modeling and simulation. However, we believe that an algorithm could be extended for a one degree of redundancy task for a serial manipulator in 3-dimensional space as well as in 2-dimensional space.

1.1.2 Research objectives

— Resolving the kinematic redundancy in order to avoid joint limits.

— Using an ANFIS system to learn the pre-computed redundancy resolution for the entire workspace and use it for any Cartesian trajectory in order to avoid computational complexity each time performing a new task.

1.1.3 Assumptions

The following assumptions were made in this research work:

A) The robotic manipulator is a dynamic system with a predictable behaviour and hence we use the principle of causality.

B) Rigid body behaviors:
Links do not deflect or deform;
There is no slack between joints and links;
Joints shafts are considered to be perfectly stiff.

C) There is no link interference, possible for planar robot;

1.2 Thesis Outline

Chapter 2 presents a literature review of basic concepts and the existing research and software is presented in Chapter 2. Chapter 3, discusses the kinematic modeling of a redundant serial planar manipulator, while Chapter 4 is dedicated to the path planning and Fuzzy logic side of the research.Chapter 5 presents the conclusion, future research.

CHAPITRE 2 THEORY AND LITERATURE REVIEW

2.1 Basic definitions and Terminology

"According to the Terminology for the Theory of Machines and Mechanisms defined by IFToMM (1991): A Kinematic chain is an assemblage of links and joints " [1]. "There are six lower kinematic pairs (joints), namely, prismatic (P), cylindrical (C), revolute (R), helical(H), spherical (S), and planar (E). Simple chains are defined as kinematic chains containing links with a degree of connectivity smaller than or equal to two" [3]. In figure, 2.1 the different kinematic pairs are illustrated.

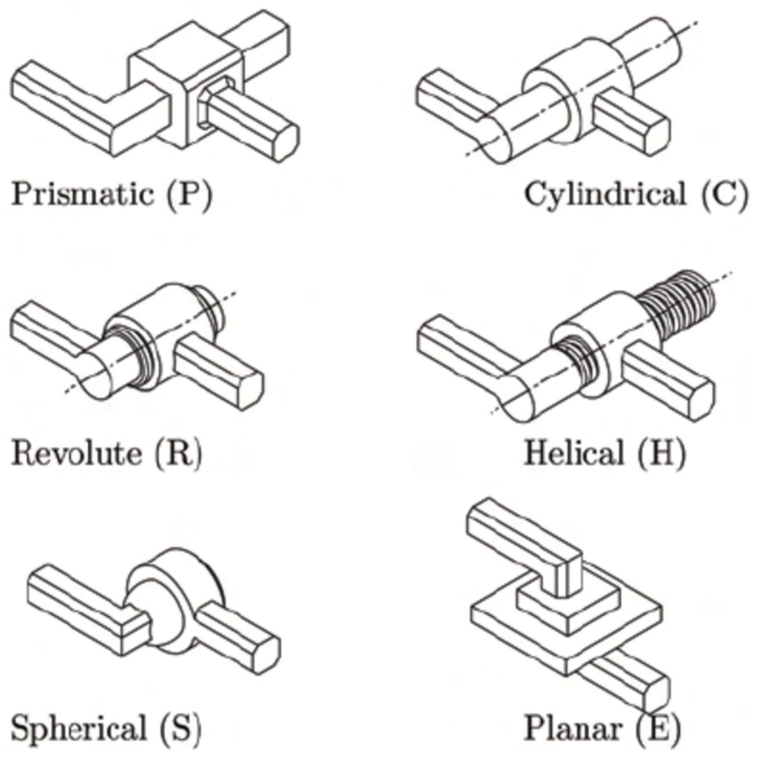


Figure 2.1 Lower kinematic pairs [4]

2.1.1 Degrees of Freedom and Kinematic Redundancy

Degree of freedom (DOF) of a mechanical system is defined as the minimum number of independent parameters that we need to completely define the configuration of the mechanical systems. In a serial manipulator the configuration of the manipulator can be specified by knowing the configuration of all the joints, hence the DOF of a manipulator is defined as the sum of DOF of each joint. On the other hand, the DOF of the end-effector is defined as the number of independent translational and orientational coordinates that we need to specify the exact end-effector configuration.

Kinematic redundancy occurs when a robotic manipulator has more degrees of freedom than the number of independent variables that is required to specify a task by its end-effector.

The end-effector of a planar manipulator, needs three independent variables to be defined: two positional and one orientational. However, we may need less variables to perform a desired task if the orientation of the last link doesn't play a role in performing the task. Some examples of these tasks are laser cutting and welding.

2.1.2 Inverse Kinematic

Inverse Kinematics (IK) is a procedure to find a set of appropriate joint parameters for performing a given kinematic task. Over the past decades different techniques have been proposed to solve IK problems [5]. In order to obtain an IK solution, we need to solve non-linear equations with transcendental functions, and it might not be possible to acquire a closed-form solution [6, 7]. Algebraic, geometric, and iterative techniques have been used to solve IK for complex manipulators; however, these techniques suffers from complicated mathematical formulation with a time consuming solving procedure [6]. In 1969 a technique has been introduced by Whitney [8] to write a linear equation between the end-effector velocity and the joint velocity. It is possible to solve IK problems as long as we have small displacements. As showing in 2.1, the Jacobian matrix is the coefficient of the linear equation, as follow:

$$\dot{\mathbf{p}} = \mathbf{J}\dot{\boldsymbol{\theta}} \tag{2.1}$$

Where $\mathbf{J}$ is the Jacobian matrix, $\dot{\mathbf{p}}$ is the velocity of a point of the end-effector, and $\dot{\boldsymbol{\theta}}$ are the joint velocities of the manipulator [7].

2.1.3 Singularities of the Serial Manipulator

Singular configuration of a manipulator is defined as the configurations in which the ability of movement of the end-effector becomes limited [9]. In these configurations, the Jacobian matrix is not full rank. An example of a singular configuration for a serial manipulator is when three of the joints axes are parallel in the same plane, so the rotation is limited to around one axis and the translation is only possible perpendicular to the common plane [10]. Figure 2.2 shows a manipulator at this singular configuration.

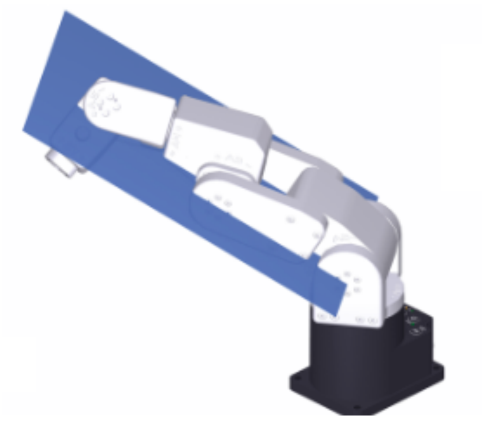


Figure 2.2 Manipulator at a singularity configuration [11]

2.2 Redundancy Resolution

As mentioned earlier redundant manipulators have more DOF than required to perform a task. This excess DOF result in unlimited number of possible joint position, while solving inverse kinematic. Redundancy resolution is defined as finding the feasible joint path for a given end-effector task [12]. Different techniques have been employed to use the extra DOF to satisfy other secondary tasks such as joint-limits, singularity avoidance, torque optimization, energy minimization, etc [13]. These techniques fall into the following main categories : pseudoinverse technique, geometrical methods, optimisation techniques, redundancy resolution by artificial intelligence (AI) [14, 15] . In the following sections we will discuss these techniques in more details.

2.2.1 Pseudoinverse Technique

One of the most used redundancy resolution method is the Pseudo inverse technique. As mentioned in inverse Kinematic section, one of the method to resolve the inverse kinematic of serial manipulators to perform a particular Cartesian trajectory, is to solve it at the velocity level. From 2.1 we can replace the velocities by small displacement as follow:

$$\Delta\mathbf{p} = \mathbf{J}\Delta\boldsymbol{\theta} \tag{2.2}$$

Where $\mathbf{J}$ is the Jacobian matrix, $\Delta\mathbf{p}$ is the small displacement in the Cartesian space and $\Delta\boldsymbol{\theta}$ is a small joint displacement. If $\mathbf{J}$ is square, we can solve the inverse kinematic [16] as:

$$\Delta\boldsymbol{\theta} = \mathbf{J}^{-1}\Delta\mathbf{p} \tag{2.3}$$

When $\mathbf{J}$ is redundant, it possess more columns than rows, and there is more than one answer for $\Delta\boldsymbol{\theta}$. Thus, equation 2.2 is solved with pseudoinverse [16,17] in order to find the minimum norm solution plus a possible projection over the null space of $\mathbf{J}$.

$$\Delta\boldsymbol{\theta} = \underbrace{\mathbf{J}^{+}\Delta\mathbf{p}}_{minimum-norm\ solution} + \underbrace{(\mathbf{I}-\mathbf{J}^{+}\mathbf{J})\mathbf{u}}_{homogeneous\ solution} \tag{2.4}$$

In that $\mathbf{I}$ is the identity matrix of the dimension $n \times n$, $\mathbf{J}^{+}$ is the pseudoinverse (generalized inverse) of $\mathbf{J}$ with the dimension $n \times m$ and vector $\mathbf{u}$ is an arbitrary vector in $\mathbf{R}^{n}$ and we can select this vector in such a way that it satisfies secondary constraints. As shown in equation 2.4, the first part of the right hand side of the equation is the minimum-norm solution which minimize $||\Delta\boldsymbol{\theta}||$ subject to $\mathbf{J}\Delta\boldsymbol{\theta} = \Delta\mathbf{p}$.

2.2.2 Optimization Techniques

Most optimization techniques are based on optimizing an objective function which is derived to satisfy a secondary task with the help of the pseudoinverse formulation. These methods are classified into two main categories: local optimization and global optimization. Local optimization is a linear optimization problem and can be implemented in real-time applications due to its simplicity [18]. Global optimization on the other hand, is useful for off-line trajectory planning in which computation time is less important and the strict optimality is needed. For example, in space application where the use of energy must be efficient, global optimization is used to acquire the minimum energy motion [19]. One of the commonly used optimization technique with which pseudoinverse formulation is used, is the Gradient Projection Method (GPM) that was used in [20] to avoid joint limits by projecting the gradient of a function of the joint angles and their limits into the null space of Jacobian matrix [13]. There also exist optimization techniques such as [21] which do not require calculation of pseudoinverse [14]. In this technique a formulation was proposed to obtain joint velocities instead of pseudoinverse calculation, while optimizing a given secondary task with gradient projection method. Genetic algorithm was also used as an optimization technique to solve redundancy resolution of hyper redundant manipulators by searching for all global optimum solutions. In [22], for example, genetic algorithm was employed to find "the best compliance" joint trajectory to satisfy bionics principle.

2.2.3 Geometrical Method

Some geometrical methods are proposed to resolve redundancy which are not based on pseudoinverse of Jacobian. As an example [23] proposed to consider the angles between each adjacent links equal. This approach prevents the manipulator from singular configurations since lining up of the joint axis will not be possible. However, the method is proposed for planar manipulators and the formulation was not extended to 3D.

2.2.4 Redundancy Resolution by AI

Artificial intelligence (AI) has also been used by researchers to solve redundancy resolution problems. Although there exist more major branches of AI , most of the focus in this area has been on using neural network and fuzzy logic based solutions or a combination of these two. In figure 2.3, neural network, fuzzy logic and Adaptive Network-based Fuzzy Inference System (ANFIS) are shown in the context of AI. In the following, some of the research work in these areas are presented.

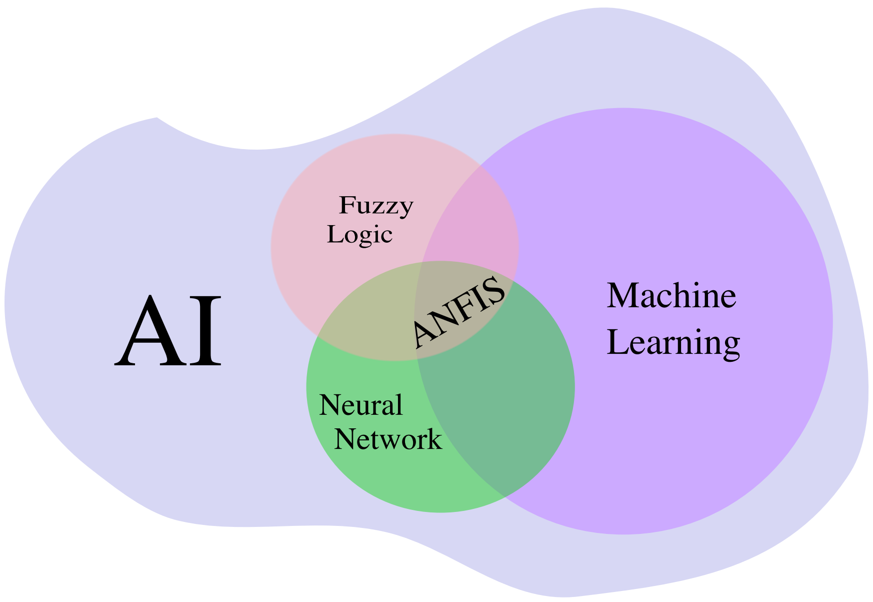


Figure 2.3 Neural Network, Fuzzy Logic and ANFIS in the context of AI

Neural Networks

Neural networks were recently used in this area as it provides us with a prediction based on learning the relationship between a given input and output of a data-set. Collision avoidance, physical constraints, obstacle avoidance, joint velocity and acceleration control has been considered as a secondary to be solved with redundancy resolution by neural networks [14]. In [24] for example, a one-layer dual neural network was proposed to solve redundancy resolution, considering joint limit of the manipulators and joint velocity limits. In another research [25] collision free motion control of redundant manipulator was solved by a recurrent neural network through tuning the problem to an optimization problem. Although there are more research work using neural network to solve a redundancy resolution problem, most of the research are concentrated on the ability of the neural network to solve an optimization problem and not on its learning ability for prediction or automation purposes.

Fuzzy logic and ANFIS

Another technique that has been used in redundancy resolution is fuzzy logic. This technique has been mostly used to make decisions based on linguistic fuzzy rules which are defined by expert knowledge. In [26] a fuzzy system was proposed to determine joint angles of the manipulator while respecting the joint limitation and avoiding obstacle. In this research fuzzy rules were defined to correct the joint angles based on two criteria: the distance between the robot links and obstacles and, the distance between a joint angle and the middle position in the joint coordinates. Similarly in [27] fuzzy rules are employed to avoid calculating pseudoinverse of Jacobian matrix and mapping the Cartesian space trajectory to the Joint space trajectory directly by making decisions on the joint angles displacement. In [28] on the other hand, a fuzzy system was designed to select between the joint trajectories which were already calculated through pseudoinverse and gradient vectors techniques. The selection was made based on the satisfaction degrees of a trajectory which was modeled by a function that reflects the distance to the singular configuration and joint angle limits. Another form of using fuzzy logic technique is when we train and tune the fuzzy sets and rules with numerical data. instead of just relying on the expert knowledge. This technique is called adaptive Fuzzy Logic. In [29] for example, adaptive Fuzzy Logic technique was used to solve redundancy resolution problem while avoiding obstacles and joint limits. In this research, a cost function was defined to reflect the error vector of the end-effector in Cartesian space and the rules are updated via an adaptive law and based on the values that minimize the cost function. For the purpose of training fuzzy sets and rules, different techniques are employed. One of these technique is utilizing neural network; thus, in some studies Adaptive Network-based Fuzzy Inference System (ANFIS) was used for redundancy resolution. In [30] for instance, ANFIS was used to solve forward and inverse kinematic of redundant 2-DOF, 3-DOF and 5-DOF manipulators. In the mentioned research, a certain number of data points are acquired by solving the forward kinematic and inverse kinematic of the manipulators analytically and then, this information was feed into ANFIS system to obtain the solution. However this research was concentrated on obtaining a possible solution for forward and inverse kinematic without considering any secondary task.

2.3 Inverse Kinematic, path planning and ANFIS

Although the concentration of this research is on the redundancy resolution by AI, solving the inverse kinematic is part of this work and it is noteworthy to mention that inverse kinematic can be also solved by ANFIS. Due to the non-linear and complicated equations associated with solving inverse kinematic specially as the motion of the system become more complicated [31], there are some research mentioning these techniques. In [32] for example, the inverse kinematic of a three DOF manipulator was calculated by ANFIS. Also in [31] the ANFIS is trained by the data generated by mathematical modeling so the complexity of system has been reduced. Also in [33] ANFIS was used with the same approach and forward kinematic data was used for training. Another research work using this approach is presented in [34]. Other similar approaches are different in input data selection and modeling the problem to minimize the error [34]. However, the solutions generated by ANFIS through this approach are approximate while there is already an analytical solution for these problems. ANFIS is also used in path planning problem such as in [35]. In these research work, ANFIS was utilized as a curve fitting tool to smooth the via-points which satisfy the requirements of the path planning such as obstacle avoidance.

2.4 Inverse Kinematic and Redundancy Resolution Applications

The redundancy resolution is still an open area to investigate and conduct studies to gain more efficient results. These studies are specially useful in off-line and online trajectory planning of robotic manipulators, robot programming software and simulation softwares including the software like Robotmaster, ROBCAD and OCTOPUZ. The application of these softwares address the need of many manufacturing applications including robotic welding, trimming, machining, polishing, cutting, etc [36]. Figures 2.4 , 2.5, 2.6 and 2.7 shows a picture of robotic manipulator simulation in Robotmaster software showing the manipulators while machining, cutting, welding and trimming.

