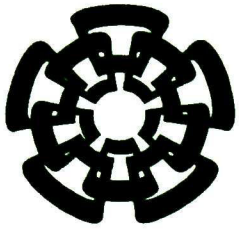
BC 674

Don. 2012

xx (185247.1)



Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional
Unidad Guadalajara

Construcción y Control de un Robot Redundante de Cinco Grados de Libertad

Tesis que presenta:
Roberto Loera Díaz

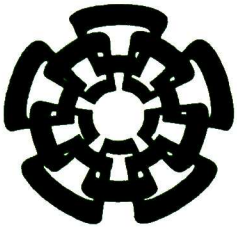
para obtener el grado de:
Maestro en Ciencias

en la especialidad de:
Ingeniería Eléctrica

Directores de Tesis
Dr. Bernardino Castillo Toledo
Dr. Maarouf Saad

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, Septiembre 2011

ID: 18 S156 -3001



Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional
Unidad Guadalajara

Construction and Control of a Five Degrees of Freedom Redundant Robot

A thesis presented by:
Roberto Loera Díaz

to obtain the degree of:
Master in Science

in the subject of:
Electrical Engineering

Thesis Advisors:
Dr. Bernardino Castillo Toledo
Dr. Maarouf Saad

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, September 2011

Construcción y Control de un Robot Redundante de Cinco Grados de Libertad

Tesis de Maestría en Ciencias en Ingeniería Eléctrica

Por:
Roberto Loera Díaz
Ingeniero en Comunicaciones y Electrónica
Universidad de Guadalajara 2002-2007

Becario de CONACYT, expediente No. 41836

Directores de Tesis:
Dr. Bernardino Castillo Toledo
Dr. Maarouf Saad

CINVESTAV del IPN Unidad Guadalajara, Septiembre 2011

Construction and Control of a Five Degrees of Freedom Redundant Robot

Master of Science Thesis
in Electrical Engineering

By:
Roberto Loera Díaz
Engineer in Communications and Electronics
Universidad de Guadalajara 2002-2007

Scholarship No. 41836 granted by CONACYT

Thesis Advisors:
Dr. Bernardino Castillo Toledo
Dr. Maarouf Saad

CINVESTAV del IPN Unidad Guadalajara, September 2011

Resumen

El uso de manipuladores robóticos se encuentra muy extendido hoy en día tanto en la industria como en el ámbito académico ya que estos pueden realizar una gran variedad de tareas. En particular los robots redundantes son de gran interés ya que al contar con una cantidad mayor de grados de libertad (DOF) de los que necesitan para realizar su tarea, son capaces de alcanzar una misma posición mediante diferentes configuraciones de sus articulaciones.

En este trabajo de tesis se estudia un prototipo de un robot redundante de cinco grados de libertad y se analizan algunas técnicas de control dinámico para el seguimiento de trayectorias: control neuronal, control difuso y control neurodifuso.

Al utilizar el control neuronal se tiene la ventaja de que no es necesario conocer el modelo matemático del sistema ya que la red neuronal se encarga de identificarlo. Se propone una red neuronal recurrente de alto orden (RHONN) entrenada por medio del filtro de Kalman para la identificación de la planta de tal modo que el error de identificación tienda a cero (identificación exacta). Se utiliza también un control difuso a partir de la construcción de un modelo difuso del tipo Takagi-Sugeno obtenido por aproximación local en partición de espacios difusos. Finalmente se utiliza un control neurodifuso por medio de un sistema de inferencia adaptativo neurodifuso (ANFIS).

Además se exploró el desempeño del control a través de Internet utilizando el protocolo de comunicaciones TCP/IP. Las señales de referencia fueron creadas en un computador central y recibidas en un computador conectado directamente al manipulador robótico.

Abstract

In today's world, robotic manipulators are very common in industry and academics since they can perform a great variety of tasks. In particular, redundant robots are of great interest because they are able to reach a position using different joint configurations. This is because redundant robots have more degrees of freedom (DOF) than the number of DOF needed to perform a task.

In this thesis a 5DOF redundant robot prototype is studied. Moreover, several control techniques for trajectory tracking are analyzed: neural control, fuzzy control and neuro-fuzzy control.

When a neural control is developed, it is not necessary, in general, to know the mathematical model of the system because the neural network performs a sort of a grey identification. In this work, a recurrent high order neural network (RHONN) is proposed in order to identify the system. This neural network is trained using the Kalman filter, thus, the error identification converges to zero (exact identification). A Takagi-Sugeno fuzzy control is also studied. It is obtained using the local approximation in fuzzy partition spaces technique. Finally, a neuro-fuzzy control is designed using an adaptive neuro-fuzzy inference system (ANFIS).

Moreover, the performance of the control was analyzed through the Internet using the TCP/IP communication protocol. The reference signals were created in a main computer and received in a computer which was directly connected to the robotic manipulator.

Dedico esta tesis a mis Padres
por todo su amor, apoyo, guía y comprensión:

Salvador Loera Vivar

María Guadalupe Díaz Velasco

This thesis is dedicated to my Parents
for all their love, support, guidance and comprehension:

Salvador Loera Vivar

María Guadalupe Díaz Velasco

Agradecimientos

A Dios por estar conmigo y ser mi sustento en todo momento.

A mis padres, Salvador Loera Vivar y María Guadalupe Díaz Velasco, por todo el amor, la comprensión, el apoyo, la guía y el ejemplo que siempre me han brindado.

A mis hermanos, Adriana Loera Díaz, María Guadalupe Loera Díaz y Salvador Loera Díaz, por todo su apoyo.

A mi asesor y director de tesis, Dr. Bernardino Castillo, por la orientación, el tiempo y el esfuerzo que me brindó para desarrollar este proyecto de tesis. A mi asesor Dr. Maarouf Saad.

A la Dra. Ofelia Begovich y el Dr. Eduardo Bayro por el tiempo y esfuerzo invertidos en la revisión de este trabajo.

A María Elena Rosales por todo su apoyo.

A Luis Ramón Martínez, Christian Pulido, Alejandro Sandoval, Giovanni Rodríguez, Concepción Murguía y todos los demás a quienes no puedo nombrar por falta de espacio, por brindarme su amistad y apoyo.

A mis compañeros de generación 'cont08', mis compañeros del 'grupo de sistemas no lineales', mis compañeros de control automático y en general mis compañeros del CINVESTAV Unidad Guadalajara: Carmen Guillén, Rocío Hernández, Blanca León, Adrián Navarro, Andrés Bueno, Axel Padilla, Carlos López, Christian Álvarez, Javier Molina, José Luis Oviedo, Luis Luque, Oscar Aroche, Oscar Carbajal, Santiago Elvira, Sergio Aguilar, Víctor Huidobro y todos los demás a quienes no puedo nombrar por falta de espacio, por todo el apoyo y amistad que me ofrecieron en esta etapa de mi vida.

A todos aquellos que de alguna forma me ayudaron en la realización de este proyecto.

Al Centro de Investigación y Estudios Avanzados del I.P.N. (CINVESTAV) Unidad Guadalajara, por el apoyo académico que me proporcionó para realizar mis estudios de maestría y durante la realización de esta tesis.

Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico otorgado que me permitió concluir este posgrado.

Acknowledgment

Thanks to God for being with me and for being my support at all times.

Thanks to my parents, Salvador Loera Vivar and María Guadalupe Díaz Velasco, for all the love, comprehension, support, guidance and examples that have given to me at all times.

Thanks to my sisters, Adriana Loera Díaz, María Guadalupe Loera Díaz and my brother, Salvador Loera Díaz, for all their support.

Thanks to my thesis advisor, Dr. Bernardino Castillo, for his guidance, time and effort that helped me to develop this thesis project. And thanks to my advisor Dr. Maarouf Saad.

Thanks to Dra. Ofelia Begovich and Dr. Eduardo Bayro for the time and effort in reviewing this thesis.

Thanks to María Elena Rosales for all her support.

Thanks to Luis Ramón Martínez, Christian Pulido, Alejandro Sandoval, Giovanni Rodríguez, Concepción Murguía and all whom I may not mention because I do not have enough space, for their friendship and support.

Thanks to my colleagues of 'cont08', my colleagues of the 'non-linear systems group', my colleagues of the automatic control area and all my colleagues from 'CINVESTAV Unidad Guadalajara': Carmen Guillén, Rocío Hernández, Blanca León, Adrián Navarro, Andrés Bueno, Axel Padilla, Carlos López, Christian Álvarez, Javier Molina, José Luis Oviedo, Luis Luque, Oscar Aroche, Oscar Carbajal, Santiago Elvira, Sergio Aguilar, Victor Huidobro and all whom I may not mention because I do not have enough space, for their support and friendship.

Thanks to all those persons that supported me to complete this goal.

Thanks to CINVESTAV (Centro de Investigación y Estudios Avanzados del I.P.N) Unidad Guadalajara, for the academic support provided during my studies and during the realization of this thesis.

Thanks to CONACYT (Consejo Nacional de Ciencia y Tecnología) for the scholarship that allowed me to complete my studies.

Contents

List of Figures

Chapter 1

Introduction

In today's world, the use of robotic manipulators (Figure 1.1) is very common in industry, academics and scientific research [8]. In industry, robots are part of the automation process. They are an important part of production lines because they make the industrial processes faster and more efficient. Robotic manipulators are also used in high-risk locations where a human operator could not easily or safely work. They are also used for many other applications [13].

A particular class of robot manipulators is represented by redundant robots. These robots have more degrees of freedom than the number of DOF needed in order to reach a specific point in their workspace. This feature is of great interest because redundant robots are capable of reaching the same point using different trajectories, thus they can optimize energy or reach a point, while avoiding obstacles in their path.

In this thesis, the construction and control of a five degrees of freedom redundant robot prototype is reported. Furthermore, improvements of mechanical and electronic aspects of a previous prototype are discussed.

In addition, some control algorithms developed for the prototype are presented: neural control, fuzzy control and neuro-fuzzy control. The neural control was developed using a Recurrent High Order Neural Network (RHONN). This neural network was trained using the Kalman filter. The fuzzy control was created using the local approximation in fuzzy partition spaces technique. The hybrid neuro-fuzzy control was developed using an Adaptive Neuro-Fuzzy Inference System (ANFIS).

The behavior of the control algorithms was also analyzed using the TCP/IP communications protocol. This protocol was used in order to drive the prototype from a remote location through the Internet.

With this work, experience was obtained in order to build and modify prototypes of these characteristics. Therefore, in the future, it would be easier to create and modify new prototypes. Mechanical and electronic parts of the system were accessed so they could be modified and new elements could be implemented. On the contrary, if a new prototype had been bought, it would not had been possible to access to some parts of the system in order to modify them or add new items.

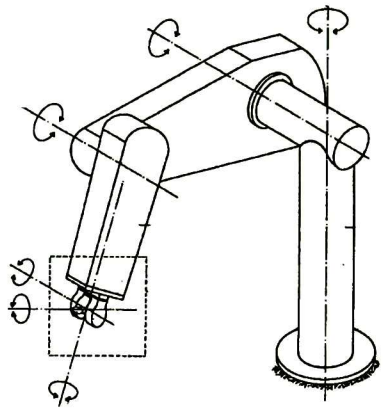


Figure 1.1: A robotic manipulator

During the construction of the prototype, engineering knowledge could be applied in order to develop and improve the system. Different types of control algorithms could be developed and simulated. Then, they could be applied into the prototype. The redundant robot properties and applications could be analyzed and implemented in simulation and real time.

1.1 Objectives

The objectives to be achieved during this work were to:

Improve of the mechanical and electronic features of the five degrees of freedom redundant robot of CINVESTAV.

Construct and implement the electronic elements needed in the prototype.

Analyze the properties of the five degrees of freedom robotic arm prototype: forward and inverse kinematics, dynamic model of the robot and state space representation of the model.

Design control algorithms for the system: neural control, fuzzy control and neuro-fuzzy control.

Simulated and analyze these control algorithms.

Implement one of these control algorithms in real time.

Operate the five degrees of freedom redundant robot via Internet using the TCP/IP communications protocol.

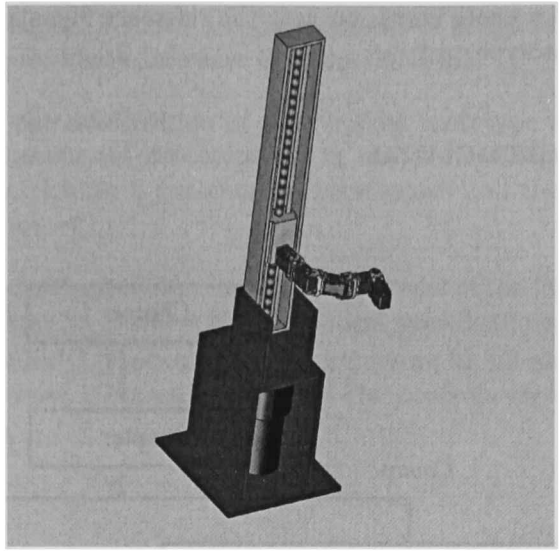


Figure 1.2: 5DOF Robot Prototype (CINVESTAV)

1.2 Contributions

The contributions realized during the development of this thesis project were:

Some modifications were made in the design of the prototype (Figure 1.2). Some changes in the five degrees of freedom robotic arm physical prototype and in the electronic interface were made in order to improve it.

A prismatic joint was implemented in the prototype. This joint allows the robotic arm to perform longitudinal movements. Moreover, mechanical features were improved. The rotational joints were changed by new motors which present many advantages with respect to previous motors.

Electronic features were also improved. The electronic interface was changed in order to control the rotational articulations easily. The previous interface was found to generate noisy signals. With the new interface this problem was solved.

A neural control was designed using a Recurrent High Order Neural Network (RHONN). This network was trained using the Kalman filter (discrete RHONN). A neural network is used to identify the system and then control it.

A Takagi-Sugeno fuzzy control was developed. This control was designed using the local approximation in fuzzy partition spaces technique. A hybrid neuro-fuzzy control was designed using an Adaptive Neuro-Fuzzy Inference System (ANFIS).

The current five degrees of freedom robotic arm prototype was used in real time to reach a trajectory. This robotic arm was also remotely controlled via Internet using the

TCP/IP protocol. A remote computer sent the reference signals to a local computer connected to the prototype.

1.3 Thesis structure

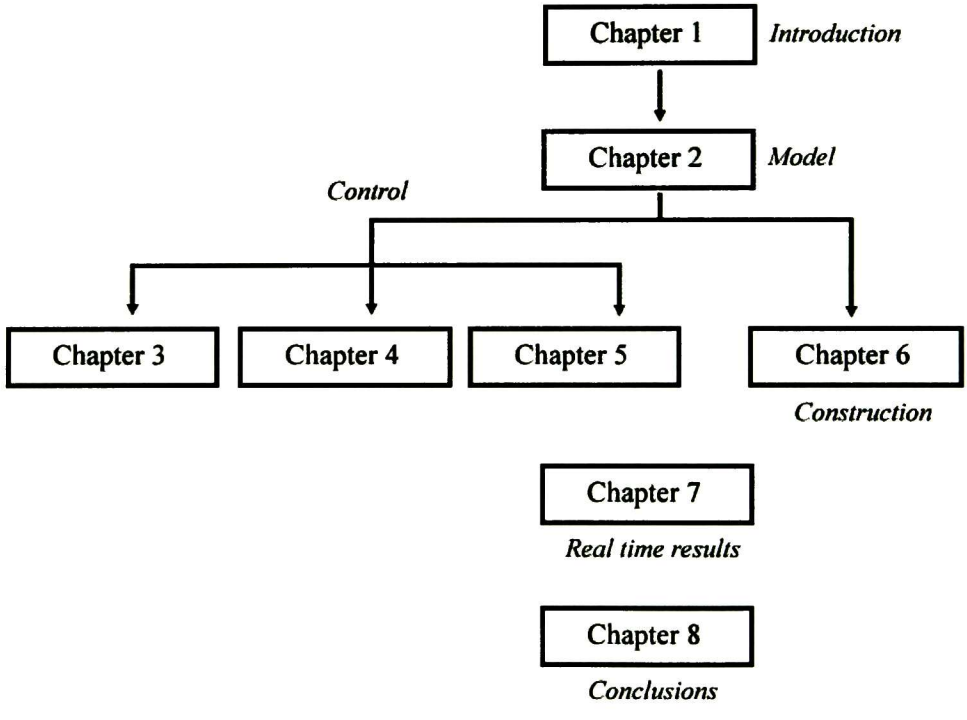


Figure 1.3: Thesis structure

This thesis is divided in eight chapters (Figure 1.3). Chapter 1 presents the introduction, justification, objectives and thesis structure. The direct kinematics and inverse kinematics of the five degrees of freedom prototype are presented in chapter 2 (robot modeling,), including the dynamic model of the system and its space state representation.

In chapters three, four and five, the design of different kinds of control algorithms is presented. Chapter 3 presents the development of a neural control. This control was designed using a Recurrent High Order Neural Networks (RHONN) trained using the Kalman filter. Chapter 4 discusses the development of a Takagi-Sugeno fuzzy control. This fuzzy control has been obtained using the local approximation in fuzzy partition

spaces technique. Chapter 5 shows the development of a hybrid neuro-fuzzy control with an Adaptive Neuro-Fuzzy Inference Systems (ANFIS).

Chapter 6 shows the construction of the physical prototype and its electronic interface. It also talks about the prototype use in real time. Chapter 7 establishes real time results. Finally, chapter 8 presents the conclusions and the work which will be developed in the future.

Moreover, Appendix A establishes the dynamic model of the five degrees of freedom redundant robot. Appendix B shows the component specifications used in the robot of CINVESTAV. Appendix C presents the robot structure in 3D made in SolidWorks® with an academic license. Appendix D shows the controls algorithms developed in Simulink®

Chapter 2

Robot modeling

2.1 Kinematics

In general, the kinematics problem for a robot system can be divided in two different subproblems: the forward kinematics and the inverse kinematics problem (Figure 2.1) [17].

The forward kinematics problem consists of finding the position and orientation of the end-effector with respect to a cartesian reference coordinate system x, given the joint coordinates vector $q = (q_1, q_2, \cdots, q_n)$ of the robotic arm and the geometric parameters of the links, where n represents the number of degrees of freedom.

The inverse kinematics problem consists of finding the joint vector q needed to positioning the end-effector (Figure 2.2) in the desired place and orientation, given the position and orientation of the end-effector with respect to the reference coordinate system and the geometric parameters of the links. Since the independent variables in a robot are the joint variables and a task is generally defined in terms of the cartesian coordinate system, it is more common to use the inverse kinematics solution in most of the applications.

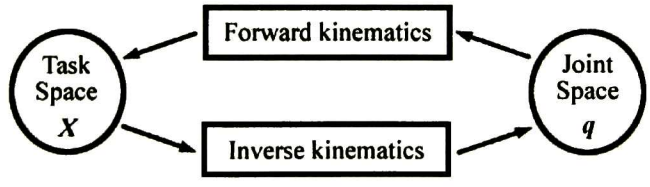


Figure 2.1: Kinematics

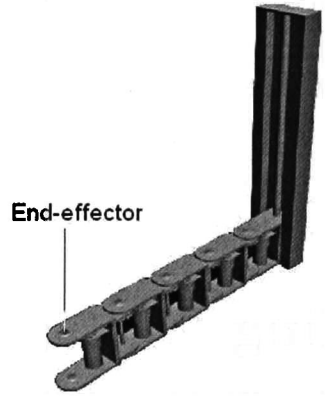


Figure 2.2: End-effector

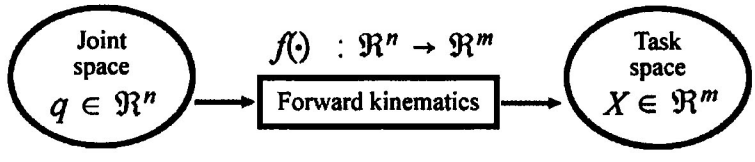


Figure 2.3: Forward kinematics

2.1.1 Forward kinematics

As mentioned before, the forward kinematics problem consists in determining the cartesian position of the end-effector given the joint variables. Let $q \in \Re^n$ be the joint variable vector and $X \in \Re^m$ the end-effector position vector, where n is the number of degrees of freedom of the robot and m is the dimension of the robot work space. The forward kinematics objective is to find a function $f(\cdot) : \Re^n \rightarrow \Re^m$ such that each vector $q \in \Re^n$ has an associated vector $X \in \Re^m$ (Figure 2.3).

The Denavit-Hartenberg (D-H) parameters are a series of values that geometrically relate the robot joints. In the traditional D-H algorithm, an $(i-1)$-th coordinate system in $\Re^3$ is associated to the i-th joint. In the modified D-H algorithm the i-th joint is associated to the i-th coordinate system.

Let $\{A\}$ be a cartesian coordinate system represented by three unitary orthogonal vectors. A position in space may be represented by a vector ${}^AP$ or by a set of three numbers. The ${}^AP$ vector is defined as:

$$ {}^AP = \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix} \tag{2.1} $$

In order to describe the orientation of a body, a coordinate system is added to that body. Then, a description is given to a reference coordinate system. If a coordinate system $\{B\}$ is added to a point in the space, then, the orientation of this point is given by a description of $\{B\}$ with respect to $\{A\}$ [1].

Let $\{A\}$ and $\{B\}$ be a pair of coordinate systems with origin in $\{x_A, y_A, z_A\}$ and $\{x_B, y_B, z_B\}$ respectively. The coordinate system $\{A\}$ may be different with respect to the coordinate system $\{B\}$ due to a translation of their origin (Figure 2.4). $\{A\}$ and $\{B\}$ can also be different by a rotation in the axis x, y, z or a combination of these (Figure 2.5). Moreover, the pair of coordinate systems $\{A\}$ and $\{B\}$ may be different by a combination of both a translation of their origins and a rotation of their axis.

Let ${}^{B}P$ be a different position in space. If we want to describe a position ${}^{B}P$ in terms of $\{A\}$, when $\{A\}$ has the same orientation than $\{B\}$, then $\{B\}$ is different of $\{A\}$ by a translations which may be represented by the ${}^{A}P_0$ vector, where ${}^{A}P_0$ is a vector located in the origin of $\{B\}$ with respect to $\{A\}$:

$$ {}^{A}P = {}^{B}P + {}^{A}P_0 \tag{2.2} $$

If we want to describe the position ${}^{B}P$ in terms of $\{A\}$ when $\{A\}$ and $\{B\}$ have the same origin but different orientation, then we may use the rotation matrix ${}^{A}_{B}R$ which describes $\{B\}$ with respect to $\{A\}$:

$$ {}^{A}P = {}^{A}_{B}R\,{}^{B}P \tag{2.3} $$

When $\{A\}$ and $\{B\}$ have different origins and a different orientation and we want to define the position ${}^{B}P$ with respect to $\{A\}$, then we can use the rotation matrix ${}^{A}_{B}R$ which describes $\{B\}$ with respect to $\{A\}$ and we can translate that new point by adding the ${}^{A}P_0$ vector.

$$ {}^{A}P = {}^{A}_{B}R\,{}^{B}P + {}^{A}P_0 \tag{2.4} $$

Expression (2.4) can be represented using the homogeneous transformation ${}^{A}_{B}T$:

$$ {}^{A}P = {}^{A}_{B}T\,{}^{B}P \tag{2.5} $$

from (2.5) we have:

$$ \begin{pmatrix} {}^{A}P \\ 1 \end{pmatrix} = \begin{pmatrix} {}^{A}_{B}R & {}^{A}P_0 \\ 0_{xyz} & 1 \end{pmatrix} \begin{pmatrix} {}^{B}P \\ 1 \end{pmatrix} \tag{2.6} $$

The matrix that relates the i-th coordinate system with respect to the $(i-1)$-th coordinate system by means of the D-H algorithm is given by:

$$ {}^{i-1}_{i}T = \begin{pmatrix} cos(\theta_i) & -sin(\theta_i) & 0 & a_{i-1} \\ sin(\theta_i)cos(\alpha_{i-1}) & cos(\theta_i)cos(\alpha_{i-1}) & -sin(\alpha_{i-1}) & -sin(\alpha_{i-1})d_i \\ sin(\theta_i)sin(\alpha_{i-1}) & cos(\theta_i)sin(\alpha_{i-1}) & cos(\alpha_{i-1}) & cos(\alpha_{i-1})d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.7} $$

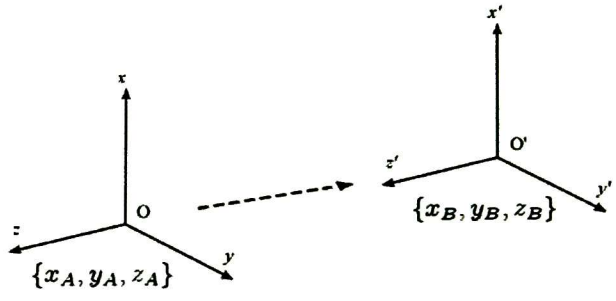


Figure 2.4: Origin translation of a coordinate system

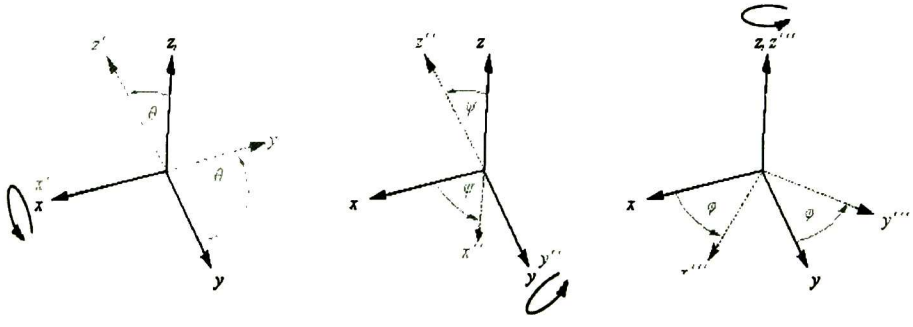


Figure 2.5: Rotation in the x, y and z axes

where α_{i-1}, a_{i-1}, θ_i and d_i are the D-H parameters.

The end-effector coordinate system is related to the base coordinate system by the homogeneous transformation matrix ${}^0_eT$, defined as:

$$ {}^0_eT = {}^0_1T\,{}^1_2T\,{}^2_3T \cdots {}^{e-1}_eT \tag{2.8} $$

In the following table the D-H parameters for the 5DOF robot are shown. Here L_1 represents the length of the first link and L represents the length of the successive ones:

Joint	a_{i-1}	a_{i-1}	θ_i	d_i
1	0	0	q_1	0
2	0	L_1	0	q_2
3	0	L	0	q_3
4	0	L	0	q_4
5	0	L	0	q_5

D-H parameters for the 5DOF robot.

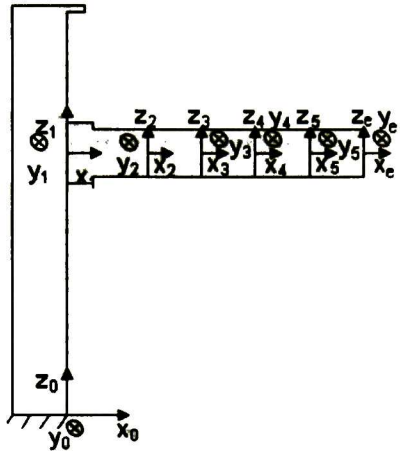


Figure 2.6: 5DOF robot

For the case of the 5DOF robot the homogeneous transformation matrix that relates the end-effector coordinate system and the base coordinate system turns to be:

$$
{}^{0}_{e}T = \begin{pmatrix}
\cos(\theta_2+\theta_3\ldots \\ +\theta_4+\theta_5) & -\sin(\theta_2+\theta_3\ldots \\ +\theta_4+\theta_5) & 0 & \begin{array}{c} L\cos(\theta_2+\theta_3+\theta_4+\theta_5)\ldots \\ +L\cos(\theta_2+\theta_3+\theta_4)\ldots \\ +L\cos(\theta_2+\theta_3)+L\cos(\theta_2)+L_1 \end{array} \\
-\sin(\theta_2+\theta_3\ldots \\ +\theta_4+\theta_5) & \cos(\theta_2+\theta_3\ldots \\ +\theta_4+\theta_5) & 0 & \begin{array}{c} L\sin(\theta_2+\theta_3+\theta_4+\theta_5)\ldots \\ +L\sin(\theta_2+\theta_3+\theta_4)\ldots \\ +L\sin(\theta_2+\theta_3)+L\sin(\theta_2) \end{array} \\
0 & 0 & 1 & q_1 \\
0 & 0 & 0 & 1
\end{pmatrix} \tag{2.9}
$$

The following matrix relates the origin of the end effector coordinate system $\{x_e, y_e, z_e\}$ with respect to the origin of the base coordinate system $\{x_0, y_0, z_0\}$:

$$
{}^{0}_{e}O = \begin{pmatrix}
L\cos(\theta_2+\theta_3+\theta_4+\theta_5)+L\cos(\theta_2+\theta_3+\theta_4)+L\cos(\theta_2+\theta_3)+L\cos(\theta_2)+L_1 \\
L\sin(\theta_2+\theta_3+\theta_4+\theta_5)+L\sin(\theta_2+\theta_3+\theta_4)+L\sin(\theta_2+\theta_3)+L\sin(\theta_2) \\
q_1
\end{pmatrix} \tag{2.10}
$$

The $\{x_e, y_e, z_e\}$ coordinate system has a rotation in the z_0 axis defined bye the Euler angle $\gamma = q_2 + q_3 + q_4 + q_5$. This angle is unique in the 5DOF robot case because the end-effector can only rotate in the z_0 axis.

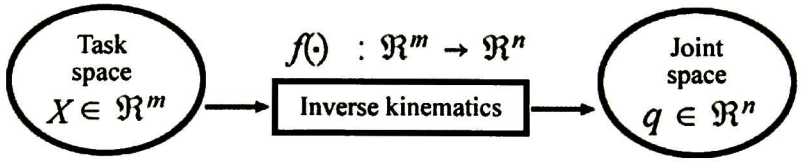


Figure 2.7: Inverse kinematics

The set of expressions ${}^0_eO$ and γ fully describes the position of the end-effector with respect to the base coordinate system $\{x_0, y_0, z_0\}$ (Figure 2.6). Therefore, the forward kinematics solution is given by:

$$X = \begin{pmatrix} {}^0_eO \\ \gamma \end{pmatrix} \tag{2.11}$$

2.1.2 Inverse kinematics

The inverse kinematics problem consists in finding the joint variables $q \in \Re^n$ from the position of the end-effector, $X \in \Re^m$ In this case, the objective is to find a function $f(\cdot) : \Re^m \rightarrow \Re^n$ such that each $X \in \Re^m$ vector has an associated vector $q \in \Re^n$.

The linear and angular velocities of the joint space q are related with the end-effector velocities X by the following expression:

$$\dot{X} = J(q)\dot{q} \tag{2.12}$$

where $\dot{q} \in \Re^n$ is the time variation of the joint positions, $\dot{X} \in \Re^m$ is the end-effector velocity and $J(q) \in \Re^{m \times n}$ is the following Jacobian matrix, where $\delta(q) = X$:

$$J(q) = \frac{\partial \delta(q)}{\partial q} \tag{2.13}$$

$$G(\dot{q}) = \dot{q}^T \dot{q} \tag{2.14}$$

$$G(\dot{q}, \lambda) = \dot{q}^T \dot{q} - \lambda^T (\dot{J}q - \dot{X}) \tag{2.15}$$

The 5DOF robot case, being a redundant robot, has a larger number of degrees of freedom than those needed. As $n > m$, the $J(q)$ Jacobian matrix is a rectangular matrix with more columns than rows, thus, it is not possible to use a common inverse matrix to obtain $\dot{q}$ from (2.12). Therefore, in order to get a good approximation to $\dot{q}$ a right pseudoinverse obtained from the least square criterion is used (2.14). Using Lagrange multiplicators in the matrix (2.15), $\dot{q}$ is found, so that it satisfies (2.12) for any $\dot{X}$ and $J(q)$ minimizing at the same time the cost function $G(\dot{q})$, where λ is a $\Re^m$ known vector.

To find the minimum value of the cost function, the following conditions have to be satisfied:

$$\frac{\partial G}{\partial \dot{q}} = 0 = 2\dot{q} - J^T\lambda \tag{2.16}$$

$$\frac{\partial G}{\partial \lambda} = 0 = J\dot{q} - \dot{X} \tag{2.17}$$

From (2.16), it can be seen that $\dot{q} = \frac{1}{2}J^T\lambda$ and replacing it in (2.17) we obtain that $(JJ^T)\lambda = 2\dot{X}$. As $J(q)$ is a row full rank matrix, then JJ^T is a nonsingular square matrix, this is, it is invertible. For this reason and removing the λ Lagrange multiplicators, it can be seen than the inverse kinematics solution is given by the following expression, where $J^T(JJ^T)^{-1}$ is the right Penrose pseudoinverse.

$$\dot{q} = J^T(JJ^T)^{-1}\dot{X} \tag{2.18}$$

The general solution for a redundant robot is given by:

$$\dot{q} = J^T(JJ^T)^{-1}\dot{X} + N\zeta \tag{2.19}$$

where $N = I - J^T(JJ^T)^{-1}J$ is the null space of $J(q)$ and ζ is a $\Re^n$ vector, which can be used to perform certain tasks, for example, obstacle avoidance such that the distance between the robot and the obstacle is minimal without collisioning. If (2.18) is premultiplied by $J(q)$ in both sides, then the right side of the equation does not affect the movement of the end-effector, namely:

$$J\dot{q} = J^T(JJ^T)^{-1}\dot{X} + (J - JJ^T(JJ^T)^{-1}J)\zeta \tag{2.20}$$

For the 5DOF robot case the Jacobian is as follows:

$$J(q) = \begin{bmatrix} 0 & J_{12} & J_{13} & J_{14} & J_{15} \\ 0 & J_{22} & J_{23} & J_{24} & J_{25} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{2.21}$$

where:

$$\begin{aligned}
J_{12} &= -L(\sin(\theta_2+\theta_3+\theta_4+\theta_5)+\sin(\theta_2+\theta_3+\theta_4)+\sin(\theta_2+\theta_3)+\sin(\theta_2)) \\
J_{13} &= -L(\sin(\theta_2+\theta_3+\theta_4+\theta_5)+\sin(\theta_2+\theta_3+\theta_4)+\sin(\theta_2+\theta_3)) \\
J_{14} &= -L(\sin(\theta_2+\theta_3+\theta_4+\theta_5)+\sin(\theta_2+\theta_3+\theta_4)) \\
J_{15} &= -L(\sin(\theta_2+\theta_3+\theta_4+\theta_5)) \\
J_{22} &= L(\cos(\theta_2+\theta_3+\theta_4+\theta_5)+\cos(\theta_2+\theta_3+\theta_4)+\cos(\theta_2+\theta_3)+\cos(\theta_2)) \\
J_{23} &= L(\cos(\theta_2+\theta_3+\theta_4+\theta_5)+\cos(\theta_2+\theta_3+\theta_4)+\cos(\theta_2+\theta_3)) \\
J_{24} &= L(\cos(\theta_2+\theta_3+\theta_4+\theta_5)+\cos(\theta_2+\theta_3+\theta_4)) \\
J_{25} &= L(\cos(\theta_2+\theta_3+\theta_4+\theta_5))
\end{aligned}$$

In the expression (2.21) the linear velocities of the end-effector are related to the first three rows of the matrix and the angular velocity of the end-effector is related to the last row.

2.2 Mathematical model

In the motion of a robot, many different forces are involved; for example, the gravitation force, friction forces and inertia forces. All of these forces have to be considered in the mathematical model. This model describes what kind of joint torque forces will be produced from the physical model of the robot.

The equation of the robot motion is described by the following dynamic equation, which has being obtained from the Lagrange method.

$$\tau = M(q)\ddot{q} + V(q,\dot{q}) + F_g(q) + F_f(q,\dot{q}) \tag{2.22}$$

where:
q represents the position vector.
$\dot{q}$ represents the velocity vector.
$\ddot{q}$ represents the acceleration vector.
$M(q)$ is an inertia positive-definite matrix.
$V(q,\dot{q})$ is the centrifugal and Coriolis forces vector.
$F_g(q)$ is the gravitational forces vector.
$F_f(q,\dot{q})$ is the friction forces vector.
τ is the torque vector forces applied to the robot joints.

Expanding the equation (2.22) for the 5DOF robot, it can be seen that:

$$\begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \\ \tau_5 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} & m_{15} \\ m_{21} & m_{22} & m_{23} & m_{24} & m_{25} \\ m_{31} & m_{32} & m_{33} & m_{34} & m_{35} \\ m_{41} & m_{42} & m_{43} & m_{44} & m_{45} \\ m_{51} & m_{52} & m_{53} & m_{54} & m_{55} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \\ \ddot{q}_4 \\ \ddot{q}_5 \end{bmatrix} + \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \end{bmatrix} + \begin{bmatrix} F_{g1} \\ F_{g2} \\ F_{g3} \\ F_{g4} \\ F_{g5} \end{bmatrix} + \begin{bmatrix} f_{f1} & 0 & 0 & 0 & 0 \\ 0 & f_{f2} & 0 & 0 & 0 \\ 0 & 0 & f_{f3} & 0 & 0 \\ 0 & 0 & 0 & f_{f4} & 0 \\ 0 & 0 & 0 & 0 & f_{f5} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \\ \dot{q}_5 \end{bmatrix} \tag{2.23}$$

The terms of the equation (2.23) are bounded [15]. If the position vector q and the velocity vector $\dot{q}$ are taken as state variables, then the following state space representation is obtained:

$$\begin{aligned} \dot{\chi}_I &= \chi_{II} \\ \dot{\chi}_{II} &= F(\chi_I,\chi_{II}) + G(\chi_I)u \end{aligned} \tag{2.24}$$

where
$\dot{\chi}_I = q$ is a $\Re^n$ vector which represents the position of each link.

$\dot{\chi}_{II} = \dot{q}$ is a $\Re^n$ which represents the velocity for each link.
$F(\chi_I, \chi_{II}) = -M^{-1}(\chi_I)(V(\chi_I, \chi_{II}) + F_g(\chi_I) + F_f(\chi_I, \chi_{II}))$
$G(\chi_I) = M^{-1}(\chi_I)$
$u = \tau$ is the torque input vector.

The model (2.24) can be given explicitly as:

$$\begin{pmatrix} \dot{\chi}_1 \\ \dot{\chi}_2 \\ \dot{\chi}_3 \\ \dot{\chi}_4 \\ \dot{\chi}_5 \\ \dot{\chi}_6 \\ \dot{\chi}_7 \\ \dot{\chi}_8 \\ \dot{\chi}_9 \\ \dot{\chi}_{10} \end{pmatrix} = \begin{pmatrix} \chi_6 \\ \chi_7 \\ \chi_8 \\ \chi_9 \\ \chi_{10} \\ F(\chi_1, \chi_6) + G(\chi_1)u \\ F(\chi_2, \chi_7) + G(\chi_2)u \\ F(\chi_3, \chi_8) + G(\chi_3)u \\ F(\chi_4, \chi_9) + G(\chi_4)u \\ F(\chi_5, \chi_{10}) + G(\chi_5)u \end{pmatrix} = \tag{2.25}$$

$$\begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \\ \dot{q}_5 \\ -m(q_1)\{v(q_1, \dot{q}_1) + f_g(q_1) + f_f(q_1, \dot{q}_1)\} \\ -m(q_2)\{v(q_2, \dot{q}_2) + f_g(q_2) + f_f(q_2, \dot{q}_2)\} \\ -m(q_3)\{v(q_3, \dot{q}_3) + f_g(q_3) + f_f(q_3, \dot{q}_3)\} \\ -m(q_4)\{v(q_4, \dot{q}_4) + f_g(q_4) + f_f(q_4, \dot{q}_4)\} \\ -m(q_5)\{v(q_5, \dot{q}_5) + f_g(q_5) + f_f(q_5, \dot{q}_5)\} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ m(q_1) \\ m(q_2) \\ m(q_3) \\ m(q_4) \\ m(q_5) \end{pmatrix} \tau$$

This representation is in the so called nonlinear control block form which has some interesting mathematical properties [5]. This representation will be used in the following chapters to obtain controllers for the 5DOF redundant robot such as a neural control (chapter 3) and a fuzzy control (chapter 4).