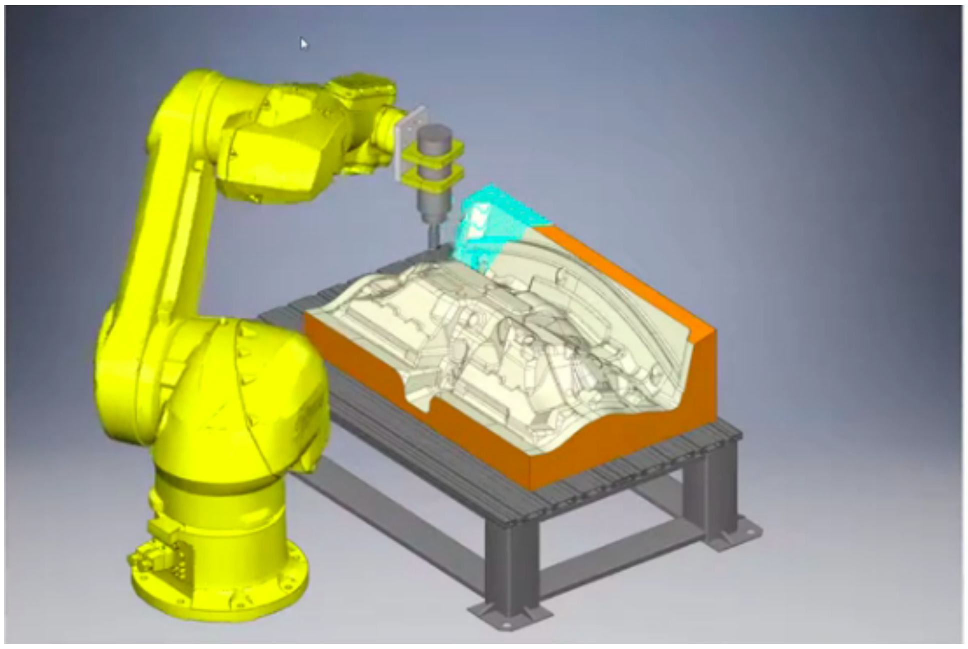


Figure 2.4 Machining-Milling of a mold using a Staubli robot- Pictures of robotic manipulators captured from simulation in Robotmaster software while machining, cutting, welding and trimming [36]

2.5 Scope of this research compared to the work of others

This research aims to address a redundancy resolution problem by proposing a technique using Pseudoinverse Technique formulation. The technique is not a conventional optimisation technique but using this formulation to provide a better understanding of the redundant space and develop an exact Inverse Kinematic solution based on this understanding. In this research work the ANFIS system is employed as an AI system to use pre-calculated bound-limits of the joints as the training data set to acquire the these limits for any desired path so that the process of finding the bound limits become easier and our inverse kinematic solution always fall into the joint limits of the manipulator.

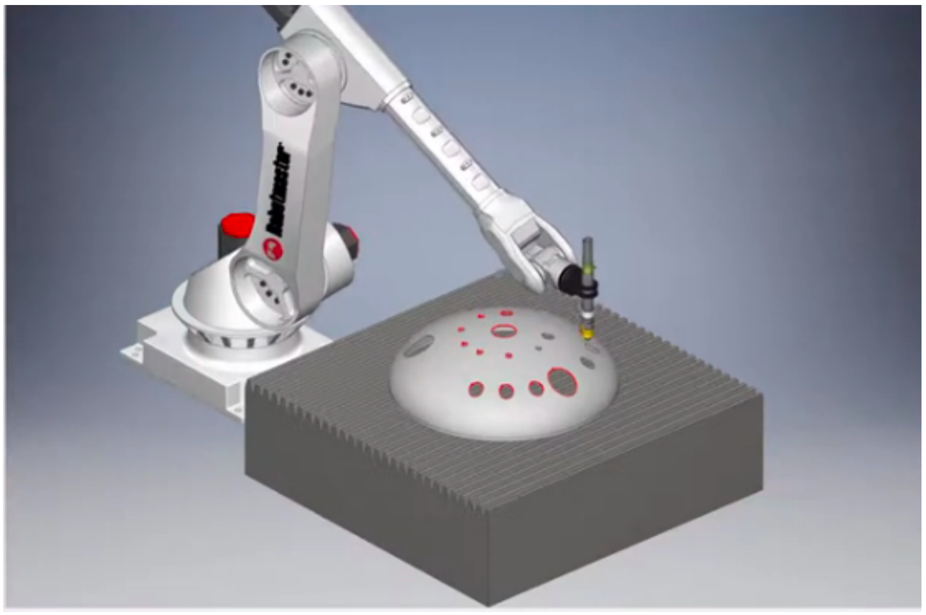


Figure 2.5 Cutting-Robotic programming for plasma cutting holes into a steel dome-Pictures of robotic manipulators captured from simulation in Robotmaster software while machining, cutting, welding and trimming [36]

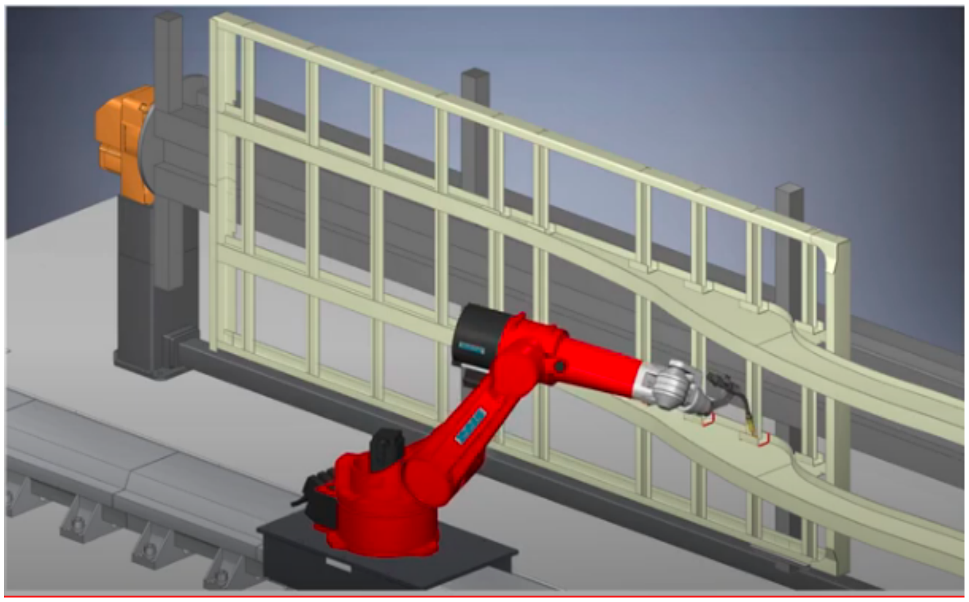


Figure 2.6 Welding the joints of a truck bed using a Reis robot- Pictures of robotic manipulators captured from simulation in Robotmaster software while machining, cutting, welding and trimming [36]

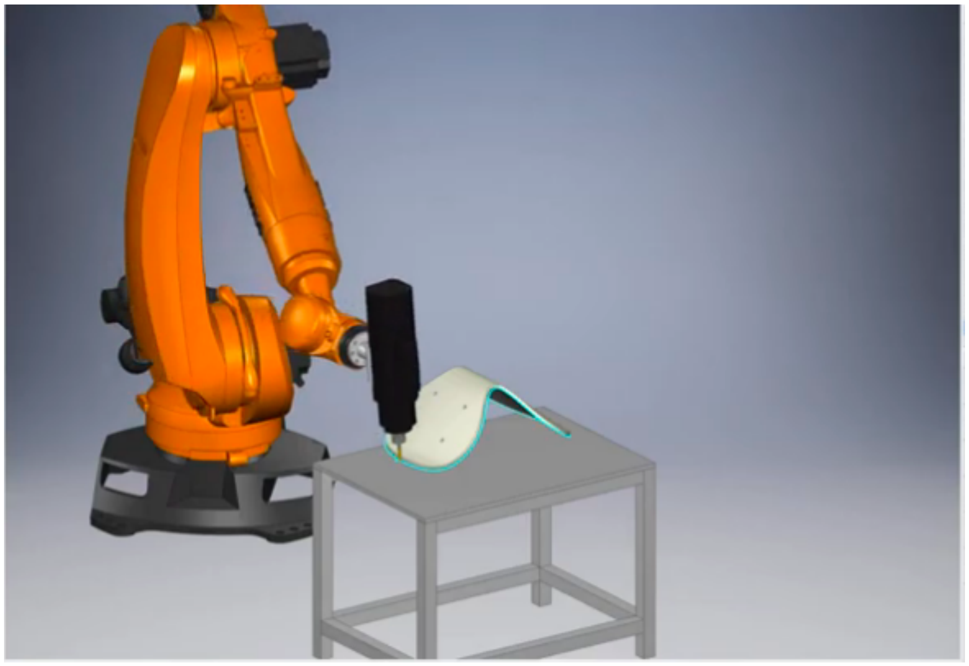


Figure 2.7 Welding the joints of a truck bed using a Reis robot- Pictures of robotic manipulators captured from simulation in Robotmaster software while machining, cutting, welding and trimming [36]

CHAPITRE 3 KINEMATICS

3.1 Manipulator, kinematic redundancy

The modeling has been conducted for a 3-DOF planar serial manipulator which consists of three revolute joints. The manipulator is intrinsically redundant in its operational space (the plane) and functionally redundant for a given task in the plane if not considering the orientation of the end-effector. Thus there are infinite joint trajectories to perform any given task within its workspace. However, every revolute joint has its own limitation and avoiding those limits gives us a smaller feasible region within the joint-space in which we can select a trajectory for each joint. This chapter investigate this selection in more details.

3.2 Task description

The area in that we can select a trajectory for a certain joint in the joint space is not only limited because of its own limitation but also bounded because of the nature of the manipulator and because of the limitations of other joints. The problem to address in this section is to find all the feasible region in the joint space for each joint in order to perform a particular path in the Cartesian-space by taking advantage of redundancy. Figure 3.1 shows the input and output of the process we propose to find the feasible regions in joint-space for a given Cartesian trajectory.

3.2.1 Problem explanation

Considering eq. (2.4), we can see that the manipulator can still perform a self-motion with no displacement for the end-effector ($\Delta\mathbf{p} = 0$) and the equation yields to:

$$\Delta\boldsymbol{\theta} = (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\mathbf{u} \tag{3.1}$$

Eq. (3.1) allows to compute the self-motion of eq. (2.4). i.e. the joint motion while maintaining the end point at the same location. According to definition [37] null space of the matrix $\mathbf{J}$ is defined as a set of all vectors $\mathbf{x}$ which satisfy the equation 3.2:

$$\mathbf{J}\mathbf{x} = 0 \tag{3.2}$$

. Thus, considering equation 2.2 ($\Delta\mathbf{p} = \mathbf{J}\Delta\boldsymbol{\theta}$), when there is no Cartesian displacement

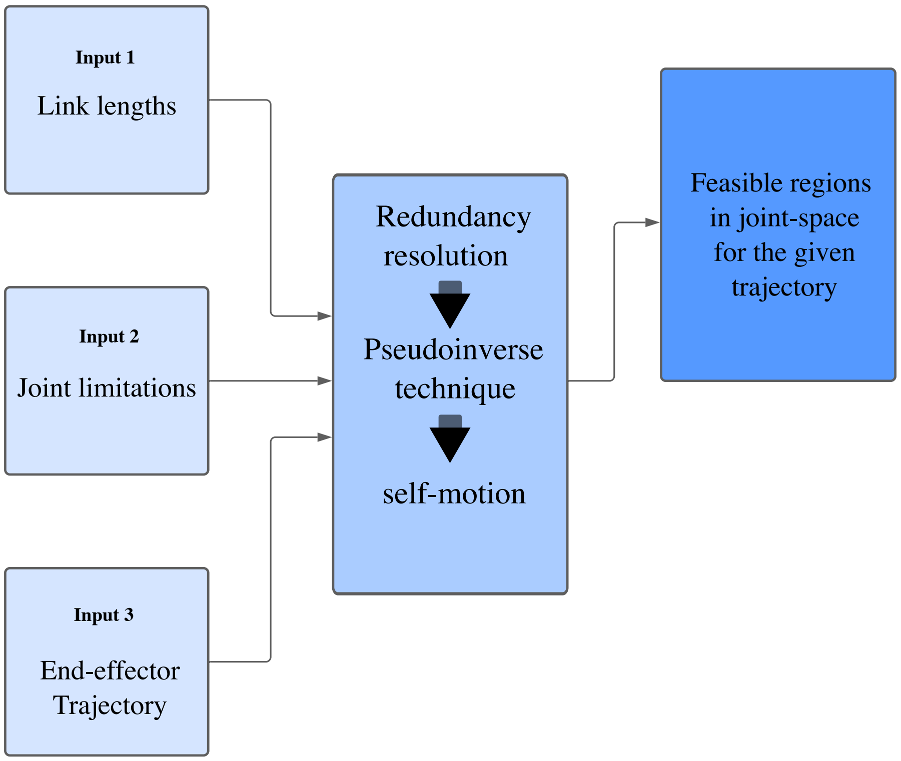


Figure 3.1 Overall strategy to find the feasible region

($\Delta \mathbf{p} = 0$) in our case, the null space of the Jacobin matrix is a set of $\Delta \boldsymbol{\theta}$ vectors .

Using the eq. (2.4) and (3.1), we have found all the feasible regions for the manipulator, considering its joint-limits by simulating the manipulator to perform certain paths in the Cartesian space. As shown in algorithm 1 in the Appendix, the manipulator starts at the point P_0 then on its way toward the last point P_1, at each point we try to minimize the error between the real position of the manipulator and the desired position in the path then, the manipulator performs a self-motion in one direction at each point of the discretized path, and until we reach a joint-limit $(Appendix\ A : s \neq 0)$. Then, while keeping the End-Effector at the same point, it performs the self motion in the opposite direction by multiplying the $\mathbf{u}$ vector of the eq. (3.2.1) by -1 in order to reach the other bound-limit. After the self-motion is done in the both direction, we move on to the next point of the task path. Figure 3.3 presents the overall process while Figure 3.2 illustrates the self motion of the manipulator at a fixed point of the end-effector. In other words, the joint displacement from the first joint limit that is reached by the manipulator to the joint limit that is reached by the manipulator in the other direction when there is no Cartesian displacement. Equation 2.4 shows a decomposition of $\Delta \boldsymbol{\theta}$ in which the second part of the right hand side of the equation is a projection of the $\mathbf{u}$

vector on a local plane tangent to the self motion line. The projected $\mathbf{u}$ vector on this plane is the local $\Delta\boldsymbol{\theta}$ at a certain joint configuration $\boldsymbol{\theta}$. That means the Figure shows the projection of the $\mathbf{u}$ vector on the null space of Jacobin when the end effector of the manipulator does not move but there is a joint displacement ($\Delta\mathbf{p} = 0$, $\Delta\boldsymbol{\theta} \neq 0$)). The $\mathbf{u}$ vector does not change along the joint trajectory however the projection $\Delta\boldsymbol{\theta}$ changes along the trajectory.

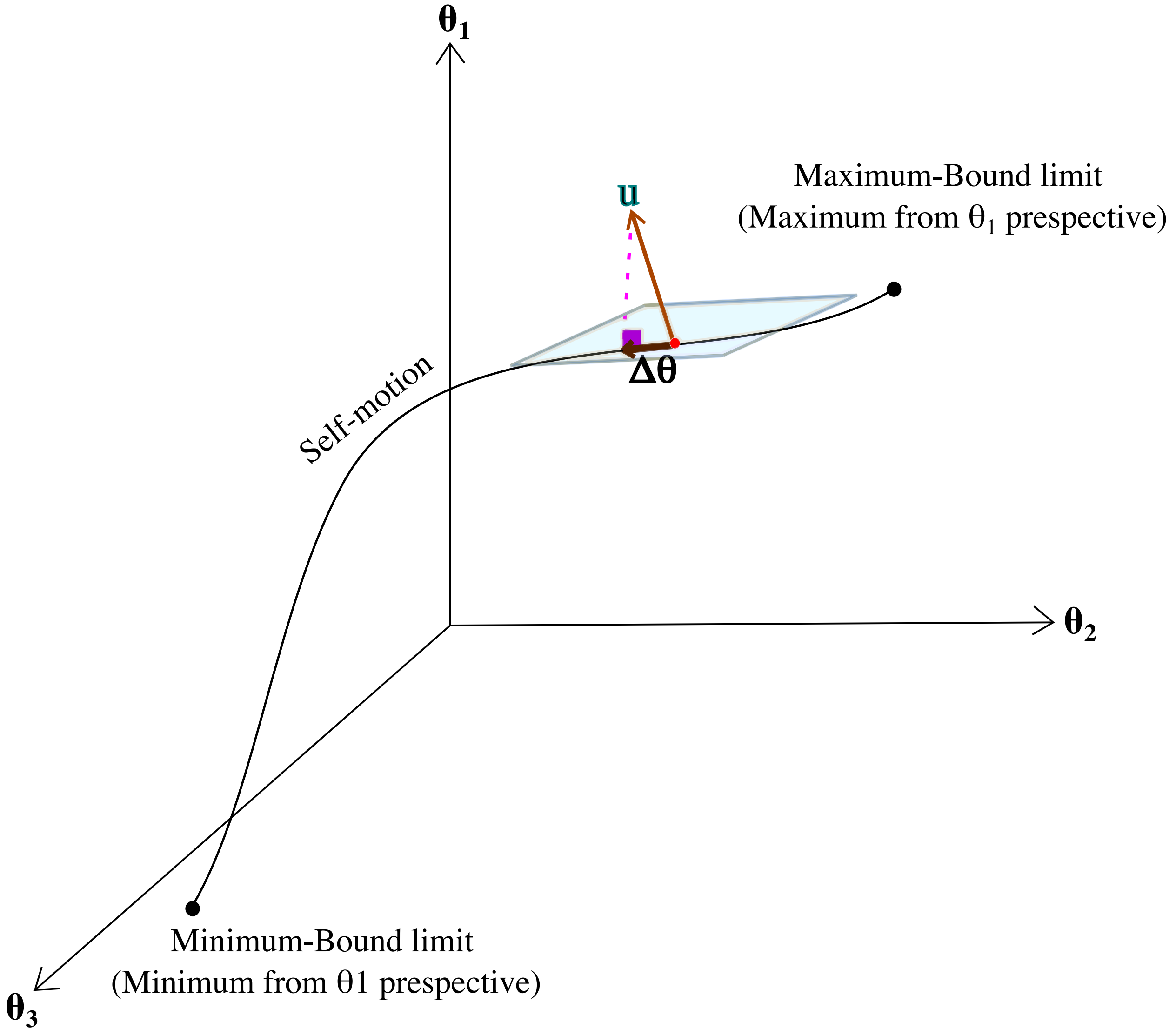


Figure 3.2 Joint displacement at a fixed point of the end-effector- Projection of vector $\mathbf{u}$ on the null space of matrix $\mathbf{J}$