The full representation of all the elements of the inertia matrix $M(q)$, the gravitational forces vector $F_g(q)$ and the Coriolis forces vector $V(q, \dot{q})$ of the 5DOF robot dynamic model is shown in appendix A.

Figure 2.8 shows the five degrees of freedom robot prototype constructed at CINVESTAV. This prototype has one prismatic joint and four revolute joints. Thus, this robotic arm is able to reach a variety of points inside a large workspace.

Figure 2.8: 5DOF Robot Prototype

Chapter 3

Neural control

3.1 Neural networks

Artificial Neural Networks (ANN) are simplified models based in biologic neural networks, which try to imitate the human brain capacities in problems resolution, such as vision, pattern recognition, moto-sensorial control, etc. An artificial neural network is a great amount of parallel distributed interconnected processors able to storage experimental data [2] [9].

The fundamental unit of biologic neural networks is the neuron (see Figure 3.1). The neuron is a cell that can be excited electrically. It processes information and transmits it by electrical and chemical signals. Neurons have three principal parts: dendrites, soma and axon. Dendrites are fibers which send electrical signals to the neurons body. The soma (or neurons body) sums the electrical signals that it receives. The axon sends signals from the neurons body to others neurons.

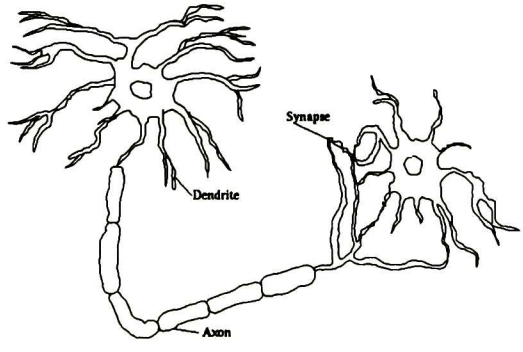


Figure 3.1: Biological neuron

In artificial neural networks the neuron (Figure 3.2) is the fundamental unit of process and information. In neurons, the following elements can be identified:

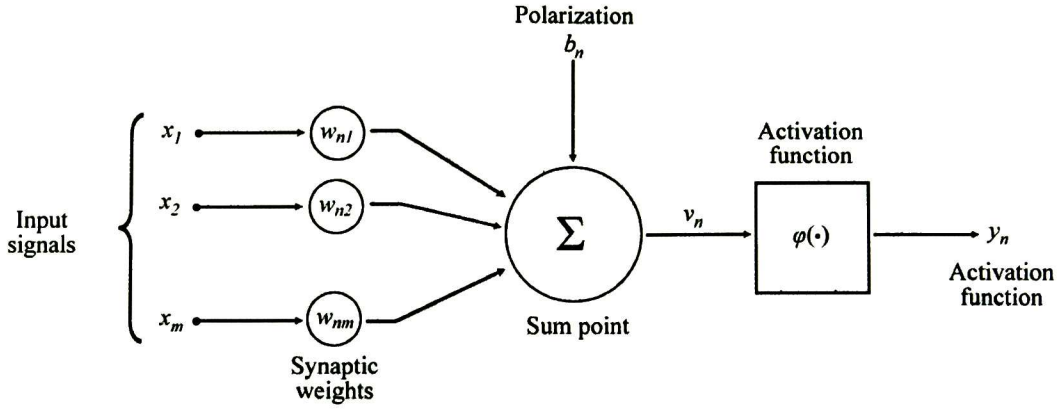


Figure 3.2: Artificial neuron structure

Link connections. These are parameterized by synaptic weights w_{nf}, where the first subscript corresponds to the receptor neuron and the second corresponds to the emitter neuron. If $w_{nj} > 0$, then it is an exciting connection; if $w_{nj} < 0$ it is an inhibit connection.

Sum ($\sum$). It sums the input signals which are multiplied by w_{nj}.

Activation function. This is a non-linear transformation. It defines the output of the neuron in function of the activation potential.

Threshold. This is a signal which displaces the original input signal where a positive or a negative scalar is added.

Artificial neural networks acquire their knowledge experimentally. In an experiment, the neurons interconnection weights (synapsis) change constantly. Neural networks have the following advantages:

Non-linearity. The neural processor is basically non-linear, thus, the neural network is also non-linear.

Input-output transformation. The learning process consists of presenting examples to the network to modify the synaptic weights according to the response.

Analysis and design uniformity. Thanks to this uniformity it is possible to guarantee precise characteristics in the neural network.

Biological networks analogy. Because artificial neural networks are based on biological neural networks, the knowledge in both areas can be used to work together.

A neuron n can be described mathematically by the following equations:

$$u_n = \sum_{j=1}^{m} w_{nj} x_j \tag{3.1}$$

$$y_n = \varphi(u_n + b_n) \tag{3.2}$$

where $x_1, x_2, \cdots, x_m$ are the input signals; $w_{n1}, w_{n2}, \cdots, w_{nm}$ are the synaptic weights of the neuron n; u_n is a linear combination of the inputs weighted by the synaptic weights; b_n is the polarization or threshold; $\varphi(\cdot)$ is the activation function; and y_n is the neuron's output signal.

If the polarization b_n is considered as another input signal $x_0 = +1$ with weight $w_{n0} = b_n$, then:

$$u_n = \sum_{j=0}^{m} w_{nj} x_j \tag{3.3}$$

$$y_n = \varphi(v_n) \tag{3.4}$$

in this case, v_n is called the activation potential (see Figure 3.3).

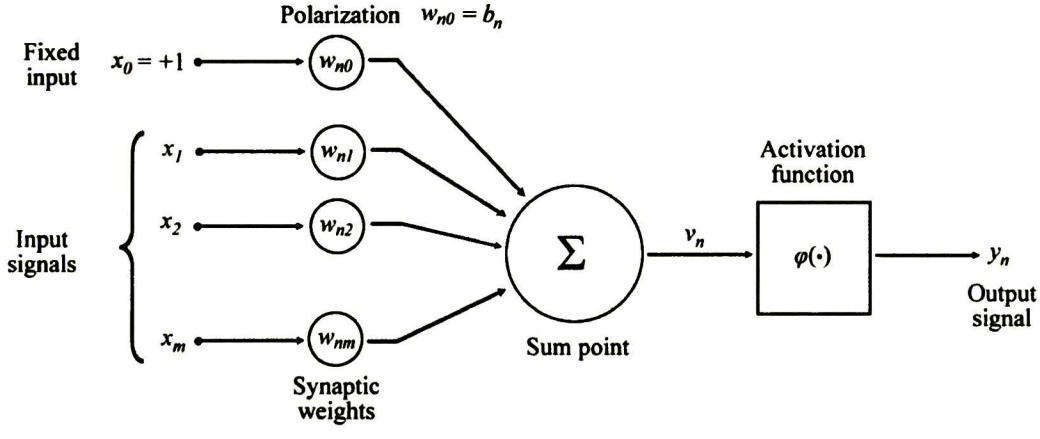


Figure 3.3: Artificial neuron structure

The activation function $\varphi(\cdot)$ defines the output of the neuron as a function of the activation potential v. Some examples of activation functions are the step function, piecewise linear function and the sigmoid function, the later being one of the most used activation functions in neural networks, defined as:

$$\varphi(v) = \frac{1}{1 + e^{-av}} \tag{3.5}$$

where a is the parameter that determines the slope of the sigmoid function.

3.2 Recurrent High Order Neural Networks (RHONN)

Recurrent high order neural networks are those having one or more feedback closed loops (Figure 3.4). The feedback can be local, this is, one neuron feedbacks itself, or global, when one neuron feedbacks other neurons of the same layer or previous layers [11].

Recurrent high order neural networks are a generalization of the first order Hopfield networks. In a RHONN, the state of a neuron can be determined by the following differential equation:

$$\dot{x}_i = -a_i x_i + b_i \sum_{j}^{L} w_{ij} y_j \tag{3.6}$$

If a neural network with n neurons and m inputs is considered, then, the state of any neuron is represented by the following differential equation:

$$\dot{x}_i = -a_i x_i + b_i \sum_{k=1}^{L} w_{ik} \prod_{j \in I_k} y_j^{d_j(k)} \tag{3.7}$$

where y is a vector given by:

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ y_{n+1} \\ \vdots \\ y_{n+m} \end{bmatrix} = \begin{bmatrix} S(x_1) \\ \vdots \\ S(x_n) \\ S(u_1) \\ \vdots \\ S(u_m) \end{bmatrix} \tag{3.8}$$

with $u = [u_1, u_2, , u_m]^T$ as the input vector to the neural network. $S(x)$ is given by:

$$S(x) = \frac{\mu}{1 + e^{-\beta x}} + \varepsilon \tag{3.9}$$

where β and μ are positive constants and ε is a small positive real number. $S(x) \in [\varepsilon, \varepsilon + 1]$

The network is represented by:

$$z_I(x,u) = \begin{bmatrix} zI_1 \\ zI_2 \\ \vdots \\ zI_L \end{bmatrix} = \begin{bmatrix} \prod_{j \in I_1} y_j^{d_j(1)} \\ \prod_{j \in I_2} y_j^{d_j(2)} \\ \vdots \\ \prod_{j \in I_L} y_j^{d_j(L)} \end{bmatrix} \tag{3.10}$$

$$\dot{x}_i = -a_i x_i + \sum_{k=1}^{L} w_{ik} z_{Ik} \tag{3.11}$$

where L is the number of high order connections; $\{I_1, I_2, \cdots, I_L\}$ is the set of L non ordered subsets $\{1, 2, \cdots, m+n\}$.

If $wi = [w_{i1} w_{i2} \cdots w_{iL}]^T$ in equation (3.11), then:

$$\dot{x}_i = -a_i x_i + w_i^T z_I(x, u) \tag{3.12}$$

Therefore, the RHONN model is given as follows:

$$\dot{x}_i = -Ax + W^T z_i(x, u) \tag{3.13}$$

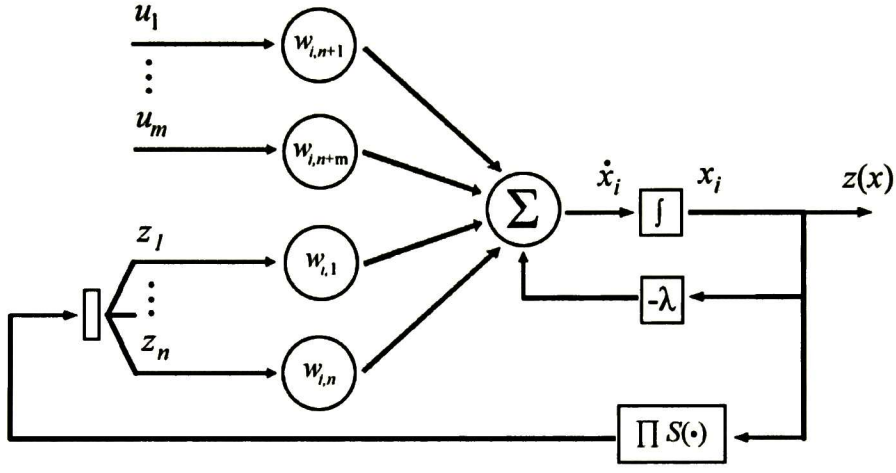


Figure 3.4: Recurrent High Order Neural Network (RHONN) structure

A RHONN allows efficiently modelling a complex dynamic systems. They are an option for identification and control, since they are easy to implement with a relatively simple structure. Moreover, they are able to modify its own parameters online by a suitable adaptation algorithm.

Similarly to continuos RHONN, discrete high order neural networks (Figure 3.5) are defined by:

$$x_i(k+1) = w_i^T z_I(x(k), u(k)),\ i = 1, \cdots, n \tag{3.14}$$

where $x_i(k)$ is the state of the i-th neuron in the k iteration; w_i is the online adaptive weight vector; $d_j(k)$ are nonnegative integers and $z_i(x(k), u(k))$ is defined by equation (3.10) as well.

Discrete neural networks have the same characteristics than the continuous RHONN. Therefore, they are good candidates for using with complex discrete dynamic systems.

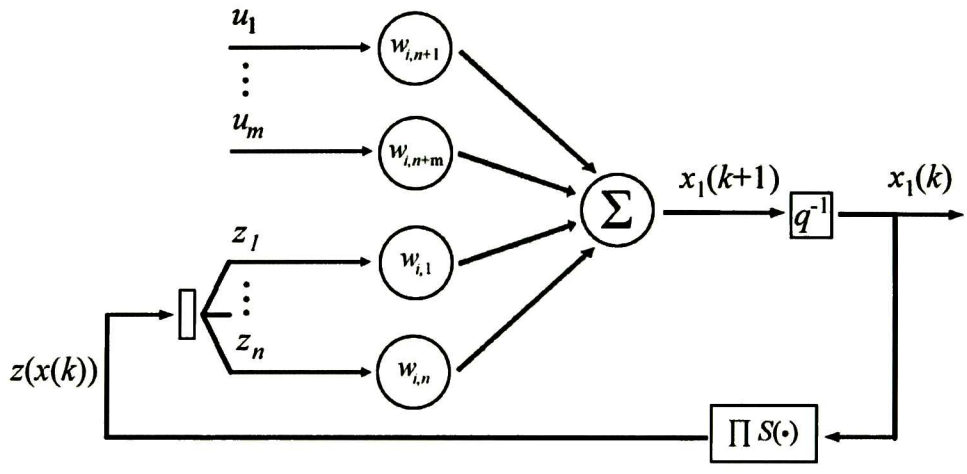


Figure 3.5: Discrete Recurrent High Order Neural Network (RHONN) structure

3.3 Identification

A network structure was proposed in order to identify the system (in this case the 5DOF rendundant robot). This network was created from the state space representation (2.24) which is in the nonlinear control block form (NLCB).

The estimated states $\hat{x}_1$, $\hat{x}_2$, $\hat{x}_3$, $\hat{x}_4$ and $\hat{x}_5$ are part of the first block of the system (2.24). The estimated states $\hat{x}_6$, $\hat{x}_7$, $\hat{x}_8$, $\hat{x}_9$ and $\hat{x}_{10}$ are part of the second block of the system.

The states of the second block are controlled by the input τ_i. The states of the first block are controlled through the states $\hat{x}_6$, $\hat{x}_7$, $\hat{x}_8$, $\hat{x}_9$ and $\hat{x}_{10}$ which are represented in the network by:

$$w_{i,2}(k)x_{i+5}(k) \tag{3.15}$$

where $i = 1, 2, 3, 4$ and 5.

The states of the second block $F(\chi_I,\chi_{II}) + G(\chi_I)u$ are represented by the following expression:

$$\begin{aligned} &w_{i+5,1}(k)\tanh[x_{i+5}(k)] + w_{i+5,2}(k)\tanh[x_i(k)]^2\tanh[x_{i+5}(k)] + w_{i+5,3}(k)\tanh[x_i(k)]^2\ldots \\ &+w_{i+5,4}(k)\tanh[x_i(k)]^2\tanh[x_{i+5}(k)] + w_{i+5,5}(k)\tau_i(k) \end{aligned} \tag{3.16}$$

were $i = 1, 2, 3, 4$ and 5.

The neural network proposed to identify the system was obtained using (3.15) and (3.16) in (2.24). This neural network is thus the following:

$$
\begin{aligned}
\hat{x}_1(k+1) &= w_{11}(k)\tanh[x_1(k)] + w_{12}(k)x_6(k), \qquad (3.17)\\
\hat{x}_2(k+1) &= w_{21}(k)\tanh[x_2(k)] + w_{22}(k)x_7(k),\\
\hat{x}_3(k+1) &= w_{31}(k)\tanh[x_3(k)] + w_{32}(k)x_8(k),\\
\hat{x}_4(k+1) &= w_{41}(k)\tanh[x_4(k)] + w_{42}(k)x_9(k),\\
\hat{x}_5(k+1) &= w_{51}(k)\tanh[x_5(k)] + w_{52}(k)x_{10}(k),\\
\hat{x}_6(k+1) &= w_{61}(k)\tanh[x_6(k)] + w_{62}(k)\tanh[x_1(k)]^2\tanh[x_6(k)] + w_{63}(k)\tanh[x_1(k)]^2\cdots\\
&\quad + w_{64}(k)\tanh[x_1(k)]^2\tanh[x_6(k)] + w_{65}(k)\tau_1(k),\\
\hat{x}_7(k+1) &= w_{71}(k)\tanh[x_7(k)] + w_{72}(k)\tanh[x_2(k)]^2\tanh[x_7(k)] + w_{73}(k)\tanh[x_2(k)]^2\cdots\\
&\quad + w_{74}(k)\tanh[x_2(k)]^2\tanh[x_7(k)] + w_{75}(k)\tau_2(k),\\
\hat{x}_8(k+1) &= w_{81}(k)\tanh[x_8(k)] + w_{82}(k)\tanh[x_3(k)]^2\tanh[x_8(k)] + w_{83}(k)\tanh[x_3(k)]^2\cdots\\
&\quad + w_{84}(k)\tanh[x_3(k)]^2\tanh[x_8(k)] + w_{85}(k)\tau_3(k),\\
\hat{x}_9(k+1) &= w_{91}(k)\tanh[x_9(k)] + w_{92}(k)\tanh[x_4(k)]^2\tanh[x_9(k)] + w_{93}(k)\tanh[x_4(k)]^2\cdots\\
&\quad + w_{94}(k)\tanh[x_4(k)]^2\tanh[x_9(k)] + w_{95}(k)\tau_4(k),\\
\hat{x}_{10}(k+1) &= w_{101}(k)\tanh[x_{10}(k)] + w_{102}(k)\tanh[x_5(k)]^2\tanh[x_{10}(k)] + w_{103}(k)\tanh[x_5(k)]^2\cdots\\
&\quad + w_{104}(k)\tanh[x_5(k)]^2\tanh[x_{10}(k)] + w_{105}(k)\tau_5(k),
\end{aligned}
$$

Representing this structure (3.17) in blocks, we have:

$$
\begin{aligned}
\hat{X}_I(k+1) &= f_I[x(k), w(k)] \qquad (3.18)\\
\hat{X}_{II}(k+1) &= f_{II}[x(k), w(k)] + g_{II}[w(k)]\tau(k)
\end{aligned}
$$

From the structure (3.17) we get the following weight matrix:

$$\begin{bmatrix} w_{11} & w_{12} & 0 & 0 & 0 \\ w_{21} & w_{22} & 0 & 0 & 0 \\ w_{31} & w_{32} & 0 & 0 & 0 \\ w_{41} & w_{42} & 0 & 0 & 0 \\ w_{51} & w_{52} & 0 & 0 & 0 \\ w_{61} & w_{62} & w_{63} & w_{64} & w_{65} \\ w_{71} & w_{72} & w_{73} & w_{74} & w_{75} \\ w_{81} & w_{82} & w_{83} & w_{84} & w_{85} \\ w_{91} & w_{92} & w_{93} & w_{94} & w_{95} \\ w_{101} & w_{102} & w_{103} & w_{104} & w_{105} \end{bmatrix}$$

These weights are used in the RHONN to identify the plant. Figure 3.6 shows the structure used in order to identify the plant using neural identification trained with the Kalman filter.

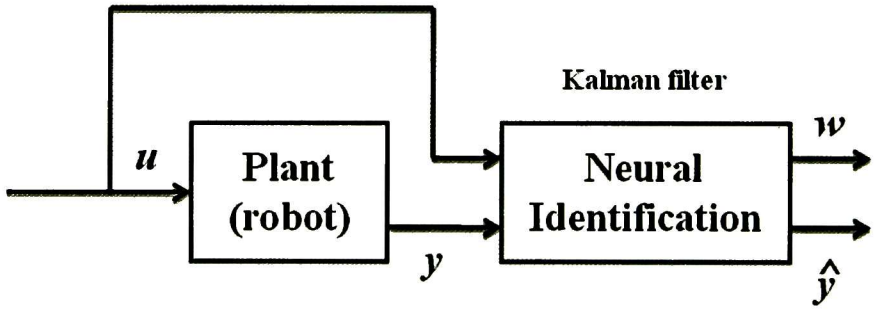


Figure 3.6: Identification structure

In this case, the Kalman filter is used in order to train the RHONN [3]. The Kalman filter equations for neural networks training [12] are:

$$K(k) = P(k)H^T(k)[R(k) + H(k)P(k)H^T(k)]^{-1} \quad (3.19)$$

$$w(k+1) = w(k) + K(k)[y(k) - \hat{y}(k)] \quad (3.20)$$

$$P(k+1) = P(k) - K(k)H(k)P(k) + Q(k) \quad (3.21)$$

$$e(k) = y(k) - \hat{y}(k)$$

where $P(k) \in \Re^{L \times L}$ and $P(k+1) \in \Re^{L \times L}$ represent the covariance error matrix of the prediction in the iterations k and $k+1$ respectively; $w \in \Re^L$ is the weight (states) vector; L is the total number of weights in the neural network; $y \in \Re^m$ is the output vector, where m is the total number of outputs; $\hat{y} \in \Re^m$ is the output of the neural network; $K \in \Re^{L \times m}$ is the matrix of Kalman gain; $Q \in \Re^{L \times L}$ is the covariance matrix

of the process noise; $R \in \Re^{m \times m}$ is the covariance matrix of the measurement noise, $e(k)$ is the error. $H \in \Re^{m \times L}$ is the matrix of the derivatives of every output of the neural network ($\hat{y}_i$) with respect to each of the (w_j) weights, defined as follows:

$$H_{ij}(k) = [\frac{\partial \hat{y}_i(k)}{\partial w_j(k)}]_{w(k)=\hat{w}(k+1)}, i = 1 \cdots m,\ j = 1 \cdots L \tag{3.22}$$

3.4 Neural control

Let Z_I be the tracking error for each joint of the first block of (3.18) at the instant k:

$$Z_I(k) = X_I(k) - X_I^d(k) \tag{3.23}$$

where:

$$X_I(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ x_4(k) \\ x_5(k) \end{bmatrix} \quad X_I^d(k) = \begin{bmatrix} x_1^d(k) \\ x_2^d(k) \\ x_3^d(k) \\ x_4^d(k) \\ x_5^d(k) \end{bmatrix} \tag{3.24}$$

The error for the instant $k+1$ of the first block is:

$$Z_I(k+1) = K_I Z_I(k) \tag{3.25}$$

where K_I is chosen to be a Schur matrix:

$$K_I = \begin{bmatrix} k_1 & 0 & 0 & 0 & 0 \\ 0 & k_2 & 0 & 0 & 0 \\ 0 & 0 & k_3 & 0 & 0 \\ 0 & 0 & 0 & k_4 & 0 \\ 0 & 0 & 0 & 0 & k_5 \end{bmatrix} \tag{3.26}$$

The block diagram for this neural control is shown in Figure 3.7. This structure has the following main elements: the robot system, the neural identification, the desired trajectory and finally the neural control.

Replacing (3.24) in (3.23) and expanding the expression, the following equation is obtained:

$$Z_I(k+1) = \begin{bmatrix} z_1(k+1) \\ z_2(k+1) \\ z_3(k+1) \\ z_4(k+1) \\ z_5(k+1) \end{bmatrix} = \begin{bmatrix} \hat{x}_1(k+1) \\ \hat{x}_2(k+1) \\ \hat{x}_3(k+1) \\ \hat{x}_4(k+1) \\ \hat{x}_5(k+1) \end{bmatrix} - \begin{bmatrix} x_1^d(k+1) \\ x_2^d(k+1) \\ x_3^d(k+1) \\ x_4^d(k+1) \\ x_5^d(k+1) \end{bmatrix} \tag{3.27}$$

In order to know the value of X_I at the instant $k+1$, the estimated values $\hat{x}_i(k+1) = w_{i1}(k)\tanh[x_i(k)] + w_{i2}(k)x_{i+5}(k)$ are replaced in (3.27), to get:

$$\begin{bmatrix} z_1(k+1) \\ z_2(k+1) \\ z_3(k+1) \\ z_4(k+1) \\ z_5(k+1) \end{bmatrix} = \begin{bmatrix} w_{11}(k)\tanh[x_1(k)] + w_{12}(k)x_6(k) - x_1^d(k+1) \\ w_{21}(k)\tanh[x_2(k)] + w_{22}(k)x_7(k) - x_2^d(k+1) \\ w_{31}(k)\tanh[x_3(k)] + w_{32}(k)x_8(k) - x_3^d(k+1) \\ w_{41}(k)\tanh[x_4(k)] + w_{42}(k)x_9(k) - x_4^d(k+1) \\ w_{51}(k)\tanh[x_5(k)] + w_{52}(k)x_{10}(k) - x_5^d(k+1) \end{bmatrix} \tag{3.28}$$

The quasi-control in the first block is given by x_6, x_7, x_8, x_9 and x_{10}. From (3.28), the quasy-control is calculated to produce the desired dynamic of the system. Therefore, the quasi-control for the first block error is given by:

$$\begin{bmatrix} x_1^d(k) \\ x_2^d(k) \\ x_3^d(k) \\ x_4^d(k) \\ x_5^d(k) \end{bmatrix} = \begin{bmatrix} w_{12}(k) & 0 & 0 & 0 & 0 \\ 0 & w_{22}(k) & 0 & 0 & 0 \\ 0 & 0 & w_{32}(k) & 0 & 0 \\ 0 & 0 & 0 & w_{42}(k) & 0 \\ 0 & 0 & 0 & 0 & w_{52}(k) \end{bmatrix}^{-1} \cdots$$
$$\begin{bmatrix} k_1 z_1(k) + x_1^d(k+1) - w_{11}(k)\tanh[x_1(k)] \\ k_2 z_2(k) + x_2^d(k+1) - w_{21}(k)\tanh[x_2(k)] \\ k_3 z_3(k) + x_3^d(k+1) - w_{31}(k)\tanh[x_3(k)] \\ k_4 z_4(k) + x_4^d(k+1) - w_{41}(k)\tanh[x_4(k)] \\ k_5 z_5(k) + x_5^d(k+1) - w_{51}(k)\tanh[x_5(k)] \end{bmatrix} \tag{3.29}$$

Let $Z_{II}(k)$ be the tracking error of the second block (3.18) at the instant k:

$$Z_{II}(k) = X_{II}(k) - X_{II}^d(k) \tag{3.30}$$

where:

$$X_{II}(k) = \begin{bmatrix} x_6(k) \\ x_7(k) \\ x_8(k) \\ x_9(k) \\ x_{10}(k) \end{bmatrix}, X_{II}^d(k) = \begin{bmatrix} x_6^d(k) \\ x_7^d(k) \\ x_8^d(k) \\ x_9^d(k) \\ x_{10}^d(k) \end{bmatrix} \tag{3.31}$$

The error tracking Z_{II} at the instant $k+1$ is:

$$Z_{II}(k+1) = X_{II}(k+1) - X_{II}^d(k+1) \tag{3.32}$$

where $Z_{II}(k+1)$ is given by:

$$Z_{II}(k+1) = \begin{bmatrix} z_6(k+1) \\ z_7(k+1) \\ z_8(k+1) \\ z_9(k+1) \\ z_{10}(k+1) \end{bmatrix} \tag{3.33}$$

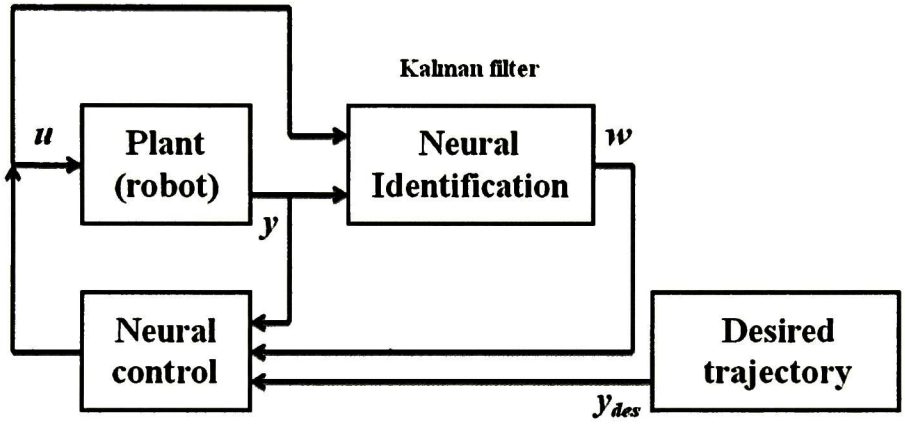


Figure 3.7: Neural control structure

Replacing the following estimated values:

$$w_{i+5,1}(k)\tanh[x_{i+5}(k)] + w_{i+5,2}(k)\tanh[x_i(k)]^2\tanh[x_{i+5}(k)]\ldots$$
$$+w_{i+5,3}(k)\tanh[x_i(k)]^2 + w_{i+5,4}(k)\tanh[x_i(k)]^2\tanh[x_{i+5}(k)]$$

in (3.32) and expanding the expression we obtain:

$$
\begin{bmatrix} z_6(k+1) \\ z_7(k+1) \\ z_8(k+1) \\ z_9(k+1) \\ z_{10}(k+1) \end{bmatrix} = \begin{bmatrix} w_{61}(k)\tanh[x_6(k)] + w_{62}(k)\tanh[x_1(k)]^2\tanh[x_6(k)]\ldots \\ +w_{63}(k)\tanh[x_1(k)]^2 + w_{64}(k)\tanh[x_1(k)]^2\tanh[x_6(k)] \\ \\ w_{71}(k)\tanh[x_7(k)] + w_{72}(k)\tanh[x_2(k)]^2\tanh[x_7(k)]\ldots \\ +w_{73}(k)\tanh[x_2(k)]^2 + w_{74}(k)\tanh[x_2(k)]^2\tanh[x_7(k)] \\ \\ w_{81}(k)\tanh[x_8(k)] + w_{82}(k)\tanh[x_3(k)]^2\tanh[x_8(k)]\ldots \\ +w_{83}(k)\tanh[x_3(k)]^2 + w_{84}(k)\tanh[x_3(k)]^2\tanh[x_8(k)] \\ \\ w_{91}(k)\tanh[x_9(k)] + w_{92}(k)\tanh[x_4(k)]^2\tanh[x_9(k)]\ldots \\ +w_{93}(k)\tanh[x_4(k)]^2 + w_{94}(k)\tanh[x_4(k)]^2\tanh[x_9(k)] \\ \\ w_{101}(k)\tanh[x_{10}(k)] + w_{102}(k)\tanh[x_5(k)]^2\tanh[x_{10}(k)]\ldots \\ +w_{103}(k)\tanh[x_5(k)]^2 + w_{104}(k)\tanh[x_5(k)]^2\tanh[x_{10}(k)] \end{bmatrix} \ldots \quad (3.34)
$$

$$
+ \begin{bmatrix} w_{12}(k+1) & 0 & 0 & 0 & 0 \\ 0 & w_{22}(k+1) & 0 & 0 & 0 \\ 0 & 0 & w_{32}(k+1) & 0 & 0 \\ 0 & 0 & 0 & w_{42}(k+1) & 0 \\ 0 & 0 & 0 & 0 & w_{52}(k+1) \end{bmatrix}^{-1} \ldots
$$

$$
\begin{bmatrix} k_1 z_1(k+1) + x_1^d(k+2) - w_{11}(k+1)\tanh[x_1(k+1)] \\ k_2 z_2(k+1) + x_2^d(k+2) - w_{21}(k+1)\tanh[x_2(k+1)] \\ k_3 z_3(k+1) + x_3^d(k+2) - w_{31}(k+1)\tanh[x_3(k+1)] \\ k_4 z_4(k+1) + x_4^d(k+2) - w_{41}(k+1)\tanh[x_4(k+1)] \\ k_5 z_5(k+1) + x_5^d(k+2) - w_{51}(k+1)\tanh[x_5(k+1)] \end{bmatrix} \ldots
$$

$$
+ \begin{bmatrix} w_{65}(k) & 0 & 0 & 0 & 0 \\ 0 & w_{75}(k) & 0 & 0 & 0 \\ 0 & 0 & w_{85}(k) & 0 & 0 \\ 0 & 0 & 0 & w_{95}(k) & 0 \\ 0 & 0 & 0 & 0 & w_{105}(k) \end{bmatrix} \begin{bmatrix} \tau_1(k) \\ \tau_2(k) \\ \tau_3(k) \\ \tau_4(k) \\ \tau_5(k) \end{bmatrix}
$$

Finally, if the error tracking $Z_{II}(k+1) = 0$ in (3.34), then we obtain the following neural control τ:

$$\tau = \begin{bmatrix} \tau_1(k) \\ \tau_2(k) \\ \tau_3(k) \\ \tau_4(k) \\ \tau_5(k) \end{bmatrix} = - \begin{bmatrix} w_{65}(k) & 0 & 0 & 0 & 0 \\ 0 & w_{75}(k) & 0 & 0 & 0 \\ 0 & 0 & w_{85}(k) & 0 & 0 \\ 0 & 0 & 0 & w_{95}(k) & 0 \\ 0 & 0 & 0 & 0 & w_{105}(k) \end{bmatrix}^{-1} \cdots \quad (3.35)$$

$$\begin{bmatrix} w_{61}(k)\tanh[x_6(k)] + w_{62}(k)\tanh[x_1(k)]^2\tanh[x_6(k)]\ldots \\ +w_{63}(k)\tanh[x_1(k)]^2 + w_{64}(k)\tanh[x_1(k)]^2\tanh[x_6(k)] \\ \\ w_{71}(k)\tanh[x_7(k)] + w_{72}(k)\tanh[x_2(k)]^2\tanh[x_7(k)]\ldots \\ +w_{73}(k)\tanh[x_2(k)]^2 + w_{74}(k)\tanh[x_2(k)]^2\tanh[x_7(k)] \\ \\ w_{81}(k)\tanh[x_8(k)] + w_{82}(k)\tanh[x_3(k)]^2\tanh[x_8(k)]\ldots \\ +w_{83}(k)\tanh[x_3(k)]^2 + w_{84}(k)\tanh[x_3(k)]^2\tanh[x_8(k)] \\ \\ w_{91}(k)\tanh[x_9(k)] + w_{92}(k)\tanh[x_4(k)]^2\tanh[x_9(k)]\ldots \\ +w_{93}(k)\tanh[x_4(k)]^2 + w_{94}(k)\tanh[x_4(k)]^2\tanh[x_9(k)] \\ \\ w_{101}(k)\tanh[x_{10}(k)] + w_{102}(k)\tanh[x_5(k)]^2\tanh[x_{10}(k)]\ldots \\ +w_{103}(k)\tanh[x_5(k)]^2 + w_{104}(k)\tanh[x_5(k)]^2\tanh[x_{10}(k)] \end{bmatrix}$$

$$+ \begin{bmatrix} w_{12}(k+1) & 0 & 0 & 0 & 0 \\ 0 & w_{22}(k+1) & 0 & 0 & 0 \\ 0 & 0 & w_{32}(k+1) & 0 & 0 \\ 0 & 0 & 0 & w_{42}(k+1) & 0 \\ 0 & 0 & 0 & 0 & w_{52}(k+1) \end{bmatrix}^{-1} \ldots$$

$$\begin{bmatrix} k_1 z_1(k+1) + x_1^d(k+2) - w_{11}(k+1)\tanh[x_1(k+1)] \\ k_2 z_2(k+1) + x_2^d(k+2) - w_{21}(k+1)\tanh[x_2(k+1)] \\ k_3 z_3(k+1) + x_3^d(k+2) - w_{31}(k+1)\tanh[x_3(k+1)] \\ k_4 z_4(k+1) + x_4^d(k+2) - w_{41}(k+1)\tanh[x_4(k+1)] \\ k_5 z_5(k+1) + x_5^d(k+2) - w_{51}(k+1)\tanh[x_5(k+1)] \end{bmatrix}$$

3.5 Simulation results

Figure 3.8 shows the identification error obtained from the neural network (3.17) proposed to identify the system (2.24). This RHONN (3.17) is trained by the Kalman filter (3.19). The results show how the identification error is close to zero, as desired.

We can see in this figure that the identification error obtained by the neural network tends to zero for all the states of the system (the five degrees of freedom redundant robot).

For this control (3.35), we use the following Schur matrix (3.36) in order to reach

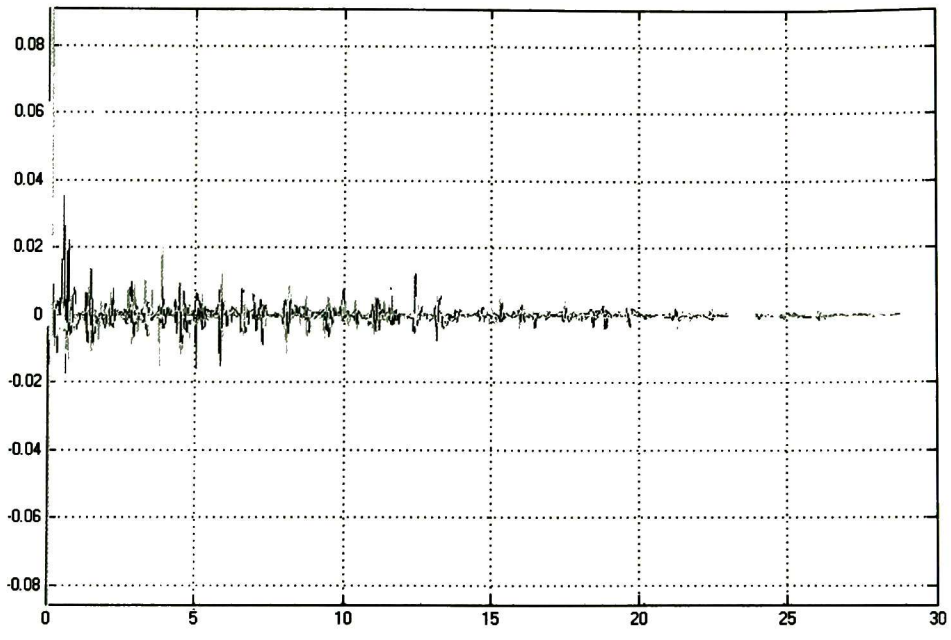


Figure 3.8: Identification error

the desired trajectory with a tracking error close to zero:

$$K_I = \begin{bmatrix} 0.99 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & 0 & 0 & 0 \\ 0 & 0 & 0.4 & 0 & 0 \\ 0 & 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0 & 0.1 \end{bmatrix} \tag{3.36}$$

The desired trajectory q_{ref} to reach, is the following:

$$q_{ref} = \begin{bmatrix} 0.15 \\ 10\frac{\pi}{180}\sin(0.5t) \\ -10\frac{\pi}{180}\cos(0.5t) \\ 6\frac{\pi}{180}\sin(0.5t) \\ 8\frac{\pi}{180}\sin(0.5t) \end{bmatrix} \tag{3.37}$$

First, we use this neural control (3.35) without disturbances and we obtain the signals of Figures 3.9 to 3.11. Figure 3.9 shows simulation results of desired positions q_{ref} (continuous line) versus output positions q (dotted line) for each of the five links

of the five degrees of freedom redundant robot, where q_1 to q_5 represent the position of each of the five links of the system. This figure shows how the real positions converge to the desired positions.

Figure 3.10 shows the control signals τ_1 to τ_5 obtained in the simulation of this neural control without disturbances for each of the five links of the robot. Finally, we can see the tracking error e_i for the neural control in Figure 3.11. This figure shows how all this tracking errors e_1 to e_5, obtained from the neural control, tend to zero.

Then, we use this neural control (3.35) with two disturbances and we obtain the signals of Figures 3.12 to 3.14. The first disturbance occurs around the second 20 and the second disturbance occurs around the second 55.

Figure 3.12 shows the desired positions q_{ref} (continuous line) and the output positions q (dotted line) for each of the five links of the robotic arm (with these two disturbances). We can see how the real positions converge to the desired positions before the first disturbance. After the first disturbance, which occurs around the second 20, the current positions are separated from the desired positions due to the disturbance, but eventually they converge to the desired positions.

Around the second 55, the current positions are separated from the desired positions again due to the second disturbance in the system. However, the current positions converge again to the desired positions.

Figure 3.13 shows the control signals for each of the five links τ_1 to τ_5. We can see in this figure how the control signals are adjusted in order to correct the disturbances around the second 20 and the second 55.

Finally, Figure 3.14 shows the tracking error for each of the five links of the system, e_1 to e_5. We can see how the tracking error tends to zero at the beginning of the simulation. Then, the tracking error increases around the second 20 due to the first disturbance. After this, the error tends to zero but around the second 55 this tracking error increases again due to the second disturbance in the system. Finally, the tracking error tends to zero.

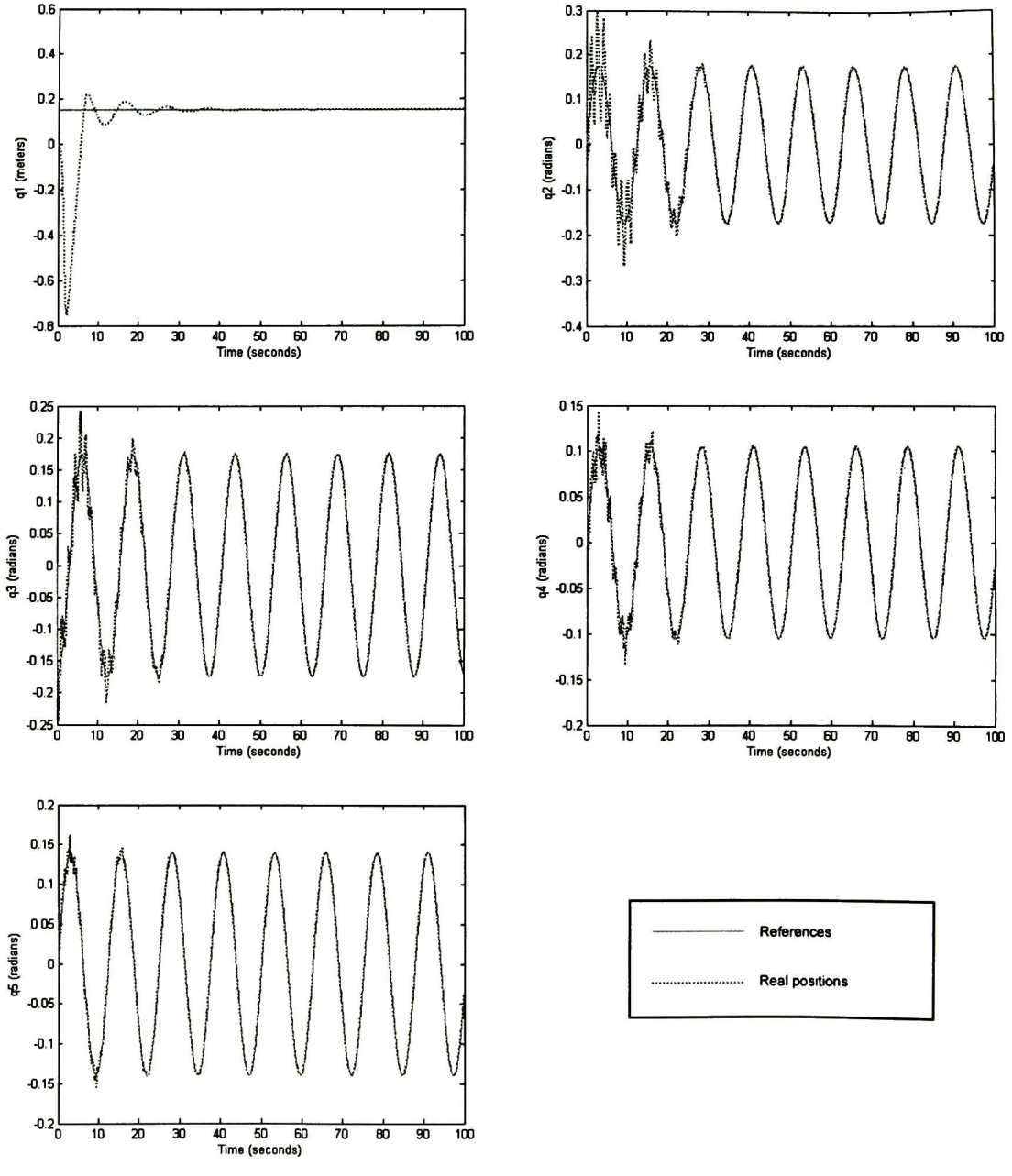


Figure 3.9: Positions (q_i vs. q_{i_ref})

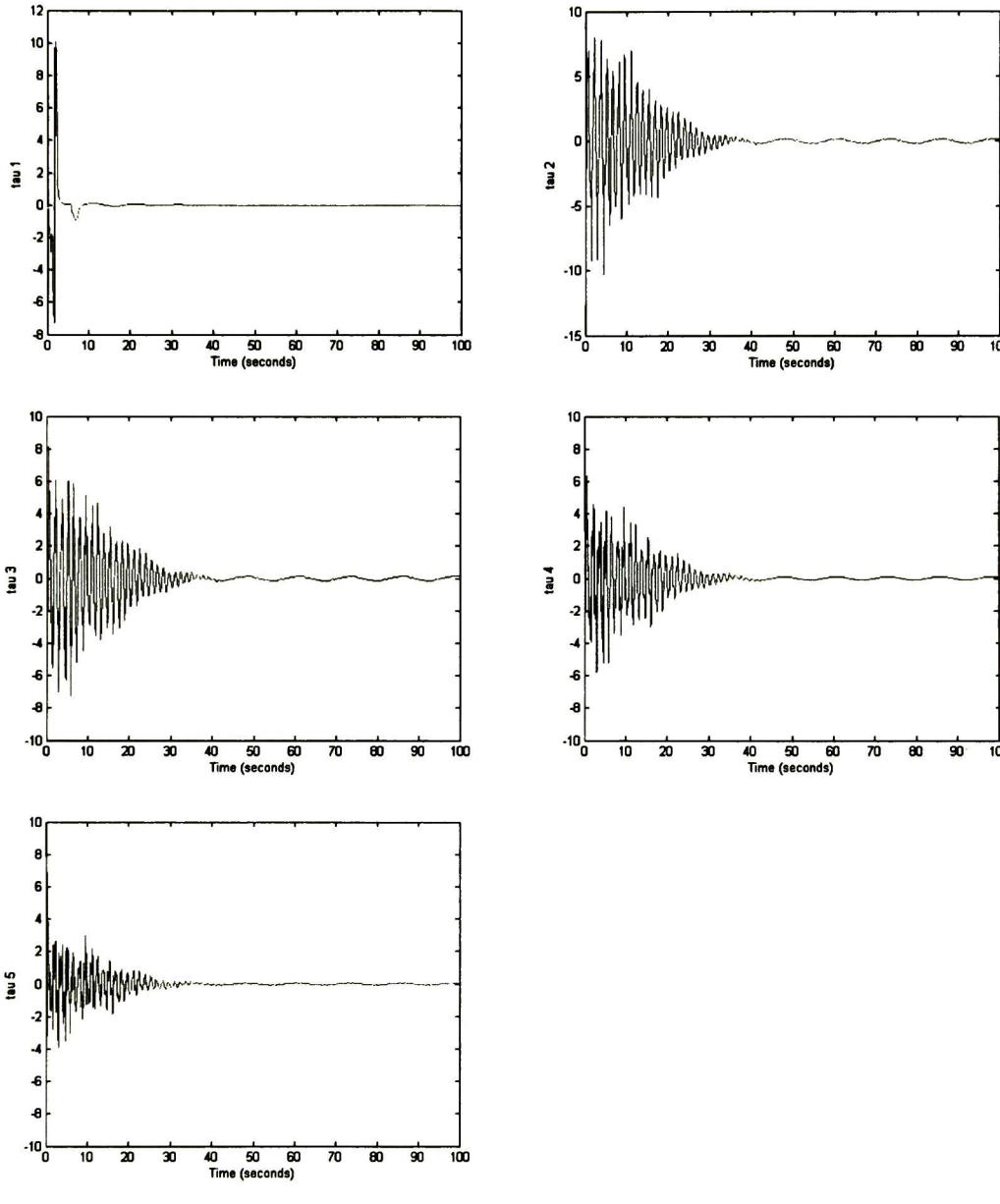


Figure 3.10: Control signals (τ_i)

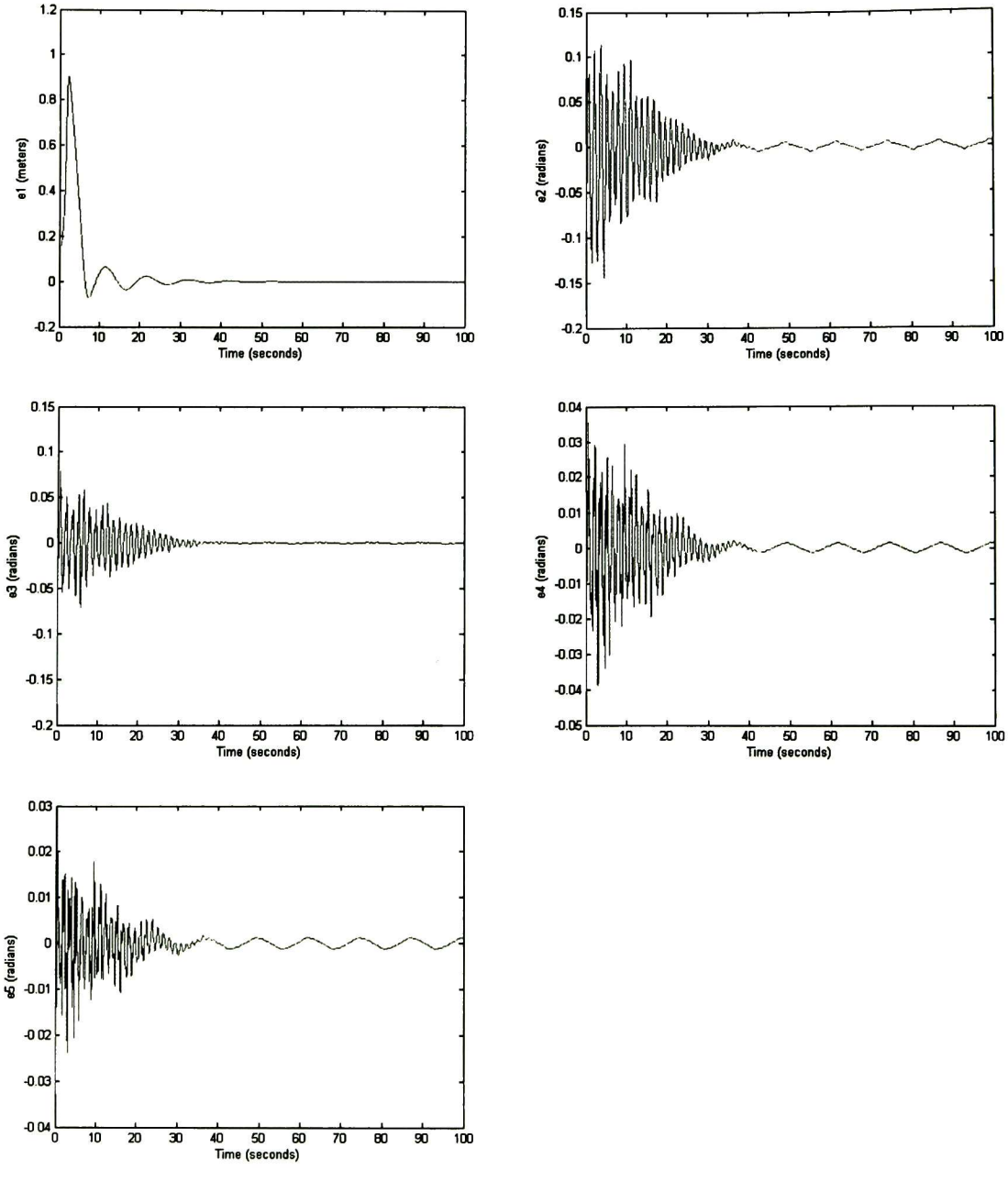


Figure 3.11: Tracking error (e_i)

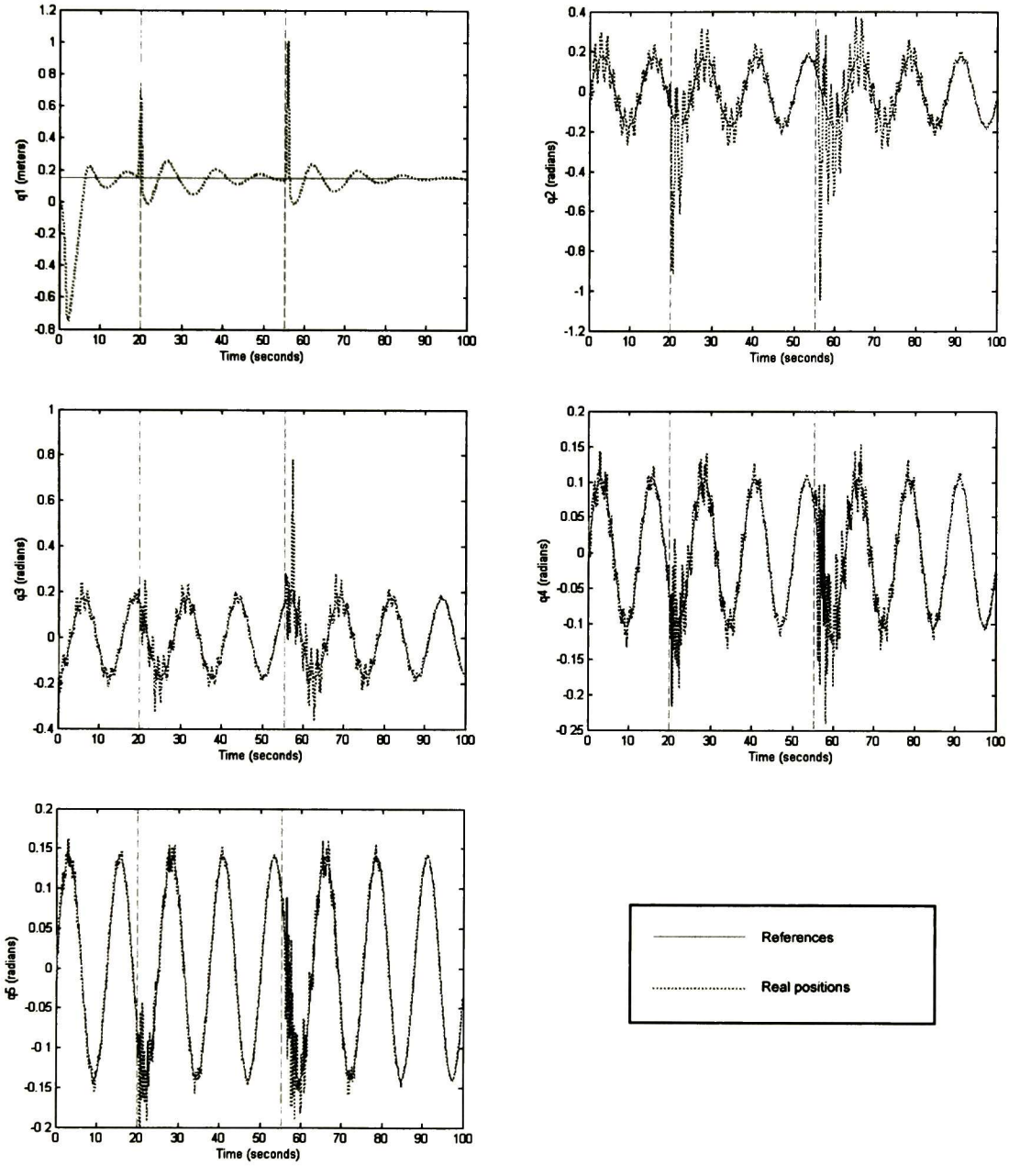


Figure 3.12: Positions (q_i vs. q_{i_ref}) with disturbances

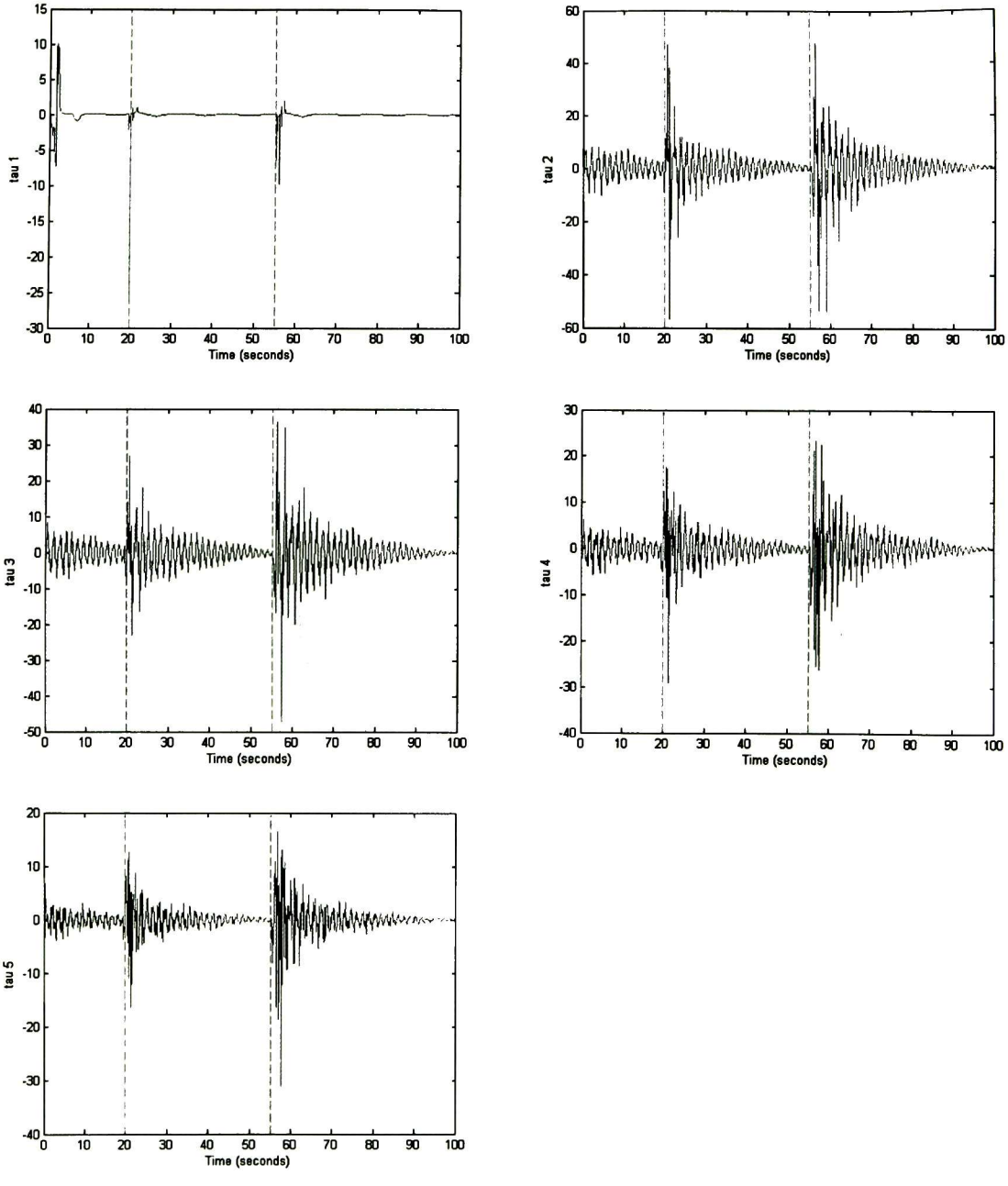


Figure 3.13: Control signals (τ_i) with disturbances

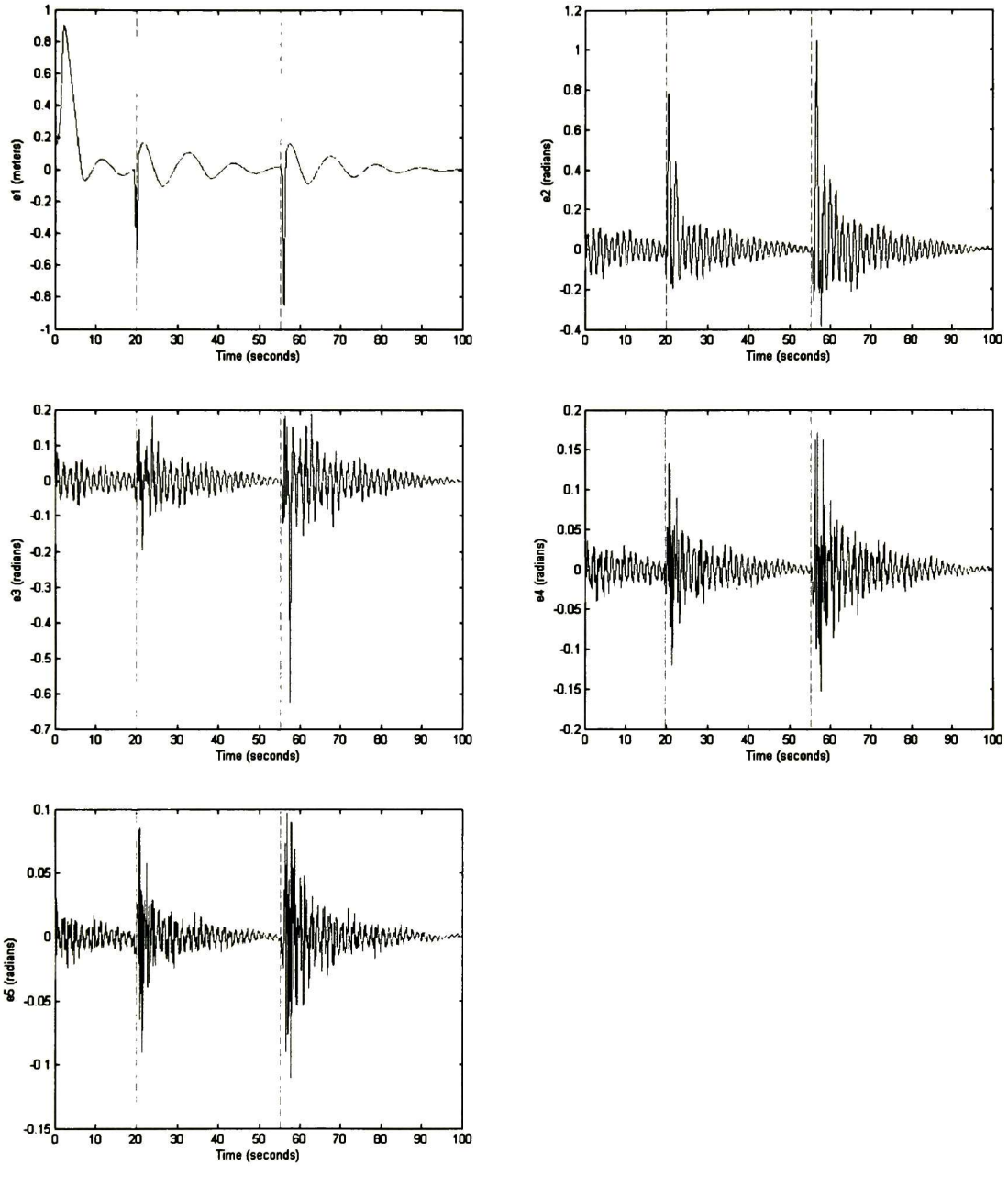


Figure 3.14: Tracking (e_i) with disturbances

Chapter 4

Fuzzy control

4.1 Background

In a classical (crisp) set, if U is the universe of discourse, i.e. contains all the possible elements in a context or a particular application. A classical set A inside the universe of discourse U may be defined by a list of its elements (list method) or by specifying the properties that an element has to satisfy in order to be part of the set (rule method). The first method can be used only in finite sets, therefore, its use is limited. Instead, the second method is more general, where a set A can be represented by:

$$A = \{x \in U | x \text{ satisfies some conditions}\} \tag{4.1}$$

There exists a third method to define a classical set A by a Membership Function (MF) taking values zero or one, denoted by $\mu_A(x)$, where:

$$\mu_A(x) = \begin{cases} 1 \text{ if } x \in A \\ 0 \text{ if } x \notin A \end{cases} \tag{4.2}$$

The set A is mathematically equivalent to the membership function $\mu_A(x)$ in the sense that knowing $\mu_A(x)$ is equivalent to know the set A itself. In a classical set, an element may belong to a set ($\mu_A(x) = 1$) or not ($\mu_A(x) = 0$).

In contrast to a classical set, a fuzzy set is a set where a sharp boundary does not exist. This is, the transition of *belonging* and *not belonging* to a set is gradual and this transition is characterized by the membership function, which give to the fuzzy system, the flexibility to model linguistic expressions, for example: *"the water is cold", "the temperature is hot"*. These examples use the words *"cold"* and *"hot"* to describe *"the water"* and *"the temperature"*, this is, the variables *"the water"* and *"the temperature"* take as values *"cold"* and *"hot"* respectively. If a variable can take words of a natural language as is values (cold, warm, hot; slow, fast, etc.), then, it is called a linguistic variable where its values are characterized in the fuzzy sets defined inside the universe of discourse, where the variables are defined [6] [16].

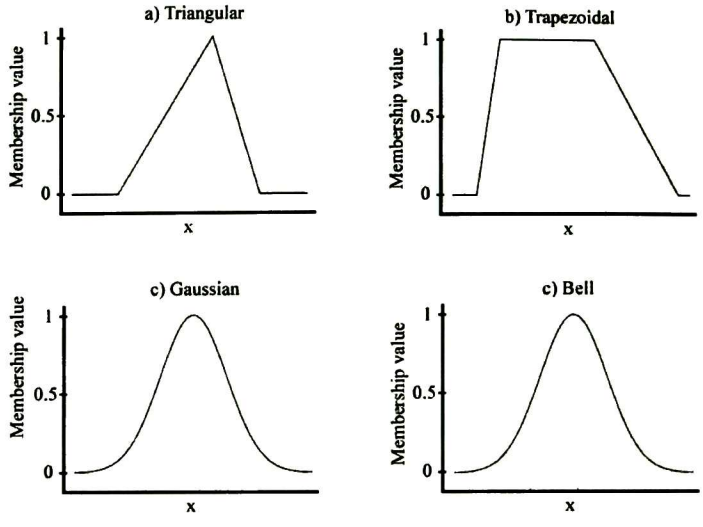


Figure 4.1: Membership functions

A linguistic variable is defined by (X, T, V, M), where X is the name of the linguistic variable (*"temperature of the water"*), T is the set of linguistic values that can be taken by X, for example: T ={cold, warm, hot}. V is the physical domain where the variable X take its quantitative values: V =[0°, 30°]. M is a semantic rule that relates each linguistic value in T with a fuzzy set in V. In this example *"cold"*, *"warm"* and *"hot"* are related by the membership functions.

If X is a collection of objects denoted by x, then the fuzzy set A in X is defined by the following ordered pair:

$$A = \{(x, \mu_A(x)) | x \in X\} \tag{4.3}$$

where the membership function $\mu_A(x)$ relates each element of X with a continuos membership value between 0 and 1. Membership functions can be: triangular, trapezoidal, Gaussian, bell shaped or sigmoidal.

Triangular and trapezoidal membership functions have been extensively used because they have simple formulas and a good computational efficiency. They are used especially in real time applications.

A **triangular** membership function is defined by three parameters {a, b, c} which determine the three corners of the triangle:

$$triangle(x; a, b, c) = max(min(\frac{x-a}{b-a}, \frac{c-x}{c-b}), 0) \tag{4.4}$$

A **trapezoidal** membership function is defined by the four parameters $\{a, b, c, d\}$ as follows:

$$trapezoid(x; a, b, c, d) = max(min(\frac{x-a}{b-a}, 1, \frac{c-x}{c-b}), 0) \tag{4.5}$$

Fuzzy sets, which are loosely described, play an important role in human thinking, in particular in the pattern recognition area, communications and abstraction.

A fuzzy system is a knowledge based or a rule based system. The center of the fuzzy system is the base of knowledge which consists in the IF-THEN fuzzy rules. These rules are IF-THEN declarations where a few words are defined by continuous membership functions. For example, *IF the temperature of the water is cold, THEN increase the heat.*

4.2 Takagi-Sugeno fuzzy model

The fuzzy model proposed by Takagi and Sugeno is composed by IF-THEN rules which represent local linear input-output relations of a nonlinear system. The main characteristic of a Takagi-Sugeno fuzzy model is to express dynamic characteristics in a local form of each implication rule by a linear model of the system.

The i rules that conform a Takagi-Sugeno model are as follows:

Rule i of the model for a fuzzy continuos system:
IF $z_1(t)$ is M_{i1} and $\cdots$ and $z_p(t)$ is M_{ip},

$$\text{THEN} \begin{cases} \dot{x}(t) = A_i x(t) + B_i u(t) \\ y(t) = C_i x(t), \end{cases} \qquad i = 1, 2, \cdots, r$$

Rule i of the model for a discrete fuzzy system:
IF $z_1(k)$ is M_{i1} and $\cdots$ and $z_p(k)$ is M_{ip},

$$\text{THEN} \begin{cases} x(k+1) = A_i x(k) + B_i u(k) \\ y(k) = C_i x(k), \end{cases} \qquad i = 1, 2, \cdots, r$$

Here, M_{ij} is the fuzzy set and r is the number of rules of the model; $\dot{x} \in R^n$ is the state vector; $u \in R^m$ is the input vector; $y \in R^q$ is the output vector. $A_i \in R^{n \times n}$, $Bi \in R^{n \times m}$, and $Ci \in R^{q \times n}$; $z_1, \cdots, z_p$ known are premise variables, which are functions of the state variables, external disturbances or time disturbances; z is a vector which contains all the independent elements $z_1, \cdots, z_p$.

The Takagi-Sugeno fuzzy model [10] is a method to approximate nonlinear systems as linear systems. Then, linear control strategies are developed, for each linear system, that stabilize each subsystem separately.

In order to design a fuzzy control (4.2), it is necessary to have the fuzzy model of the nonlinear system. Therefore, the construction of the fuzzy model represents an important and basic step.

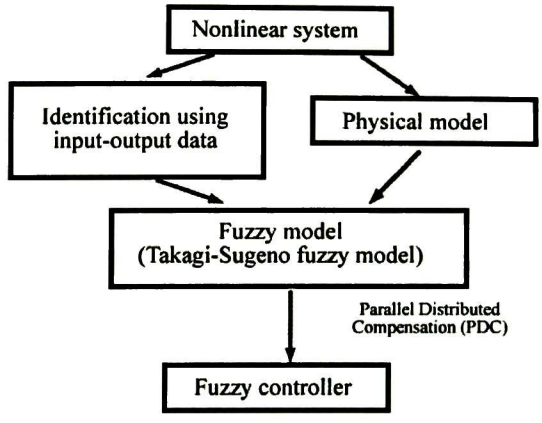


Figure 4.2: Diagram for obtaining a fuzzy controller.

In general, there exist two approaches to build a fuzzy model:
1. Identification of the fuzzy model by using input-output data.
2. To obtain the fuzzy model from the nonlinear equations of the model.

The second approach uses the idea of *nonlinear sector*, *local approximation* or a combination of both in order to get the fuzzy model.

The nonlinear sector idea to design a fuzzy model appeared for first time in [6]. A non-linear sector is based in the following concept: Let $\dot{x}(t) = f(x(t))$ be a simple nonlinear system, where $f(0) = 0$. The objective is to find a global sector which satisfies the following expression:

$$\dot{x}(t) = f(x(t)) \in [a_1, a_2]x(t)$$

Figure 4.3 [14] shows the nonlinear sector approach. This approach guarantees an exact fuzzy model construction.

Even though, sometimes it is difficult to find a global sector in general nonlinear systems. In this case, local nonlinear sectors are considered. This is because variables of physical systems are always bounded. Figure 4.4 [14] shows the local nonlinear sector, where two lines make the local nonlinear sector with $-d < x(t) < d$. The fuzzy model exactly represents the nonlinear system in the region [14], this is:

$$-d < x(t) < d$$

Another approach to obtain T-S fuzzy models is the so-called Local Approximation in Fuzzy Partition Spaces. The spirit of the approach is to approximate nonlinear terms by judiciously chosen linear terms. This procedure leads to reduction of the number

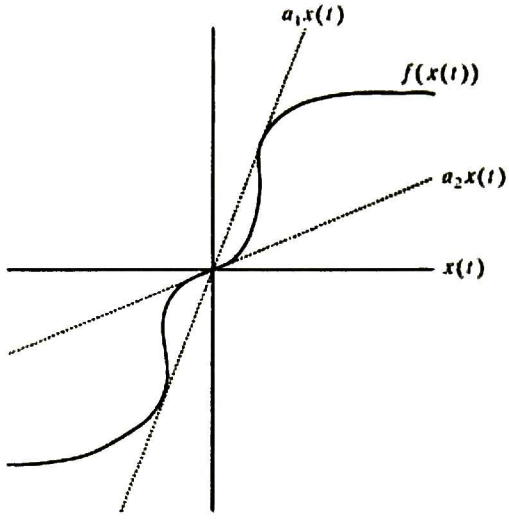


Figure 4.3: Global sector nonlinearity

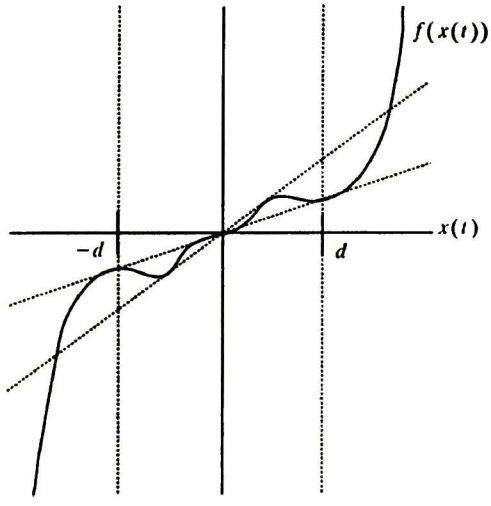


Figure 4.4: Local sector nonlinearity

of model rules. For instance, a the fuzzy model constructed using the nonlinear sector approach may have more rules in comparison, to a fuzzy model constructed using the nonlinear sector idea.

The number of model rules is directly related to complexity of analysis. This is because the number of rules for the overall control system is basically the combination of the model rules and control rules.

Although local approximation technique leads to the reduction of the number of rules for fuzzy models, designing control laws based on the approximated fuzzy model may not guarantee the stability of the original nonlinear systems under such control laws. One of the approaches to alleviate the problem is to introduce robust controller design.

The so-called parallel distributed compensation (PDC) began with a model-based design procedure proposed by Kang and Sugeno. The design procedure is named "parallel distributed compensation". The PDC offers a procedure to design a fuzzy controller from a given T-S fuzzy model. To realize the PDC, a controlled object (nonlinear system) is first represented by a T-S fuzzy model. In the PDC design, each control rule is designed from the corresponding rule of a T-S fuzzy model The designed fuzzy controller shares the same fuzzy sets with the fuzzy model in the premise parts. For a fuzzy models, we may construct the following fuzzy controller via the PDC [14]:

Control Rule i:
IF $z_1(t)$ is M_{i1} and $\cdots$ and $z_p(t)$ is M_{ip},
THEN $u(t) = -F_{ix}(t)$, $i = 1,2,\cdots,r$

The fuzzy control rules have a linear controller (state feedback laws in this case) in the consequent parts. The overall fuzzy controller is represented by:

$$u(t) = \frac{\sum_{i=1}^{r} w_i(z(t))F_i x(t)}{\sum_{i=1}^{r} w_i(z(t))} = \sum_{i=1}^{r} h_i(z(t))F_i x(t) \tag{4.6}$$

The fuzzy controller design is to determine the local feedback gains F_i in the consequent parts. With the PDC we have a simple and natural procedure to handle nonlinear control systems. Other nonlinear control techniques require special and rather involved knowledge.

4.3 Application to the 5DOF robot model

The local approximation in fuzzy partition spaces is used to obtain the fuzzy model of the 5DOF robot (2.25). Therefore, the nonlinear model of the robot is linearized and

evaluated in some operations points. Then, we have the matrices A and B that will be used to create the fuzzy controller.

We chose the following operation points: $-\pi$, $-\frac{\pi}{2}$, 0, $+\frac{\pi}{2}$ and $+\pi$ radians. We linearize and evaluate all the states of the 5DOF model (2.25) in these operation points. When the 5DOF model is linearized and evaluated in $+\frac{\pi}{2}$ and $-\frac{\pi}{2}$ we obtain the same matrices A and B. These matrices evaluated in $\pm\frac{\pi}{2}$ are called $A_{\pm\pi/2}$ and $B_{\pm\pi/2}$. In the same way, when this model is linearized and evaluated in $+\pi$ and $-\pi$ the same matrices A and B are obtained. We call these matrices evaluated in $\pm\pi$ as $A_{\pm\pi}$ and $B_{\pm\pi}$. Finally, the matrices A and B linearized and evaluated in 0 radians are called A_0 and B_0.

Therefore, the Takagi-Sugeno fuzzy model of the 5DOF is defined by the operation points: $-\pi$, $-\frac{\pi}{2}$, 0, $+\frac{\pi}{2}$ and $+\pi$ radians, and by the matrices: A_0, B_0, $A_{\pm\pi/2}$, $B_{\pm\pi/2}$, $A_{\pm\pi}$ and $B_{\pm\pi}$.