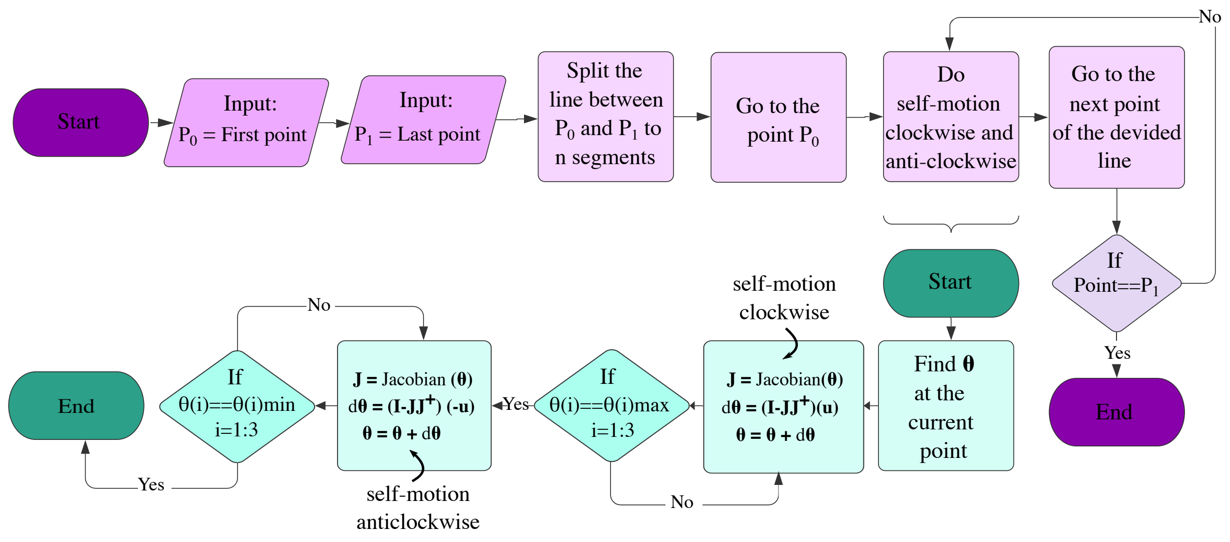


Figure 3.3 The algorithm to find the feasible joint-space region for a certain manipulator performing a line from P_0 to P_1

3.3 Simulation Results

Figure 3.4 shows the planar 3-DOF manipulator used for simulation, while table 3.1 gives its geometrical parameters. The simulation has been done for a square path and a rectangle path. For each path, we find a figures for θ_1, θ_2 and θ_3, which show the bound limits and the feasible region for each of the joints.

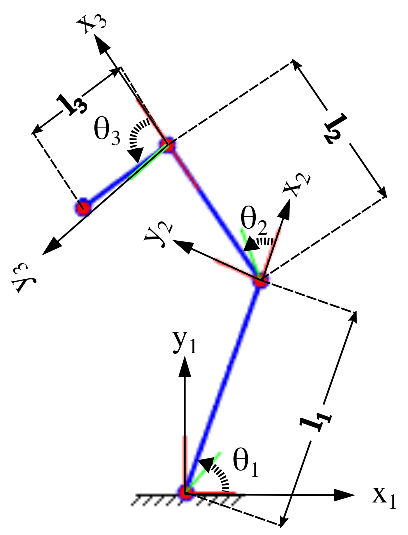


Figure 3.4 planar 3-DOF serial manipulator

Table 3.1 Manipulator Parameters.

	$\theta_{Min.}$	$\theta_{Max.}$	l_i
$i = 1$	0	$\pi/2$	2
$i = 2$	0	$\pi/2$	1.5
$i = 3$	0	π	1

3.3.1 Trajectory A: Square

The coordinates of the square trajectory are specified In the table 3.2. The manipulator moves from P_0 to the other coordinates of the path at P_1 then P_2, P_3 and finally returns to P_4 which is the same point as the starting point.

Table 3.2 Coordinates of trajectory A.

P_i	P_0	P_1	P_2	P_3	P_4
x	-1	1	1	-1	-1
y	2	2	4	4	2

Figure 3.5 shows the manipulator at its initial position along trajectory A. The configuration of the manipulator being the one obtain by the minimum -norm solution of eq. 2.4. The green lines shows the orientation of the link at the mid-joint position, while the red small lines shows the joint limits. Apparently from Fig. 3.5 the manipulation is away from its joint limits as found by the minimum-norm solution. Our algorithm now performs a self motion of the manipulator (meaning that the end-point will not move in Cartesian space) in any of the upper side or lower side with an arbitrary vector u and -u up to any joint exceed its limits. The joint position are recorded for the upper and lower limits together with the minimum-norm solution.

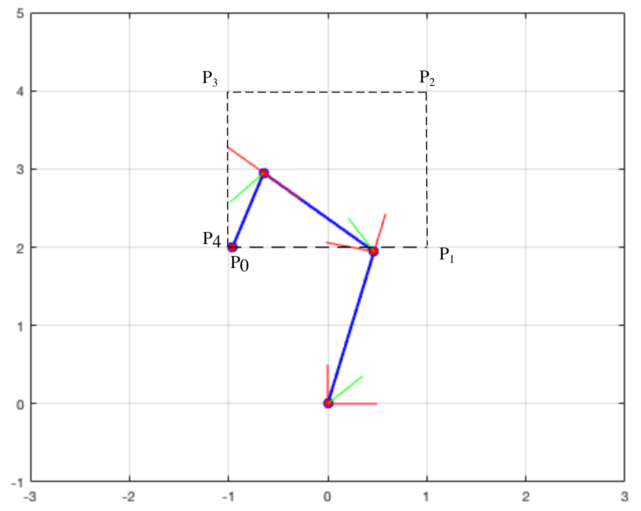


Figure 3.5 Manipulator at its initial position, minimum-norm-solution

Figure 3.6 shows the feasible θ_1 regions for trajectory A. From left to right we have the percentage of completion of the trajectory as the 4 segments of the square trajectory. The white region with red boarders is the feasible θ_1 solutions allowed by the redundancy along the trajectory. Any joint path of θ_1 within that region can be chosen to preform the Cartesian trajectory and avoiding the limits of every joints. The 2 dashed horizontal black lines are the lower and upper limits of θ_1. Obviously, the feasible region needs to site between those 2 horizontal lines. When a red boarder of the feasible region touch one of these 2 horizontal lines, it means that the limit of θ_1 is reached. When the red boarders are inside these 2 horizontal lines, it means that the manipulator cannot go further like the upper black region between 5% to 28%, otherwise 1 or 2 of the other joint limit is reached. This figure is very helpful for the planning of the joint path, since the vertical axis shows the redundant capacity of the manipulator, while the horizontal axis is different end-point position along the Cartesian trajectory. This 2D figure can be produce for any manipulator having 1 degree of kinematic redundancy. Of course the kinematic relationships would be different based on the task mobility of the end-effector.

Figure 3.7 shows 4 configurations of the manipulator along its self motion while keeping the end-point at 79% of the trajectory A. Figure 3.7 a) shows the manipulator at its lower-bound of the feasible region because of θ_3; b,c) shows the manipulator at intermediate configurations And d) shows the manipulator at its upper-bound of the feasible region. Configuration 3.7 a, 3.7b and 3.7 d can be seen on Figure 3.6 along a line at 79%.

Again, Figure 3.8 shows 4 configurations of the manipulator along its self motion while keeping the end-point at 10% of the trajectory A. Figure 3.8 a) shows the manipulator at its lower-bound of the feasible region because of singularity b,c) shows the manipulator at intermediate configurations And d) shows the manipulator at its upper-bound of the feasible region caused by θ_2 limitations. Points a' , b', c' and d' corresponds to a, b, c and d poses in Figure 3.8.

Figure 3.9 shows the corresponding feasible θ_2 regions for trajectory A. The 2 dashed horizontal black lines are the lower and upper limits of θ_2. Obviously, the feasible region needs to site between those 2 horizontal lines. When a red boarder of the feasible region touch one of these 2 horizontal lines, it means that the limit of θ_2 is reached. When the red boarders are inside these 2 horizontal lines, it means that the manipulator cannot go further like the upper black region between 5% to 28%, otherwise 1 or 2 of the other joint limit is reached. This figure is very helpful for the planning of the joint path, since the vertical axis shows the redundant capacity of the manipulator, while the horizontal axis is different end-point position along the Cartesian trajectory. This figure can be used as an alternative to Figure 3.6, since once you pick a value of any of θ_1, θ_2 or θ_3, the other 2 joint values can be explicitly computed. The same thing applies to Figure 3.10.

Consequently, Figures 3.6, 3.9 and 3.10 are useful tools for performing the path planning of Cartesian trajectories in the context of kinematic redundancy.

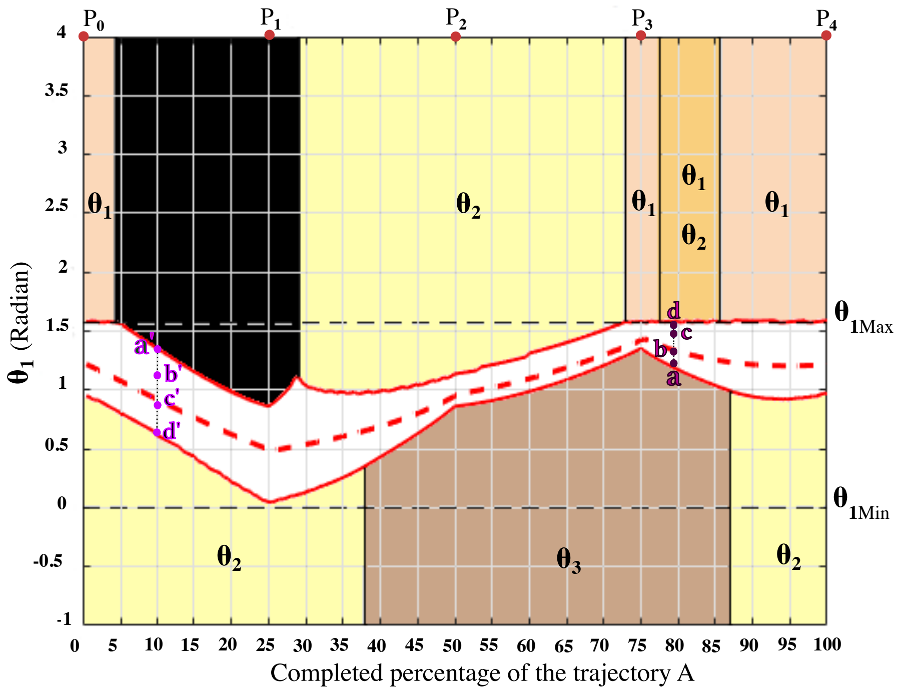


Figure 3.6 Feasible θ_1 region for trajectory A

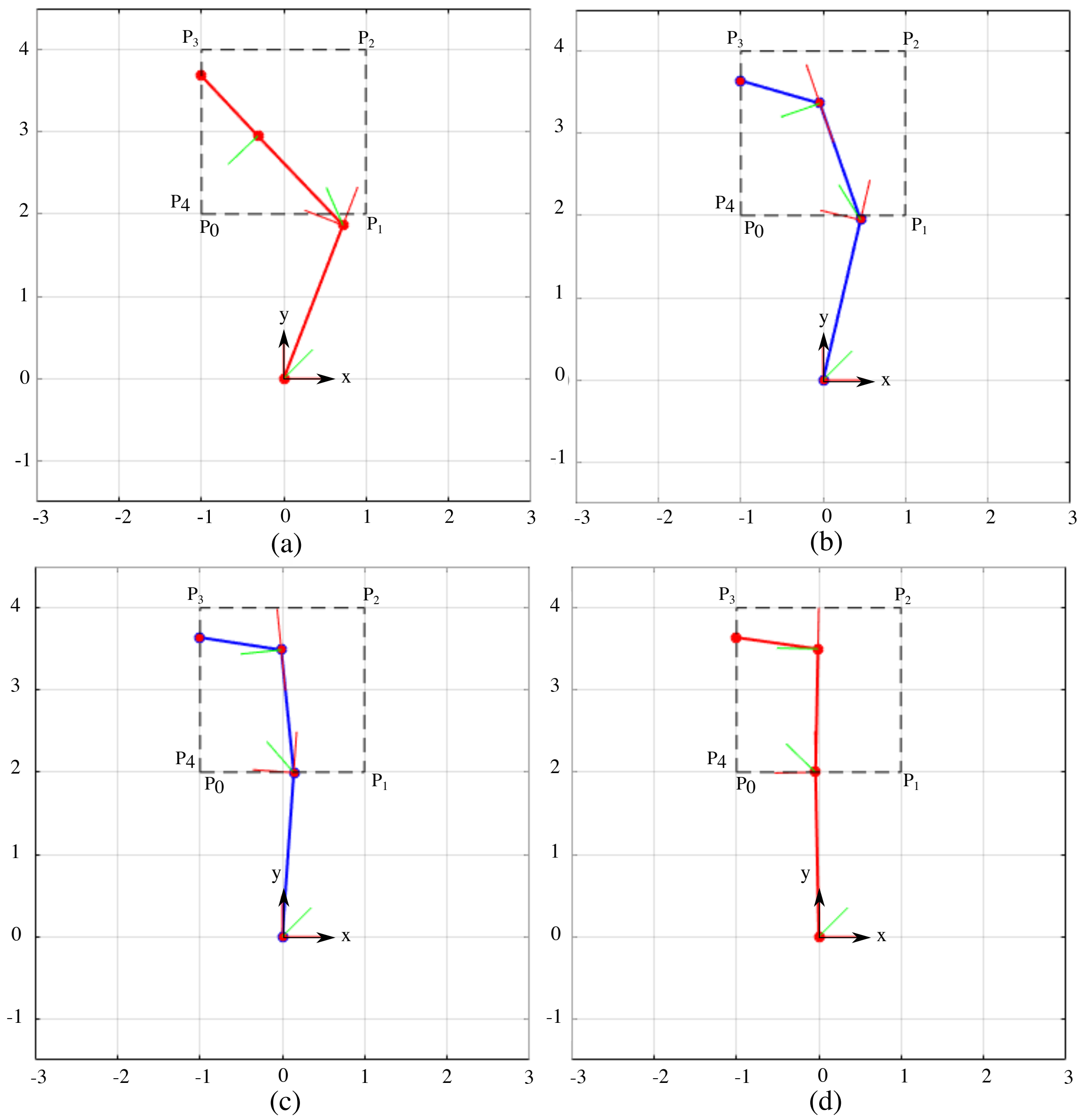


Figure 3.7 Simulation of the manipulator at 79% of the trajectory A: a) The lower bound – bounded to θ_3 limitation b, c) intermediate configurations d)The upper-bound– bounded to θ_1 and θ_2 limitations

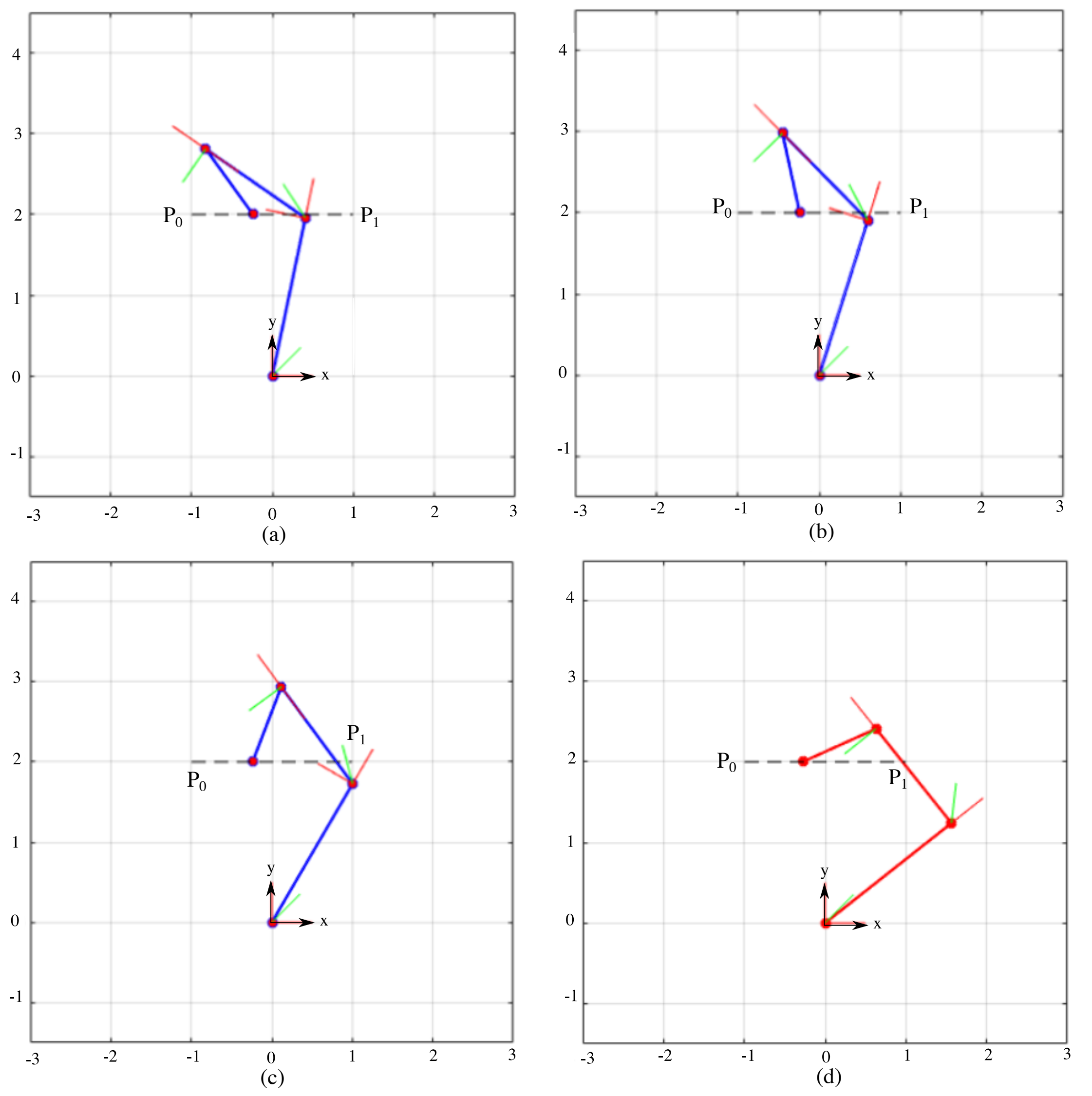


Figure 3.8 Simulation of the manipulator at 10% of the trajectory A: a)The bound limitation caused by singularity b, c) intermediate configurations d)The upper-bound caused by θ_2 limitations