When linearizing and evaluating the matrices A and B of the 5DOF robot model (2.25) in 0 radians, we get:

$$A_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 \\ 0 & 0 & 0 & 0 & 0 & -9.9051 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.1918 & 0.1976 & -0.0023 & -0.0030 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1976 & -0.3975 & 0.2000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.0023 & 0.2000 & -0.3976 & 0.1999 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.0030 & 0 & 0.1999 & -0.3971 \end{bmatrix}$$

$$B_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.4255 & 0 & 0 & 0 & 0 \\ 0 & 1.9182 & -1.9759 & 0.0233 & 0.0303 \\ 0 & -1.9759 & 3.9752 & -1.9997 & 0.0004 \\ 0 & 0.0233 & -1.9997 & 3.9755 & -1.9992 \\ 0 & 0.0303 & 0.0004 & -1.9992 & 3.9712 \end{bmatrix}$$

When the matrices A and B from (2.25) are linearized and evaluated in $-\frac{\pi}{2}$ or in

$+\frac{\pi}{2}$, we obtain:

$$A_{\pm\pi/2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 \\ 0 & 0 & 0 & 0 & 0 & -9.9051 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.1917 & 0.1917 & -0.0035 & 0.0035 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1917 & -0.3856 & 0.1975 & -0.0040 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.0035 & 0.1975 & -0.3903 & 0.1968 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0035 & -0.0040 & 0.1968 & -0.3962 \end{bmatrix}$$

$$B_{\pm\pi/2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.4255 & 0 & 0 & 0 & 0 \\ 0 & 1.9165 & -1.9165 & 0.0354 & -0.0354 \\ 0 & -1.9165 & 3.8557 & -1.9746 & 0.0397 \\ 0 & 0.0354 & -1.9746 & 3.9029 & -1.9680 \\ 0 & -0.0354 & 0.0397 & -1.9680 & 3.9622 \end{bmatrix}$$

Finally, when we linearize and evaluate A and B from (2.25) in $-\pi$ or $+\pi$ radians, we obtain:

$$A_{\pm\pi} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 \\ 0 & 0 & 0 & 0 & 0 & -9.9051 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.1918 & 0.1861 & 0.0092 & -0.0038 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1861 & -0.3744 & 0.1815 & 0.0077 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0092 & 0.1815 & -0.3836 & 0.1921 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.0038 & 0.0077 & 0.1921 & -0.3954 \end{bmatrix}$$

$$B_{\pm\pi} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.4255 & 0 & 0 & 0 & 0 \\ 0 & 1.9182 & -1.8606 & -0.0920 & 0.0384 \\ 0 & -1.8606 & 3.7444 & -1.8148 & -0.0773 \\ 0 & -0.0920 & -1.8148 & 3.8364 & -1.9213 \\ 0 & 0.0384 & -0.0773 & -1.9213 & 3.9544 \end{bmatrix}$$

4.4 Fuzzy control

We use the matrices of the Takagi-Sugeno fuzzy model of the 5DOF robot, obtained from the local approximation in fuzzy partition spaces (A_0, B_0, $A_{\pm\pi/2}$, $B_{\pm\pi/2}$, $A_{\pm\pi}$ and $B_{\pm\pi}$), to create a fuzzy control for this system. We use the input-output technique in order to design the Takagi-Sugeno fuzzy control. We design one control for each of the three cases of the fuzzy model (0, $\pm\frac{\pi}{2}$, $\pm\pi$).

From the 5DOF robot model (2.25) we define the following system:

$$
\begin{aligned}
\dot{x}_1 &= x_6 \\
\dot{x}_2 &= x_7 \\
\dot{x}_3 &= x_8 \\
\dot{x}_4 &= x_9 \\
\dot{x}_5 &= x_{10} \\
\dot{x}_6 &= a_{66}x_6 + a_{67}x_7 + a_{68}x_8 + a_{69}x_9 + a_{610}x_{10} + b_{61}u_1 \\
&\quad + b_{62}u_2 + b_{63}u_3 + b_{64}u_4 + b_{65}u_5 \\
\dot{x}_7 &= a_{76}x_6 + a_{77}x_7 + a_{78}x_8 + a_{79}x_9 + a_{710}x_{10} + b_{71}u_1 \\
&\quad + b_{72}u_2 + b_{73}u_3 + b_{74}u_4 + b_{75}u_5 \\
\dot{x}_8 &= a_{86}x_6 + a_{87}x_7 + a_{88}x_8 + a_{89}x_9 + a_{810}x_{10} + b_{81}u_1 \\
&\quad + b_{82}u_2 + b_{83}u_3 + b_{84}u_4 + b_{85}u_5 \\
\dot{x}_9 &= a_{96}x_6 + a_{97}x_7 + a_{98}x_8 + a_{99}x_9 + a_{910}x_{10} + b_{91}u_1 \\
&\quad + b_{92}u_2 + b_{93}u_3 + b_{94}u_4 + b_{95}u_5 \\
\dot{x}_{10} &= a_{106}x_6 + a_{107}x_7 + a_{108}x_8 + a_{109}x_9 + a_{1010}x_{10} + b_{101}u_1 \\
&\quad + b_{102}u_2 + b_{103}u_3 + b_{104}u_4 + b_{105}u_5
\end{aligned}
\tag{4.7}
$$

The tracking error is defined by:

$$\varepsilon_{i1} = x_i - x_{i,ref} \text{ ; for } i = 1, \cdots, 5 \tag{4.8}$$

$$\dot{\varepsilon}_{i1} = \dot{x}_i - \dot{x}_{i,ref} \tag{4.9}$$

$$\dot{\varepsilon}_{i1} = x_{(5+i)} - \dot{x}_{i,ref} = \varepsilon_{i2} \tag{4.10}$$

$$\varepsilon_{i2} = x_{(5+i)} - \dot{x}_{i,ref} \text{ ; for } i = 1, \cdots, 5 \tag{4.11}$$

$$\dot{\varepsilon}_{i2} = \dot{x}_{(5+i)} - \ddot{x}_{i,ref} \tag{4.12}$$

$$\dot{\varepsilon}_{i2} = a_{(5+i)6}x_6 + a_{(5+i)7}x_7 + a_{(5+i)8}x_8 + a_{(5+i)9}x_9 + a_{(5+i)10}x_{10} \cdots + b_{(5+i)1}u_1 + b_{(5+i)2}u_2 + b_{(5+i)3}u_3 + b_{(5+i)4}u_4 + b_{(5+i)5}u_5 - \ddot{x}_{i,ref} \tag{4.13}$$

The following condition will be defined in order to reduce the error:

$$\dot{\varepsilon}_{i2} = -\alpha_{i1}\varepsilon_{i1} - \alpha_{i2}\varepsilon_{i2} \text{ ; for } i = 1, \cdots, 5 \tag{4.14}$$
$$\text{with } \alpha_{i1},\ \alpha_{i2} > 0$$

Using (4.10) and (4.13) we obtain the following system:

$$b_{61}u_1 + b_{62}u_2 + b_{63}u_3 + b_{64}u_4 + b_{65}u_5 = -\alpha_{11}\varepsilon_{11} - \alpha_{12}\varepsilon_{12} + \ddot{x}_{1ref} - a_{66}x_6 - a_{67}x_7 - a_{68}x_8 - a_{69}x_9 - a_{610}x_{10} \tag{4.15}$$

$$b_{71}u_1 + b_{72}u_2 + b_{73}u_3 + b_{74}u_4 + b_{75}u_5 = -\alpha_{21}\varepsilon_{21} - \alpha_{22}\varepsilon_{22} + \ddot{x}_{2ref} - a_{76}x_6 \cdots - a_{77}x_7 - a_{78}x_8 - a_{79}x_9 - a_{710}x_{10}$$

$$b_{81}u_1 + b_{82}u_2 + b_{83}u_3 + b_{84}u_4 + b_{85}u_5 = -\alpha_{31}\varepsilon_{31} - \alpha_{32}\varepsilon_{32} + \ddot{x}_{3ref} - a_{86}x_6 \cdots - a_{87}x_7 - a_{88}x_8 - a_{89}x_9 - a_{810}x_{10}$$

$$b_{91}u_1 + b_{92}u_2 + b_{93}u_3 + b_{94}u_4 + b_{95}u_5 = -\alpha_{41}\varepsilon_{41} - \alpha_{42}\varepsilon_{42} + \ddot{x}_{4ref} - a_{96}x_6 \cdots - a_{97}x_7 - a_{98}x_8 - a_{99}x_9 - a_{910}x_{10}$$

$$b_{101}u_1 + b_{102}u_2 + b_{103}u_3 + b_{104}u_4 + b_{105}u_5 = -\alpha_{51}\varepsilon_{51} - \alpha_{52}\varepsilon_{52} + \ddot{x}_{5ref} - a_{106}x_6 \cdots - a_{107}x_7 - a_{108}x_8 - a_{109}x_9 - a_{1010}x_{10}$$

In order to simplify these expressions, let P be a vector defined as follows:

$$P = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \end{bmatrix} \tag{4.16}$$

where:

$$
\begin{aligned}
P_1 &= -\alpha_{11}\varepsilon_{11} - \alpha_{12}\varepsilon_{12} \\
P_2 &= -\alpha_{21}\varepsilon_{21} - \alpha_{22}\varepsilon_{22} \\
P_3 &= -\alpha_{31}\varepsilon_{31} - \alpha_{32}\varepsilon_{32} \\
P_4 &= -\alpha_{41}\varepsilon_{41} - \alpha_{42}\varepsilon_{42} \\
P_5 &= -\alpha_{51}\varepsilon_{51} - \alpha_{52}\varepsilon_{52}
\end{aligned}
$$

From (4.15) we obtain the following expression:

$$
\begin{bmatrix}
b_{61} & b_{62} & b_{63} & b_{64} & b_{65} \\
b_{71} & b_{72} & b_{73} & b_{74} & b_{75} \\
b_{81} & b_{82} & b_{83} & b_{84} & b_{85} \\
b_{91} & b_{92} & b_{93} & b_{94} & b_{95} \\
b_{101} & b_{102} & b_{103} & b_{104} & b_{105}
\end{bmatrix}
\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} =
\begin{bmatrix}
P_1 + \ddot{x}_{1ref} - a_{66}x_6 - a_{67}x_7 - \cdots \\
\quad a_{68}x_8 - a_{69}x_9 - a_{610}x_{10} \\
P_2 + \ddot{x}_{2ref} - a_{76}x_6 - a_{77}x_7 - \cdots \\
\quad a_{78}x_8 - a_{79}x_9 - a_{710}x_{10} \\
P_3 + \ddot{x}_{3ref} - a_{86}x_6 - a_{87}x_7 - \cdots \\
\quad a_{88}x_8 - a_{89}x_9 - a_{810}x_{10} \\
P_4 + \ddot{x}_{4ref} - a_{96}x_6 - a_{97}x_7 - \cdots \\
\quad a_{98}x_8 - a_{99}x_9 - a_{910}x_{10} \\
P_5 + \ddot{x}_{5ref} - a_{106}x_6 - a_{107}x_7 - \cdots \\
\quad a_{108}x_8 - a_{109}x_9 - a_{1010}x_{10}
\end{bmatrix}
\tag{4.17}
$$

Therefore, from 4.17,the input vector u for each of the three cases of the Takagi-Sugeno fuzzy model of the 5DOF redundant robot is:

$$
\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} =
\begin{bmatrix}
b_{61} & b_{62} & b_{63} & b_{64} & b_{65} \\
b_{71} & b_{72} & b_{73} & b_{74} & b_{75} \\
b_{81} & b_{82} & b_{83} & b_{84} & b_{85} \\
b_{91} & b_{92} & b_{93} & b_{94} & b_{95} \\
b_{101} & b_{102} & b_{103} & b_{104} & b_{105}
\end{bmatrix}^{-1}
\begin{bmatrix}
P_1 + \ddot{x}_{1ref} - a_{66}x_6 - a_{67}x_7 - \cdots \\
\quad a_{68}x_8 - a_{69}x_9 - a_{610}x_{10} \\
P_2 + \ddot{x}_{2ref} - a_{76}x_6 - a_{77}x_7 - \cdots \\
\quad a_{78}x_8 - a_{79}x_9 - a_{710}x_{10} \\
P_3 + \ddot{x}_{3ref} - a_{86}x_6 - a_{87}x_7 - \cdots \\
\quad a_{88}x_8 - a_{89}x_9 - a_{810}x_{10} \\
P_4 + \ddot{x}_{4ref} - a_{96}x_6 - a_{97}x_7 - \cdots \\
\quad a_{98}x_8 - a_{99}x_9 - a_{910}x_{10} \\
P_5 + \ddot{x}_{5ref} - a_{106}x_6 - a_{107}x_7 - \cdots \\
\quad a_{108}x_8 - a_{109}x_9 - a_{1010}x_{10}
\end{bmatrix}
\tag{4.18}
$$

The block diagram for this fuzzy control is shown in Figure 4.5. The controls for these three cases are interpolated by the membership functions shown in Figure 4.6.

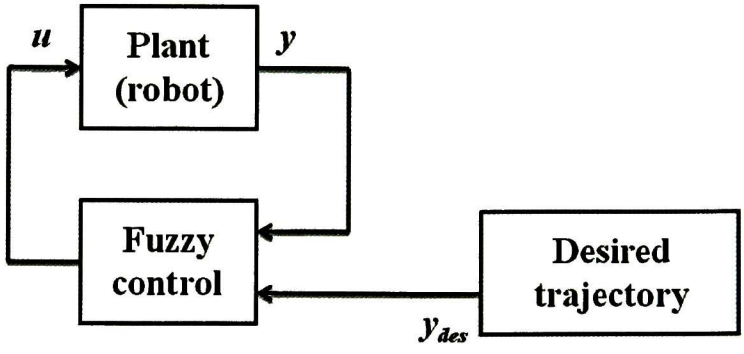


Figure 4.5: Fuzzy control block diagram

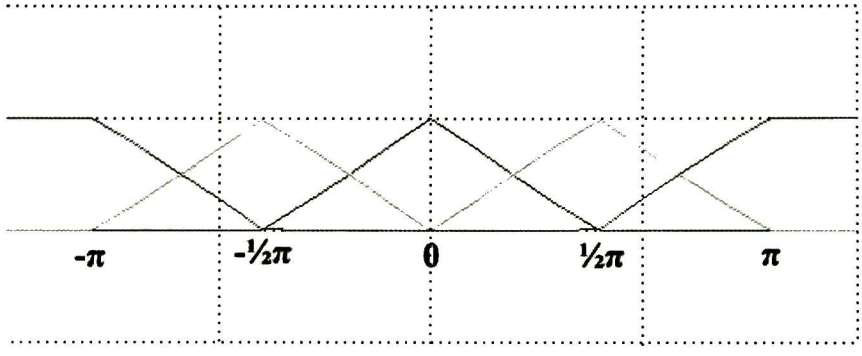


Figure 4.6: Membership functions

4.5 Simulation

For this control, we use the following values:

$$\begin{array}{ll} \alpha_{11} = 500 & \alpha_{12} = 500 \\ \alpha_{21} = 500 & \alpha_{22} = 500 \\ \alpha_{31} = 500 & \alpha_{32} = 500 \\ \alpha_{41} = 500 & \alpha_{42} = 500 \\ \alpha_{51} = 500 & \alpha_{52} = 500 \end{array}$$

These α values and the values of the matrices: A_0, $A_{\pm\pi/2}$, $A_{\pm\pi}$, B_0, $B_{\pm\pi/2}$ and $B_{\pm\pi}$ of the fuzzy model, are used in order to design a control for each of the following cases: 0, $-\frac{\pi}{2}$, $+\frac{\pi}{2}$, $-\pi$ and $+\pi$ radians. As we mentioned before, the case with $+\frac{\pi}{2}$ and $-\frac{\pi}{2}$ radians have the same behavior. In the same way, the case with $+\pi$ and $-\pi$ radians also have the same behavior.

From 4.18, for the matrices A_0 and B_0 we have:

$$\begin{aligned}
\tau_{10} &= (2.3502)f_{10} + (0)f_{20} + (0)f_{30} + (0)f_{40} + (0)f_{50} \\
\tau_{20} &= (0)f_{10} + (2.1232)f_{20} + (1.5750)f_{30} + (1.0331)f_{40} + (0.5037)f_{50} \\
\tau_{30} &= (0)f_{10} + (1.5750)f_{20} + (1.5487)f_{30} + (1.0225)f_{40} + (0.5026)f_{50} \\
\tau_{40} &= (0)f_{10} + (1.0331)f_{20} + (1.0225)f_{30} + (1.0120)f_{40} + (0.5015)f_{50} \\
\tau_{10} &= (0)f_{10} + (0.5037)f_{20} + (0.5026)f_{30} + (0.5015)f_{40} + (0.5004)f_{50}
\end{aligned}$$

where

$$\begin{aligned}
f_{10} &= -(-9,9051)x_6 - (0)x_7 - (0)x_8 - (0)x_9 - (0)x_{10} + P_1 + \ddot{x}_{1ref} \\
f_{20} &= (0)x_6 - (-0.1918)x_7 - (0.1976)x_8 - (-0.0023)x_9 - (-0.0030)x_{10} + P_2 + \ddot{x}_{2ref} \\
f_{30} &= (0)x_6 - (0.1976)x_7 - (-0.3975)x_8 - (0.2000)x_9 - (0)x_{10} + P_3 + \ddot{x}_{3ref} \\
f_{40} &= (0)x_6 - (-0.0023)x_7 - (0.2000)x_8 - (-0.3976)x_9 - (0.1999)x_{10} + P_4 + \ddot{x}_{4ref} \\
f_{50} &= (0)x_6 - (-0.0030)x_7 - (0)x_8 - (0.1999)x_9 - (-0.3971)x_{10} + P_5 + \ddot{x}_{5ref}
\end{aligned}$$

In the same way, from 4.18, for the matrices $A_{\pm\pi/2}$,and $B_{\pm\pi/2}$, we obtain:

$$\begin{aligned}
\tau_{11} &= (2.3502)f_{11} + (0)f_{21} + (0)f_{31} + (0)f_{41} + (0)f_{51} \\
\tau_{21} &= (0)f_{11} + (2.0264)f_{21} + (1.5138)f_{31} + (0.9993)f_{41} + (0.4993)f_{51} \\
\tau_{31} &= (0)f_{11} + (1.5138)f_{21} + (1.5232)f_{31} + (1.0087)f_{41} + (0.4993)f_{51} \\
\tau_{41} &= (0)f_{11} + (0.9993)f_{21} + (1.0087)f_{31} + (1.0098)f_{41} + (0.5004)f_{51} \\
\tau_{51} &= (0)f_{11} + (0.4993)f_{21} + (0.4993)f_{31} + (0.5004)f_{41} + (0.5004)f_{51}
\end{aligned}$$

where

$$\begin{aligned}
f_{11} &= -(-9,9051)x_6 - (0)x_7 - (0)x_8 - (0)x_9 - (0)x_{10} + P_1 + \ddot{x}_{1ref} \\
f_{21} &= (0)x_6 - (-0.1917)x_7 - (0.1917)x_8 - (-0.0035)x_9 - (0.0035)x_{10} + P_2 + \ddot{x}_{2ref} \\
f_{31} &= (0)x_6 - (0.1917)x_7 - (-0.3856)x_8 - (0.1975)x_9 - (-0.0040)x_{10} + P_3 + \ddot{x}_{3ref} \\
f_{41} &= (0)x_6 - (-0.0035)x_7 - (0.1975)x_8 - (-0.3903)x_9 - (0.1968)x_{10} + P_4 + \ddot{x}_{4ref} \\
f_{51} &= (0)x_6 - (0.0035)x_7 - (-0.0040)x_8 - (0.1968)x_9 - (-0.3962)x_{10} + P_5 + \ddot{x}_{5ref}
\end{aligned}$$

Finally, from 4.18, for the matrices $A_{\pm\pi}$ and $B_{\pm\pi}$, we get:

$$
\begin{aligned}
\tau_{12} &= (2.3502)f_{12} + (0)f_{22} + (0)f_{32} + (0)f_{42} + (0)f_{52} \\
\tau_{22} &= (0)f_{12} + (2.0144)f_{22} + (1.4998)f_{32} + (1.0079)f_{42} + (0.4995)f_{52} \\
\tau_{32} &= (0)f_{12} + (1.4998)f_{22} + (1.5071)f_{32} + (0.9996)f_{42} + (0.5006)f_{52} \\
\tau_{42} &= (0)f_{12} + (1.0079)f_{22} + (0.9996)f_{32} + (1.0078)f_{42} + (0.4994)f_{52} \\
\tau_{52} &= (0)f_{12} + (0.4995)f_{22} + (0.5006)f_{32} + (0.4994)f_{42} + (0.5005)f_{52}
\end{aligned}
$$

where

$$
\begin{aligned}
f_{12} &= -(-9,9051)x_6 - (0)x_7 - (0)x_8 - (0)x_9 - (0)x_{10} + P_1 + \ddot{x}_{1ref} \\
f_{22} &= (0)x_6 - (-0.1918)x_7 - (0.1861)x_8 - (0.0092)x_9 - (-0.0038)x_{10} + P_2 + \ddot{x}_{2ref} \\
f_{32} &= (0)x_6 - (0.1861)x_7 - (-0.3744)x_8 - (0.1815)x_9 - (0.0077)x_{10} + P_3 + \ddot{x}_{3ref} \\
f_{42} &= (0)x_6 - (0.0092)x_7 - (0.1815)x_8 - (-0.3836)x_9 - (0.1921)x_{10} + P_4 + \ddot{x}_{4ref} \\
f_{52} &= (0)x_6 - (-0.0038)x_7 - (0.0077)x_8 - (0.1921)x_9 - (-0.3954)x_{10} + P_5 + \ddot{x}_{5ref}
\end{aligned}
$$

The membership functions used in the implementation of this control are shown in Figure 4.6. The interpolation of these three controls is given by these membership functions, thus, the control signal τ of the fuzzy control is given by:

$$
\tau = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \\ \tau_5 \end{bmatrix} = \begin{bmatrix} \tau_{10}(FM_0) + \tau_{11}(FM_1) + \tau_{12}(FM_2) \\ \tau_{20}(FM_0) + \tau_{21}(FM_1) + \tau_{22}(FM_2) \\ \tau_{30}(FM_0) + \tau_{31}(FM_1) + \tau_{32}(FM_2) \\ \tau_{40}(FM_0) + \tau_{41}(FM_1) + \tau_{42}(FM_2) \\ \tau_{50}(FM_0) + \tau_{51}(FM_1) + \tau_{52}(FM_2) \end{bmatrix} \tag{4.19}
$$

In this simulation the following references q_{ref} are used:

$$
q_{ref} = \begin{bmatrix} 0.15 \\ 10\frac{\pi}{180}\cos(0.5t) \\ -10\frac{\pi}{180}\cos(0.5t) \\ 6\frac{\pi}{180}\cos(0.5t) \\ 8\frac{\pi}{180}\cos(0.5t) \end{bmatrix} \tag{4.20}
$$

First we use the fuzzy control τ (4.19) in order to track the desired trajectory q_{ref} and we obtain the simulation results shown in Figures 4.7 to 4.9. The position of each of the five link of the five degrees of freedom redundant robot is shown in Figure 4.7. The continuous line represents the desired positions q_{ref} and the dotted line represents the output position q obtained from the fuzzy control. This figure shows how the output positions converge to the desired positions for each of the five links of the five degrees of freedom robot redundant.

The control signals τ_1 to τ_5 generated by the fuzzy control for each link of the five degrees of freedom redundant robot in the system without disturbances are shown in Figure 4.8. The trajectory tracking error e for the fuzzy control applied to the robotic arm can be seen in the Figure 4.9. We can see how the tracking error with this fuzzy control tends to zero.

In addition, Figures 4.10 to 4.12 show the simulation results of the fuzzy control for the system with three disturbances. The first disturbance occurs around the second 20, the second disturbance occurs around the second 55 and the last disturbance occurs around the second 90. Figure 4.10 shows the desired positions (continuous line) and the output positions (dotted line) q_1 to q_5, obtained with the fuzzy control with these three disturbances in the system.

We can see in this results how the output positions converge to the desired positions. Around the second 20 the current positions are separated from the desired positions due to the first disturbance but eventually they converge. Later, around the second 55, the current positions are separated from the desired positions again due to the second disturbance. Finally, this occurs one last time with the third disturbance around the second 90. Then the output positions converge again in the desired positions.

The control signals τ_1 to τ_5 generated for each of the links for this simulation with three disturbances are shown in Figure 4.11. This figure shows how the control signals are adjusted in order to correct the errors. These errors are caused by the three disturbances which occur around the second 20, 55 and 90.

Finally, Figure 4.12 shows the tracking error for the system with three perturbations around the second 20, 55 and 90. At the beginning of the simulation the error tends to zero, but when the first disturbance occurs around the second 20 the tracking error increases. Then, the error tends to zero again until the second disturbance occurs around the second 55. Finally, the tracking error tends to zero again after the third disturbance around the second 90.

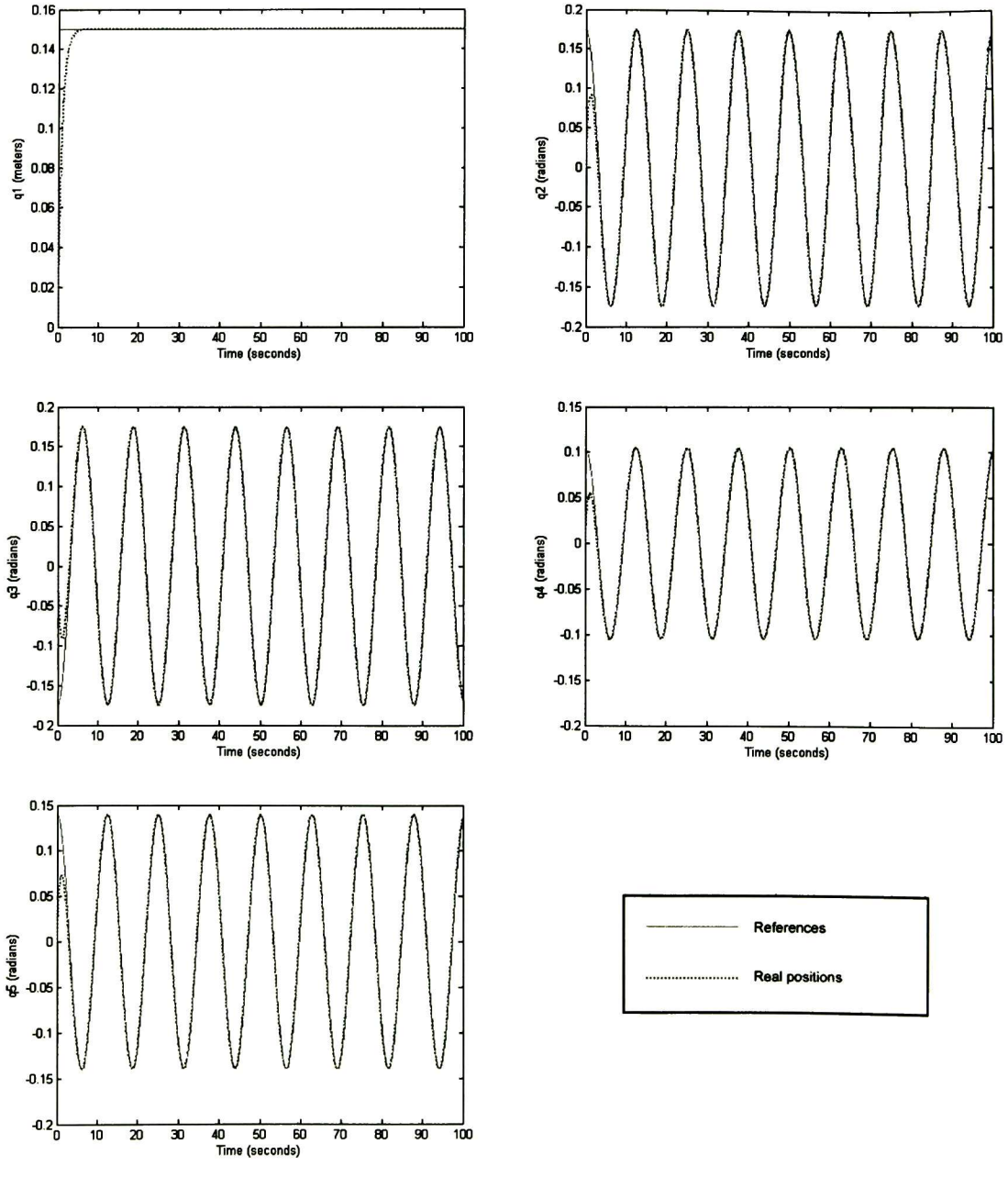


Figure 4.7: Positions (q_i vs. q_{i_ref})

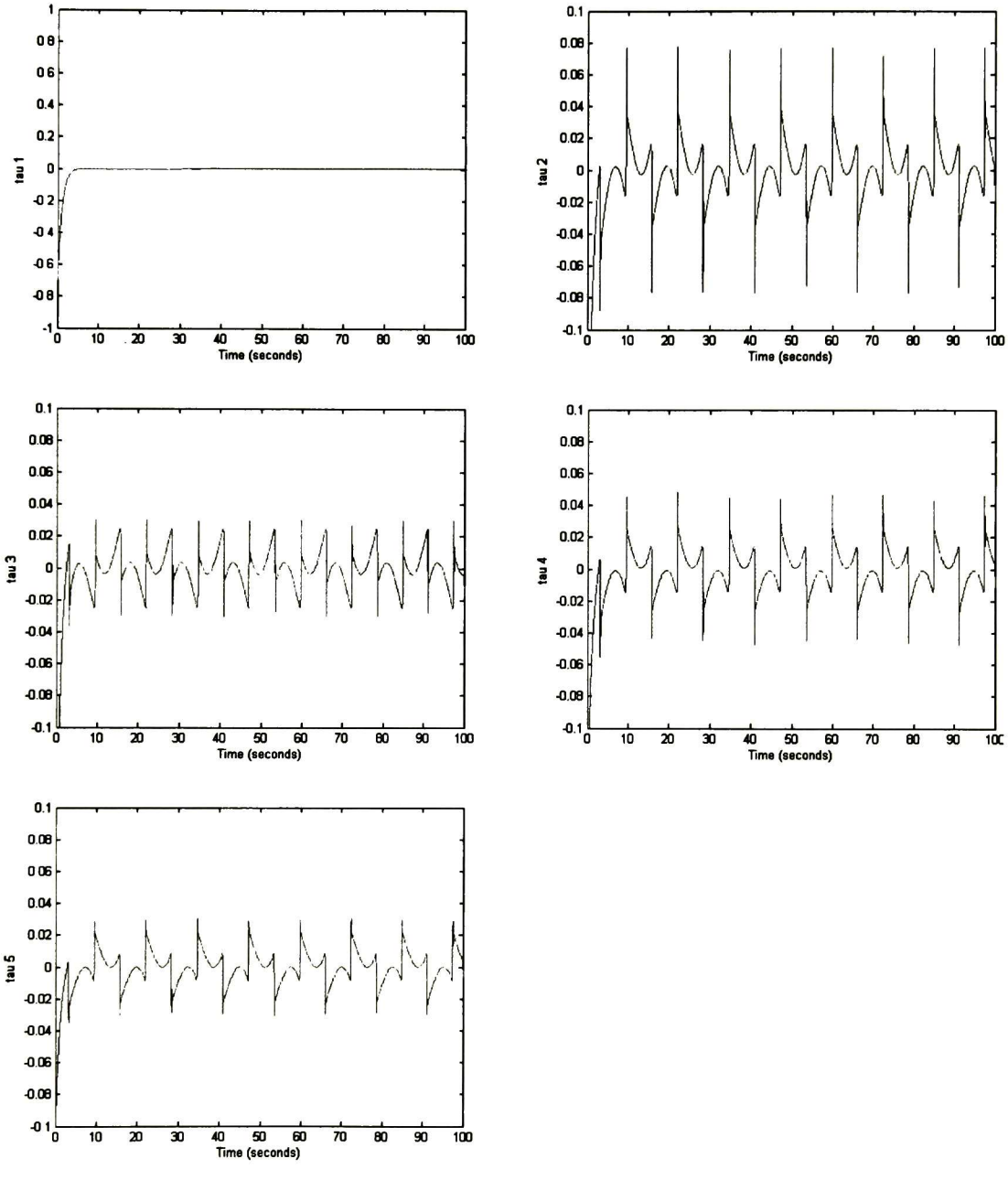


Figure 4.8: Control signals (τ_i)

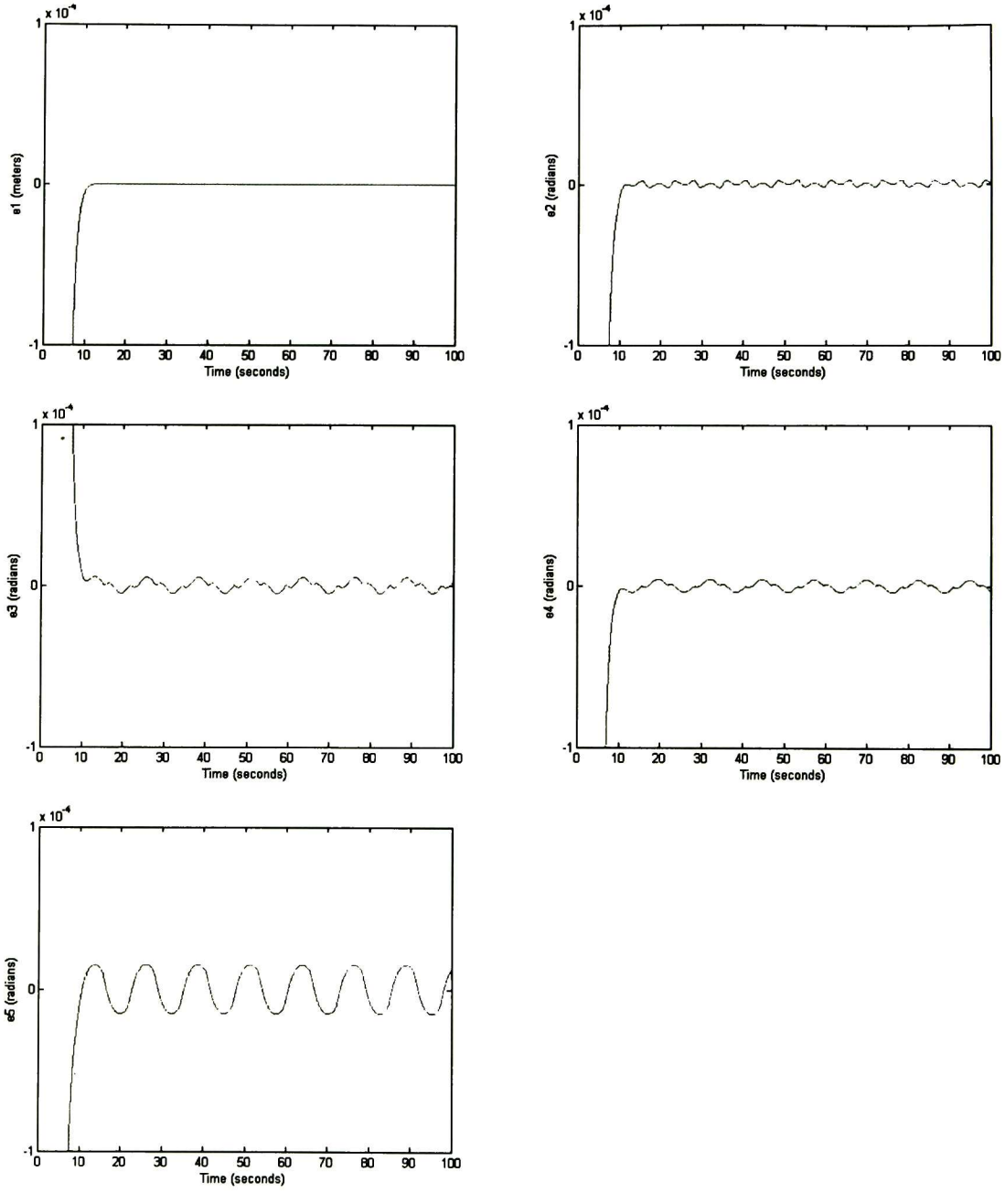


Figure 4.9: Tracking error (e_i)