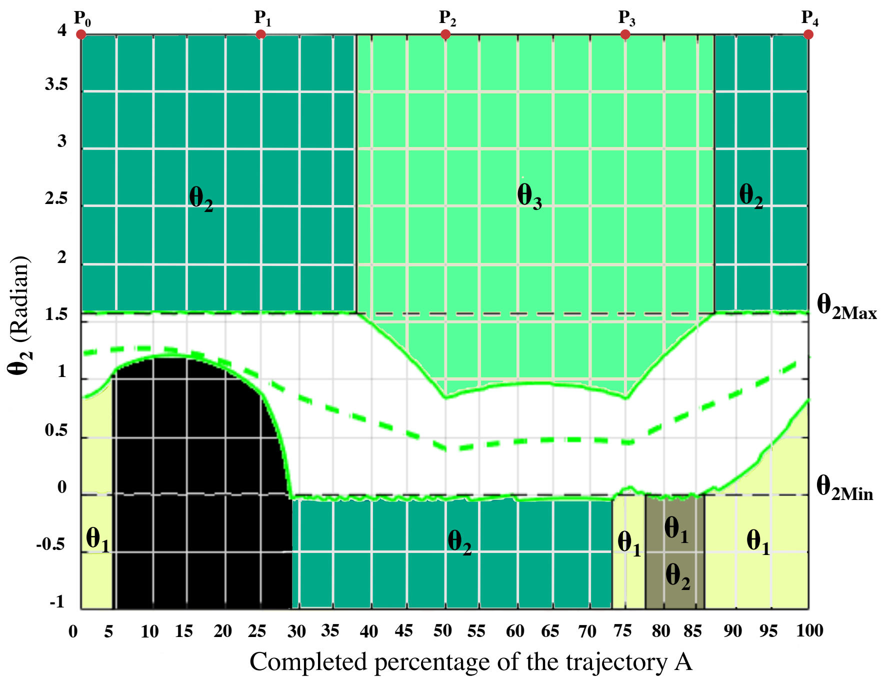


Figure 3.9 Feasible θ_2 region for trajectory A

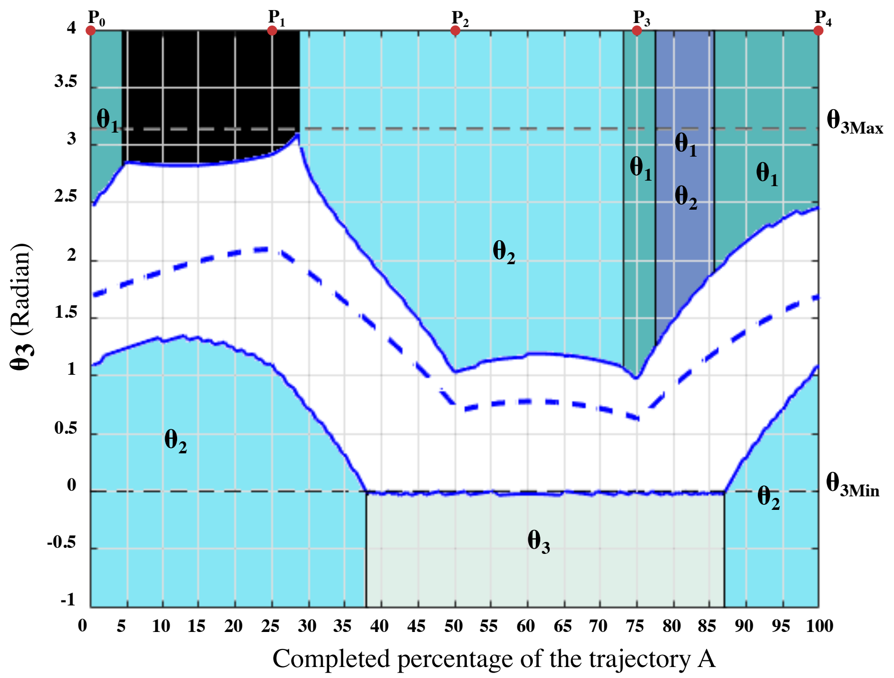


Figure 3.10 Feasible θ_3 region for trajectory A

3.3.2 Trajectory B: Rectangle

Figure 3.11 shows the trajectory B chosen to be more challenging with corners outside the reachable workspace of the manipulator. As it is shown the starting point of this trajectory is not in the workspace of the manipulator and at this point the limit of θ_1 will be passed. In table 3.3 the coordinates of the square trajectory are specified.

The manipulator moves from P_0 to the other Coordinates of the trajectory B at P_1 then P_2, P_3 and it finally returns to P_4 which is the same point as the starting point.

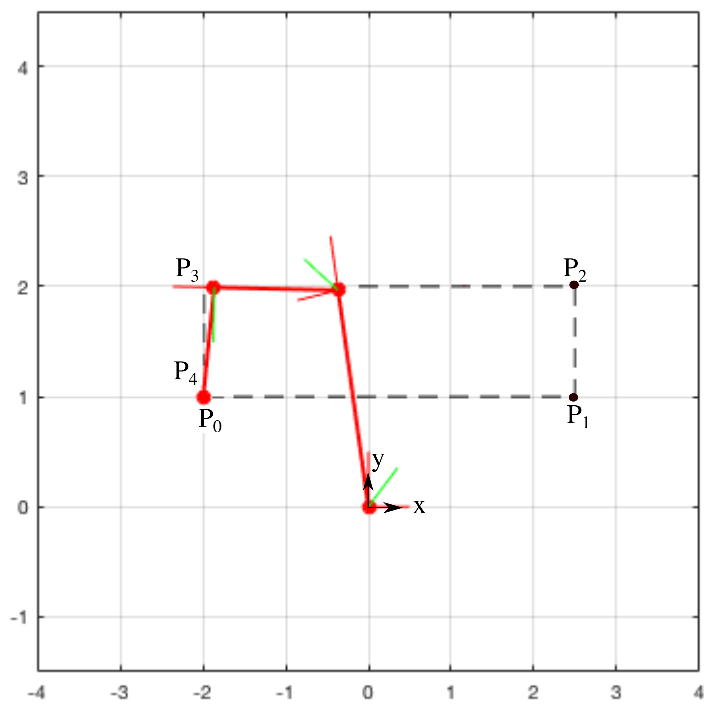


Figure 3.11 The manipulator configuration at P_0 in which θ_1 limitation is already exceeded. Black dash line is representing the trajectory and P_0 to P_4 are the trajectory corners

Table 3.3 Coordinates of trajectory B

P_i	P_0	P_1	P_2	P_3	P_4
x	-2	2.5	2.5	-2	-2
y	1	1	2	2	1

Figures 3.13, 3.14, 3.15 also show the bound limits and the possible region for θ_1, θ_2 and θ_3. As we can see, in many areas it is not possible for the manipulator to perform the trajectory

e.g. from simulation point 0 to 2.5%, 2.5% to 17%, 22% to 35%, 75% to 100s% in that θ_1, θ_2, θ_1 and θ_1 in order, are the limits to perform the trajectory. In figure 3.12 the manipulator pose at the simulation point around 19% to 22% of the path is shown. In this point the upper-bound (θ_1, θ_2, θ_2) is because of the singularity (figure 3.12.a) and the lower-bound is when $\theta_1 = 0$ is passed (figure 3.12.d).

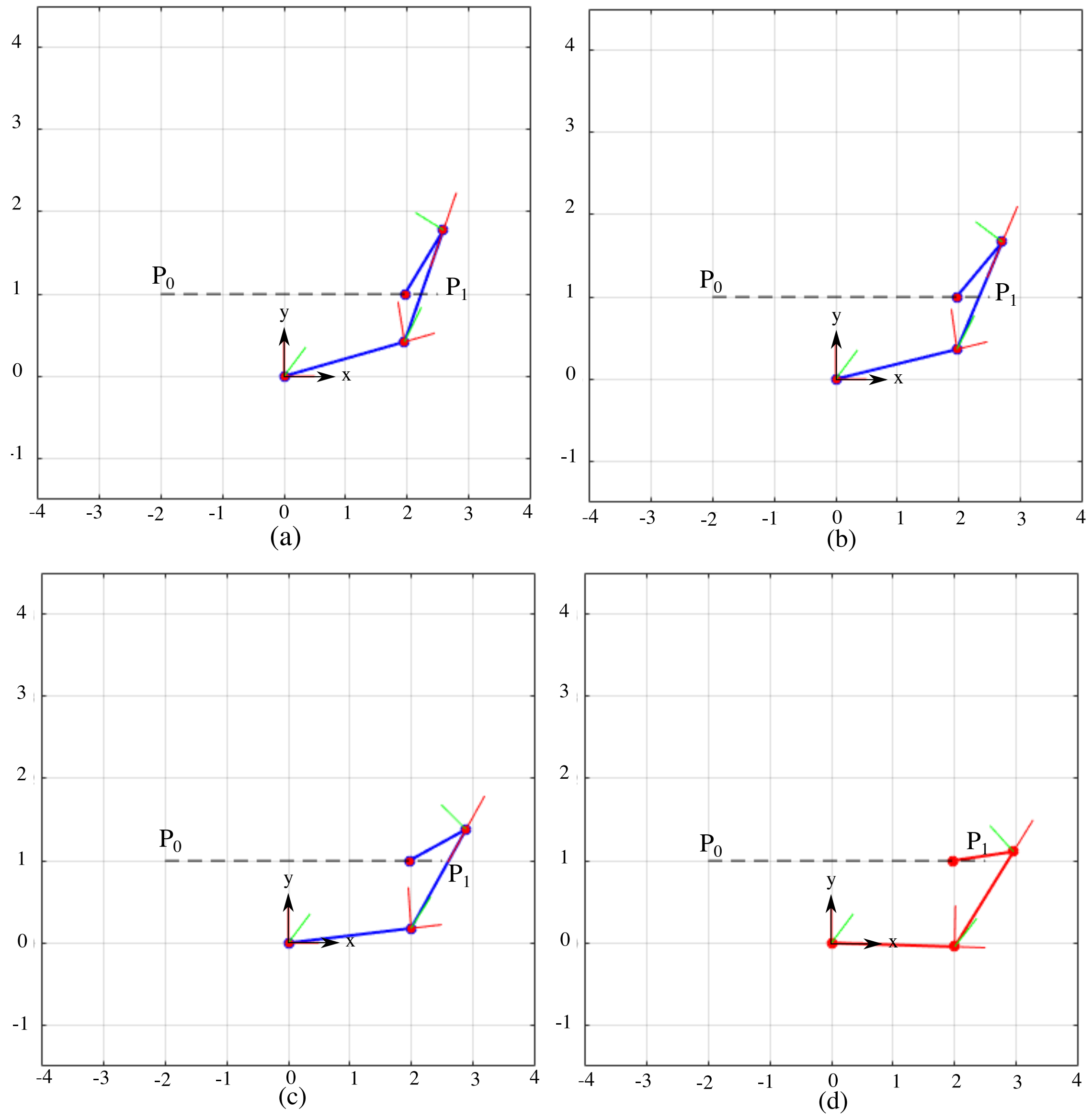


Figure 3.12 Simulation of the manipulator at 20% of the trajectory A: a)The bound limitation caused by singularity b, c) intermediate configurations d) The lower-bound caused by θ_1 limitation

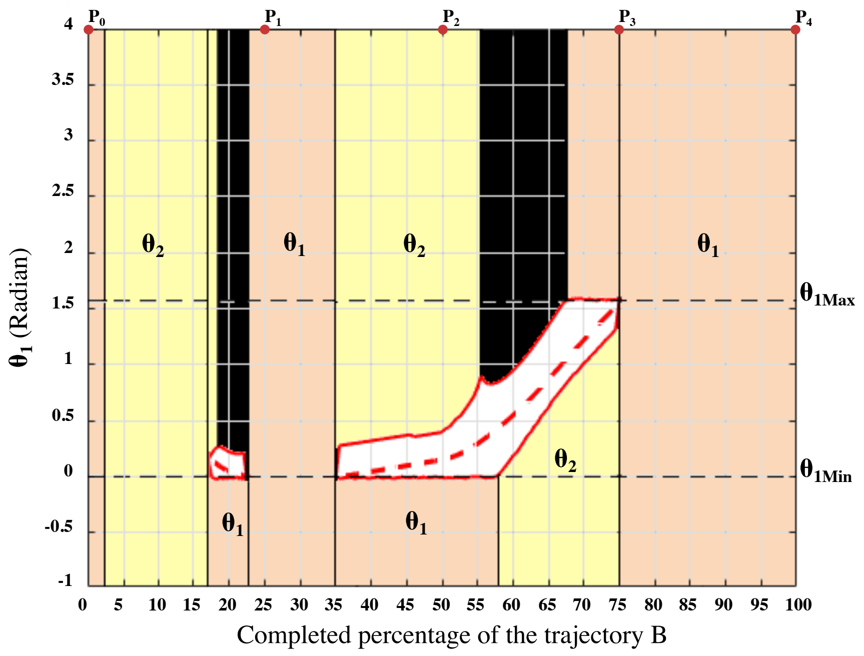


Figure 3.13 Feasible θ_1 region for trajectory B

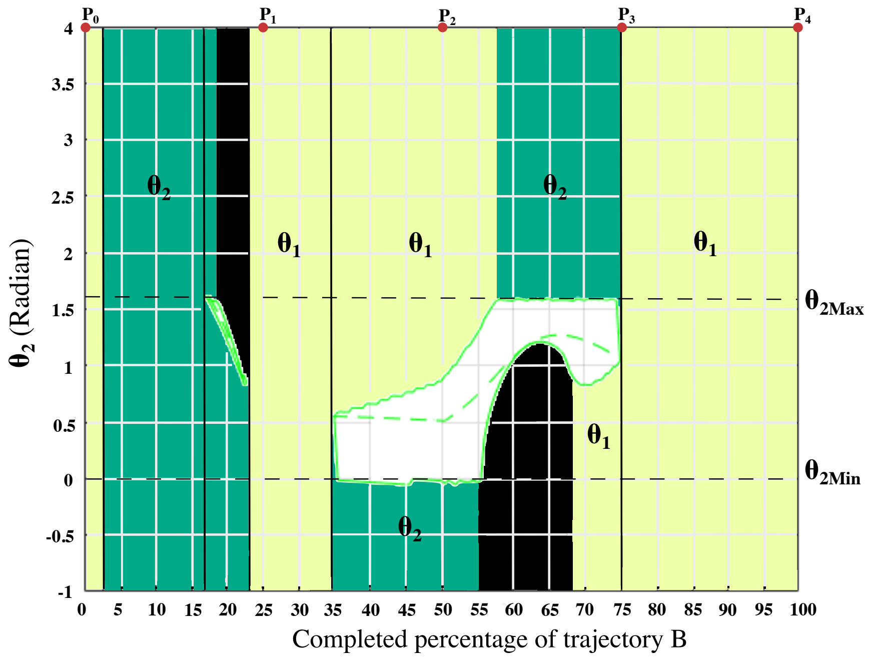


Figure 3.14 Feasible θ_2 region for trajectory B

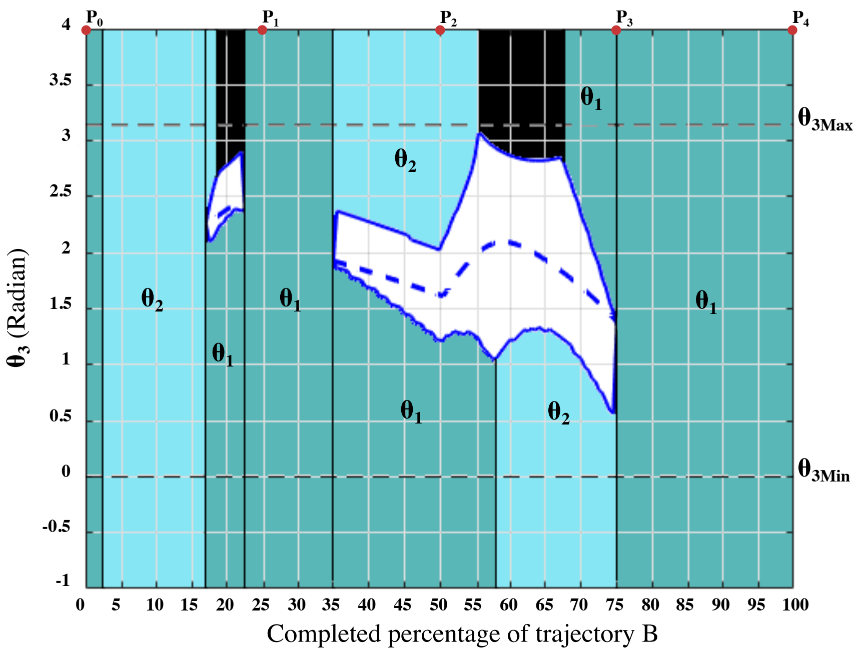


Figure 3.15 Feasible θ_3 region for trajectory B

3.3.3 Mid-point of the feasible region

In this section we investigate the correspondence between a point in the feasible region of θ_1 with these in θ_2 and θ_3 feasible region. Figures 3.16 show the corresponding points in θ_2 and θ_3 when we are at the mid-point of $theta_1$ at 61% of trajectory A. Apparently, the mid-point of θ_1 does not correspond to the mid-point of θ_2 and θ_3. Figures 3.17, 3.18, 3.19 also demonstrates middle point of θ_2 and θ_3 in comparison with the corresponding trajectory when we are at the middle point of θ_1 at all the points of trajectory A.