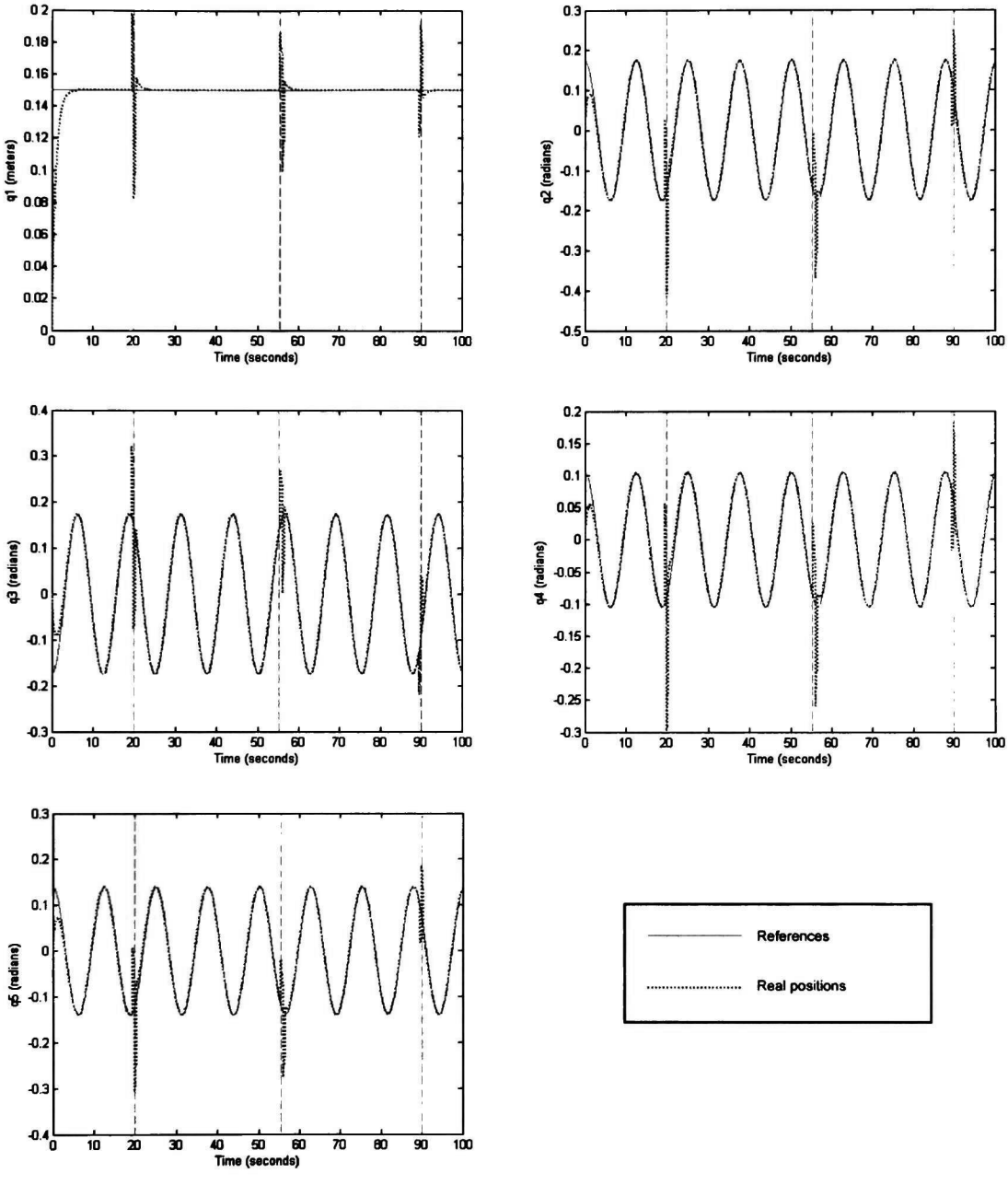


Figure 4.10: Positions (q_i vs. q_{i_ref}) with disturbances

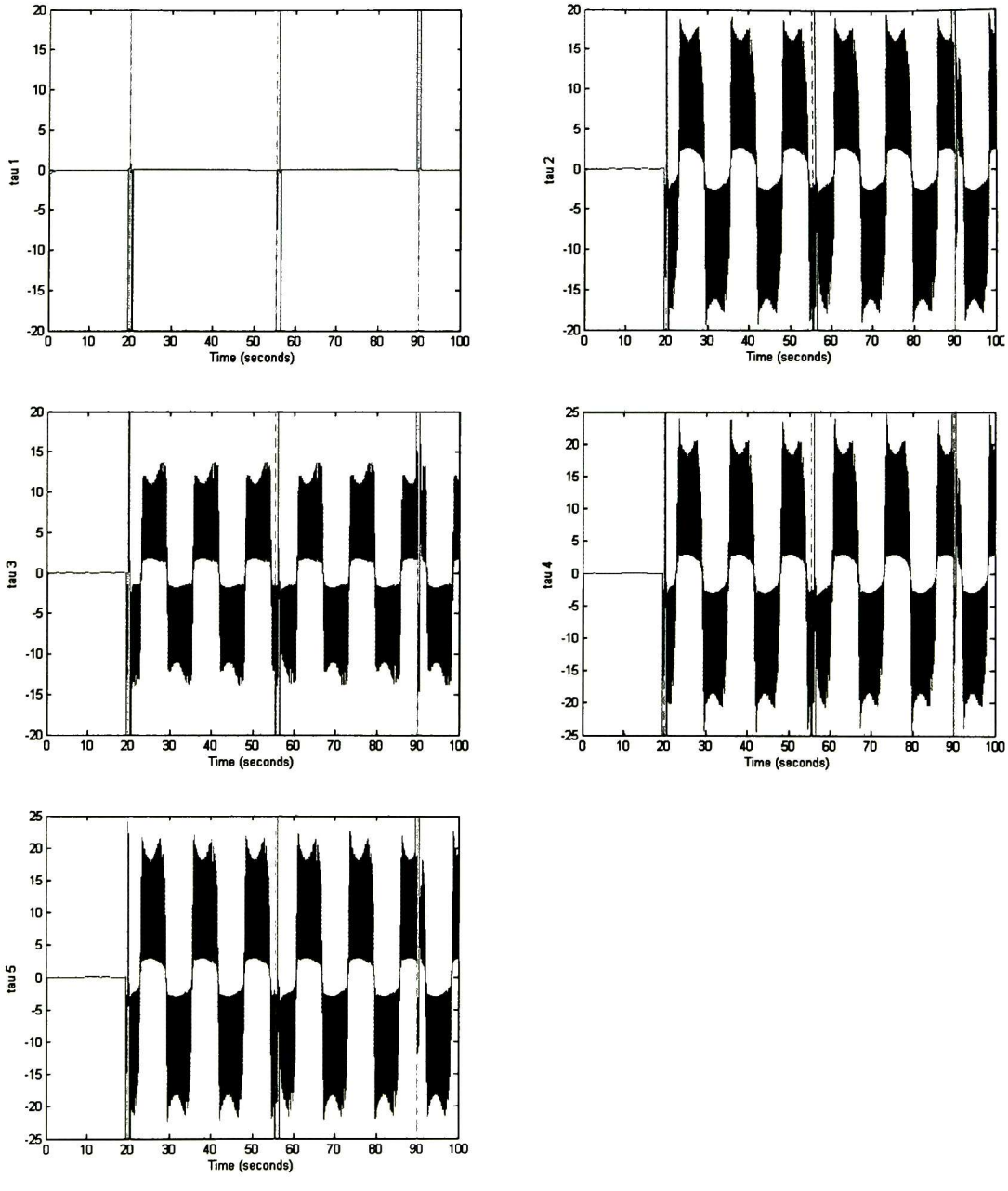


Figure 4.11: Control signals (τ_i) with disturbances

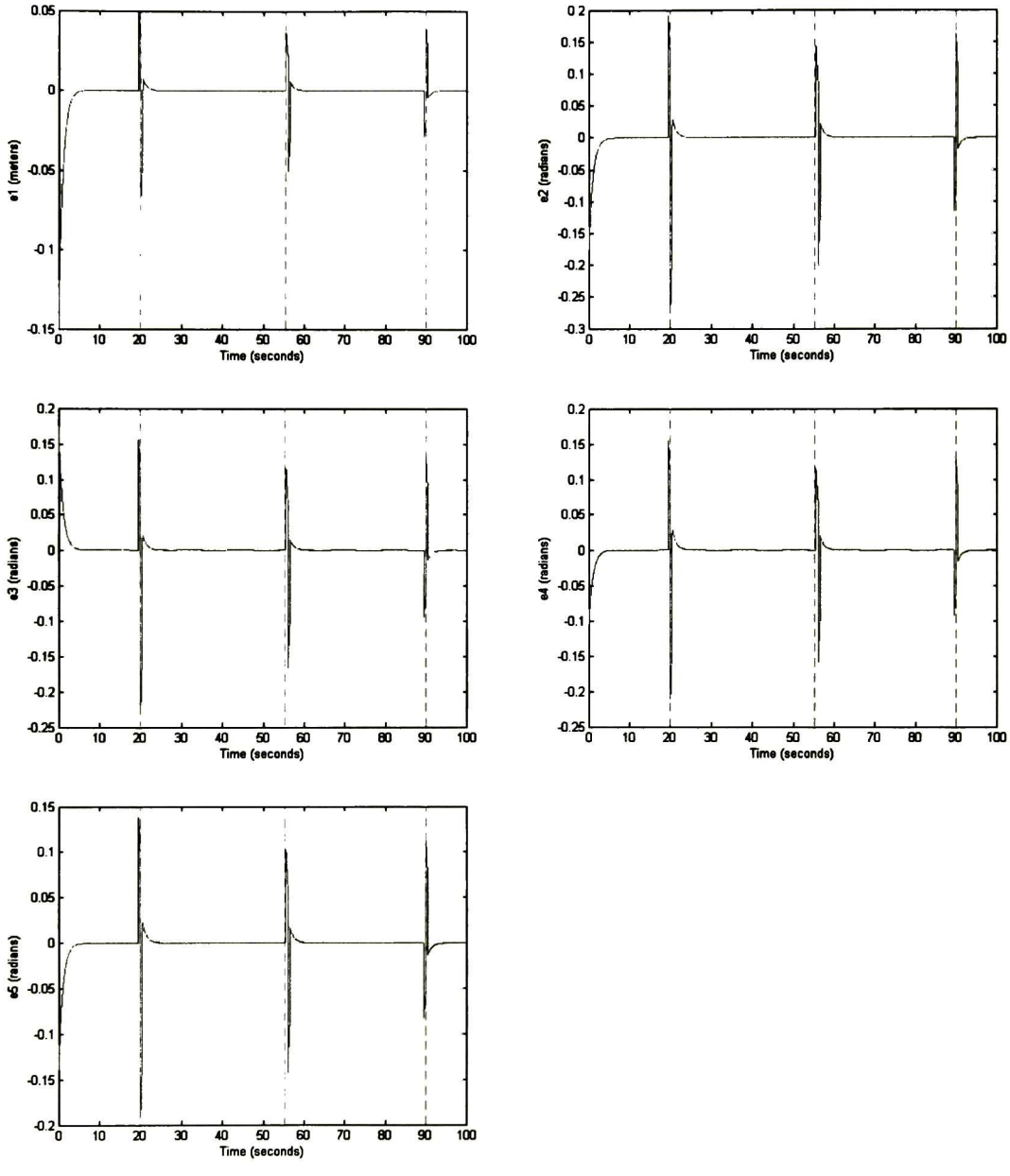


Figure 4.12: Tracking error (e_i) with disturbances

Chapter 5

Neuro-fuzzy control

5.1 Neuro-fuzzy hybrid techniques

Fuzzy logic and neural networks have properties that make them appropriate to some specific problems. Neuro-fuzzy hybrid systems allow using the advantages of learning from the neural networks with the power of linguistic interpretation from the fuzzy systems.

Fuzzy systems have the advantage that they can represent the knowledge in an explicit form, moreover, verification and optimization of this form to represent knowledge is easy and efficient. However, training for fuzzy systems can not be explicitly defined.

Neural networks have the advantage that they train themselves from a set of data. However, the way that they represent knowledge can not be easily interpreted or modified.

Neuro-fuzzy controllers can be divided into three different areas (Figure 5.1):
1. Concurrent models: this models are fuzzy systems and neural networks that work together, although each one does not determine the parameters of the other.
2. Cooperative models: these models are defined as a fuzzy system with neuronal training, this is, theses models have two stages, one of training and another one of

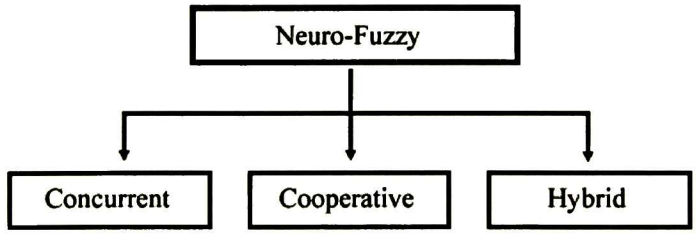


Figure 5.1: Neuro-fuzzy controllers

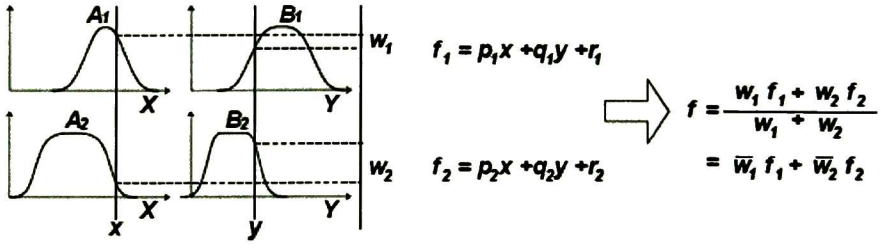


Figure 5.2: Sugeno fuzzy model

operation. Neural networks are used to determine the parameters of operation for the fuzzy control.

3. Hybrid models: these models can be defined as the neural networks and fuzzy systems working together. There are two different kinds of hybrid models: a neural system with a fuzzy behavior or a fuzzy system with distributed parameters.

The controller model of an ANFIS (adaptive neuro-fuzzy inference system) is an hybrid model which is derived from the behavior of an adaptive network of forward propagation [4].

5.2 Adaptive Neuro-Fuzzy Inference Systems (ANFIS)

An Adaptive Neuro-Fuzzy Inference System (ANFIS) is a method that allows determining the rule base of a fuzzy system by the training of an artificial neural network from a data collection. With its training the neural network obtains the values that represent the best the membership functions of the fuzzy model. The ANFIS architecture is functionally equivalent to a Sugeno fuzzy model (Figure 5.2) [4], as follows:

$$\begin{aligned} f_1 &= p_1 x + q_1 y + r_1 \\ f_2 &= p_2 x + q_2 y + r_2 \\ f &= \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2} = \bar{w}_1 f_1 + \bar{w}_2 f_2 \end{aligned} \tag{5.1}$$

For simplicity, the fuzzy inference system architecture is described under the consideration that it has two inputs: x, y and the output z. For a first order Sugeno fuzzy model, a typical set of two IF-THEN rules (Figure 5.3) [4] can be expressed as
Rule 1: If x is A_1 and y is B_1, then $f_1 = p_1 x + q_1 y + r_1$,
Rule 2: If x is A_2 and y is B_2, then $f_2 = p_2 x + q_2 y + r_2$,

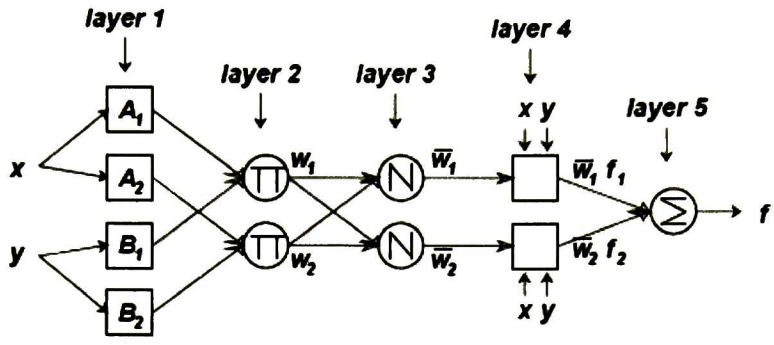


Figure 5.3: ANFIS architecture

Figure 5.2 shows a Sugeno fuzzy model of two rules and two inputs x, y. The architecture for an ANFIS is shown in Figure 5.3, where nodes of the same layer have similar functions as described below, where the exit of the node i in the layer l is denoted as $O_{l,i}$.The layers in an ANFIS are:

Layer 1: Each node in this layer is an adaptive node, with an output node defined by:

$$\begin{aligned} O_{1,i} &= \mu_{A_i}(x), \text{ for } i = 1,2 \\ O_{1,i} &= \mu_{B_{i-2}}(y), \text{ for } i = 3,4 \end{aligned} \tag{5.2}$$

where x and y are the inputs to the node; A_i and B_{i-2} are fuzzy sets associated with this node. In other words, in this layer are the membership values of the premises. They can be defined for any kind of membership function, for example, a bell function:

$$\mu_A(x) = \frac{1}{1 + [(\frac{x-c_i}{a_i})^2]b_i} \tag{5.3}$$

here $\{a_i, b_i, c_i\}$ is the parameter set. The parameters of this layer are known as premise parameters.

Layer 2: Each node in this layer is a fixed node with the label $\prod$, which multiplies the input signals. Its output is the result of this multiplication.

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), i = 1,2. \tag{5.4}$$

The output of each node represents the firing strength of a rule. In fact, any T-norm operator that performs fuzzy AND operations may be used as the function in the nodes of this layer.

Layer 3: Each node in this layer i is a node labeled N. The i-th node calculates the firing strength norm of the previous layer:

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, i = 1, 2. \tag{5.5}$$

For convenience, outputs of this layer will be called normalized firing strengths.

Layer 4: Each node in this layer is an adaptive node with a function:

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \tag{5.6}$$

where $\bar{w}_i$ is the output of layer 3 and $\{p_{ix} + q_{iy} + r_i\}$ is the set of parameters. Parameters in this layer will be referred to as consequent parameters.

Layer 5: The only node in this layer is a fixed node labeled $\sum$, which sums all the input signals:

$$O_{5,i} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{5.7}$$

Therefore, with these five layers, an adaptive network that has exactly the same function than a Sugeno fuzzy model has been created. But this structure is not the only one, since layers 3 and 4 can be easily combined to obtain an equivalent network of only four layers. In a similar way, the firing strengths normalization may be perform in the last layer.

The architecture of an ANFIS that is equivalent to a Sugeno fuzzy model of first order with two inputs and with two rules is shown in Figure 5.3. This ANFIS is equivalent to the Sugeno fuzzy model of Figure 5.2.

Figure 5.4 [4] shows the architecture of an ANFIS that is equivalent to a Sugeno fuzzy model of first order with two inputs and with nine rules. It also shows the partition of the input space into nine fuzzy regions.

5.3 Neuro-fuzzy control

An Adaptive Neuro-Fuzzy Inference System (ANFIS) is trained with an input-output data collection that describes the behavior of the 5DOF redundant robot in a certain region of operation. This data collection has been obtained from the neural control simulation from Chapter 3. In order to describe the system there were considered the values of the position of the joints q, the position error e and the control signal u of each of the five joints of the system. The neural network of this neuro-fuzzy control is trained in order to obtain the values that represent the best the membership functions of the Sugeno fuzzy model of the 5DOF redundant robot.

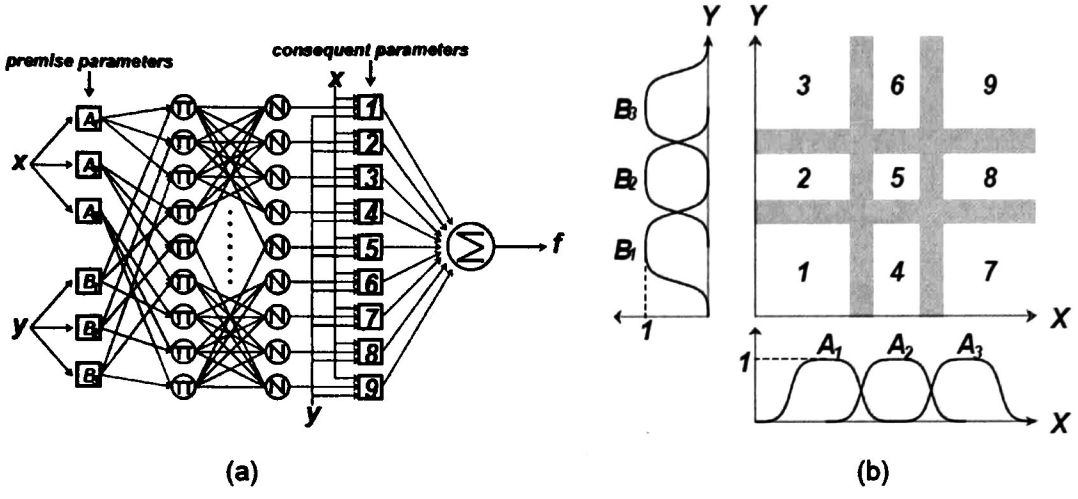


Figure 5.4: ANFIS architecture (2 inputs, 9 rules)

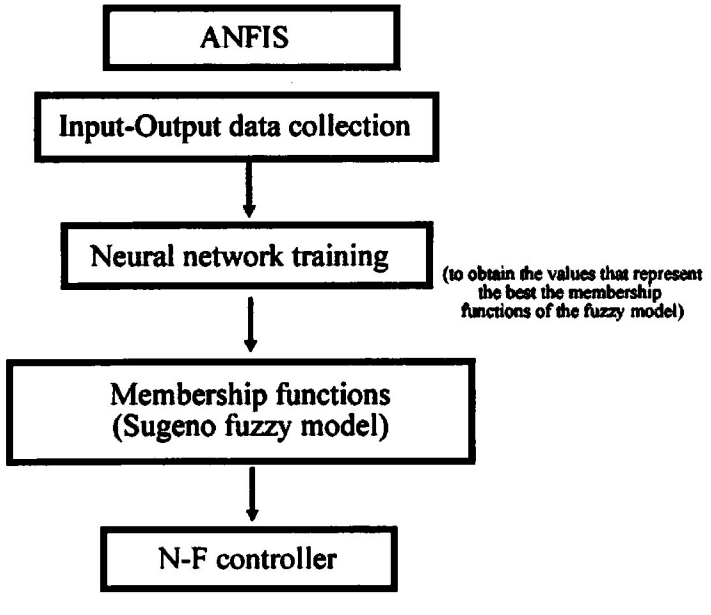


Figure 5.5: Neuro-fuzzy control procedure

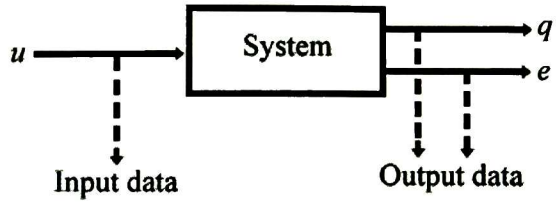


Figure 5.6: Prototype Data used to train the Neural Network

The procedure flowchart used in order to create the neuro-fuzzy controller for the five degrees of freedom redundant robot is shown in Figure 5.5.

Figure 5.6 shows the origin of the data collection obtained from the neural control simulation. This input-output data is used in order to train the ANFIS. The input data is obtained from control signals u applied into the system. The output data is obtained from the position signals q and the tracking error signals e which are produced with the corresponding control signals.

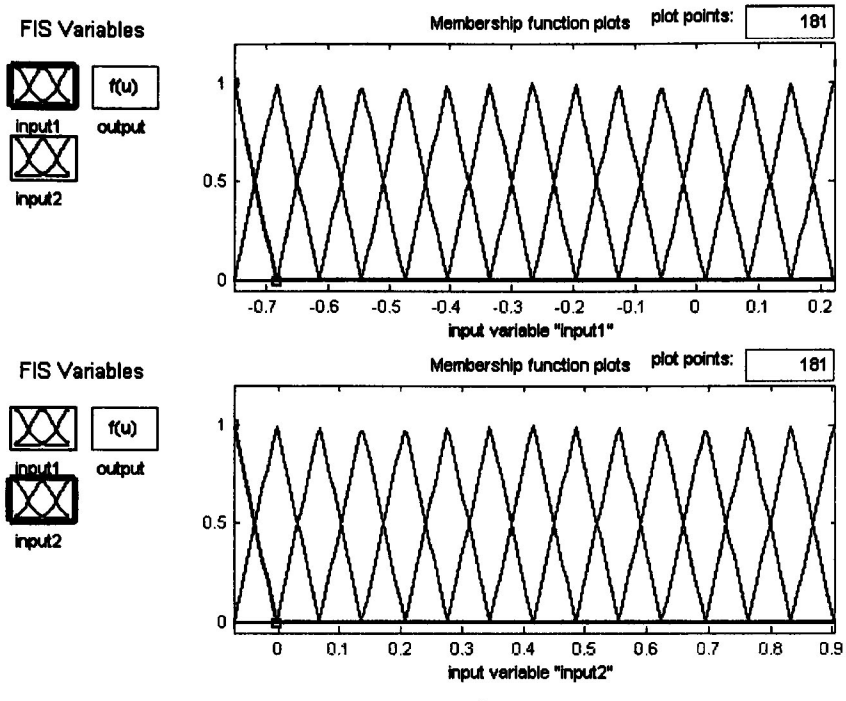
There are proposed fifteen triangular membership functions to describe the position of each of the five joints and fifteen triangular membership functions for the error position in each of them. This number of membership functions for each variable was selected because it shows a satisfactory result. The values for this membership functions will be obtained from the neural network.

The fifteen membership functions considered for the position q of the joints are: $A1$, $A2$, $A3$, $A4$, $A5$, $A6$, $A7$, $A8$, $A9$, $A10$, $A11$, $A12$, $A13$, $A14$ and $A15$. In a similar way, the membership functions that describe the error e are: $B1$, $B2$, $B3$, $B4$, $B5$, $B6$, $B7$, $B8$, $B9$, $B10$, $B11$, $B12$, $B13$, $B14$ and $B15$.

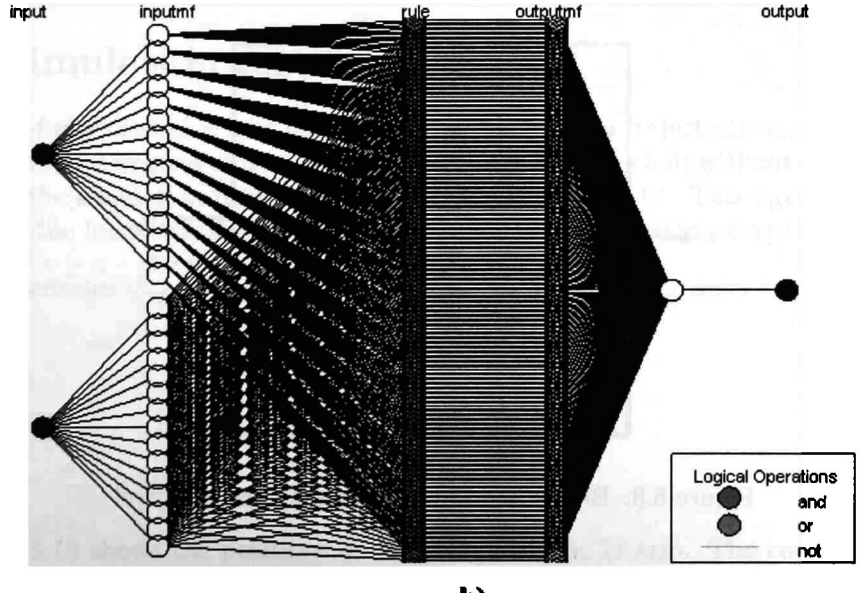
All the membership functions of the position and the error of the system for one of the joints can be seen in Figure 5.7a. The membership functions in the top of the figure belong to the position, and the other ones belong to the tracking error.

Triangular membership functions for the Sugeno fuzzy model of the system were used because, as it was mentioned before, they have simple formulas and they have a good computational efficiency. For this reason, they are commonly used in real time applications. A fewer quantity of Gaussian membership functions would be needed in order to describe the same system, but Triangular membership have better computational efficiency than Gaussian memberships.

In order to control the system there exist 15^2 control rules which have the following form:



a)



b)

Figure 5.7: a) MFs (MATLAB); b) ANFIS architecture (MATLAB)

- If position q is $A1$ and the tracking error e is $B1$ then the control signal is $C1$
- If position q is $A2$ and the tracking error e is $B1$ then the control signal is $C2$
- If position q is $A3$ and the tracking error e is $B1$ then the control signal is $C3$

$\vdots$

- If position q is $A13$ and the tracking error e is $B15$ then the control signal is $C223$
- If position q is $A14$ and the tracking error e is $B15$ then the control signal is $C224$
- If position q is $A15$ and the tracking error e is $B15$ then the control signal is $C225$

Figure 5.7b shows the ANFIS architecture for this two-input system with all the control rules of the system for one of the joints in Matlab®. All the membership functions for each of the inputs can be seen in the figure. As mentioned before, with the neural network training we obtain the values that represent the best the membership functions of the Sugeno fuzzy model for the 5DOF redundant robot.

The block diagram of this control is shown in Figure 5.8. Each of the five fuzzy blocks is responsible to control each of the five joints of the system.

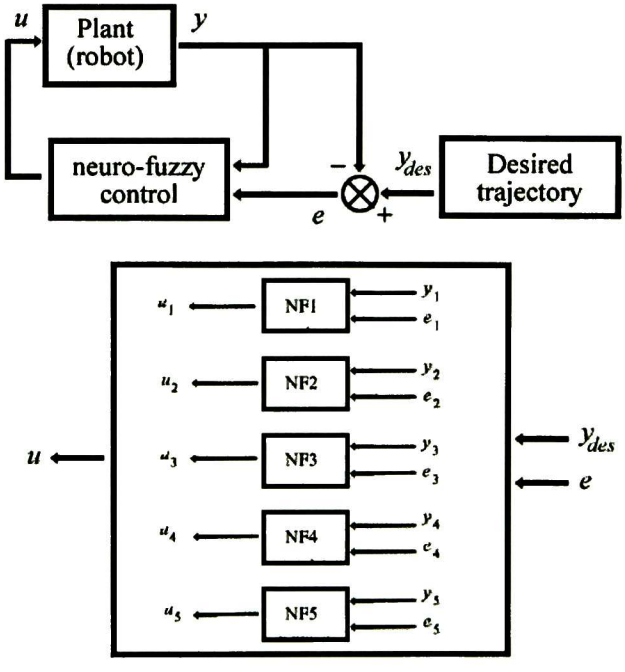


Figure 5.8: Block diagram of the neuro-fuzzy control

Since this control was obtained from input-output data in a certain operating region of the system, the control is not robust to parametric variations and stability out of the operating region may not be warranted. Although it has certain inconveniences, it has

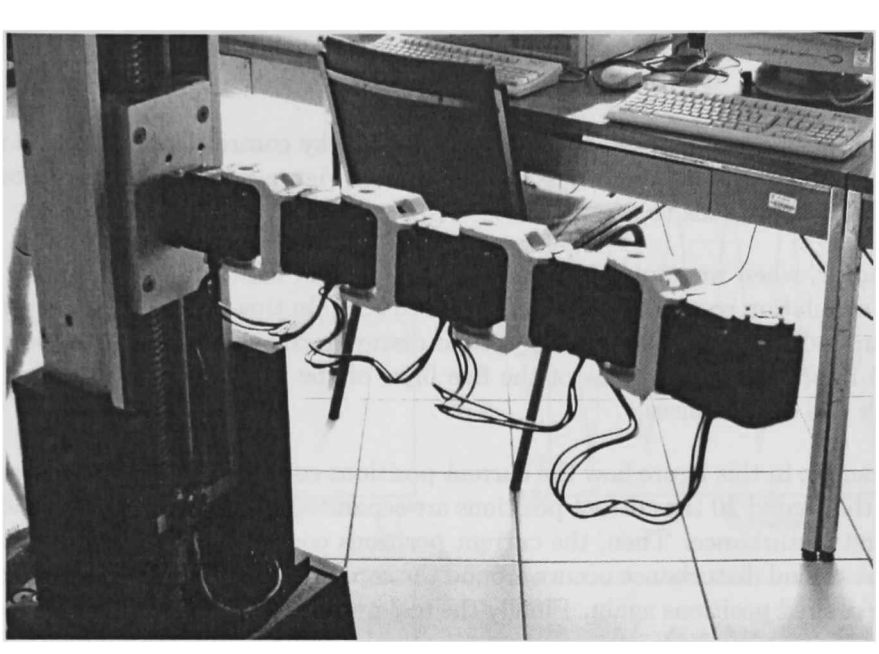


Figure 5.9: 5DOF Redundant Robot

the advantage that it does not require the mathematical model of the system in order to create the control.

5.4 Simulation results

This neuro-fuzzy control is used in order to reach the desired trajectories q_{ref}. Using this control in the five degrees of freedom redundant robot (Figure 5.9) without disturbances we obtain the simulation results shown in Figures 5.10 to 5.12. This figures show the position of the links, tracking errors and control signals generated using this control.

The references q_{ref} used in order to obtain this simulations results are:

$$q_{ref} = \begin{bmatrix} 0.15 \\ 10\frac{\pi}{180}\sin(0.5t) \\ 10\frac{\pi}{180}\sin(0.5t) \\ 6\frac{\pi}{180}\sin(0.5t) \\ 8\frac{\pi}{180}\sin(0.5t) \end{bmatrix} \tag{5.8}$$

Figure 5.10 shows the positions q_1 to q_5 for the robotic arm. The continuous line shows the desired position q_{ref} while the dotted line shows the output position q of the neuro-fuzzy control.

Figure 5.11 shows the control signals generated by the neuro-fuzzy control applied to the five degrees of freedom redundant robot in order to reach the reference signals

q_{ref} (5.8). In this figure, we can see each of the five signals τ_1 to τ_5 for the five links of the system without disturbances.

The tracking error e obtained with this neuro-fuzzy control, applied to the robotic arm, in this case without disturbances, is shown in Figure 5.12. The figure shows how the tracking error tends to zero.

Moreover, when we apply this control in this system with two disturbances, we obtain the simulation results of the Figures 5.13 to 5.15. In this case, the first disturbance occurs around the second 20 and the second disturbance occurs around the second 90. Figure 5.13 shows the positions of the five links of the robotic arm, q_1 to q_5, for this case with two disturbances.

We can see in this figure how the current positions converge to the desired positions. Around the second 20 the current positions are separated from the desired positions due to the first disturbance. Then, the current positions converge to the desired positions. When the second disturbance occurs around the second 90, the current positions diverge from the desired positions again. Finally the real positions converge again in the desired positions.

Figure 5.14 we can see the five control signals for each of the five links of the system τ_1 to τ_5. We can see in the figure how the control signals are adjusted around the second 20 and the second 90 in order to compensate the two disturbances.

Finally, in Figure 5.15 we can see the tracking errors e_1 to e_5 obtained with this neuro-fuzzy control. We can see that the error tends to zero until around the second 20 when the first disturbance occurs. Then, the error tends to zero again until around the second 90 when the second disturbance occurs. Finally, the error tends to zero again.

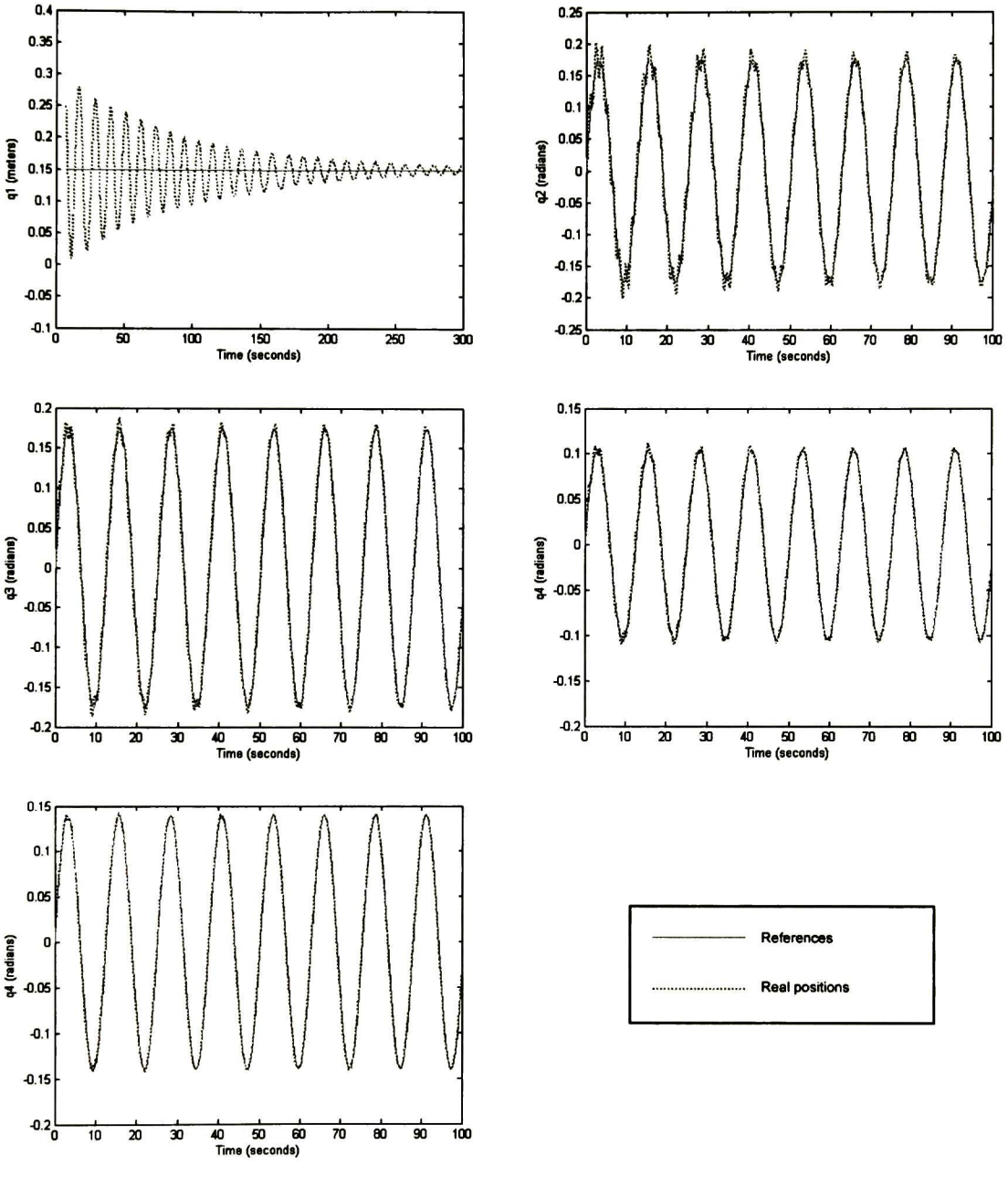


Figure 5.10: Positions (q_i vs. q_{i_ref})

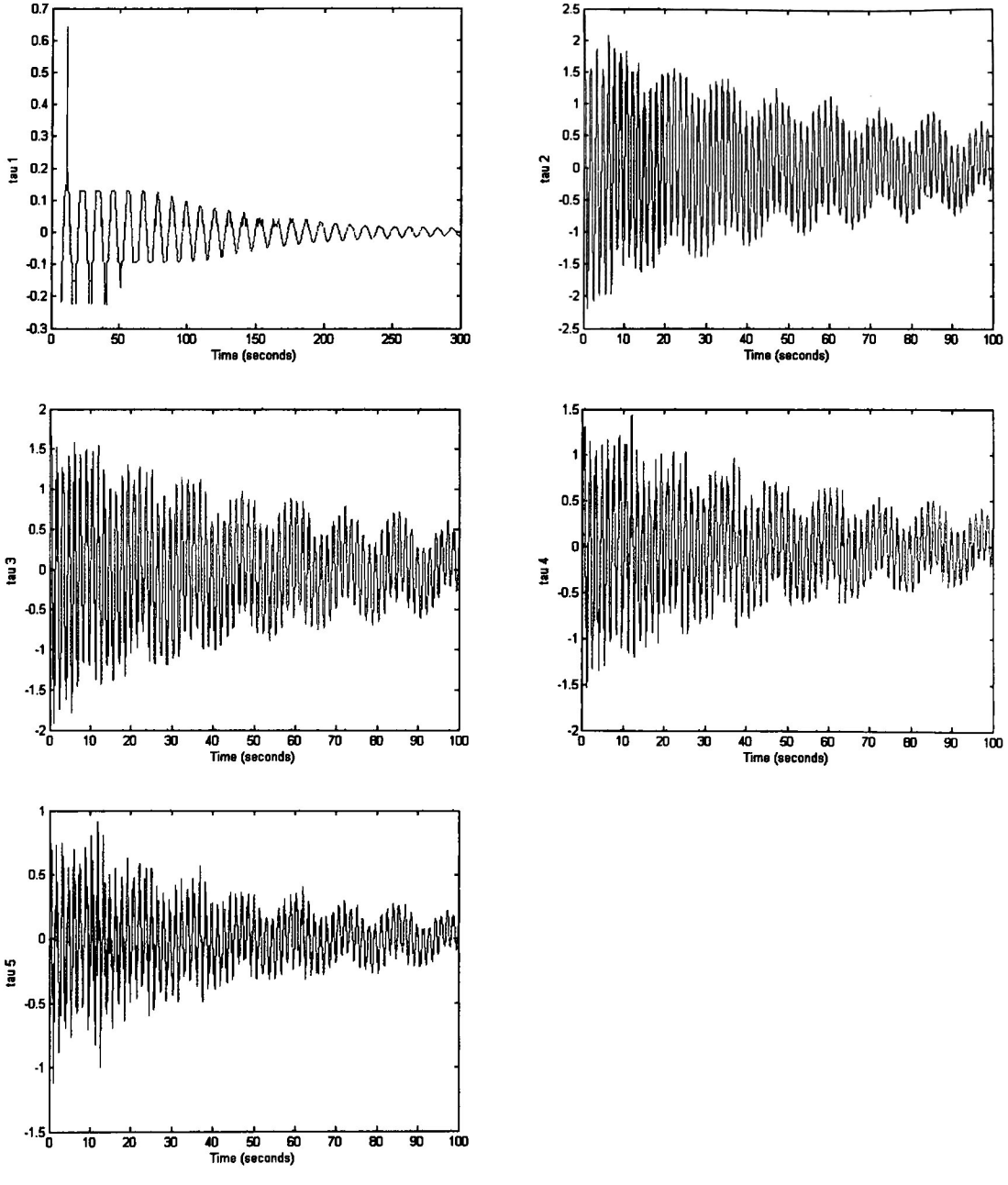


Figure 5.11: Control signals (τ_i)