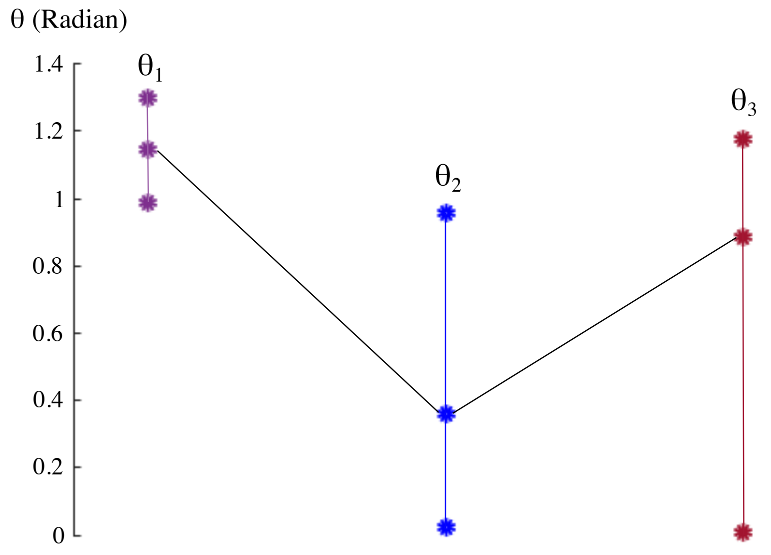


Figure 3.16 θ_1, θ_2 and θ_3 upper-bound, lower-bound, and their value at mid-point of θ_1 when we are at 61% of trajectory A

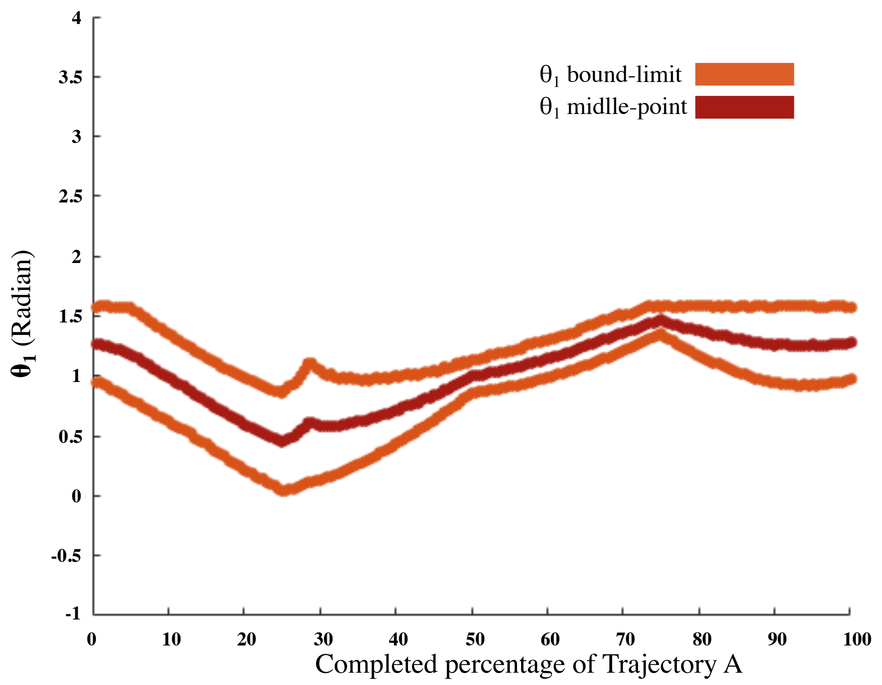


Figure 3.17 Trajectory A- Bound limits on θ_1- middle point of bound limit solution

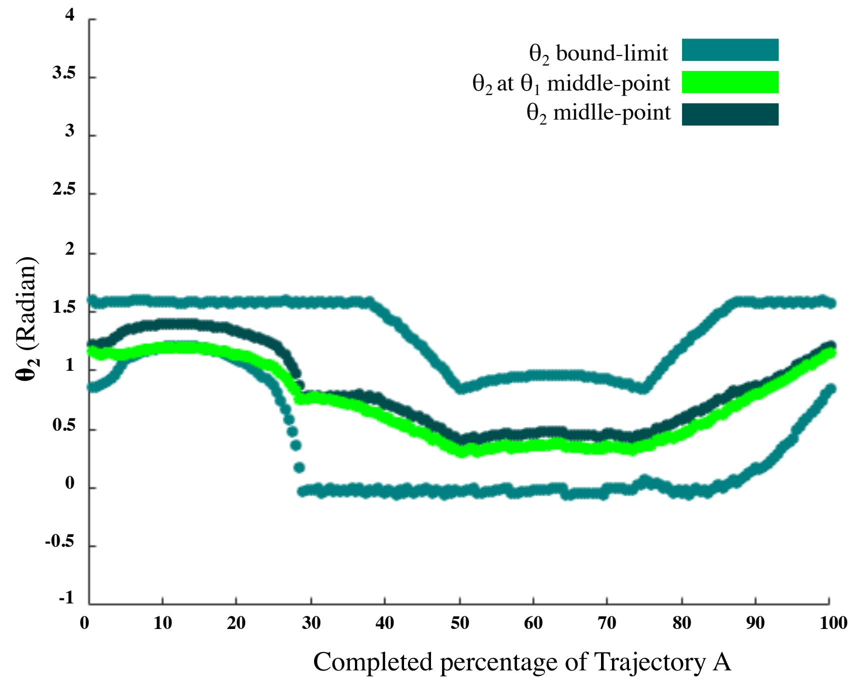


Figure 3.18 Trajectory A- Bound limits on θ_2- middle point of θ_2 bound limit versus middle point of θ_1 bound limit

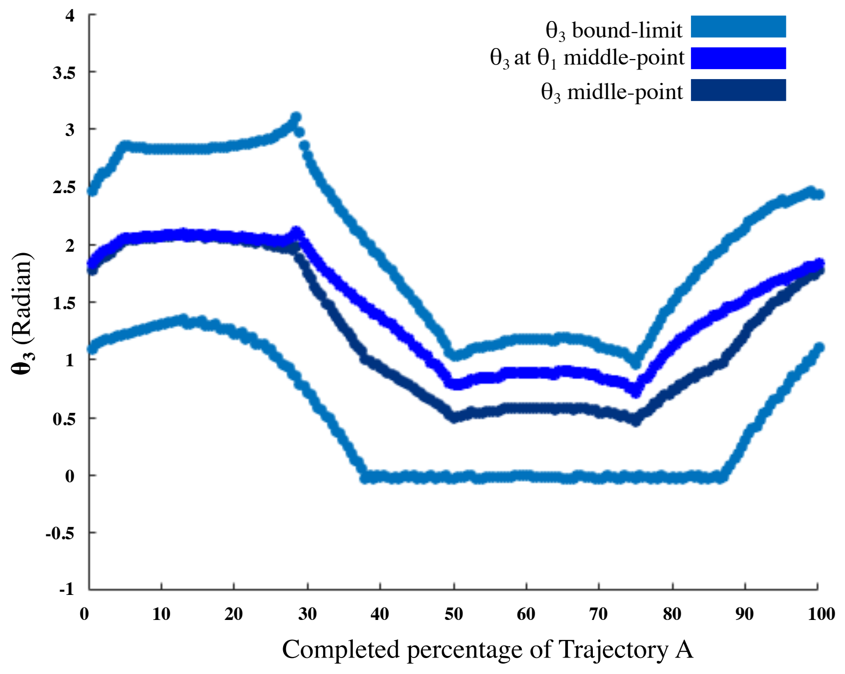


Figure 3.19 Trajectory A- Bound limits on θ_3- middle point of θ_3 bound limit versus middle point of θ_1 bound limit

3.4 Conclusion

In this chapter we have projected the joints limitations on each joint and that enabled us to also identify the joint limitations along the end-effector trajectory. We have also analyzed the middle-point of the joints bound-limits in more detail and its correspondence to the other two joints and we have concluded that although the mid-point of the bound-limit of one joint corresponds to the points within the bound limit of the other joints, it does not correspond to the middle-point of the other two joints. In the next chapter we will investigate the possibility of using this result in an AI system to predict and automate redundancy resolution.

CHAPITRE 4 ANFIS BASED REDUNDANCY RESOLUTION

4.1 Problem description

Computing feasible region for each given trajectory is a time consuming task. In this section we propose an Adaptive Network-based Fuzzy Inference System (ANFIS) as a predicting system to ease the process of finding the feasible region for a particular manipulator at each given trajectory. Figure 4.1 illustrates the process. It is also noteworthy to mention that in this method we only need to predict the feasible region for one joint and the limits of the other joints are computed in the feasible region.

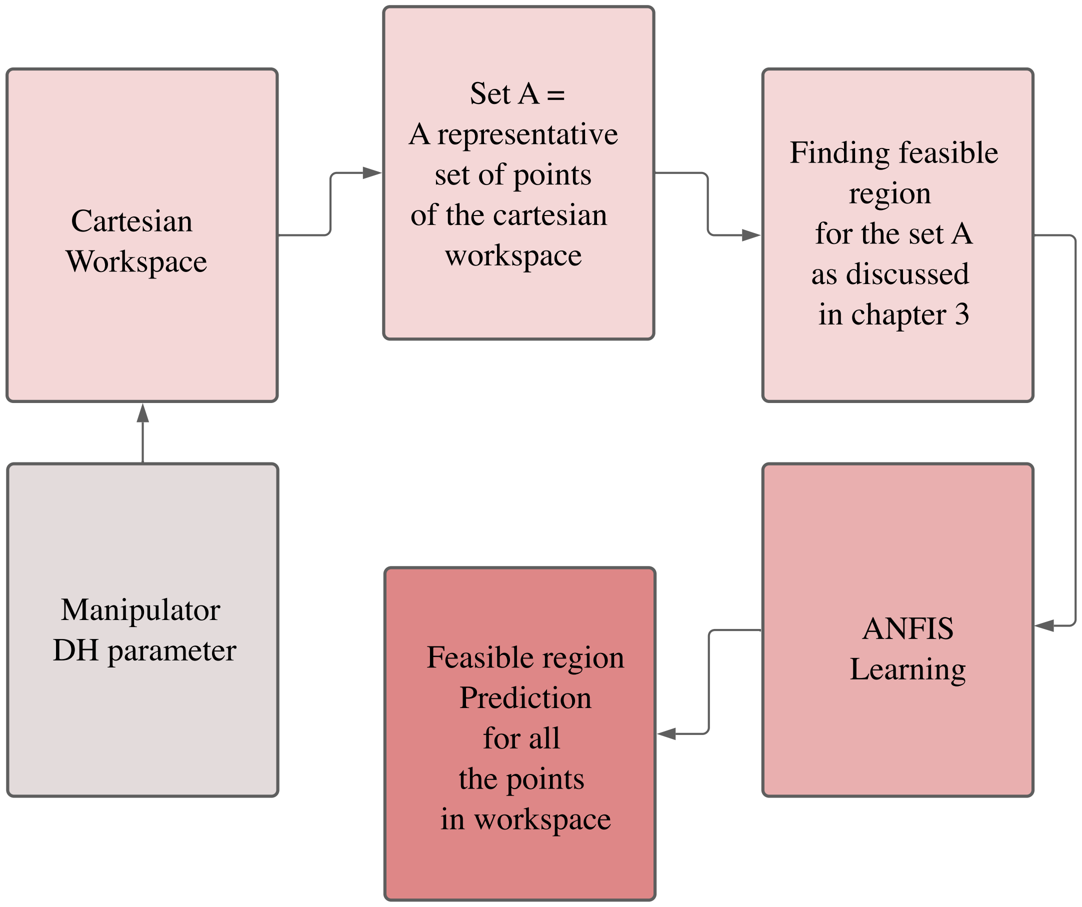

Set A =
A representative set of points of the cartesian workspace
Cartesian Workspace
Finding feasible region for the set A as discussed in chapter 3
Manipulator DH parameter
ANFIS Learning
Feasible region Prediction for all the points in workspace

Figure 4.1 The entire process of acquiring the feasible region for the workspace

4.2 ANFIS

Adaptive Network-based Fuzzy Inference System is an artificial neural network with hybrid learning procedure which combines the principle of fuzzy set theory and fuzzy inference systems with the learning ability of the neural networks [38].

ANFIS can be used for modeling nonlinear functions and predicting chaotic dynamical systems [39]. ANFIS structure consists of multiple layers. Figure 4.2 shows an example of the ANFIS structure. In our work three different ANFIS systems were trained for the prediction of θ_1-upper-bound, θ_1-lower-bound and θ_1-minimum-norm-solution.

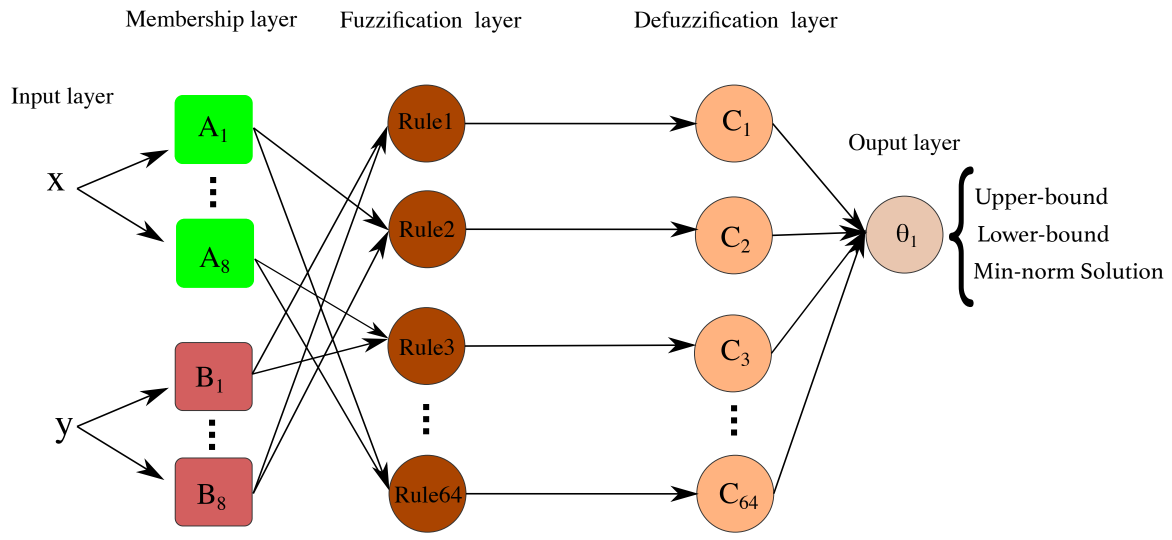


Figure 4.2 Graphical depiction of ANFIS structure

4.2.1 Input Layer

Input layer is the first layer of an ANFIS structure. In this layer the input variables are determined. In our work the input variables are the (x , y) Cartesian coordinates of the end-point position of our end-effector.

Data preparation

In order to prepare a data set for our ANFIS system, we have once scanned all the workspace of the learning manipulator with the same algorithm presented in chapter 3 to calculate upper limit, lower limit and min-norm solution for all the working space. In order to have more accurate result and avoiding from jumping over the workspace in the area that is not in the workspace, the scanning has been done in three separate areas which are shown in figure 4.3. For a better representation, the manipulator and joint limits are also shown in figure 4.3. After scanning the workspace, the ANFIS system was fed with the entire data set.

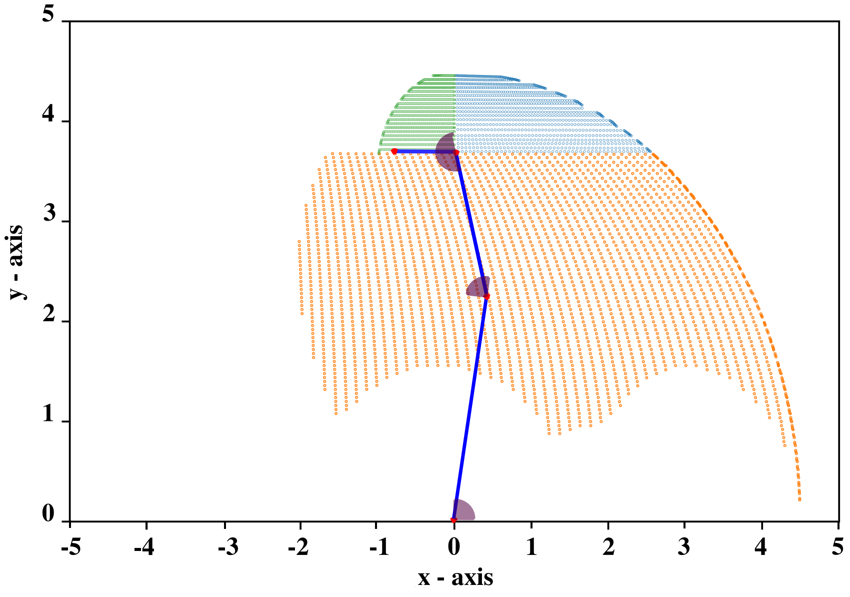


Figure 4.3 Workspace of the manipulator divided into three different areas in which our data set was prepared

4.2.2 Membership Layer

In the membership layer, each input is assigned a few number of membership functions. These membership functions help us to convey fuzzy sets visually [40]. The shape and the number of the membership functions plays an important role in ANFIS performance. However, there is no rule for the selection of these parameters and the selection is dependent upon the input data and the parameter choice requires expert knowledge. [41]. The membership function can be Sigmoidal z-shape, s-shape, triangular, Gaussian, and trapezoidal [42]. In our work, we have tried multiple membership functions and we find the best result using 8 Gaussian membership functions for each input. However, our research is not on finding the best ANFIS modeling but rather to show that pre-computed redundancy data set can be approximated by such AI algorithm.

Assigning degrees of membership

In the first step, each element of all inputs (x and y in our case) is assigned 8 degree of membership between 0 and 1 (since we have 8 membership functions for each input). Imagine we want to assign a degrees of membership to the input element (x_i) which belongs to the input x as follow:

$$x = x_1, x_2, ..., x_i \tag{4.1}$$

As explained in the last section, 8 fuzzy sets has been assigned to input x. These fuzzy sets are shown in figure 4.8 and are named as $MF1, MF2, MF3, ..., MF8$. If we consider the Gaussian membership functions in figure 4.4 to represent fuzzy set $MF4$ for example, then we can calculate a degree of membership ($\mu_{MF4}(x_i)$) for each element of input (x_i) as in equation 4.2 in which c is the center of the Gaussian function on the x-axis and $\gamma > 0$ is the standard deviation of the Gaussian function [40].

$$\begin{cases} \mu_A(x_i) = \exp\left(-\frac{(x_i-c)^2}{2(\gamma)^2}\right) \\ {\scriptstyle A=MF1,MF2\ ,\ \cdots,MF8} \end{cases} \tag{4.2}$$

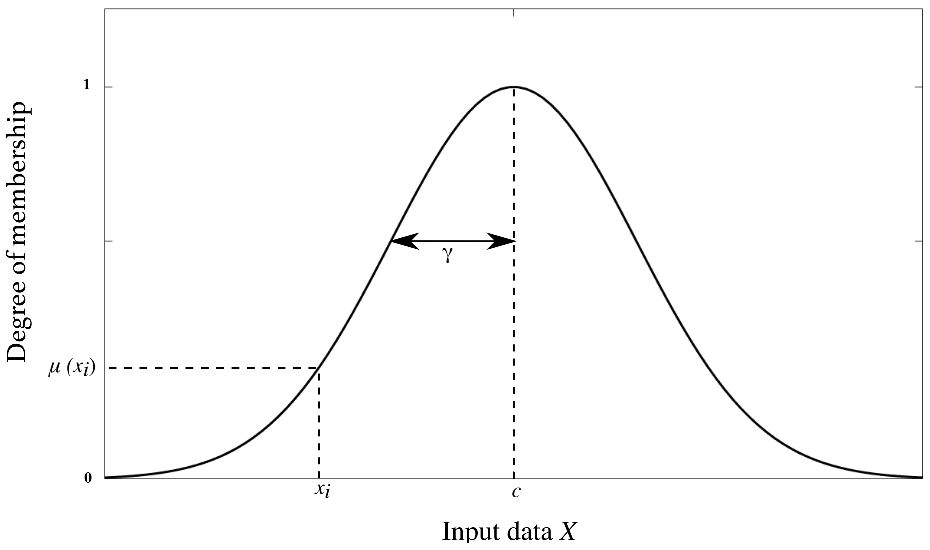


Figure 4.4 Gaussian membership function for x_i

4.2.3 Fuzzification Layer

In this layer, fuzzy rules are defined and fuzzy operators (AND/OR) are applied to the output of the previous step in order to produce the outputs. Two methods are commonly used for AND operator: minimum and product, while maximum and probor (probor(a,b) = a + b - ab) are used for OR operator [43]. A fazzy rule as shown in eq. 4.3:

$$if \underbrace{\ x\ is\ A_1\ AND/OR\ y\ is\ B_2\ }_{antecedent} Then \underbrace{\theta_1\ is\ C}_{consequent} \tag{4.3}$$

Where x and y are elements of the input sets, A_1 and B_2 and C are called linguistic values defined by fuzzy sets, and $_1$ is an element in the output set. The left side of the statement is called antecedent part and the right side is called consequent part.

4.2.4 Defuzzification Layer

Output membership function

There are two types of Fuzzy inference system based on the output membership functions: Mamdani type FIS and Sugeno type FIS. In Mamdani-type FIS output membership functions are defined to be fuzzy sets while in Sugeno-type FIS, membership functions are either defined linear or constant [43].In this work Fuzzy inference systems are considered to be Sugeno-type FIS since using ANFIS Toolbox in MATLAB, this toolbox only operates on this type of FIS [43]. The statement 4.4 shows the form of a if-then Sugeno-type fuzzy rule for a linear output [43]:

$$if\ x\ is\ A\ AND/OR\ y\ is\ B\ Then\ \theta_1 = \alpha x + \beta y + \gamma \tag{4.4}$$

Defuzzification process

In defuzzification layer, the consequent results of all the rules (C part in statement 4.3) either they are fuzzy sets, linear or constant, are combined and then a single crisp number is found for each output variable. This process is illustrated in figure 4.5.

Defuzzification methods

There are many methods for defuzzification: middle of maximum, bisector, centroid, weighted sum, weighted average (wtaver), etc [43]. In our work wtaver was used as the defuzzification

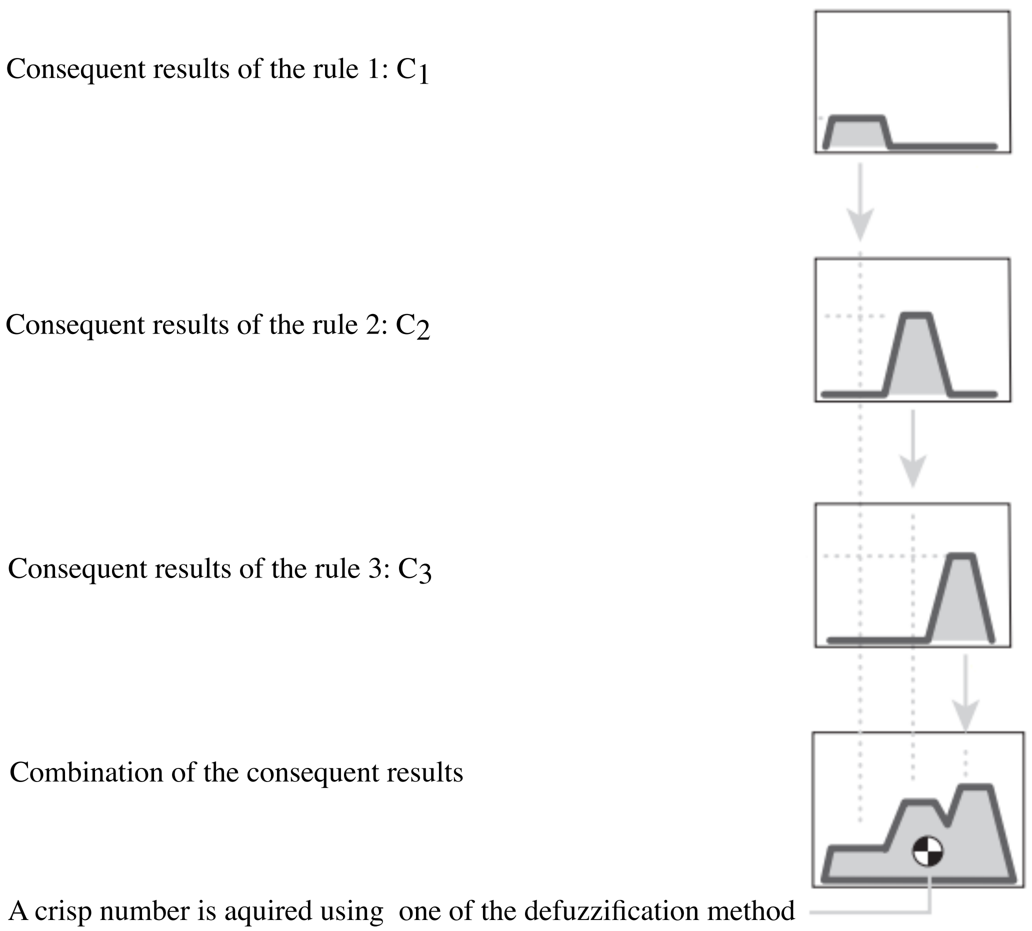

Consequent results of the rule 1: C1
Consequent results of the rule 2: C2
Consequent results of the rule 3: C3
Combination of the consequent results
A crisp number is aquired using one of the defuzzification method

Figure 4.5 Defuzzification process [43].

method. The equation 4.5 shows how in this technique [44] the final crisp number θ_1 is calculated .

$$\theta_1 = \frac{\Sigma\mu(\tilde{\theta}_1).\tilde{\theta}_1}{\Sigma\mu(\tilde{\theta}_1)} \tag{4.5}$$

Where Σ denotes the algebraic sum and it is applied for all the membership functions of an output, $\tilde{\theta}_1$ is the middle (centroid) of the membership function and $\mu(\theta_1)$ is the degree of membership of $\tilde{\theta}_1$.

4.3 ANFIS Toolbox in MATLAB

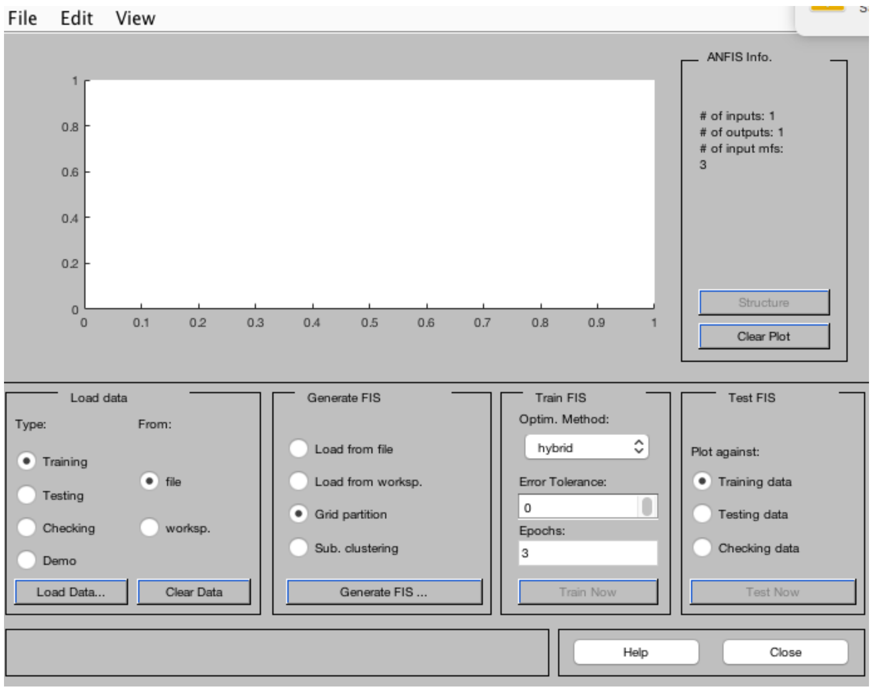
The Fuzzy Logic Toolbox in MATLAB with the ability to learn from the modeling input/output data is also called ANFIS. As mentioned in MATLAB booklet : "In general, this type of modeling works well if the training data presented to anfis for training membership function parameters is fully representative of the features of the data that the trained FIS is intended to model" [43].

4.3.1 Implementation in MATLAB

In order to work with ANFIS toolbox in MATLAB, one should write the "anfisedit()" command, then a window will open. Figure 4.6 shows the window. It can be seen in this picture that training and testing data can be loaded at the load-data part in the left bottom of the picture.Then, in the generate FIS section, the type of FIS can be selected. In our case, grid partition is selected. After selecting grid partition, by pressing generate FIS button another window opens. In figure 4.7 the new window is shown. In this figure we can select the number of membership function for each input, Membership type and weather the output is constant or linear. Figure 4.6 also shows that we can select optimization method, error tolerance and the number of epochs. After selecting all these parameter, by selecting the train- now button the ANFIS starts to be trained. After the training the plotting area can show the training error,training data, testing and checking data.

4.3.2 Training

In ANFIS toolbox the learning process consist of training the fuzzy system to determine and tune the best membership function parameters in order to enable the system to track the given input/output data [43]. In figure 4.8 the membership function of the three trained ANFIS system is illustrated. In this research work, 3356 data points was used for training and 200 data point was tested for each ANFIS systems.



ANFIS Info.
# of inputs: 1
# of outputs: 1
# of input mfs:
3

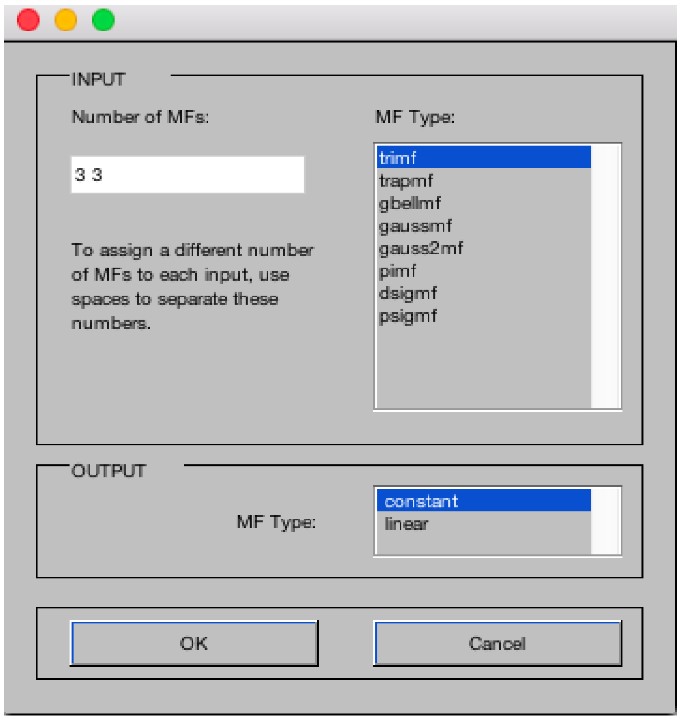
Load data
Type: [✓ Training] Testing Checking Demo
From: [✓ file] worksp.

Generate FIS
Load from file
Load from worksp.
[✓ Grid partition]
Sub. clustering

Train FIS
Optim. Method: hybrid
Error Tolerance: 0
Epochs: 3

Test FIS
Plot against: [✓ Training data] Testing data Checking data

Figure 4.6 ANFIS toolbox in MATLAB



INPUT
Number of MFs: 3 3

To assign a different number of MFs to each input, use spaces to separate these numbers.

MF Type: [✓ trimf] trapmf gbellmf gaussmf gauss2mf pimf dsigmf psigmf

OUTPUT
MF Type: [✓ constant] linear

Figure 4.7 ANFIS toolbox - generate FIS

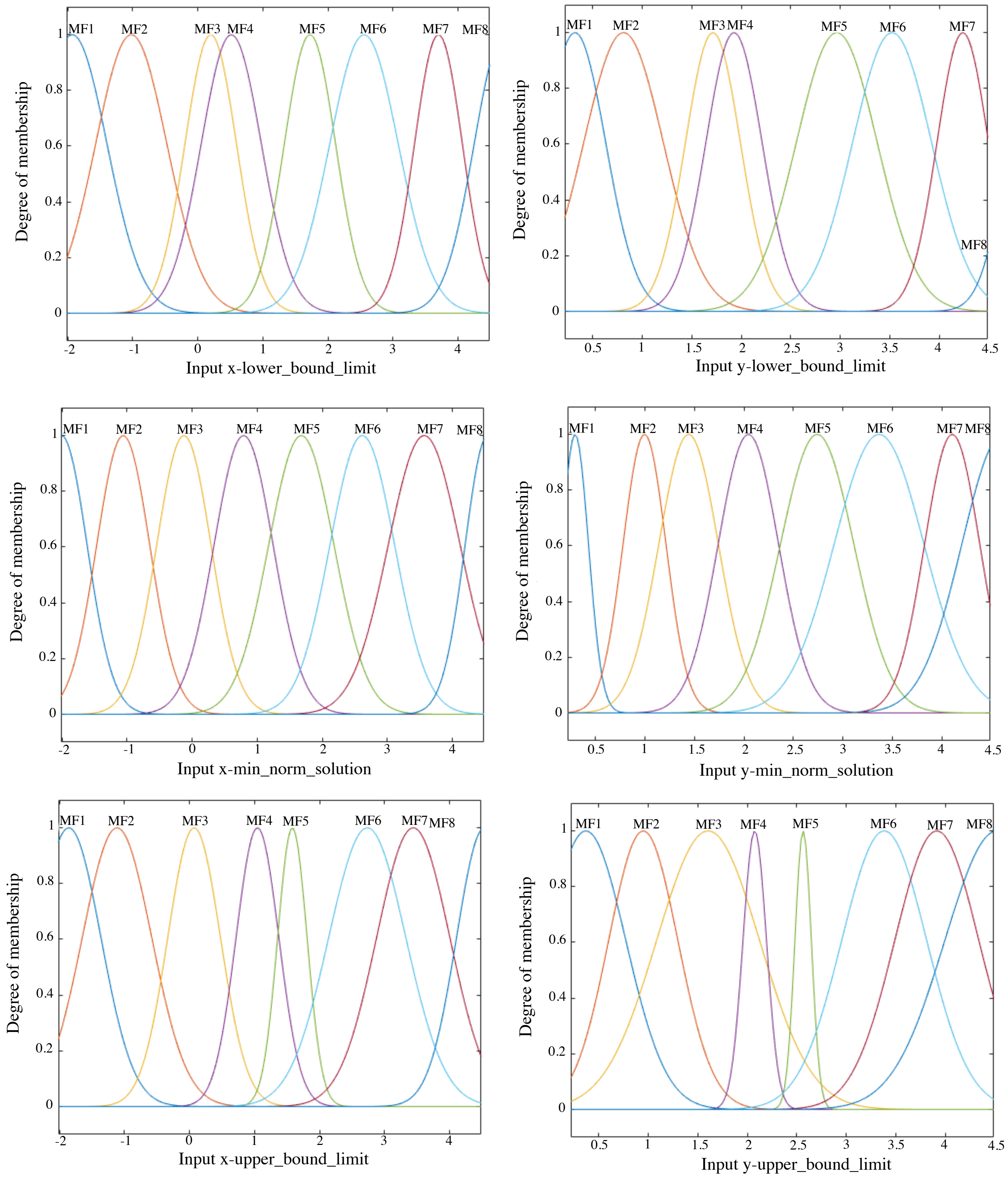


Figure 4.8 Membership functions for three trained ANFIS systems

Training optimization method

In the learning process gradient vector is used to measure how close is our modeling to input/output data pattern [43]. In the ANFIS toolbox the optimization is done using either back-propagation gradient descent method or a hybrid optimization method which is a combination of using back-propagation for the input membership functions parameters and least squares estimation for the output membership functions parameters [43].