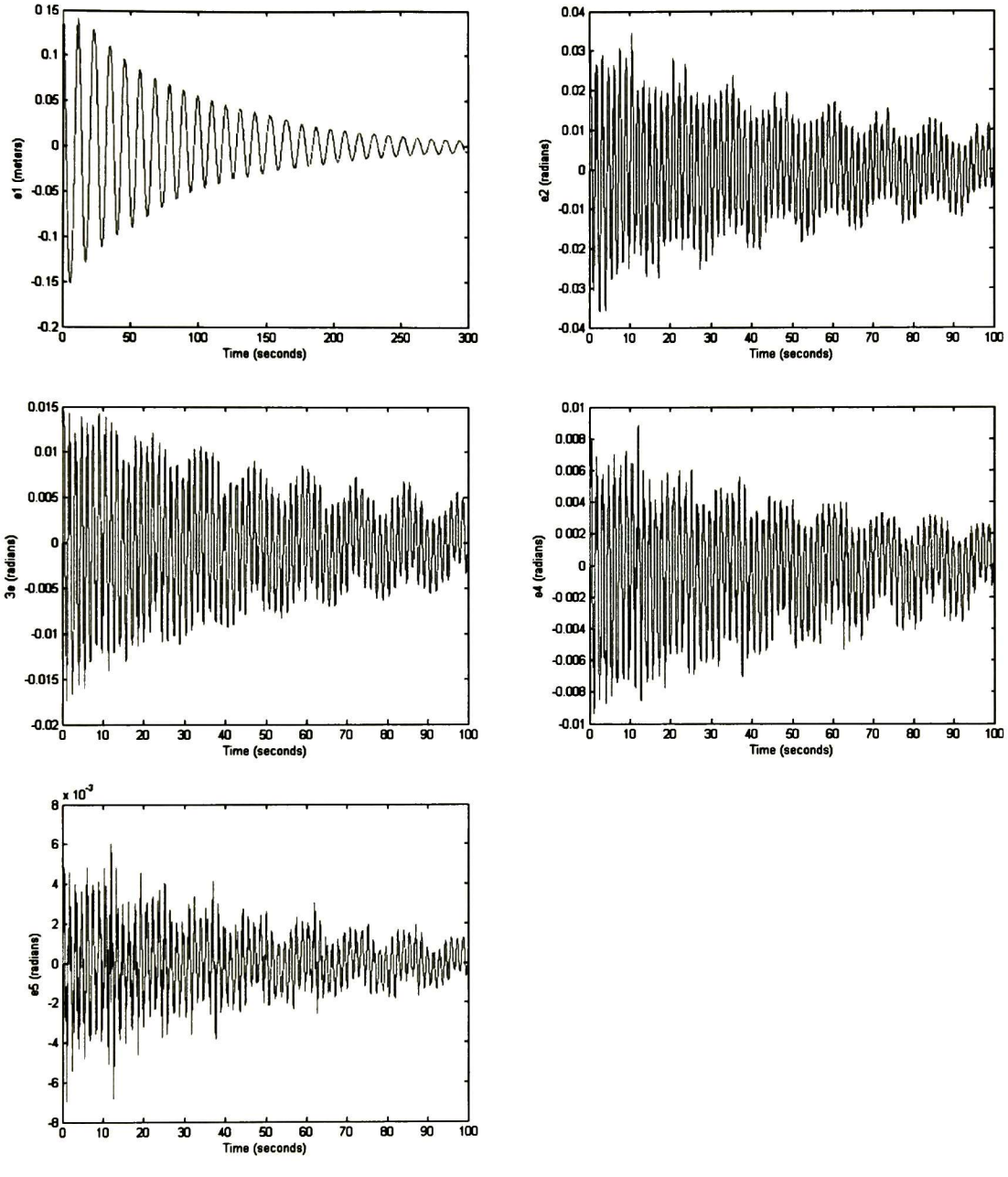


Figure 5.12: Tracking error (e_i)

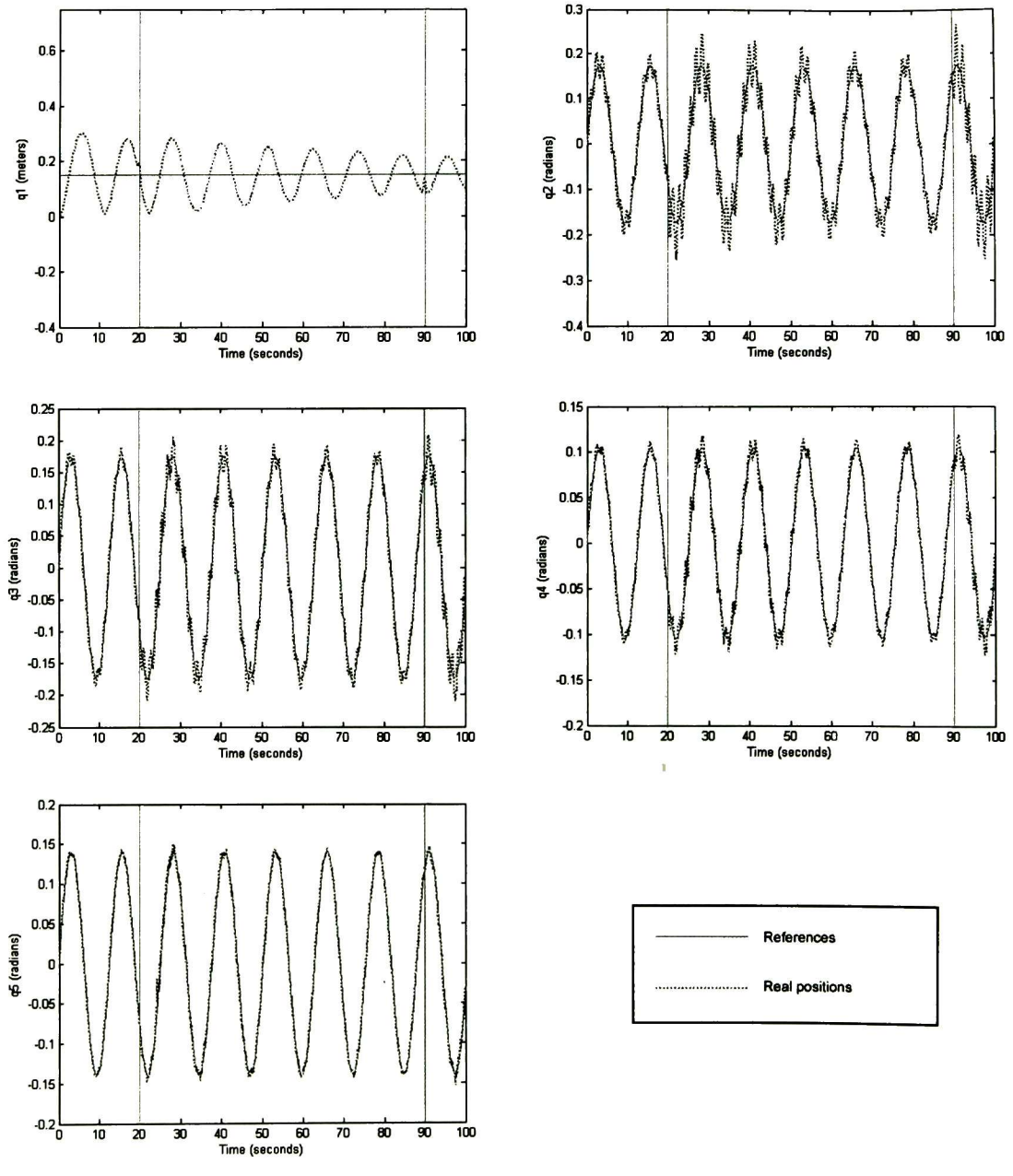


Figure 5.13: Positions (q_i vs. q_{i_ref}) with disturbances

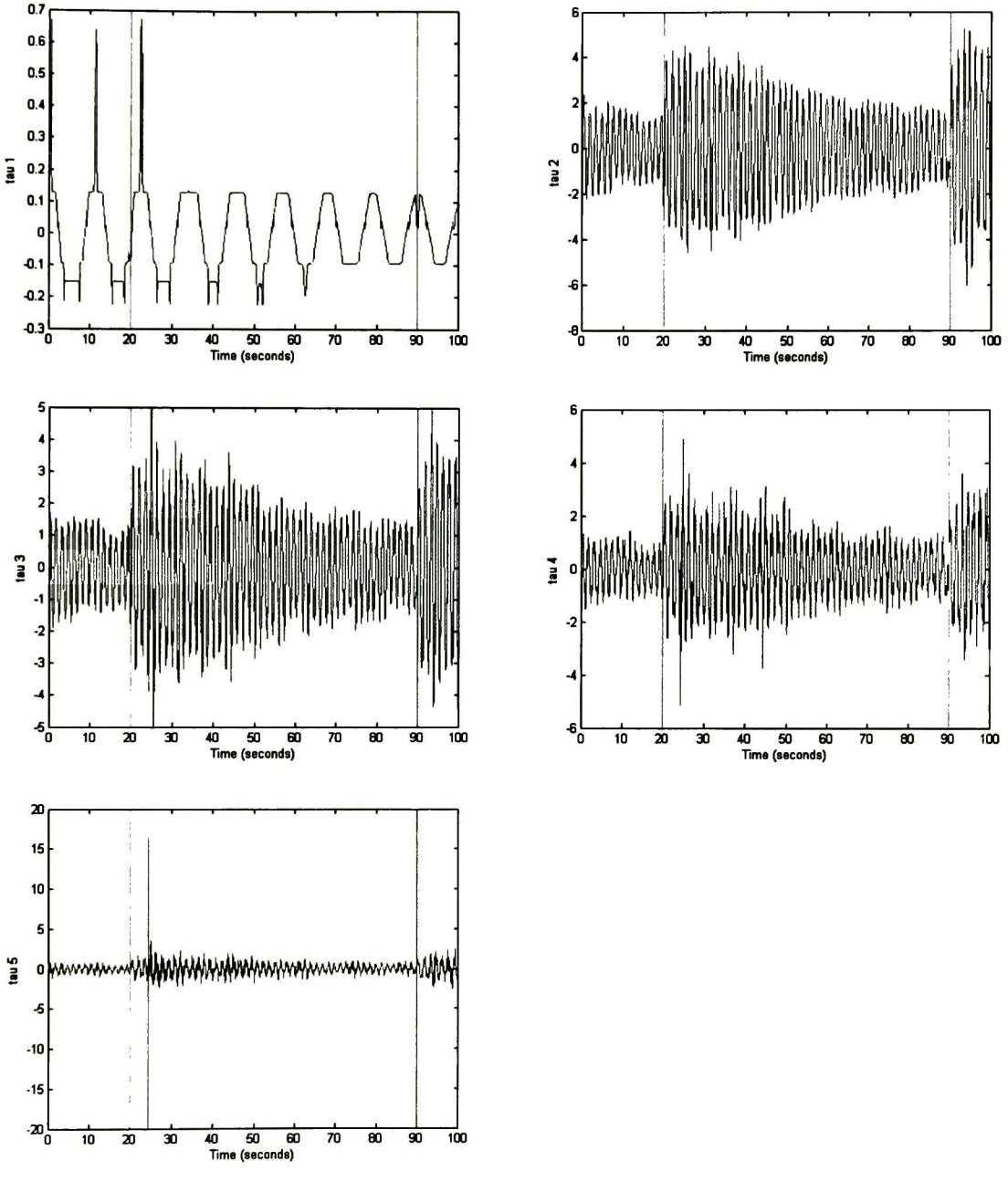


Figure 5.14: Control signals (τ_i) with disturbances

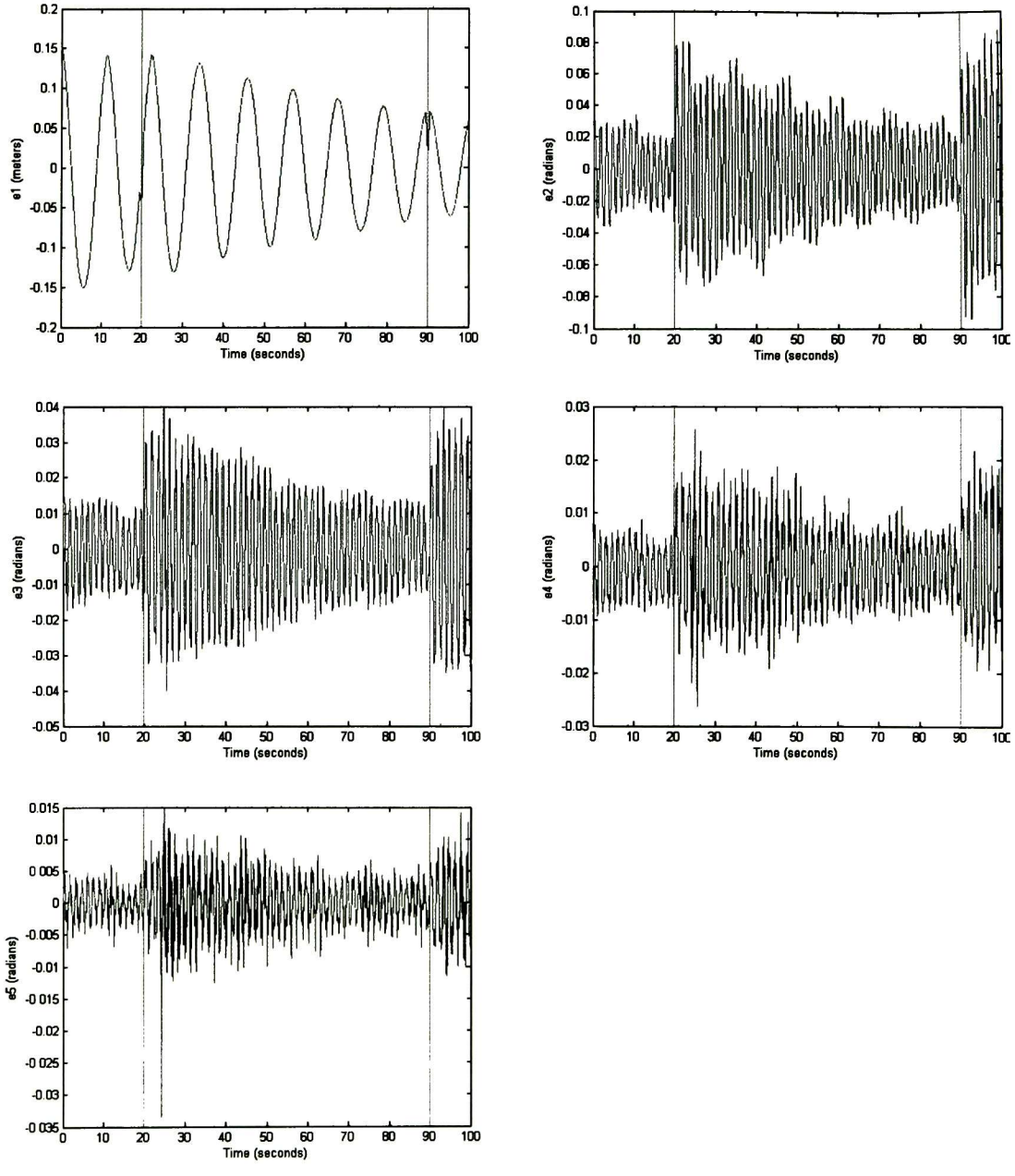


Figure 5.15: Tracking error (e_i) with disturbances

Chapter 6

Prototype construction and implementation

6.1 Original Prototype

The CINVESTAV original prototype was originally based on the seven degrees of freedom ANAT (Articulated Nimble Adaptable Trunk) robot which is located in the ETS (École de Technologie Supérieure) in Montreal, Canada. Figure 6.1 shows the ANAT robot located in Canada.

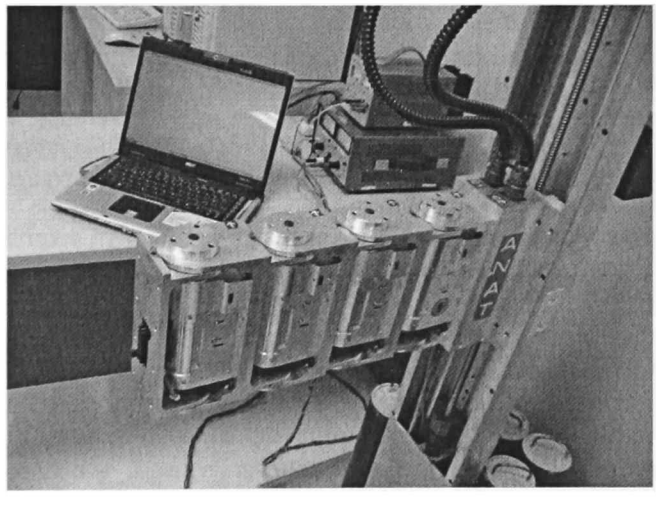


Figure 6.1: ANAT

The original prototype of CINVESTAV had five degrees of freedom. Four of them are rotational joints (angular movement) and the fifth is a prismatic joint (linear movement). The rotational joints were constructed with EYQF-63600-751 Barber Colman motors whereas the prismatic joint was constructed with a 2424P-PSL Baldor motor.

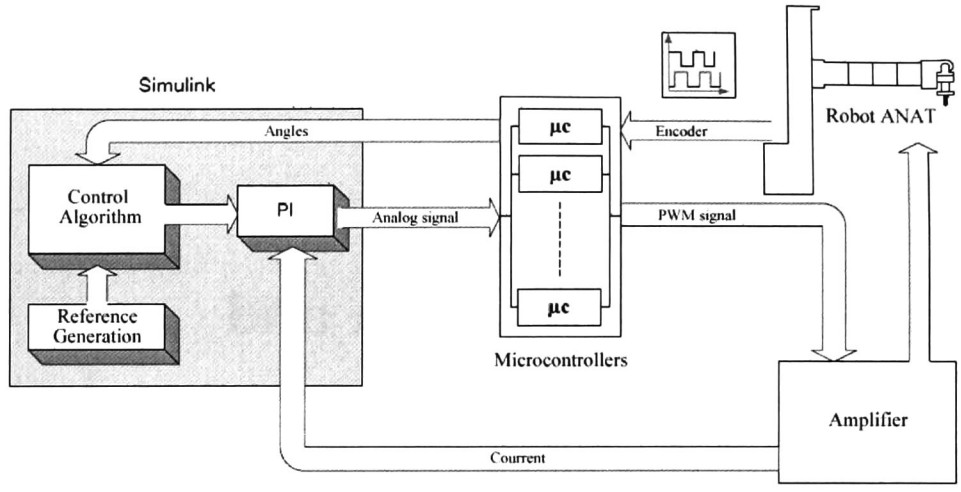


Figure 6.2: Global scheme of the 5DOF robot control architecture

The characteristics of the motors used in the rotational joints and the prismatic joints are:

Joint	Mark	Type	Characteristics
Prismatic	Baldor	2424P-PSL	42rpm; 90VDC; 280in-Lbs
Rotational	Barber Colman	EYQF-63600-751	20.67rpm; 24VDC; 110.86oz-in

The control signals are created in a computer with the Matlab/Simulink software with five control signals, one for each joint. Then, these control signals are sent from the computer to the prototype through a National Instruments data acquisition board (NI DAQ Card-6042E). Figure 6.2 shows the scheme of the prototype control architecture scheme.

Since the data acquisition board used in the prototype is limited to the number of analog outputs, the five control signal are sent through only one analog output of the DAQ from the Matlab/Simulink software using Time-division multiplexing (TDM) (Figure 6.3). In order to do this, a digital counter is used, which counts from 1 to N, where N is the biggest number of the control signals to be multiplexed. For the 5DOF robot case, N is equal to 5 because there are five control signals. Also, a reset digital signal is used in combination with the digital counter to detect each time that the counter is reseted (Figure 6.4).

In the demultiplexing stage, five microcontrollers are used in order to reconstruct all of the five control signals. Each microcontroller reconstructs one of the five signals using the counter and the reset digital signals. When the value of the counter is the same as the value of the joint which has been assigned to each of the microcontrollers, then that

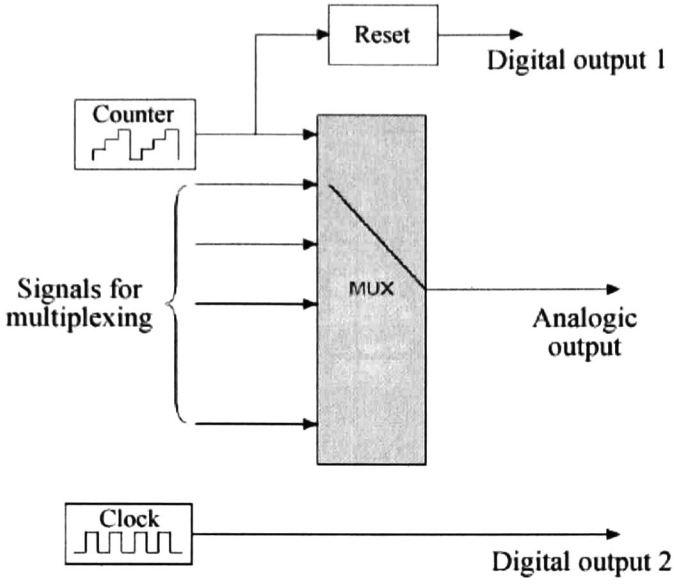


Figure 6.3: Multiplexing the N control signals

microcontroller uses the analog signal to generate a PWM (Pulse Width Modulation) signal. Figure 6.5 the circuit for one of the microcontrollers which generates one of the PWM signals for one of the joints.

The microcontrollers generate all the PWM signals. Then, these signals are amplified and sent to each joint of the robotic arm prototype (four motors for the rotational joints and one motor for the prismatic joint).

6.2 Current Prototype

In order to improve the original CINVESTAV prototype, some modifications have been done. The motors used in the rotational joints have been changed and the motor of the prismatic joint has been connected.

The EYQF-63600-751 Barber Colman motors were replaced for Dynamixel AX-12 in all the four rotational joints. The electronic interphase was changed because the communication protocol of the Dynamixel AX-12 is different than the one of the previous motors. The multiplexing, demultiplexing and amplifying stages, which were used to create the motors control signals, were removed. Therefore, the electronic interphase was simplified.

The control signal for the prismatic joint is obtained from a computer which uses the NI DAQ Card-6042E to send an analog signal to a PWM servo driver. This driver

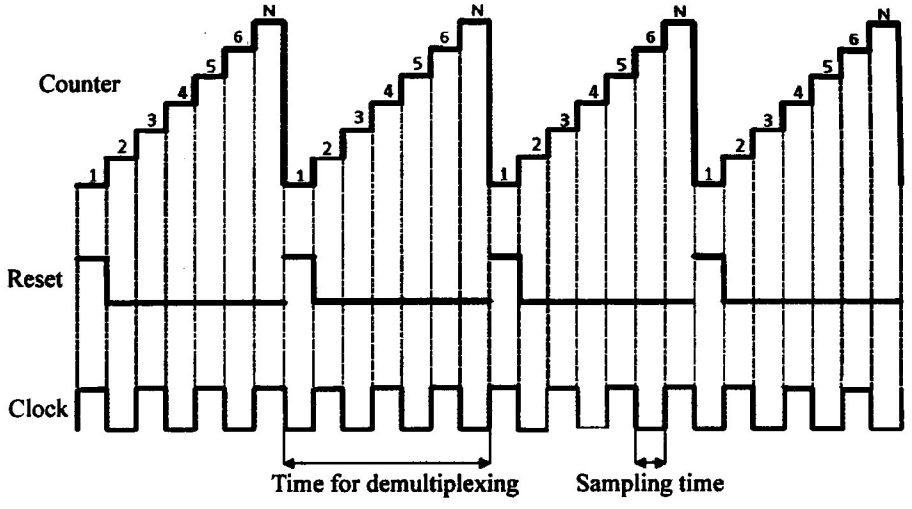


Figure 6.4: Demultiplexing

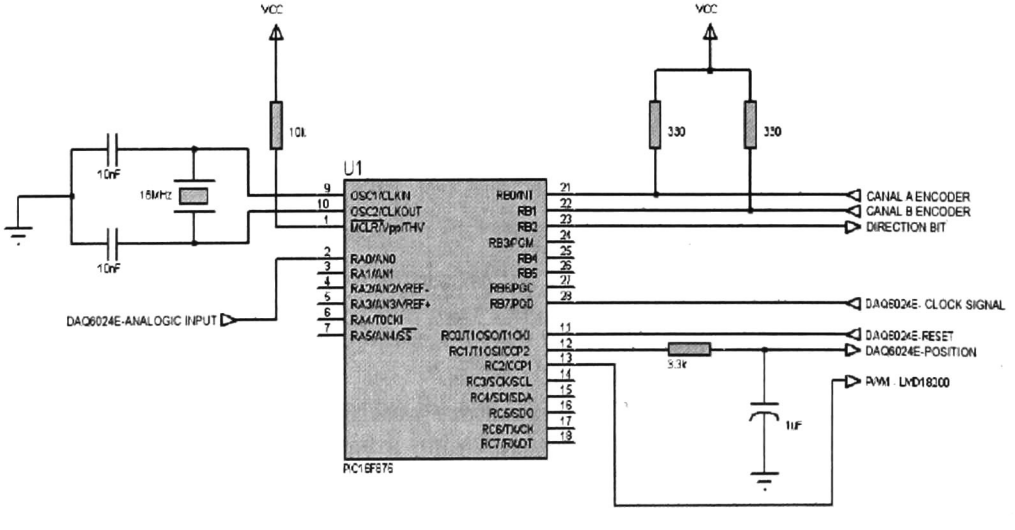


Figure 6.5: Microcontroller diagram

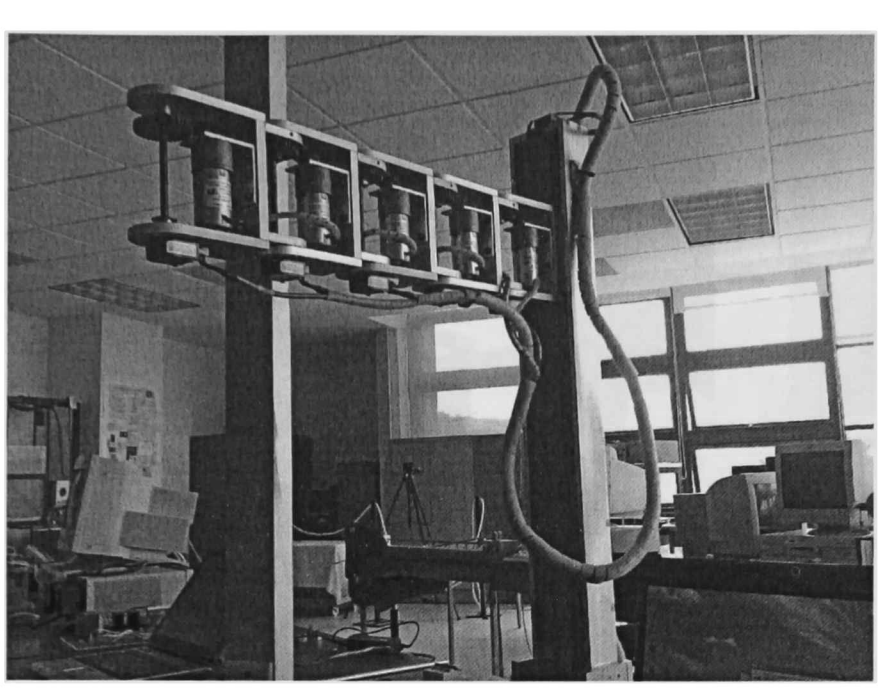


Figure 6.6: Original CINVESTAV Prototype

generates the PWM signal which in turn is sent to the 2424P-PSL Baldor motor of the prismatic joint in order to generate the liner movement of the robotic arm.

The rotational joints were implemented using four Dynamixel AX-12 motors. A computer program made in Simulink or in LabView® generates the instruction packets for all of theses motors in order to make them move. Since they use a serial protocol, the signals for all of the motors connected in series are sent by the same connection (Figure 6.8).

The same computer that generates the control signals for the rotational joints also generates the reference signal to control the prismatic joint. This signal is sent to the PWM servo driver through one of the analog outputs of the NI DAQCard-6024E. Then, the PWM servo driver sends the PWM control signal to the prototype.

All the components used in the current five degrees of freedom redundant robot from CINVESTAV are shown in appendix B. Moreover, all the component specifications are also listed in this appendix.

6.3 Remote use

These prototypes may have a lot of applications when they are controlled remotely. They may have medical, industrial or scientific applications depending of the end ef-

Figure 6.7: Current CINVESTAV Prototype

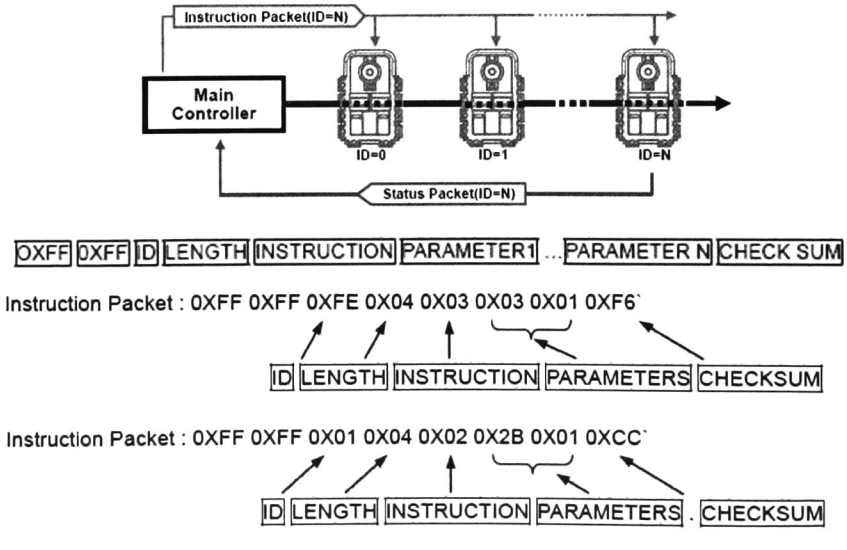


Figure 6.8: Dynamixel AX-12 instruction packets

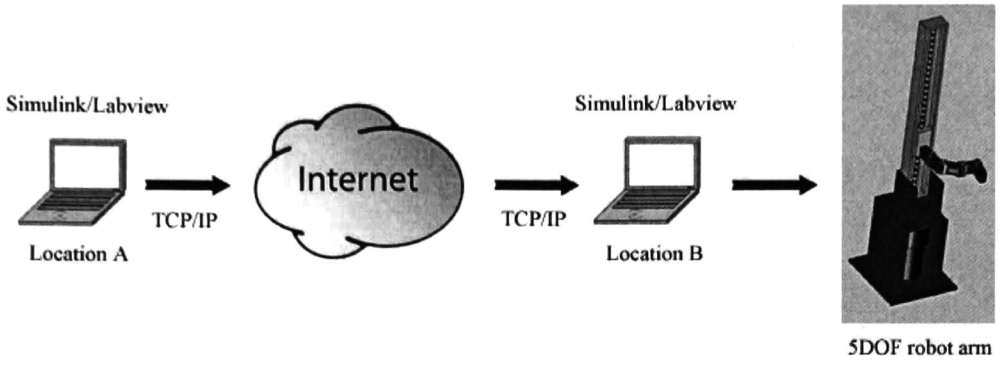


Figure 6.9: TCP/IP protocol

fector that is used. For that reason, it is of great interest to be able to manipulate these prototypes remotely. Therefore, it is proposed to control the CINVESTAV prototype using the TCP/IP communication protocol, Transmission Control Protocol (TCP) and the Internet Protocol (IP).

Using this protocol it is possible to send the control signals via Internet from one computer which may be in one particular location to another computer located in a remote location which is connected directly to the robotic arm prototype.

In order to do this, one computer generates the control signals (created in a Simulink or in a LabView program) and acts as a transmitter. These signals are sent to the remote computer which has also Simulink or LabView programs and acts as the receiver. Then, the later sends the control signals to the physical prototype.

Figure 6.9 shows how the control signals are generated in a computer which is in the location A. Then theses signals are sent over the Internet using the TCP/IP protocol to the computer in the location B, which is connected directly to the 5DOF robot arm.

The transmitter and receiver, which use the TCP/IP commutations protocol in Simulink, are shown in Figure 6.10a. Figure 6.10b shows the use of the TCP/IP communications protocol in LabView.

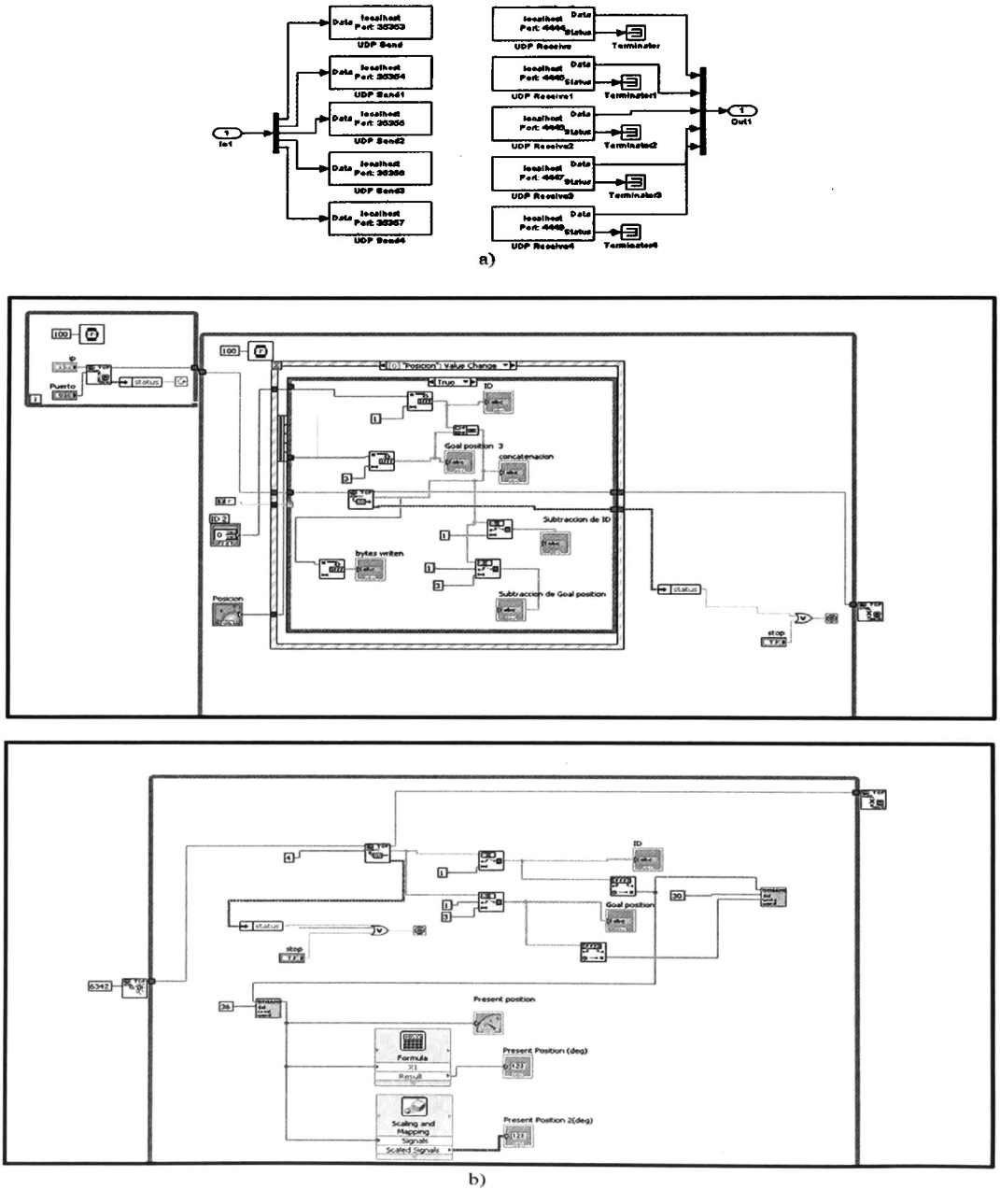


Figure 6.10: TCP/IP protocol in: a) simulink; b) labview

Chapter 7

Real time results

This chapter shows real-time implementations for the five degrees of freedom redundant robot of CINVESTAV. First we analyze an implementation using the inverse kinematics technique, discussed in Chapter 2. Finally, we show the implementation of one of the controls designed for this prototype presented in Chapters 3, 4 and 5.

The Dynamixel motors, used in the rotational links of the robot, can be used in two ways: either by position or by speed. First, we use the inverse kinematics technique in order to follow a specific trajectory given the angular positions of the links. In this first implementation the Dynamixel motors are controlled by angular positions. Finally, we use a neuro-fuzzy control (discussed in Chapter 5) in order to follow a specific trajectory. In this implementation, the Dynamixel motors are controlled by angular speed.

For real-time implementation using the technique of inverse kinematics and the control designed for the five degrees of freedom redundant robot we used the Matlab/Simulink software.

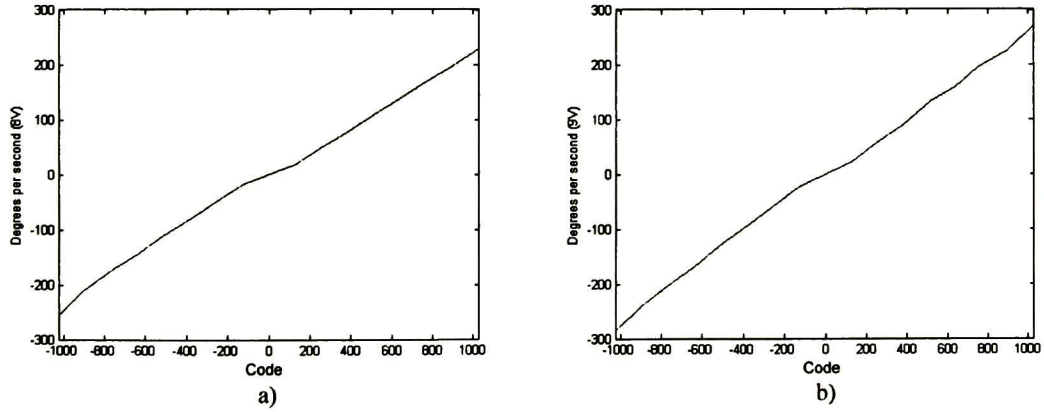


Figure 7.1: Dynamixel speed: a) 8 Volts; b) 9 Volts

Appendix D shows the embedded code used in Simulink in order to read and write real-time data in the Dynamixel motors. In this case, this code sets either the desired position or the desired speed to be implemented in the motor. Moreover, this code gets the current speed and the current position in the four Dynamixel motors used in the rotational links of the robotic arm.