Training epochs and error

Training epochs determines how many steps the learning process repeats to acquire more accurate modeling result. In figures 4.9, 4.10 and 4.11 the training error by epochs was shown for upper-bound limit, lower-bound limit and min-norm solution models.

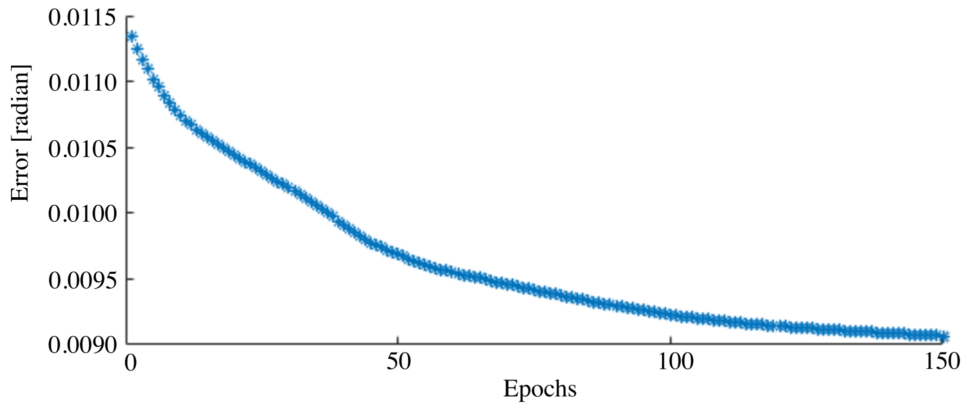


Figure 4.9 Training error upper-bound limit

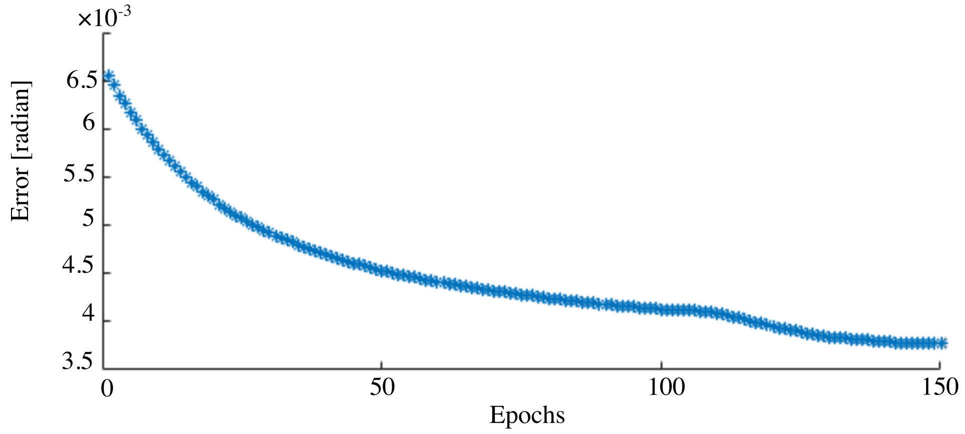


Figure 4.10 Training error lower-bound limit

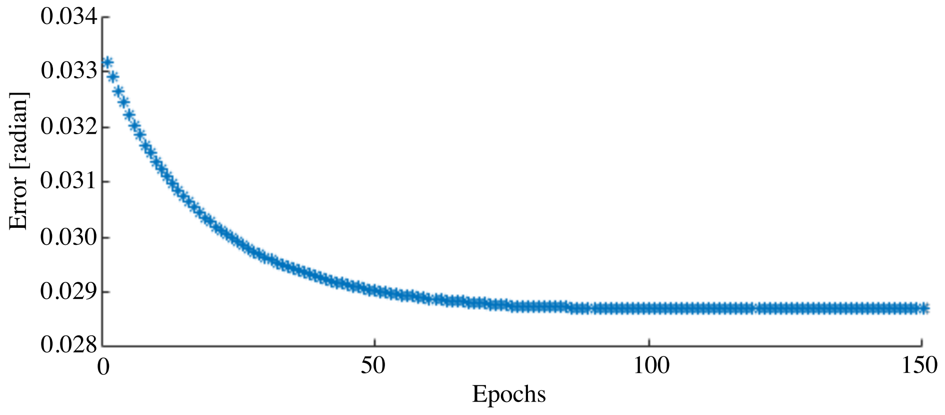


Figure 4.11 Training error min-norm solution

4.4 Validation tests

The trained ANFIS systems have been first tested for prediction of θ_1-upper-bound, θ_1-lower-bound and θ_1-minimum-norm-solution in trajectory A as presented in chapter 3.

Figure 4.12 shows the ANFIS toolbox prediction output and the testing data output upper and lower Bound limit. Figure 4.13 shows the ANFIS output and the testing data output for the minimum norm solution. Table 4.1 shows the ANFIS system rules and parameters used for each of the trained ANFIS system.

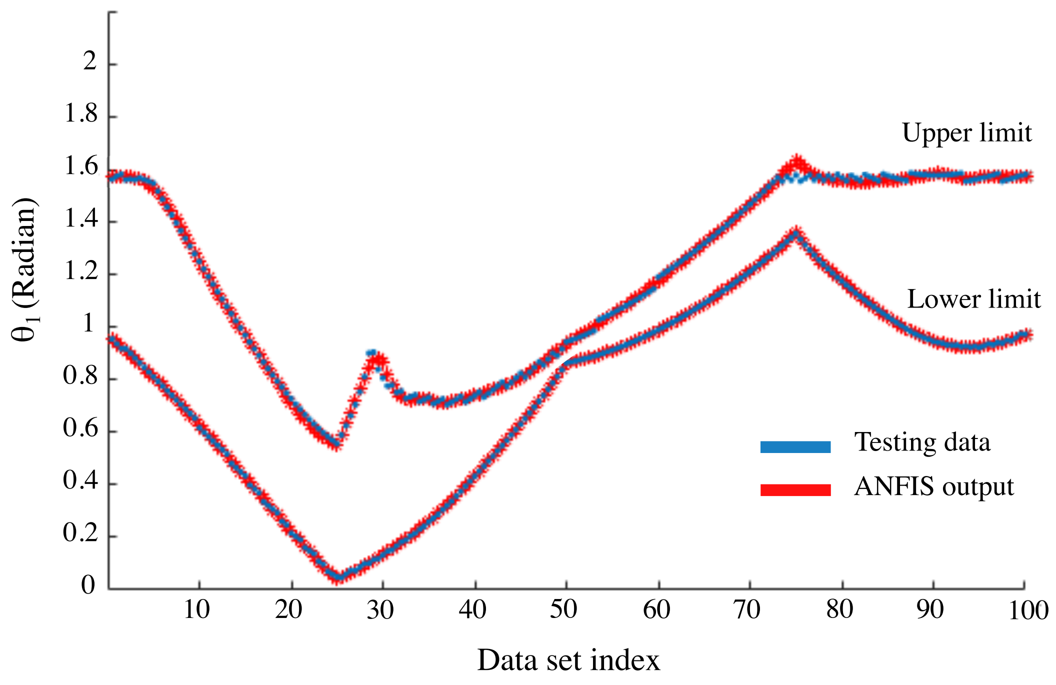


Figure 4.12 The ANFIS output for θ_1 upper-bound and lower-bound comparison with the testing data output upper-bound and lower-bound

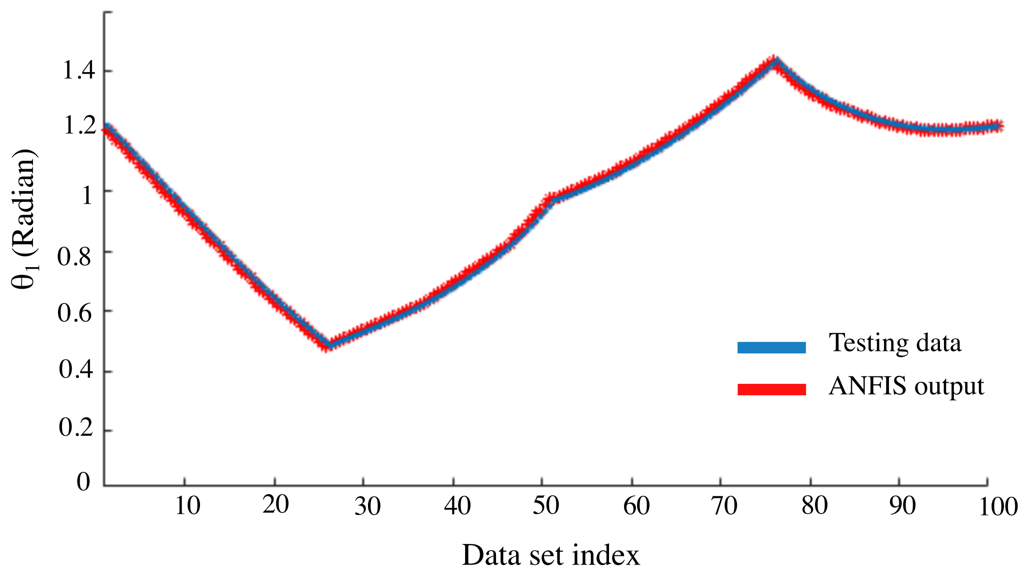


Figure 4.13 Minimum-norm solution, ANFIS output comparison with testing data

Table 4.1 ANFIS Parameters

	UPPER-bound	LOWER-bound	MIN-NORM solution
No. of MF for input x	8	8	8
MF type for input x	Gaussian	Gaussian	Gaussian
No. of MF for input y	8	8	8
MF type for input y	Gaussian	Gaussian	Gaussian
Output MF type	Linear	Linear	Linear
FIS type	Sugeno	Sugeno	Sugeno
FIS input	(x , y)	(x , y)	(x , y)
FIS output	θ_1-upper-bound	θ_1-lower-bound	θ_1-min-norm-solution
FIS AND Method	prod	prod	prod
FIS OR Method	probor	probor	probor
FIS defuzzification Method	wtaver	wtaver	wtaver
No.of FIS rule	64	64	64
ANFIS optimization method	hybrid	hybrid	hybrid
ANFIS Training epochs	150	150	150

4.5 Utilization of the ANFIS result for solving of the IK and redundancy resolution

Having the upper-limit and lower-limit of all the joints, we can select the joint which is less relaxed (θ_1 in our case) to choose as our reference to solve the inverse kinematic since by knowing one of the joints trajectory we would have enough equations to find other joints trajectory as we only have one degree of redundancy. Although any joint trajectory for θ_1 within the lower and upper limit is possible we decided to choose the middle point between lower and upper limit to be our reference for solving the inverse kinematic problem. It was also discussed in the section 3.3.3 that although middle point of θ_1 does not correspond to the middle point of other joints, certainly the answer would still be in the feasible joint area. As it is shown in the previous section the upper-limit and lower-limit of θ_1 was predicted precisely by ANFIS. Having this result, we are able to solve the inverse kinematic equations. Figure4.14 represents the entire process. The learning part of the process is off-line while calculating the Inverse kinematic part is an online process. ANFIS can also be utilized to learn minimum-norm solution at all points of the Cartesian workspace. As it is shown in previous section, the minimum-norm solution was predicted by ANFIS precisely. Having this result, we are able to automate solving the inverse kinematic problem without a need for calculating the pseudo-inverse of the each time preforming a new Cartesian trajectory. It is note worthy to mention that using an artificial intelligent system like ANFIS helps us using a limited memory to keep the result (ANFIS system) whereas using a look-up table and keeping all the data -specially when we are dealing with a large workspace and/or in 3 dimensional space- needs a higher memory space. Thus, although it does not make a big difference in our planer robot, this research was intended to test the capacity of ANFIS system for further utilization. In the next section we will go through all this steps for a new Cartesian trajectory as a validation result for ANFIS selection.

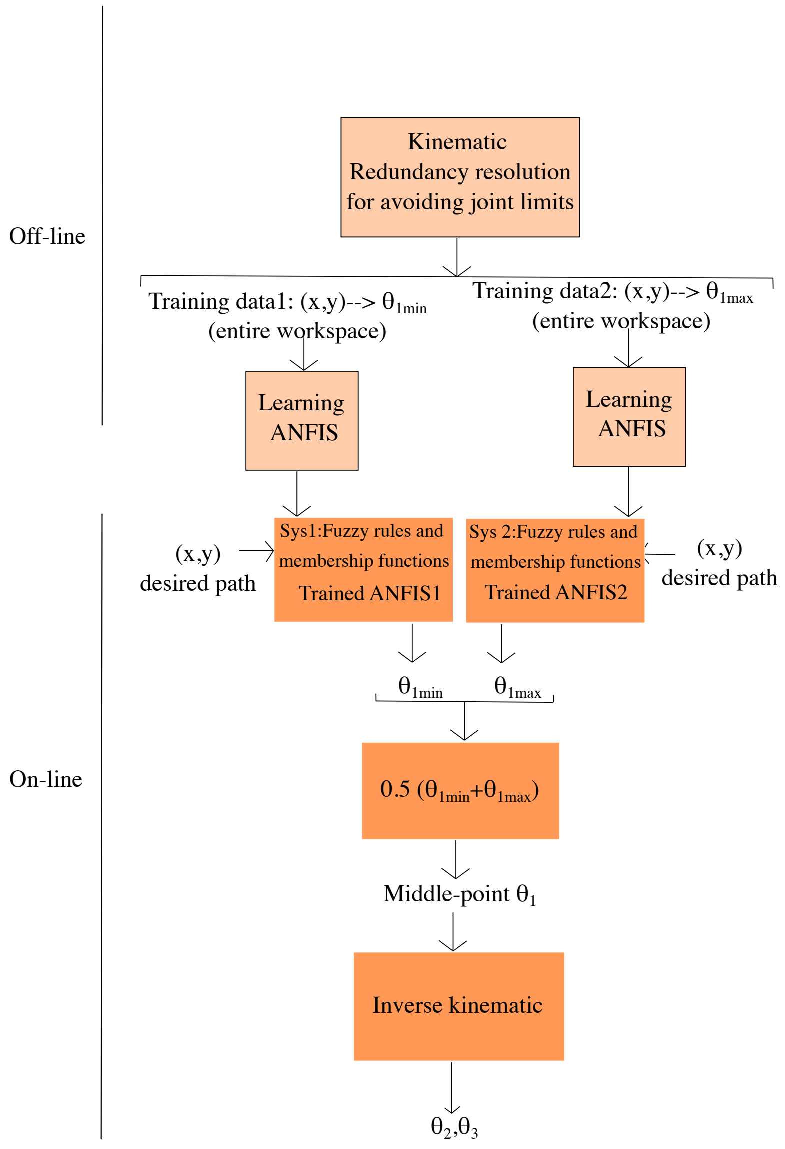

Off-line
On-line
Kinematic Redundancy resolution for avoiding joint limits
Training data1: (x,y)--> θ1min (entire workspace)
Training data2: (x,y)--> θ1max (entire workspace)
Learning ANFIS
Learning ANFIS
(x,y) desired path
Sys1:Fuzzy rules and membership functions Trained ANFIS1
Sys 2:Fuzzy rules and membership functions Trained ANFIS2
(x,y) desired path
θ1min
θ1max
0.5 (θ1min+θ1max)
Middle-point θ1
Inverse kinematic
θ2,θ3

Figure 4.14 The entire process of using ANFIS in Kinematic redundancy resolution

4.5.1 Validation result

In order to validate the result we have chosen an end effector trajectory C within the workspace. The trajectory goes from point P_0 to P_3, and P_0 and P_3 are located at the same location as shown in Fig. 4.15.

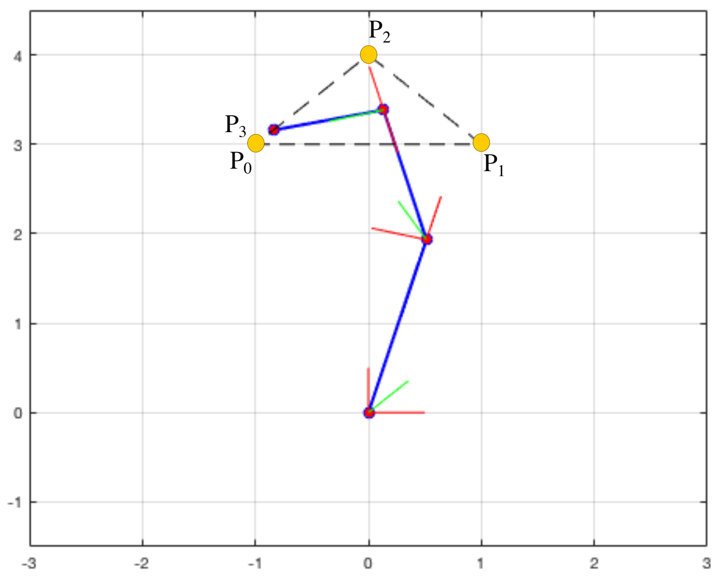


Figure 4.15 Black dash line is representing the trajectory and P_0 to P_3 are the trajectory corners

The same ANFIS system that was used in the previous section was employed for this Cartesian trajectory to select θ_1 trajectory. The selected θ_1 trajectory is the middle-point of lower-bound and upper-bound of θ_1 as it is depicted in Figure 4.16.