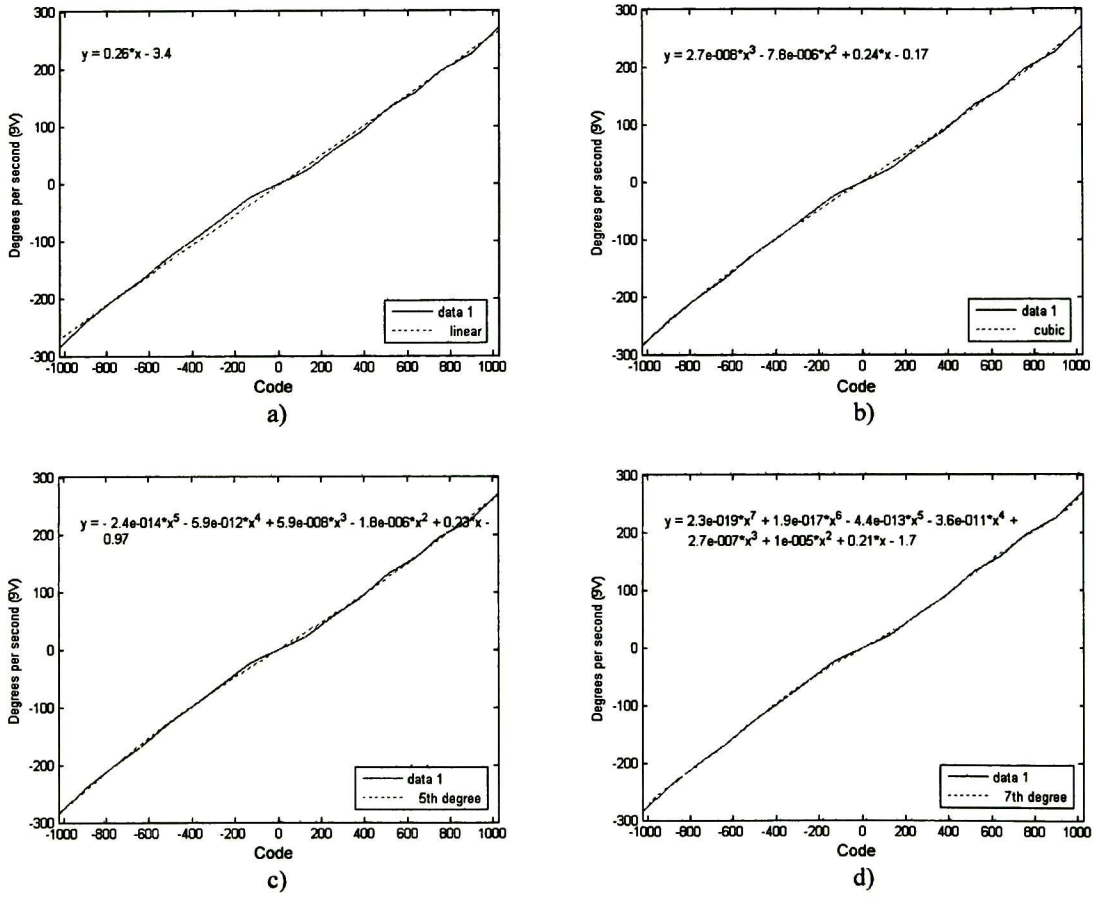


Figure 7.2: Approximation of the dynamixel speed at 9V: a) linear; b) cubic; c) 5th degree; d) 7th degree

When we use the Dynamixel motors by angular position, there exists a linear relationship between the code used and the position angle obtained in the motor. Moreover, the angular position of the Dynamixel motor does not depend on the voltage supplied.

When we control these motors by angular speed and we want to indicate the desired clockwise speed on the Dynamixel motor, then we use a Dynamixel speed code (in one of the Dynamixel registers) from 0 to 1023. 1023 represents the maximum angular speed and 0 represents no angular speed in the motor. Similarly, in order to indicate the

desired counter-clockwise angular speed we use a Dynamixel speed code from 1024 to 2047, where 2047 represents the maximum counter-clockwise angular speed and where 1024 represents no angular speed.

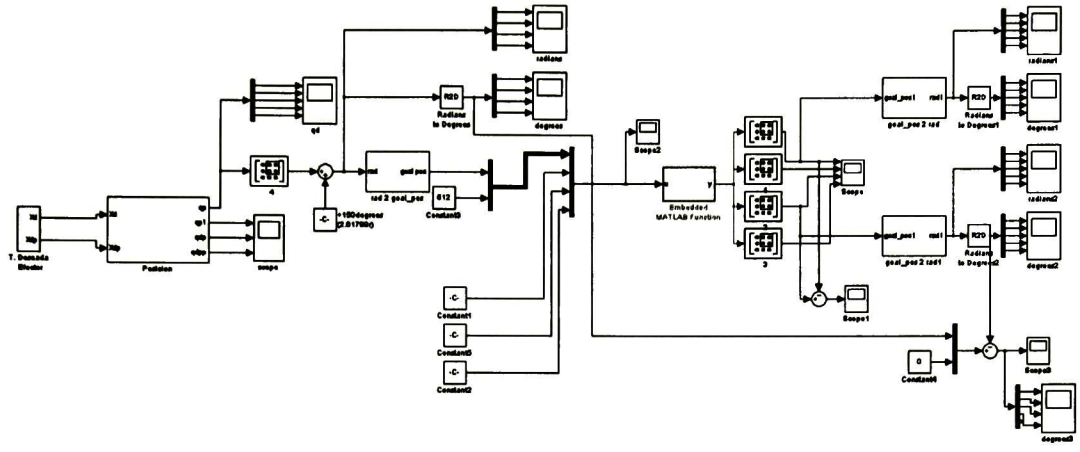


Figure 7.3: Simulink program (Inverse Kinematics)

The Dynamixel motor speed depends on the voltage supplied, this is, with a higher voltage (e.g. 9V) the maximum velocity represented by 1023 will be much greater than the maximum speed with a lower voltage (e.g. 8V). This is illustrated in Figure 7.1 which shows the angular velocities obtained experimentally for a Dynamixel motor energized with 8 volts (Figure 7.1a) and 9 volts (Figure 7.1b).

The motor speed (obtained for 1024 values of the Dynamixel speed code for the clockwise case and 1024 values for the counter-clockwise case) is not represented by a linear function. Figure 7.2 shows four approaches that represent the speed function of the Dynamixel speed at 9 Volts.

Figure 7.2a shows an approximation by the linear function $y = 0.26x - 3.4$. We may see that this first approach is quite different from the actual speed function obtained experimentally. Figure 7.2b shows the approximation by the cubic function $y = 2.7\ 10^{-8}x^3 - 7.8 \cdot 10^{-6}x^2 + 0.24x - 0.17$ which is a little better than the first approach. Figure 7.2c shows the approximation by the polynomial of 5th order $y = -2.4 \cdot 10^{-14}x^5 - 5.9 \cdot 10^{-12}x^4 + 5.9\ \ 10^{-8}x^3 - 1.8 \cdot 10^{-6}x^2 + 0.23x - 0.97$. Finally, Figure 7.2d shows the approach through the 7th order polynomial $y = 2.3 \cdot 10^{-19}x^7 + 1.9 \cdot 10^{-17}x^6 - 4.4\ 10^{-13}x^5 - 3.6 \cdot 10^{-11}x^4 + 2.7\ \ 10^{-7}x^3 + 1\ \ 10^{-5}x^2 + 0.21x - 1.7$. We may see that this latter approach better represents the real speed function obtained experimentally.

This angular speed ϖ will have a torque force N related to the power of the motor P as follows:

$$P = N\varpi$$

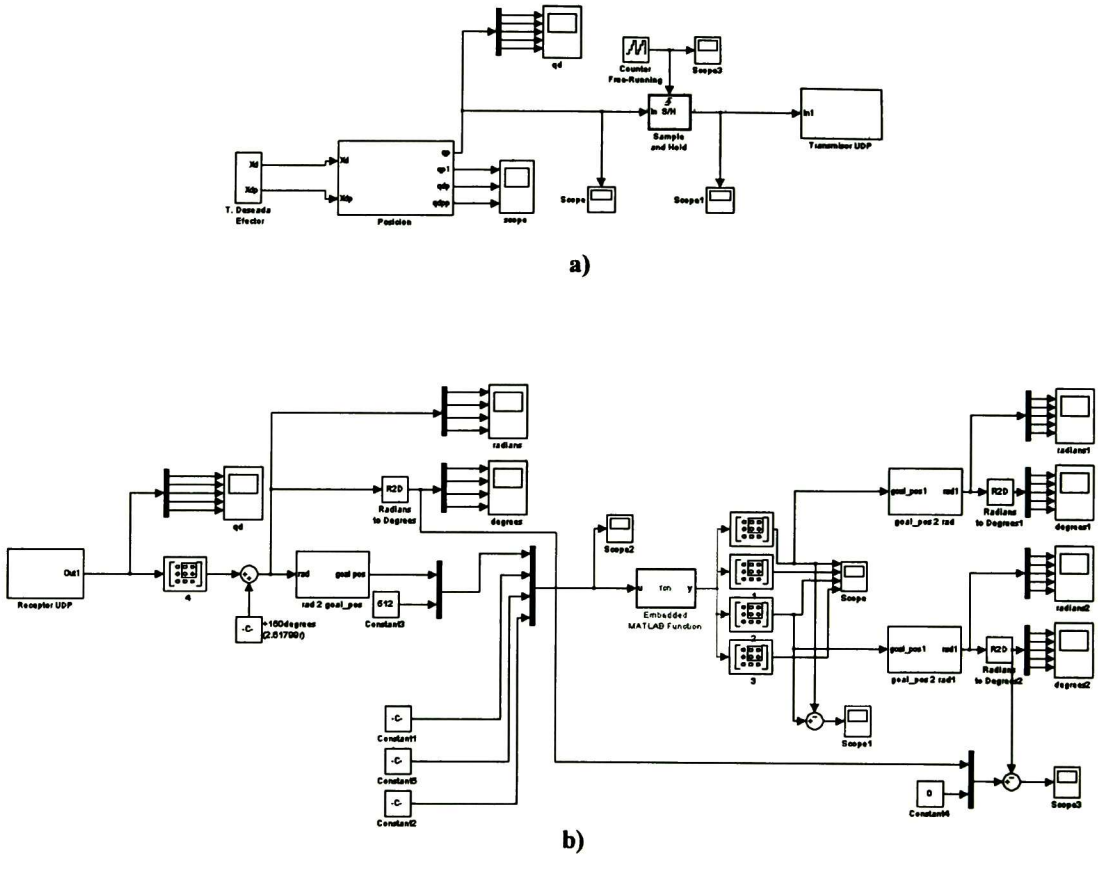


Figure 7.4: a) Trasmitter; b) Receiver

7.1 Inverse kinematics

The inverse kinematics for the five degrees of freedom redundant robot was discussed in Chapter 2. This implementation uses the inverse kinematics in order to make the robotic arm to follow a specific trajectory.

Moreover, this implementation uses the TCP/IP protocol in real-time in order to manipulate the prototype from a remote computer through the Internet. The remote computer generates the desired signals which are received by the computer connected to the prototype.

The Matlab/Simulink program which uses the inverse kinematics technique (discussed in Chapter 2) in the five degrees of freedom redundant robot is shown in Figure 7.3. The prototype follows a desired trajectory, in a plane of a cartesian reference system, with this program which uses the inverse kinematics in real time.

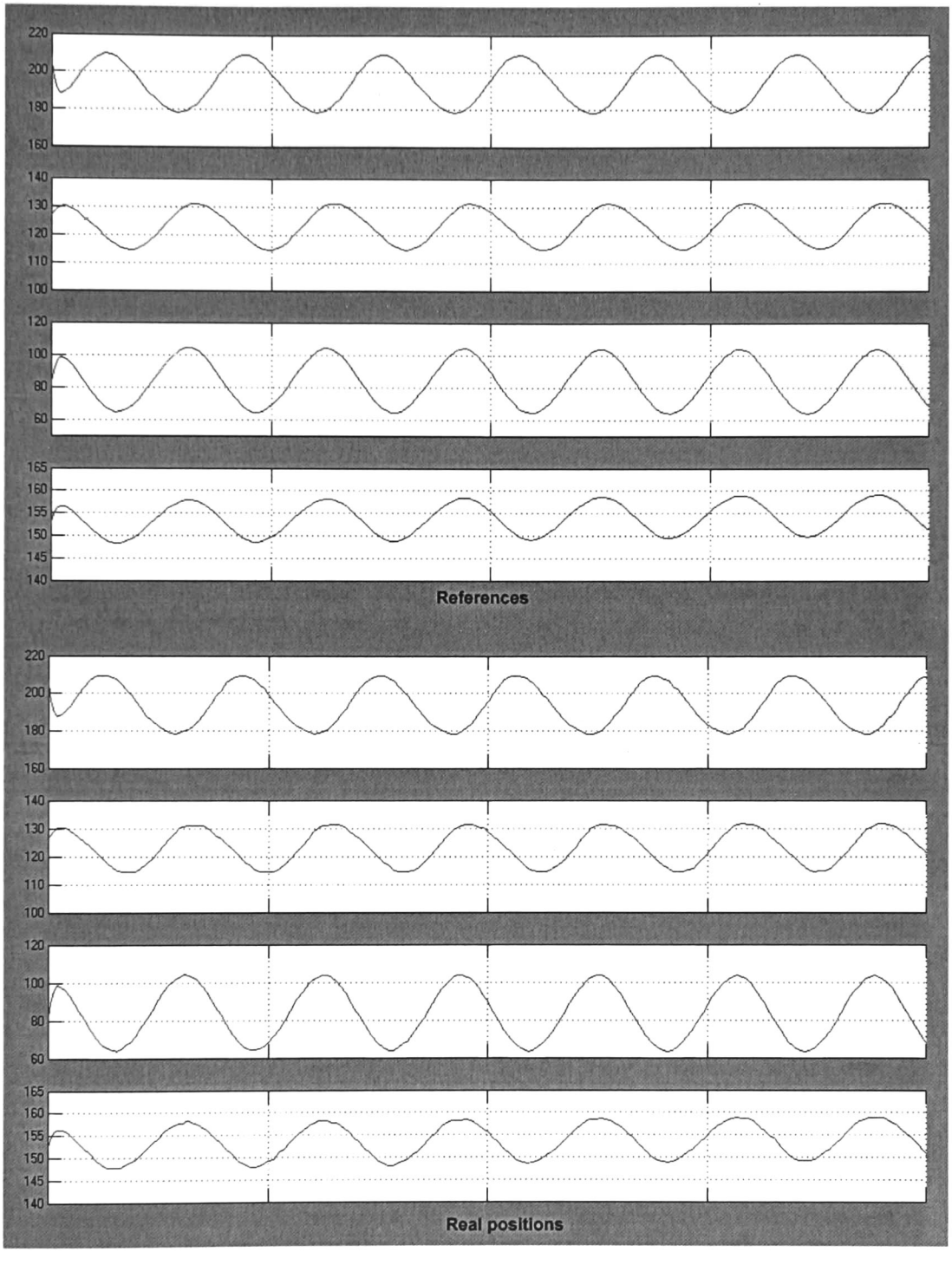


Figure 7.5: References and Real Positions (Inverse Kinematics)

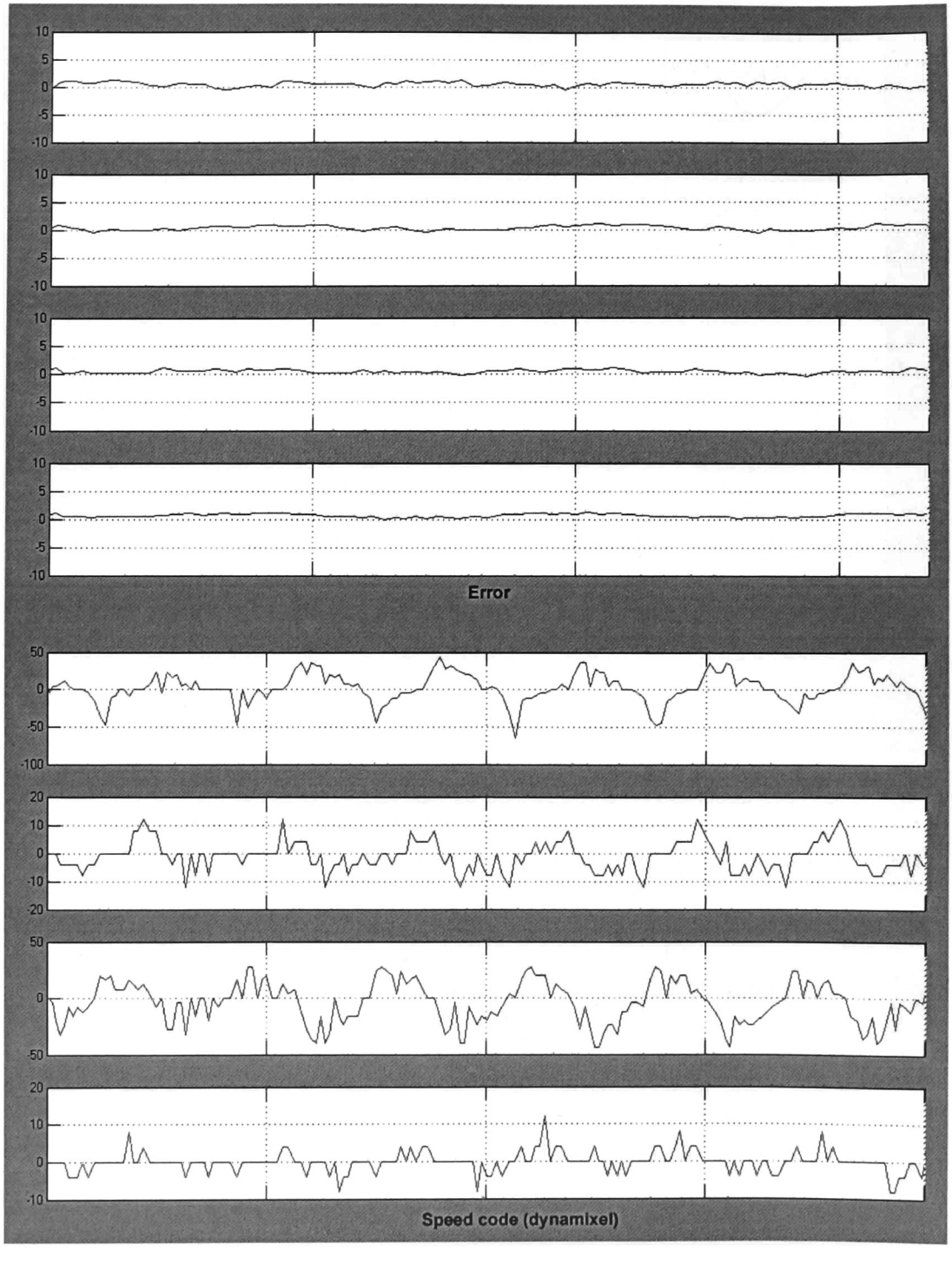


Figure 7.6: Tracking Error and Dynamixel Speed Code (Inverse Kinematics)

Figure 7.4 shows the Matlab/Simulink programs that use the inverse kinematics technique which use the TCP/IP protocol in order to manipulate the redundant protocol through the Internet. Figure 7.4a shows the program on the remote computer, which is responsible for transmitting through the Internet the signals generated using the inverse kinematics technique.

Subsequently, the computer connected directly to the five degrees of freedom prototype receives these signals via the Internet, which are sent to the prototype. Figure 7.4b shows the Matlab/Simulink program used as receptor, which is responsible for receiving these signals from the Internet and send them to the redundant robot.

The desired trajectories for each of the four rotational links of the redundant robot are shown in Figure 7.5. This Figure also shows the real trajectories obtain in the prototype for the four rotational links.

Finally, Figure 7.6 shows the tracking error obtained from the implementation of this technique in real time of the five degrees of freedom redundant robot of CINVESTAV. We can see in this figure how the error generated from this technique (inverse kinematics) is close to zero for each link in the prototype. The Dynamixel speed signal implemented on each motor is also shown in this figure.

7.2 Neuro-fuzzy control

In Chapter 5, neuro-fuzzy controls are discussed. In addition, a neuro-fuzzy control was designed and simulated for the five degrees of freedom redundant robot. This implementation uses a neuro-fuzzy control in order to make the robotic arm to follow a specific trajectory.

In order to create the neuro-fuzzy control, an Adaptive Neuro-Fuzzy Inference System (ANFIS) is trained with an input-output data collection that describes the behavior of the 5DOF redundant robot in a certain region of operation for a specific trajectory. This data collection has been obtained experimentally from the CINVESTAV prototype in real time for this specific trajectory. The ANFIS is responsible for generating the membership functions used to control the system.

The data includes values of the position of the rotational joints, the position error and the signals used in the Dynamixel motors in order to set an angular speed for this specific trajectory. These speed signals produce a torque force on the Dynamixel motors. We use angular speed because it is easier to control the Dynamixel motors by angular speed than by torque forces.

A neural network is trained with this data collection in order to obtain the values that represent the best the membership functions of a Sugeno fuzzy model for the 5DOF redundant robot. Figure 7.7 shows the membership functions of the Sugeno fuzzy model. These membership functions are triangular for this real time application

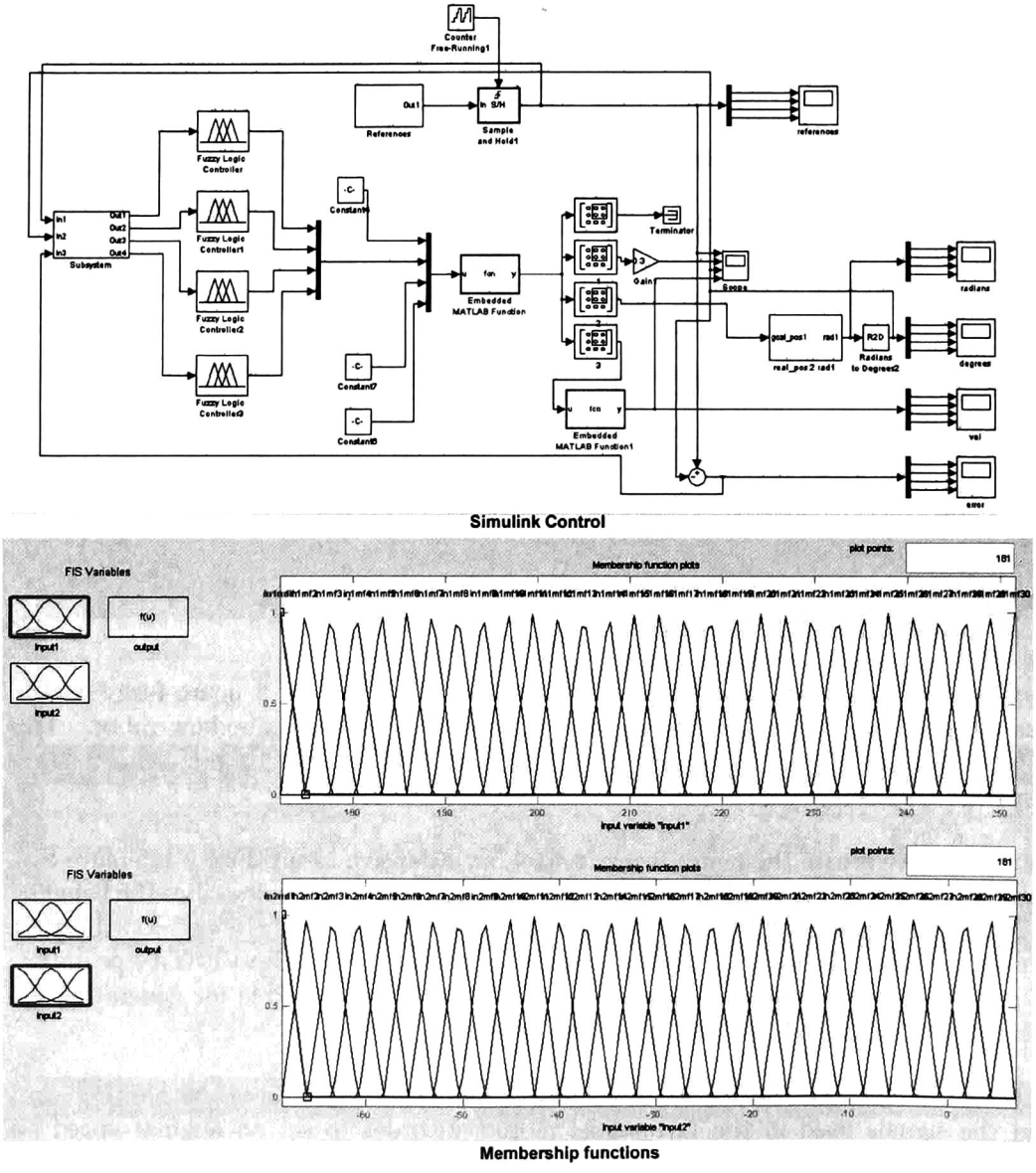


Figure 7.7: Neuro Fuzzy Control; Membership Functions

because they have simple formulas and they have a good computational efficiency (as mentioned before in Chapter 5).

The neuro-fuzzy control used in Matlab/Simulink designed in order to control the redundant robot of CINVESTAV in real time is shown in Figure 7.7. The embedded code shown in Appendix D is used in order to send and read data on the Dynamixel motors in real time.

The reference signals (in a plane of a cartesian reference system) used in the robotic arm are shown in Figure 7.8. This figure also shows the real positions obtained in real time from the neuro-fuzzy control applied in the prototype.

Figure 7.9 show the Dynamixel speed code used in order to follow the desired references for the four rotational links of the redundant robot. Figure 7.9 also shows the tracking error obtained with the neuro-fuzzy control applied in the prototype in real time.

As mentioned before, this control is not robust to parametric variations and stability out of the operating region may not be warranted because it was obtained from input-output data in a certain operating region of the system. However, this neuro-fuzzy control has the advantage that it does not require the mathematical model of the system in order to create the control.

As future work, this neuro-fuzzy control could be improved in order to make it robust to parametric variations, and also to warranty stability out of the range of operation.

In this case, we control the Dynamixel motors by angular speed, but they could be controlled by torque forces adding a current sensor to the motors or using a torque observer.

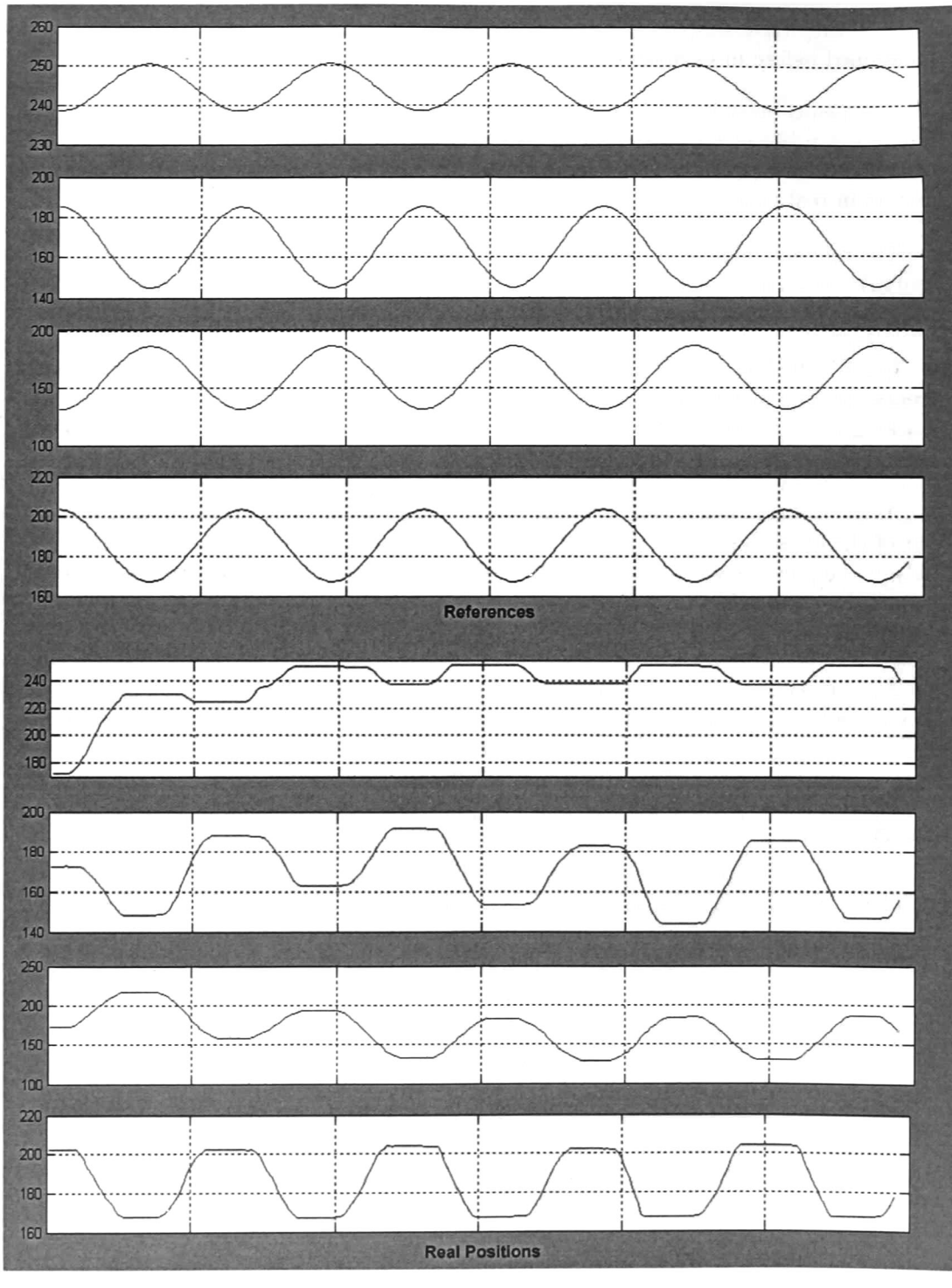


Figure 7.8: References and Real Positions

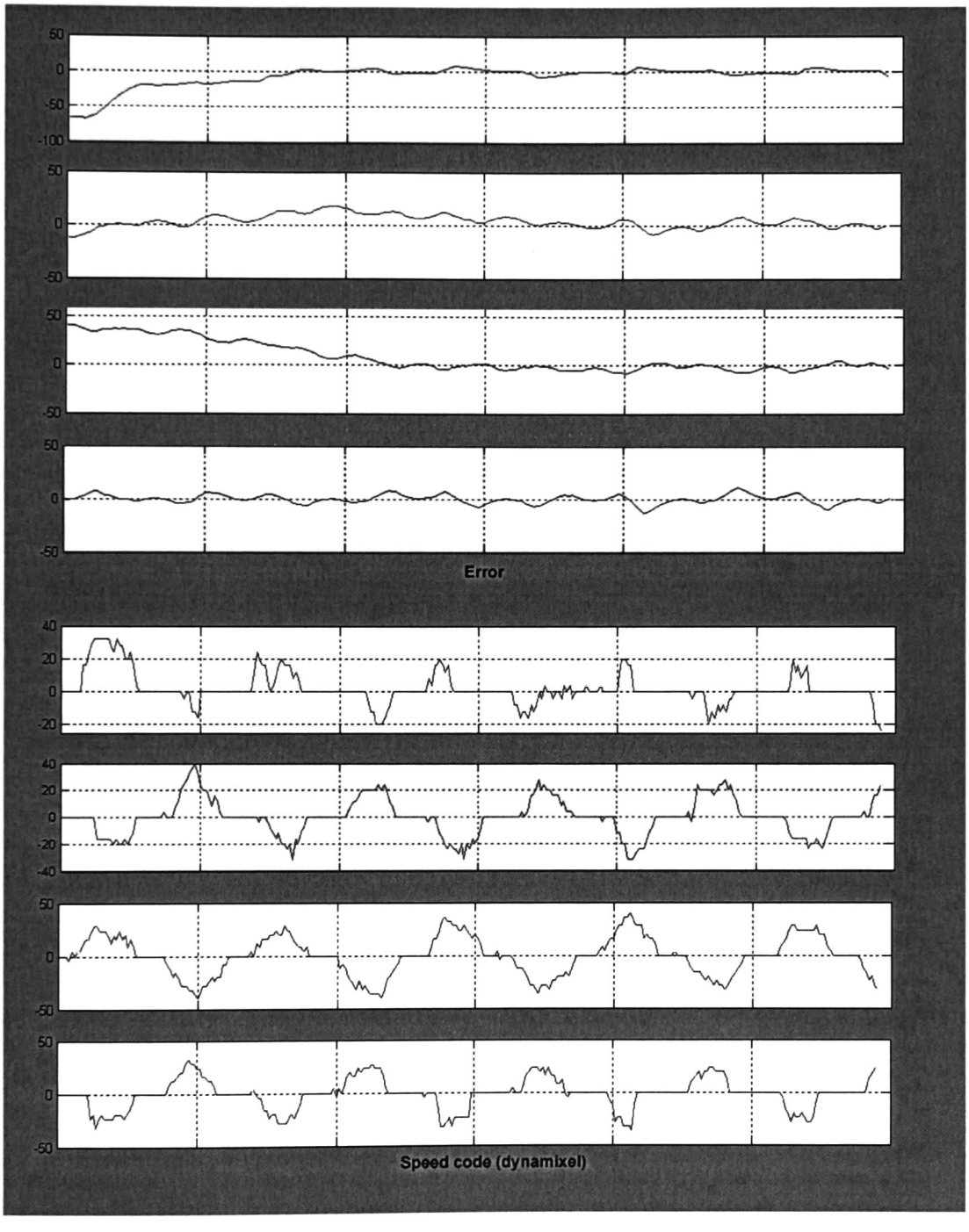


Figure 7.9: Tracking Error and Dynamixel Speed Code

Chapter 8

Conclusions and future work

8.1 Conclusions

Using the forwards kinematics and the inverse kinematics it is possible to determine the position of the end effector from the angles of a robotic model and vice-versa. This is very useful when it is desirable that the end effector of a robot prototype follows a specific trajectory.

It is also possible to make the end effector of the prototype follow a specific trajectory using the state space representation of the model and then apply a control technique, such as neural control or fuzzy control.

The neural control applied to the prototype used a Discrete Recurrent High Order Neural Network (RHONN) which was trained using the Kalman filter. Since the Kalman filter make the error reach to zero, the identification of the system turns to be exact. A great advantage of using a neural network to identify the system and control it, is that it is not necessary to know the exact mathematical model of the system.

The fuzzy control used in the robotic arm has been designed using the local approximation in fuzzy partition spaces technique. Using this technique, the number of control rules is smaller in comparison to the nonlinear sector technique, but it has the disadvantage than it is not robust to parametric variations. Moreover, the stability of the system is not warranted out of the range of operation, although it has the advantages that it does not required the exact mathematical model of the system and it is easy to implement and modify.

The neuro-fuzzy control that has been applied to the prototype using an Adaptive Neuro-Fuzzy Inference Systems (ANFIS) has the advantage that the fuzzy rules can be set empirically but they are improved by the training of the neural network which represents them. This method also has the advantage that the exact mathematical model of the system is not required. Unfortunately, this technique is not robust to parametric variations and the stability is not warranted out of the range of operation.

The physical prototype of the five degrees of freedom robot arm has been improved by means of some modifications. The motors of the rotational joints have been changed,

and the electronic interphase was altered and simplified with respect to the electronic interphase of the previous prototype.

The remote control of this class of robots is of great interest because there are a lot of applications. For these applications, this prototype has been controlled from a remote computer which sends the control signals via Internet using the TCP/IP communications protocol to a different computer which is located in a different location. This second computer is directly connected to the robot.

8.2 Future work

For future work the fuzzy control will be improved in order to make it robust enough to control parametric variations and also to warranty stability out of the range of operation.

In the same way, the neuro-fuzzy control will also be improved in order to make it robust to parametric variations, and also to warranty stability out of the range of operation.

The neuronal control obtained using a RHONN trained using the Kalman filter and the fuzzy control obtained using the local approximation in fuzzy partition spaces technique will be also implemented in real time on the five degrees of freedom redundant robot of CINVESTAV.

More types of control algorithms will be developed for the five degrees of freedom CINVESTAV robot arm prototype. For example: a sliding modes control [7].

Moreover, these new controls designed for the robotic arm (for example: sliding modes control) will be implemented in real time on the prototype of CINVESTAV.

An end effector will be added to the prototype in order to make it able to perform a specific task. For example: a camera, forceps or any other type of end effector.

Bibliography

[1] Craig, John J. (1986). *"Introduction to Robotics: Mechanics and Control"*. Addison Wesley, Massachusetts.

[2] Haykin, S. (1999). *"Neural Networks: a Comprehensive Foundation"* Prentice Hall. New Jersey, USA.

[3] Haykin, S. (2001). *"Kalman Filtering and Neural Networks"* John Wiley & Sons. NY, USA.

[4] Jyh-Shing Roger Jang; Chuen-Tsai Sun (1995). *"Neuro-Fuzzy Modeling and Control"*.

[5] Kailath, T. (1980). *"Linear Systems"*. Prentice Hall.

[6] Kawamoto, S. (1992). *"An Approach to Stability Analysis of Second Order Fuzzy Systems"* First IEEE International Conference on Fuzzy Systems, Vol. 1.

[7] Khalil, H. K. (1996). *"Nonlinear Systems"*. 3th Edition, Prentice Hall, New Jersey, USA

[8] Lee C. S. G.; Gonzalez R. C.; Fu K. S. (1983). *"Tutorial on Robotics"*.

[9] Norgaard, M.; O. Ravn; N. K. Poulsen and L. K. Hansen (1999). *"Neural Networks for Modeling and Control of Dynamic Systems"*. Springer-Verlag. Berlin, Germany.

[10] Passino, K. and Yurkovich, S.(1998). *"Fuzzy Control"* Addison Wesley. California, USA.

[11] Rovithakis, George A.; Christodoulou, Manolis A. (2000). *"Adaptive Control with Recurrent High-Order Neural Networks"* Springer-Verlag, London, Great Britain.

[12] Sanchez Camperos, Edgar Nelson; Alanis García, Alma Yolanda (2006). *"Redes neuronales: conceptos fundamentales y aplicaciones a control automático"*. Prentice Hall. Madrid, Spain.

[13] Stadler, Wolfram (1995). *"Analytical Robotics and Mechatronics"*. McGraw-Hill International Editions.

[14] Tanaka, K., Wang, H. O. (2001). *"Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality approach"*, John Wiley & Sons Inc, New York, USA.

[15] Jurgen Guldner Vadim Utkin and Jingxin Shi (1999). *"Sliding Mode Control in Electromechanical Systems"*. Taylor and Francis, London.

[16] Wang, Li-Xin (1997). *"A course in fuzzy systems and control"*. Prentice Hall. USA.

[17] Xie, Ming (2003). *"Fundamentals of Robotics: Linking Perception to Action"*. Singapure-MIT Alliance & Nanyang Technological University, Singapore.