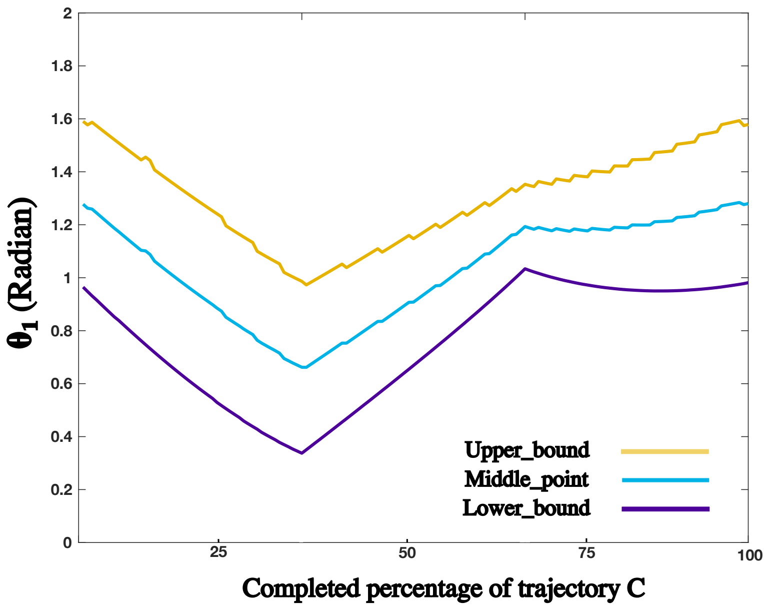


Figure 4.16 Upper-bound, lower-bound and middle-point of θ_1 at the Cartesian trajectory C

Having θ_1, the other two joints θ_2 and θ_3 are calculated as shown in Figure 4.17 and 4.18. As you can see in these figures the limitations of all the joints are respected as expected.

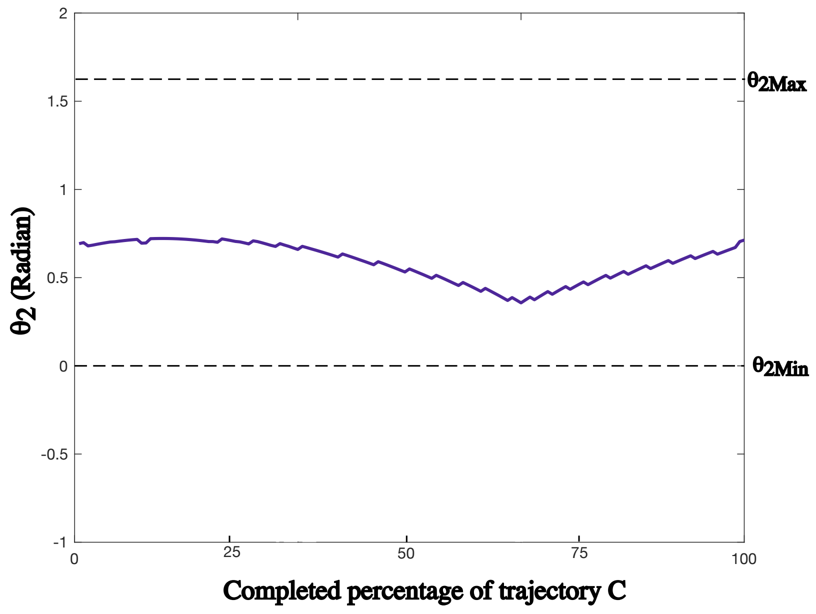


Figure 4.17 Calculated θ_2 based on the selected θ_1 joint trajectory

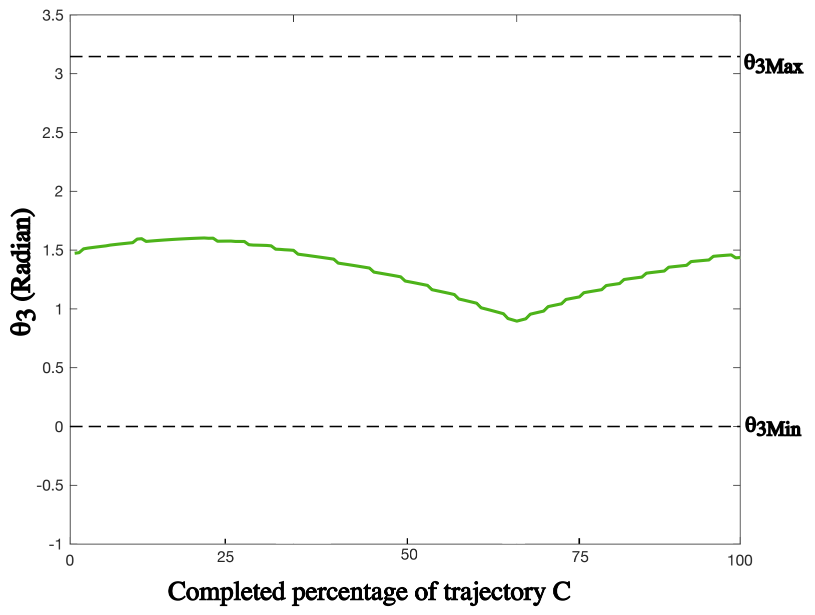


Figure 4.18 Calculated θ_3 based on the selected θ_1 joint trajectory

4.6 Validation result conclusion

The validation test has successfully shown the effectiveness of the technique and ANFIS system can be utilized to learn the workspace data and provide us with an acceptable result for each given path inside the workspace.

CHAPITRE 5 CONCLUSION

Based on the result of our simulation, we can conclude that when we are planning a joint motion trajectory, is it possible to use ANFIS modeling to help us in finding an exact inverse kinematic solution by taking advantage of the redundancy resolution. The consideration of the joint limitation, is also feasible and the joint trajectory can be selected within the bound limits of one of the desired joint e.g. middle point of θ_1. This middle point is not located at the middle point of other joints however it satisfies the limitations of other joints, since we have projected the joint limits on each individual joins and acquired the whole joint space in chapter 3. We could also used this result in our AI system to predict and automate redundancy resolution. In addition, the joint that limits the motion at each point of the end-effector trajectory was identified. In short, the proposed technique can help us find an exact inverse kinematic solution for a redundant manipulator while satisfying the limitation of the joints. In other words, this research makes a technical contribution by:

1- Presenting the projection of redundant feasible space of the manipulator on the each individual joint and

2-Examining the possibility of using an AI algorithm to learn the pre-calculated limits and predict it for any given task.

5.1 Summary of Works

Redundancy of the manipulator gave us the possibility to take advantage of the additional degree of freedom to identify all the possible region in joint space by proposing an algorithm based on pseudoinverse technique to perform a self-motion. The manipulator performed a self-motion while keeping the end-effector at each point of the path and recorded the angle in which the limitation of each joint was passed the algorithm has also record minimum norm solution at each point of the workspace. Then the result is fed to an ANFIS system to be learned. Once the maximum and minimum of the possible region is learned, we were able to use the middle point to solve the inverse kinematic equation. We are also able to use the minimum norm solution to automate kinematic inversion by training three different ANFIS system for each joint.

5.2 Future Research

Clearly further studies are needed to expand our understanding and, other techniques can be examined. Here are some of the future possible research proposition:

— The proposed technique can be simulated for a higher DOF manipulator.

— A systematic selection technique will be needed to investigated and select the best ANFIS structure. (e.g. number of membership function, type of membership function, output type, optimization method, etc)

— Other artificial intelligent algorithm can also be examined.

— An experimental study can be conducted to examine the technique in actual practice.

REFERENCES

[1] I. Bonev. (2007, 2) General terminology related to parallel mechanisms. [Online]. Available: https://www.parallemic.org/Terminology/General.html

[2] What is a robot manipulator? [Online]. Available: https://www.robots.com/faq/what-is-a-robot-manipulator

[3] L. Baron, "Contributions to the estimation of rigid-body motion under sensor redundancy," Ph.D. dissertation, Department of Mechanical Engineering , McGill University, 1998.

[4] X. Wang and L. Baron, *Topology and Geometry of Serial and Parallel Manipulators*, 04 2008.

[5] A. Aristidou and J. Lasenby, "Inverse kinematics: a review of existing techniques and introduction of a new fast iterative solver," 2009.

[6] T. P. Singh, D. P. Suresh, and D. S. Chandan, "Forward and inverse kinematic analysis of robotic manipulators," *International Research Journal of Engineering and Technology*, vol. 04, pp. 1459–1469, 2017.

[7] L. Huo, "Inverse kinematics of robotized functionally and intrinsically redundant tasks," Ph.D. dissertation, Departement de Genie Mecanique Ecole Polytechnique de Montreal, 2006.

[8] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Transactions on Man-Machine Systems*, vol. 10, no. 2, pp. 47–53, 1969.

[9] A. Palmeri, G. Chen, L. Zhang, Q. Jia, and H. Sun, "Singularity analysis of redundant space robot with the structure of canadarm2," *Mathematical Problems in Engineering*, vol. 2014, p. 735030, 06 2014.

[10] P. Donelan, "Singularity-theoretic methods in robot kinematics," *Robotica*, vol. 25, pp. 641–659, 11 2007.

[11] I. Bonev. (2019, 08) What are singularities in a six-axis robot arm? [Online]. Available: https://www.mecademic.com/en/what-are-singularities-in-a-six-axis-robot-arm

[12] E. S. Conkur, "Path planning using potential fields for highly redundant manipulators," *Robotics and Autonomous Systems*, vol. 52, no. 2, pp. 209–228, 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889005000552

[13] S. Assal, K. Watanabe, and K. Izumi, "Neural network-based kinematic inversion of industrial redundant robots using cooperative fuzzy hint for the joint limits avoidance," *IEEE/ASME Transactions on Mechatronics*, vol. 11, no. 5, pp. 593–603, 2006.

[14] S. Yahya, M. Moghavvemi, and H. Mohamed, "Redundant manipulators kinematics inversion." *Scientific Research and Essays*, vol. 6, pp. 5462–5470, 11 2011.

[15] A. Jasour and M. Farrokhi, "Path tracking and obstacle avoidance for redundant robotic arms using fuzzy nmpc," 07 2009, pp. 1353 – 1358.

[16] L. Huo and L. Baron, "The joint-limits and singularity avoidance in robotic welding," *Industrial Robot: An International Journal*, vol. 35, 08 2008.

[17] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 7, no. 12, pp. 842–868, 1977.

[18] N. N. Dragomir, "Redundancy resolution through local optimization: A review," *Journal of Robotic Systems*, vol. 6, no. 6, 1989.

[19] Y. Nakamura, *Advanced Robotics:Redundancy and Optimization*. Addison-Wesley, 1991.

[20] A. Liégeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," 1977.

[21] R. Dubey, J. Euler, and S. Babcock, "Real-time implementation of an optimization scheme for seven-degree-of-freedom redundant manipulators," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 5, pp. 579–588, 1991.

[22] Y. Yang, G. Peng, Y. Wang, and H. Zhang, "A new solution for inverse kinematics of 7-dof manipulator based on genetic algorithm," in *2007 IEEE International Conference on Automation and Logistics*, 2007, pp. 1947–1951.

[23] H. A. F. Mohamed, S. Yahya, M. Moghavvemi, and S. S. Yang, "A new inverse kinematics method for three dimensional redundant manipulators," in *2009 ICCAS-SICE*, 2009, pp. 1557–1562.

[24] Y. Zhang, J. Wang, and Y. Xia, "A dual neural network for redundancy resolution of kinematically redundant manipulators subject to joint limits and joint velocity limits," *IEEE Transactions on Neural Networks*, vol. 14, no. 3, pp. 658–667, 2003.

[25] Y. Zhang and J. Wang, "Obstacle avoidance for kinematically redundant manipulators using a dual neural network," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, pp. 752–759, 2004.

[26] R. Palm, "Collision avoidance of a redundant manipulator using fuzzy rules," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 1992, pp. 1719–1726.

[27] S.-W. Kim and J.-J. Lee, “Resolved motion rate control of redundant robots using fuzzy logic,” in *[Proceedings 1993] Second IEEE International Conference on Fuzzy Systems*, 1993, pp. 333–338 vol.1.

[28] G. M. Kun, W. D. Wei, and S. Y. Chai, “A fuzzy logic approach for kinematic control of redundant manipulators,” in *Proceedings 1993 Asia-Pacific Workshop on Advances in Motion Control*, 1993, pp. 94–99.

[29] M. Beheshti and A. Tehrani, “Obstacle avoidance for kinematically redundant robots using an adaptive fuzzy logic algorithm,” in *Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251)*, vol. 2, 1999, pp. 1371–1375 vol.2.

[30] L. Das, J. Nanda, and S. Mahapatra, “A comparative study of prediction of inverse kinematics solution of 2-dof, 3-dof and 5-dof redundant manipulators by anfis,” *IJCSN International Journal of Computer Science and Network, Volume 3, Issue 5, October 2014 ISSN (Online) : 2277-5420 www.IJCSN.org*, 10 2014.

[31] D. Deshmukh, D. K. Pratihar, A. K. Deb, H. Ray, and A. Ghosh, “Anfis-based inverse kinematics and forward dynamics of 3 dof serial manipulator,” in *Hybrid Intelligent Systems*, A. Abraham, T. Hanne, O. Castillo, N. Gandhi, T. Nogueira Rios, and T.-P. Hong, Eds. Cham: Springer International Publishing, 2021, pp. 144–156.

[32] S. Manjaree, V. Agarwal, and B. Nakra, “Kinematic analysis using neuro fuzzy intelligent technique for robotic manipulator,” *International Journal of Engineering Research and Technology*, vol. 6, no. 4, pp. 557–562, 2013.

[33] A.-V. Duka, “Anfis based solution to the inverse kinematics of a 3dof planar manipulator,” *Procedia Technology*, vol. 19, pp. 526–533, 12 2015.

[34] S. Alavandar and M. J. Nigam, “Inverse kinematics solution of 3dof planar robot using anfis,” *International Journal of Computers, Communications and Control*, vol. 3, 01 2008.

[35] N. Vu, N. Tran, and N. Nguyn, “Adaptive neuro-fuzzy inference system based path planning for excavator arm,” *Journal of Robotics*, vol. 2018, pp. 1–7, 12 2018.

[36] (2021). [Online]. Available: https://www.robotmaster.com/en

[37] I. Lankham, B. Nachtergaele, and A. Schilling, *Linear Algebra As an Introduction to Abstract Mathematics Lecture Notes for MAT67*, 11 2016.

[38] Y. Kurtgoz and E. Deniz, “Chapter 1.8 - comparison of ann, regression analysis, and anfis models in estimation of global solar radiation for different climatological locations,” in *Exergetic, Energetic and Environmental Dimensions*, I. Dincer, C. O. Colpan, and O. Kizilkan, Eds. Academic Press, 2018, pp. 133–148. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128137345000081

[39] J.-S. Jang, "Anfis: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.

[40] S. K. Alonso, C. T. Blanc, and M. G. S. Torrubia. Membership functions. [Online]. Available: http://www.dma.fi.upm.es/recursos/aplicaciones/logica_borrosa/web/fuzzy_inferencia/funpert_en.htm

[41] N. Talpur, M. Salleh, and K. Hussain, "An investigation of membership functions on performance of anfis for solving classification problems," 2017.

[42] M. Rajabi, B. Bohloli, and E. Gholampour Ahangar, "Intelligent approaches for prediction of compressional, shear and stoneley wave velocities from conventional well log data: A case study from the sarvak carbonate reservoir in the abadan plain (southwestern iran)," *Computers Geosciences*, vol. 36, no. 5, pp. 647–664, 2010.

[43] *Fuzzy Logic Toolbox™ User's Guide.* MathWorks, Inc., 1995–2016.

[44] A. Morim, L. E. Sa Fortes, P. Reis, C. A. Cosenza, F. Doria, and A. Goncalves, "Think fuzzy system : Developing new pricing strategy methods for consumer goods using fuzzy logic," *International Journal of Fuzzy Logic Systems*, vol. 7, pp. 1–17, 01 2017.

APPENDIX A ALGORITHM 1

Algorithm 1: How to find the feasible region in joint-space while performing a line

```
$P_0 \leftarrow$First point of the desired path e.g. first coordinates of the square or rectangle ;
$P_L \leftarrow$Last point of the path in a line e.g. second coordinates of the square or rectangle ;
$n \leftarrow$ Number of segments
$e \leftarrow (P_L - P_0)$ ;    /* Distance between the first point and the last point */
$e \leftarrow e/n$ ;                         /* Split the displacement in n segments  */
for $i \leftarrow 1$ to $n$ do
    $P_i \leftarrow P_0 + i * e$ ;                       /* Going step by step toward $P_l$  */
    $m \leftarrow 25$ ;                                    /* First loop counting number  */
    while $m>0$ AND $error>0.00001$ do
        $dP \leftarrow (P_i - P_{i-1})$ ;                /* Required Cartesian displacement */
        $d\theta \leftarrow J^+ * dP$;                                /* Joint displacement */
        $\theta \leftarrow \theta + d\theta$;
        $P \leftarrow$Forward kinematics $(\theta)$ ;
        $error \leftarrow norm(P_i - P)$ ;
        $m \leftarrow m - 1$ ;
    $tmin \leftarrow$ Minimum joint limits ;
    $tmax \leftarrow$ Maximum joint limits ;
    $s \leftarrow 0$;
    if $\theta(1) < tmin(1)$ OR $tmax(1) < \theta(1)$ then
        $s \leftarrow s + 1$;                                /* If we pass $\theta_1$ limitation */
    if $\theta(2) < tmin(2)$ OR $tmax(2) < \theta(2)$ then
        $s \leftarrow s + 2$;                                /* If we pass $\theta_2$ limitation */
    if $\theta(3) < tmin(3)$ OR $tmax(3) < \theta(3)$ then
        $s \leftarrow s + 4$;                                /* If we pass $\theta_3$ limitation */
    $u \leftarrow$ An arbitrary vector;
    $k \leftarrow 50$ ;                                   /* Second loop counting number  */
    while $k > 0$ AND $s == 0$ do
        $d\theta \leftarrow (I - J(\theta) * J(\theta)^+) * u$ ;
        $\theta \leftarrow \theta + *d\theta$;
        $k \leftarrow k - 1$;
    $k^{'} \leftarrow 50$ ;                                  /* Third loop counting number  */
    while $k' > 0$ AND $s == 0$ do
        $d\theta \leftarrow (I - J(\theta) * J(\theta)^+) * (-1) * u$ ;
        $\theta \leftarrow \theta + *d\theta$;
        $k^{'} \leftarrow k^{'} - 1$;
```