Appendix A

Dynamic model

The equation of the Dynamic model is as follows:

$$\tau = M(q)\ddot{q} + V(q,\dot{q}) + F_g(q) + F_f(q,\dot{q}) \tag{A.1}$$

Expanding the equation A.1 for the case of the 5DOF robot, we have:

$$\begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \\ \tau_5 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} & m_{15} \\ m_{21} & m_{22} & m_{23} & m_{24} & m_{25} \\ m_{31} & m_{32} & m_{33} & m_{34} & m_{35} \\ m_{41} & m_{42} & m_{43} & m_{44} & m_{45} \\ m_{51} & m_{52} & m_{53} & m_{54} & m_{55} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \\ \ddot{q}_4 \\ \ddot{q}_5 \end{bmatrix} + \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \end{bmatrix} \cdots + \begin{bmatrix} F_{g1} \\ F_{g2} \\ F_{g3} \\ F_{g4} \\ F_{g5} \end{bmatrix} + \begin{bmatrix} f_{f1} & 0 & 0 & 0 & 0 \\ 0 & f_{f2} & 0 & 0 & 0 \\ 0 & 0 & f_{f3} & 0 & 0 \\ 0 & 0 & 0 & f_{f4} & 0 \\ 0 & 0 & 0 & 0 & f_{f5} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \\ \dot{q}_5 \end{bmatrix} \tag{A.2}$$

The expansion of all the elements in A.2 is listed below:

Elements of the inertia matrix $M(q)$:

$m_{11} = m_1 + m_2 + m_3 + m_4 + m_5$

$m_{12} = 0$

$m_{13} = 0$

$m_{14} = 0$

$m_{15} = 0$

$m_{22} = m_5 Lx_5 L\cos(q_3+q_4+q_5) + (m_4 Lx_4 L + m_5 L^2)\cos(q_3+q_4) + m_5 Lx_5 L\cos(q_4+q_5) + ((m_5+m_4)L + Lx_3 m_3)L\cos(q_3) + (m_4 Lx_4 L + m_5 L^2)\cos(q_4) + m_5 Lx_5 L\cos(q_5) + (m_3 + 3m_5 + m_4)L^2 + m_2 Lx_2^2 + m_3 Lx_3^2 + Iz_5 + m_4 Lx_4^2 + Iz_2 + m_5 Lx_5^2 + Iz_4 + Iz^3$

$m_{23} = m_5 Lx_5 L\cos(q_3+q_4+q_5) + (m_5 L^2 + m_4 Lx_4 L)\cos(q_3+q_4) + m_5 Lx_5 L\cos(q_4+q_5) + ((m_5+m_4)L + Lx_3 m_3)L\cos(q_3) + (m_4 Lx_4 L + m_5 L^2)\cos(q_4) + m_5 Lx_5 L\cos(q_5) + (m_4+m_5)L^2 + m_5 Lx_5^2 + m_4 Lx_4^2 + Iz_5 + m_3 Lx_3^2 + Iz_3 + Iz_4$

$m_{24} = m_5 Lx_5 L\cos(q_3+q_4+q_5) + (m_5L^2 + m_4 Lx_4 L)\cos(q_3+q_4) + m_5 Lx_5 L\cos(q_4+q_5) + (m_5L^2 + m_4 Lx_4 L)\cos(q_4) + m_5 Lx_5^2 + m_4 Lx_4^2 + m_5 L^2 + m_5 Lx_5 L\cos(q_5) + Iz_4 + Iz_5$

$m_{25} = m_5 Lx_5 L\cos(q_5) + m_5 Lx_5 L\cos(q_4+q_5) + m_5 Lx_5 L\cos(q_3+q_4+q_5) + Iz_5 + m_5 Lx_5^2$

$m_{33} = m_5 Lx_5 L\cos(q_4+q_5) + (m_4 Lx_4 L + m_5 L^2)\cos(q_4) + m_5 Lx_5 L\cos(q_5) + (m_4 + m_5)L^2 + Iz_3 + m_4 Lx_4^2 + Iz_5 + Iz_4 + m_5 Lx_5^2 + m_3 Lx_3^2$

$m_{34} = m_5 Lx_5 L\cos(q_4+q_5) + (m_5L^2 + m_4 Lx_4 L)\cos(q_4) + m_5 Lx_5 L\cos(q_5) + (Lx_5^2 + L^2)m_5 + Iz_4 + m_4 Lx_4^2 + Iz_5$

$m_{35} = Iz_5 + m_5 Lx_5 L\cos(q_5) + m_5 Lx_5 L\cos(q_4+q_5) + m_5 Lx_5^2$

$m_{44} = m_5 Lx_5 L\cos(q_5) + (Lx_5^2 + L^2)m_5 + Iz_4 + m_4 Lx_4^2 + Iz_5$

$m_{45} = Iz_5 + m_5 Lx_5^2 + m_5 Lx_5 L\cos(q_5)$

$m_{55} = m_5 Lx_5^2 + Iz_5$

Elements of the gravitational forces vector $F_g(q)$:
$F_{g1} = (m_1 + m_2 + m_3 + m_4 + m_5)g$
$F_{g2} = 0$
$F_{g3} = 0$
$F_{g4} = 0$
$F_{g5} = 0$

Elements of the Coriolis forces vector $V(q, \dot{q})$:
$V_1 = 0$

$V_2 = -L(Lx_5 m_5(\dot{q}_3+\dot{q}_4+\dot{q}_5)(\dot{q}_3+\dot{q}_5+2\dot{q}_2+\dot{q}_4)\sin(q_3+q_4+q_5) + (\dot{q}_3+\dot{q}_4)(\dot{q}_3+\dot{q}_4+2\dot{q}_2)(m_4 Lx_4 + Lm_5)\sin(q_3+q_4) + 2(\dot{q}_5/2+\dot{q}_3+\dot{q}_2+\dot{q}_4/2)(\dot{q}_4+\dot{q}_5)Lx_5 m_5\sin(q_4+q_5) + \dot{q}_3(Lm_5 + m_4 L + Lx_3 m_3)(2\dot{q}_2+\dot{q}_3)\sin(q_3) + 2(\dot{q}_3+\dot{q}_2+\dot{q}_4/2)\dot{q}_4(m_4 Lx_4 + Lm_5)\sin(q_4) + 2\dot{q}_5(\dot{q}_5/2+\dot{q}_4+\dot{q}_3+\dot{q}_2)\sin(q_5)Lx_5 m_5)$

$V_3 = -(m_5\sin(q_2)\dot{q}_2 Lx_5(\dot{q}_2+\dot{q}_3+\dot{q}_4+\dot{q}_5)\cos(q_2+q_3+q_4+q_5) - m_5\cos(q_2)\dot{q}_2 Lx_5 (\dot{q}_2+\dot{q}_3+\dot{q}_4+\dot{q}_5)\sin(q_2+q_3+q_4+q_5) - \cos(q_2)\dot{q}_2(\dot{q}_2+\dot{q}_3+\dot{q}_4)(m_4 Lx_4 + Lm_5)\sin(q_2+q_3+q_4) + \sin(q_2)\dot{q}_2(\dot{q}_2+\dot{q}_3+\dot{q}_4)(m_4 Lx_4 + Lm_5)\cos(q_2+q_3+q_4) + m_5 Lx_5\dot{q}_2(\dot{q}_3+\dot{q}_4+\dot{q}_5)\sin(q_3+q_4+q_5) + 2(\dot{q}_5/2+\dot{q}_3+\dot{q}_2+\dot{q}_4/2)(\dot{q}_4+\dot{q}_5)Lx_5 m_5\sin(q_4+q_5) - \cos(q2)\dot{q}_2(Lm5 + m4L + Lx3m3)(\dot{q}_2+\dot{q}_3)\sin(q_2+q_3) + \sin(q_2)\dot{q}_2(Lm_5 + m_4 L + Lx_3 m_3)(\dot{q}_2+\dot{q}_3)\cos(q_2+q_3) + \dot{q}_2(\dot{q}_3+\dot{q}_4)(m_4 Lx_4 + Lm_5)\sin(q_3+q_4) + 2(\dot{q}_3+\dot{q}_2+\dot{q}_4/2)\dot{q}_4(m_4 Lx_4 + Lm_5)\sin(q_4) + 2\dot{q}_5(\dot{q}_5/2+\dot{q}_4+\dot{q}_3+\dot{q}_2)\sin(q_5)Lx_5 m_5 + \dot{q}_3\dot{q}_2\sin(q_3)(Lm_5 + m_4 L + Lx_3 m_3))L$

$V_4 = -L((\dot{q}_2+\dot{q}_3+\dot{q}_4+\dot{q}_5)((\dot{q}_2+\dot{q}_3)\sin(q_2+q_3) + \sin(q_2)\dot{q}_2)Lx_5 m_5\cos(q_2+q_3+q_4+q_5) - ((\dot{q}_2+\dot{q}_3)\cos(q_2+q_3) + \dot{q}_2\cos(q_2))(\dot{q}_2+\dot{q}_3+\dot{q}_4+\dot{q}_5)Lx_5 m_5\sin(q_2+q_3+q_4+q_5) - ((\dot{q}_2+\dot{q}_3)\cos(q_2+q_3) + \dot{q}_2\cos(q_2))(m_4 Lx_4 + Lm_5)(\dot{q}_2+\dot{q}_3+\dot{q}_4)\sin(q_2+q_3+q_4) + (m_4 Lx_4 + Lm_5)((\dot{q}_2+\dot{q}_3)\sin(q_2+q_3) + \sin(q_2)\dot{q}_2)(\dot{q}_2+$

$\dot{q}_3 + \dot{q}_4)\cos(q_2 + q_3 + q_4) + m_5 L x_5 \dot{q}_2(\dot{q}_3 + \dot{q}_4 + \dot{q}_5)\sin(q_3 + q_4 + q_5) + L x_5 m_5(\dot{q}_4 + \dot{q}_5)(\dot{q}_2 + \dot{q}_3)\sin(q_4 + q_5) + \dot{q}_2(\dot{q}_3 + \dot{q}_4)(m_4 L x_4 + L m_5)\sin(q_3 + q_4) + \dot{q}_4(m_4 L x_4 + L m_5)(\dot{q}_2 + \dot{q}_3)\sin(q_4) + 2\dot{q}_5(\dot{q}_5/2 + \dot{q}_4 + \dot{q}_3 + \dot{q}_2)\sin(q_5) L x_5 m_5)$

$V_5 = -L((\dot{q}_2 + \dot{q}_3 + \dot{q}_4 + \dot{q}_5)((\dot{q}_2 + \dot{q}_3 + \dot{q}_4)\sin(q_2 + q_3 + q_4) + (\dot{q}_2 + \dot{q}_3)\sin(q_2 + q_3) + \sin(q_2)\dot{q}_2)\cos(q_2 + q_3 + q_4 + q_5) - (\dot{q}_2 + \dot{q}_3 + \dot{q}_4 + \dot{q}_5)((\dot{q}_2 + \dot{q}_3 + \dot{q}_4)\cos(q_2 + q_3 + q_4) + (\dot{q}_2 + \dot{q}_3)\cos(q_2 + q_3) + \dot{q}_2\cos(q_2))\sin(q_2 + q_3 + q_4 + q_5) + \dot{q}_2(\dot{q}_3 + \dot{q}_4 + \dot{q}_5)\sin(q_3 + q_4 + q_5) + (\dot{q}_2 + \dot{q}_3)(\dot{q}_4 + \dot{q}_5)\sin(q_4 + q_5) + \sin(q_5)\dot{q}_5(\dot{q}_2 + \dot{q}_3 + \dot{q}_4))m_5 L x_5$

Appendix B

Component specifications

The components used in the construction and control of the prototype are: Dynamixel AX-12 motors, a USB connector (for the four rotational joints), a 2424P-PSL Baldor motor, a PWM servo driver, a NI DAQ Card-6042E (for the prismatic joint) and a NI CB-68LP Connector Block for the DAQ Card.

Dynamixel AX-12 motors were used for the rotational joints (Figure B.1a). These motors are not as heavy as the previous ones, so they do not generate as much weight to the prismatic joint.

The Dynamixel AX-12 motors use a serial communication protocol, so the instructions of many motors connected in series can be sent by only one connection. This communication connection is bidirectional. Therefore, it is possible to read and write data in any of the motors (Figure B.2).

A PWM servo driver is used in order to move the 2424P-PSL Baldor motor (Figure B.1b). This driver receives a reference and produces a proportional PWM signal which is necessary in order to move the prismatic joint.

Then, a NI DAQCard-6024E is used to make the communication between the PC and the PWM servo driver (Figure B.1c). Also, a NI CB-68LP Connector Block is used in order to send the control signal to the prismatic joint.

Figure B.4 shows some specifications of the 12A8 PWM servo driver. Figures B.5 and B.6 show some specifications of the Dynamixel AX-12. Figures B.7 and B.8 show some specifications of the NI DAQCard-6024E. Finally, Figure B.3 shows the pin outs of the NI CB-68LP Connector Block.

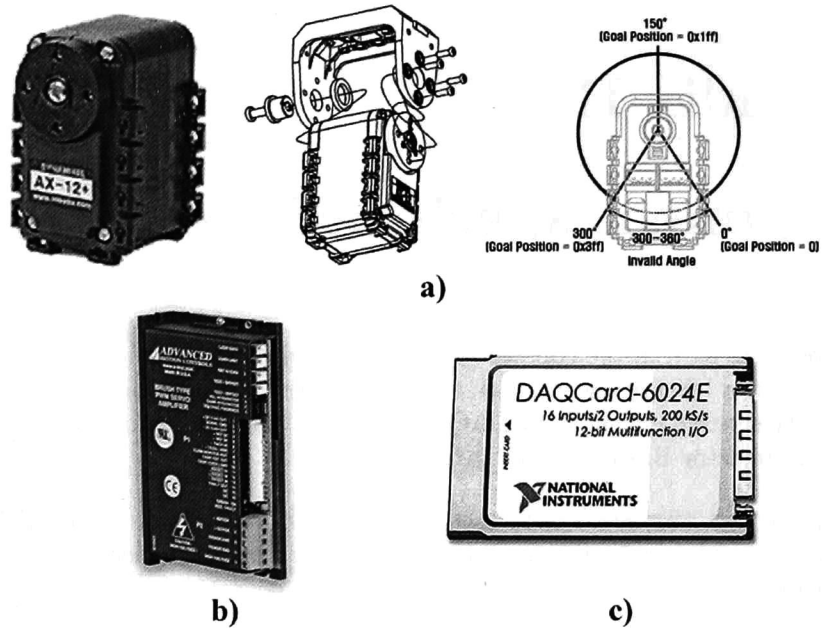


Figure B.1: a) Dynamixel AX-12; b) PWM servo driver; c) DAQCard 6024E

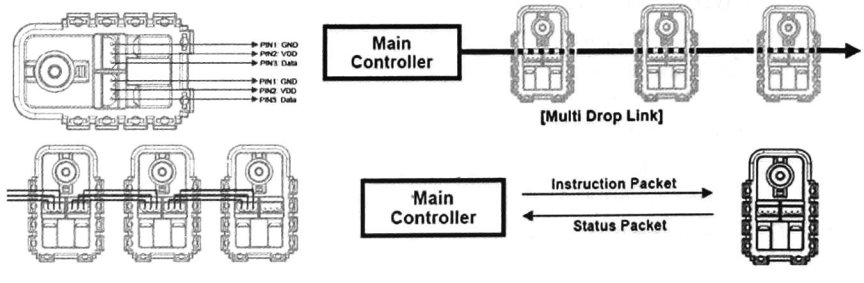
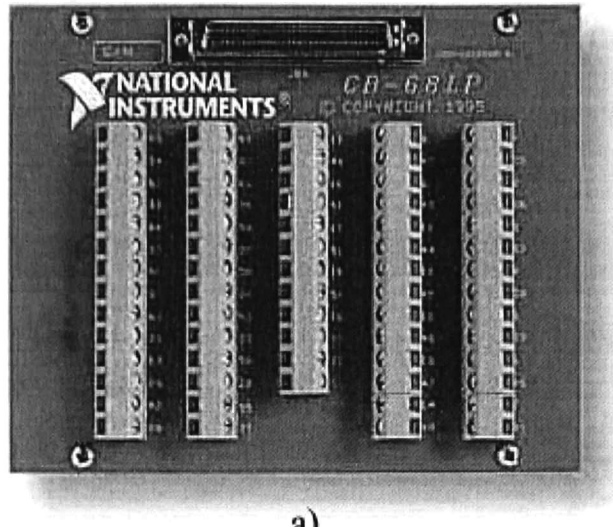


Figure B.2: Dynamixel AX-12 connections



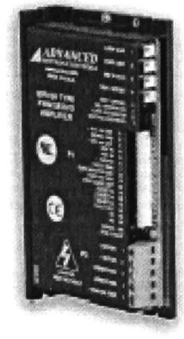
a)

ADVANCED MOTION CONTROLS

Analog Servo Drive **12A8**

Description
The 12A8 PWM servo drive is designed to drive brush type DC motors at a high switching frequency. A single red/green LED indicates operating status. The drive is fully protected against over-voltage, under voltage, over-current, over-heating and short-circuits across motor, ground and power leads. Furthermore, the drive can interface with digital controllers or be used stand-alone and requires only a single unregulated DC power supply. Loop gain, current limit, input gain and offset can be adjusted using 14-turn potentiometers. The offset adjusting potentiometer can also be used as an on-board input signal for testing purposes.

Power Range	
Peak Current	12 A
Continuous Current	6 A
Supply Voltage	20 - 80 VDC



b)

Figure B.3: a) NI CB-68LP Connector Block; b) PWM Driver

SPECIFICATIONS

Power Specifications		
Description	Units	Value
DC Supply Voltage Range	VDC	20 - 80
DC Bus Over Voltage Limit	VDC	86
Maximum Peak Output Current[1]	A	12
Maximum Continuous Output Current	A	6
Maximum Power Dissipation at Continuous Current	W	24
Minimum Load Inductance (Line-To-Line)[2]	µH	200
Switching Frequency	kHz	36
Control Specifications		
Description	Units	Value
Command Sources	-	±10 V Analog
Feedback Supported	-	Tachometer
Modes of Operation	-	Current, IR Compensation, Tachometer Velocity, Voltage
Motors Supported	-	Single Phase (Brushed, Voice Coil, Inductive Load)
Hardware Protection	-	Over Current, Over Temperature, Over Voltage, Short Circuit (Phase-Phase & Phase-Ground)
Mechanical Specifications		
Description	Units	Value
Agency Approvals	-	CE Class A (EMC), CE Class A (LVD), cUL, RoHS, UL
Size (H x W x D)	mm (in)	129.3 x 75.8 x 25.1 (5.1 x 3 x 1)
Weight	g (oz)	280 (9.9)
Heatsink (Base) Temperature Range[3]	°C (°F)	0 - 65 (32 - 149)
Storage Temperature Range	°C (°F)	-40 - 85 (-40 - 185)
Form Factor	-	Stand Alone
P1 Connector	-	16-pin, 2.54 mm spaced, friction lock header
P2 Connector	-	5-port, 5.08 mm spaced, screw terminal

Notes

1. Maximum duration of peak current is ~2 seconds.
2. Lower inductance is acceptable for bus voltages well below maximum. Use external inductance to meet requirements.
3. Additional cooling and/or heatsink may be required to achieve rated performance.

PIN FUNCTIONS

P1 - Signal Connector			
Pin	Name	Description / Notes	I/O
1	+5V 3mA OUT	±5 V @ 3 mA low power supply for customer use. Short circuit protected. Reference ground common with signal ground.	O
2	SIGNAL GND		GND
3	-5V 3mA OUT		O
4	+REF IN	Differential Reference Input (±10 V Operating Range, ±15 V Maximum Input)	I
5	-REF IN		I
6	-TACH IN	Negative Tachometer Input (Maximum ±60 V). Use signal ground for positive input.	I
7	+TACH / GND	Positive Tachometer Input and Signal Ground	GND
8	CURRENT MONITOR	Current Monitor. Analog output signal proportional to the actual current output. Scaling is 2.2 A/V. Measure relative to signal ground.	O
9	CURR REF OUT	Measures the command signal to the internal current-loop. This pin has a maximum output of ±7.25 V when the drive outputs maximum peak current. Measure relative to signal ground.	O
10	CONT CURRENT LIMIT	Can be used to reduce the factory-preset maximum continuous current limit without affecting the peak current limit by attaching an external current limiting resistor between this pin and signal ground. See pin details for resistor values.	I
11	$\overline{\text{INHIBIT IN}}$	TTL level (+5 V) inhibit/enable input. Leave open to enable drive. Pull to ground to inhibit drive. Inhibit turns off all power devices.	I
12	+INHIBIT IN	Positive Direction Inhibit (Does Not Cause A Fault Condition)	I
13	-INHIBIT IN	Negative Direction Inhibit (Does Not Cause A Fault Condition)	I
14	FAULT OUT	TTL level (+5 V) output becomes high when power devices are disabled due to at least one of the following conditions: inhibit, output short circuit, over voltage, over temperature, power-up reset.	O
15	NC	Not Connected (Reserved)	-
16	NC	Not Connected (Reserved)	-

P2 - Power Connector			
Pin	Name	Description / Notes	I/O
1	-MOT	Negative Motor Output	O
2	+MOT	Positive Motor Output	O
3	POWER GND	Power Ground (Common With Signal Ground)	GND
4	POWER GND		GND
5	HIGH VOLTAGE	DC Power Input	I

Pin Details

CONT CURRENT LIMIT (P1-10)

This pin can be used to reduce the continuous current limit without affecting the peak current limit by connecting an external current limiting resistor between this pin and signal ground. See table below.

Current Limit Resistor	15 kΩ	6.6 kΩ	3.4 kΩ	2.1 kΩ	1.2 kΩ	810Ω	500 Ω	250 Ω	0 kΩ
Continuous Current Limit	90%	80%	70%	60%	50%	40%	30%	20%	10%

Note: These values are secondary to the continuous/peak ratio set by the DIP switches.

Figure B.4: 12A8 PWM servo driver

DYNAMIXEL **AX-12** ROBOTIS

Goal Speed Setting

BIT	15~11	10	9	8	7	6	5	4	3	2	1	0
Value	0	Turn Direction	Speed Value									

Turn Direction = 0 : CCW Direction Turn, Load Direction = 1: CW Direction Turn

The following Instructions are available.

Instruction	Function	Value	Number of Parameter
PING	No action. Used for obtaining a Status Packet	0x01	0
READ DATA	Reading values in the Control Table	0x02	2
WRITE DATA	Writing values to the Control Table	0x03	2 ~
REG WRITE	Similar to WRITE_DATA, but stays in standby mode until the ACION instruction is given	0x04	2 ~
ACTION	Triggers the action registered by the REG_WRITE instruction	0x05	0
RESET	Changes the control table values of the Dynamixel actuator to the Factory Default Value settings	0x06	0
SYNC WRITE	Used for controlling many Dynamixel actuators at the same time	0x83	4~

[Control Table Data Range and Length for Writing]

Write Address	Writing Item	Length (bytes)	Min	Max
3(0X03)	ID	1	0	253(0xfd)
4(0X04)	Baud Rate	1	0	254(0xfe)
5(0X05)	Return Delay Time	1	0	254(0xfe)
6(0X06)	CW Angle Limit	2	0	1023(0x3ff)
8(0X08)	CCW Angle Limit	2	0	1023(0x3ff)
11(0X0B)	the Highest Limit Temperature	1	0	150(0x96)
12(0X0C)	the Lowest Limit Voltage	1	50(0x32)	250(0xfa)
13(0X0D)	the Highest Limit Voltage	1	50(0x32)	250(0xfa)
14(0X0E)	Max Torque	2	0	1023(0x3ff)
16(0X10)	Status Return Level	1	0	2
17(0X11)	Alarm LED	1	0	127(0x7f)
18(0X12)	Alarm Shutdown	1	0	127(0x7f)
19(0X13)	(Reserved)	1	0	1
24(0X18)	Torque Enable	1	0	1
25(0X19)	LED	1	0	1
26(0X1A)	CW Compliance Margin	1	0	254(0xfe)
27(0X1B)	CCW Compliance Margin	1	0	254(0xfe)
28(0X1C)	CW Compliance Slope	1	1	254(0xfe)
29(0X1D)	CCW Compliance Slope	1	1	254(0xfe)
30(0X1E)	Goal Position	2	0	1023(0x3ff)
32(0X20)	Moving Speed	2	0	1023(0x3ff)
34(0X22)	Torque Limit	2	0	1023(0x3ff)
44(0X2C)	Registered Instruction	1	0	1
47(0X2F)	Lock	1	1	1
48(0X30)	Punch	2	0	1023(0x3ff)

Figure B.5: Dynamixel AX-12

DYNAMIXEL **AX-12** ROBOTIS

3-4. Control Table

	Address	Item	Access	Initial Value
EEPROM Area	0(0X00)	Model Number(L)	RD	12(0x0C)
	1(0X01)	Model Number(H)	RD	0(0x00)
	2(0X02)	Version of Firmware	RD	?
	3(0X03)	ID	RD,WR	1(0x01)
	4(0X04)	Baud Rate	RD,WR	1(0x01)
	5(0X05)	Return Delay Time	RD,WR	250(0xFA)
	6(0X06)	CW Angle Limit(L)	RD,WR	0(0x00)
	7(0X07)	CW Angle Limit(H)	RD,WR	0(0x00)
	8(0X08)	CCW Angle Limit(L)	RD,WR	255(0xFF)
	9(0X09)	CCW Angle Limit(H)	RD,WR	3(0x03)
	10(0x0A)	(Reserved)	-	0(0x00)
	11(0X0B)	the Highest Limit Temperature	RD,WR	85(0x55)
	12(0X0C)	the Lowest Limit Voltage	RD,WR	60(0X3C)
	13(0X0D)	the Highest Limit Voltage	RD,WR	190(0xBE)
	14(0X0E)	Max Torque(L)	RD,WR	255(0XFF)
	15(0X0F)	Max Torque(H)	RD,WR	3(0x03)
	16(0X10)	Status Return Level	RD,WR	2(0x02)
	17(0X11)	Alarm LED	RD,WR	4(0x04)
	18(0X12)	Alarm Shutdown	RD,WR	4(0x04)
	19(0X13)	(Reserved)	RD,WR	0(0x00)
	20(0X14)	Down Calibration(L)	RD	?
	21(0X15)	Down Calibration(H)	RD	?
	22(0X16)	Up Calibration(L)	RD	?
	23(0X17)	Up Calibration(H)	RD	?
RAM Area	24(0X18)	Torque Enable	RD,WR	0(0x00)
	25(0X19)	LED	RD,WR	0(0x00)
	26(0X1A)	CW Compliance Margin	RD,WR	0(0x00)
	27(0X1B)	CCW Compliance Margin	RD,WR	0(0x00)
	28(0X1C)	CW Compliance Slope	RD,WR	32(0x20)
	29(0X1D)	CCW Compliance Slope	RD,WR	32(0x20)
	30(0X1E)	Goal Position(L)	RD,WR	[Addr36]value
	31(0X1F)	Goal Position(H)	RD,WR	[Addr37]value
	32(0X20)	Moving Speed(L)	RD,WR	0
	33(0X21)	Moving Speed(H)	RD,WR	0
	34(0X22)	Torque Limit(L)	RD,WR	[Addr14] value
	35(0X23)	Torque Limit(H)	RD,WR	[Addr15] value
	36(0X24)	Present Position(L)	RD	?
	37(0X25)	Present Position(H)	RD	?
	38(0X26)	Present Speed(L)	RD	?
	39(0X27)	Present Speed(H)	RD	?
	40(0X28)	Present Load(L)	RD	?
	41(0X29)	Present Load(H)	RD	?
	42(0X2A)	Present Voltage	RD	?
	43(0X2B)	Present Temperature	RD	?
	44(0X2C)	Registered Instruction	RD,WR	0(0x00)
	45(0X2D)	(Reserved)	-	0(0x00)
	46(0x2E)	Moving	RD	0(0x00)
	47(0x2F)	Lock	RD,WR	0(0x00)
	48(0x30)	Punch(L)	RD,WR	32(0x20)
	49(0x31)	Punch(H)	RD,WR	0(0x00)

Figure B.6: Dynamixel AX-12

Table 1. I/O Terminals (Continued)

Terminal Name	Terminal Type and Direction	Impedance Input/ Output	Protection (V) On/Off	Source (mA at V)	Sink (mA at V)	Rise Time (ns)	Bias
+5 V	—	0.1 Ω	Short-circuit to ground	1 A fused	—	—	—
P0.<0..7>	DIO	—	V_{CC} + 0.5	13 at (V_{CC} – 0.4)	24 at 0.4	1.1	50 kΩ pu
P1.<0..7>‡	DIO	—	V_{CC} + 0.5	2.5 at 3.0 min	2.5 at 0.4	5	100 kΩ pu
P2.<0..7>‡	DIO	—	V_{CC} + 0.5	2.5 at 3.0 min	2.5 at 0.4	5	100 kΩ pu
P3.<0..7>‡	DIO	—	V_{CC} + 0.5	2.5 at 3.0 min	2.5 at 0.4	5	100 kΩ pu
AI HOLD COMP or AI HOLD	DO	—	—	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
EXT STROBE*	DO	—	—	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 0/ (AI START TRIG)	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 1/ (AI REF TRIG)	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 2/ (AI CONV CLK)*	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 3/ CTR 1 SOURCE	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 4/CTR 1 GATE	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
CTR 1 OUT	DO	—	—	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 5/ (AO SAMP CLK)*	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} -0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 6/ (AO START TRIG)	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 7/ (AI SAMP CLK)	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 8/ CTR 0 SOURCE	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 9/CTR 0 GATE	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu

Figure B.7: NI DAQCard-6024E

Table 1. I/O Terminals

Terminal Name	Terminal Type and Direction	Impedance Input/ Output	Protection (V) On/Off	Source (mA at V)	Sink (mA at V)	Rise Time (ns)	Bias
AI <0..15>	AI	100 GΩ in parallel with 100 pF	42/35				±200 pA
AI SENSE	AI	100 GΩ in parallel with 100 pF	42/35	—	—	—	±200 pA
AI GND							
AO 0*	AO	0.1 Ω	Short-circuit to ground	5 at 10	5 at −10	10 V/µs	—
AO 1*	AO	0.1 Ω	Short circuit to ground	5 at 10	5 at 10	10 V/µs	
AO GND	—	—	—	—	—	—	—
D GND	—	—	—	—	—	—	—

Terminal Name	Terminal Type and Direction	Impedance Input/ Output	Protection (V) On/Off	Source (mA at V)	Sink (mA at V)	Rise Time (ns)	Bias
PFI 0/P0.0	DIO	—	—	4.5 at (V_{cc} − 0.4)	5 at 0.4	1.5	50 kΩ pu
+REQ 0/1	DO			4.5 at (V_{cc} − 0.4)	5 at 0.4	1.5	50 kΩ pu

* Indicates active low.
† NI 6024/6025E only.
‡ NI 6025E only.
AI = Analog Input DIO = Digital Input/Output pu = pull-up
AO = Analog Output DO = Digital Output
Note: The tolerance on the 50 kΩ pull-up resistors is large. Actual value might range between 17 GΩ and 100 kΩ.

Figure B.8: NI DAQCard-6024E

Appendix C

3D Robot structure

Figures C.1, C.2 and C.3 show different points of view of a 3D structure of the five degrees of freedom CINVESTAV robotic arm prototype. This 3D structures were developed in the SolidWorks software.

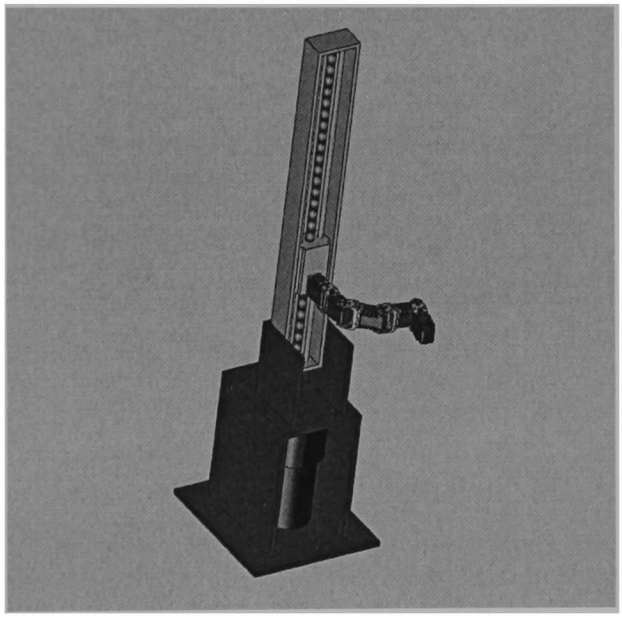


Figure C.1: 3D structure of the 5DOF CINVESTAV prototye

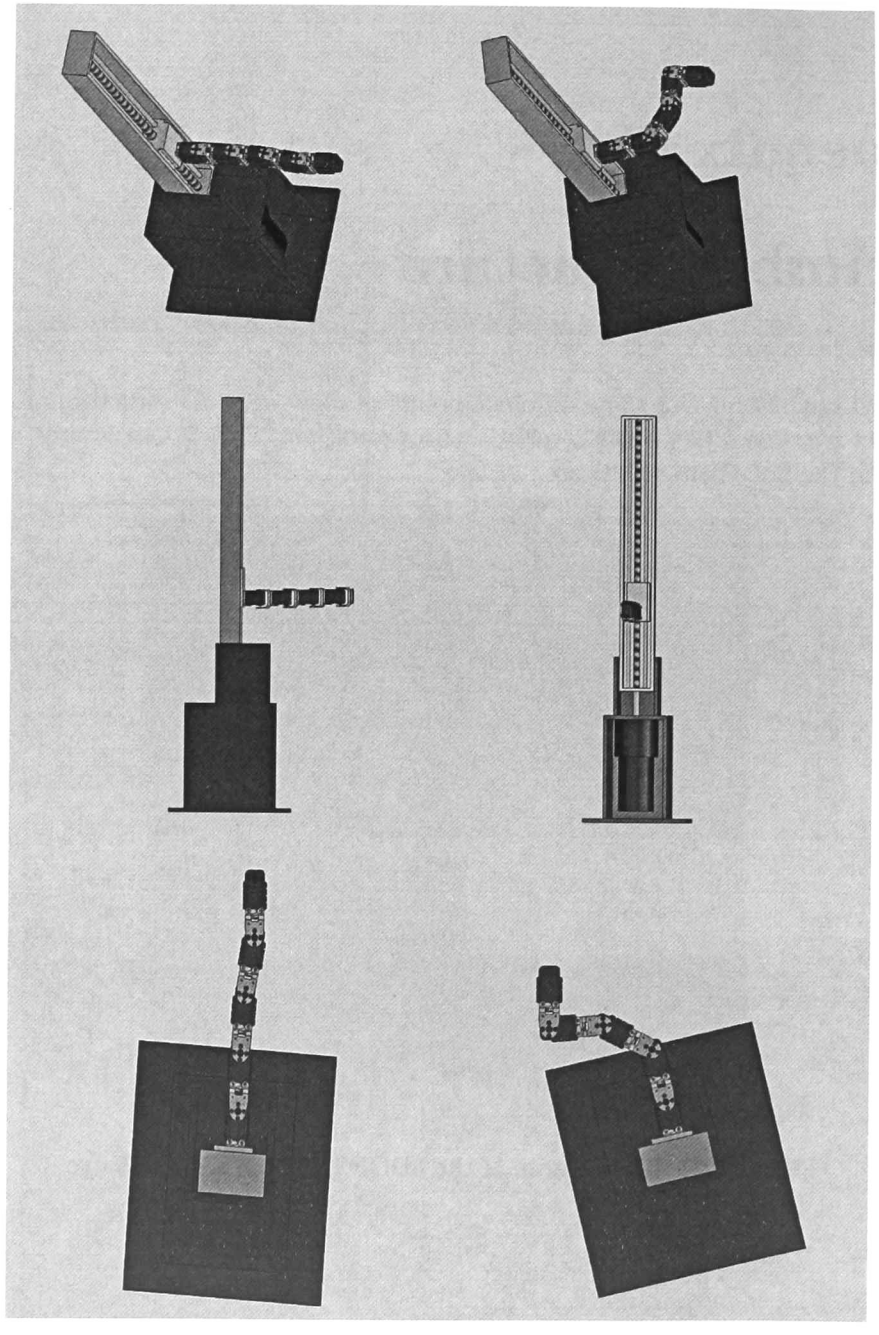


Figure C.2: 3D structure of the 5DOF CINVESTAV prototye

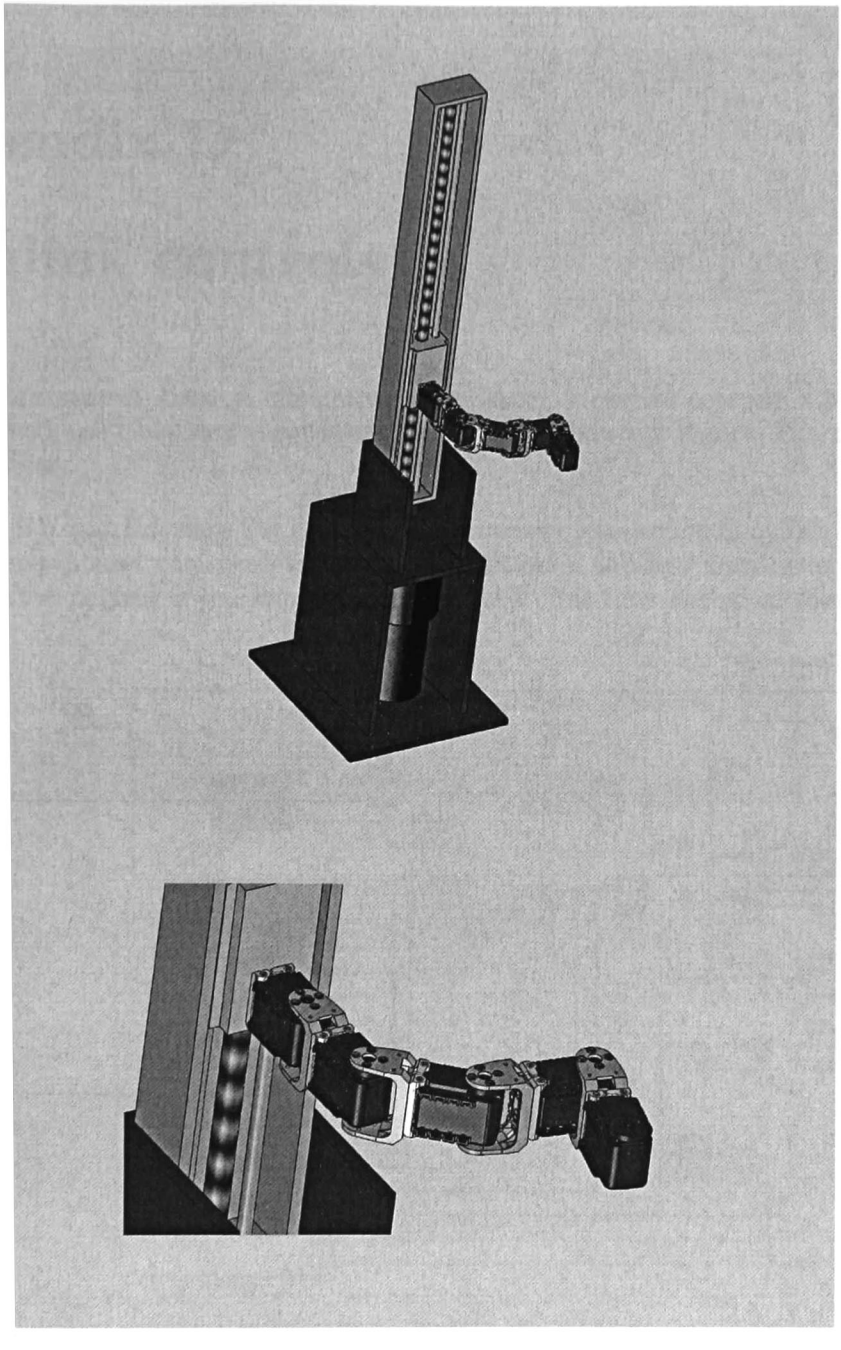


Figure C.3: 3D structure of the 5DOF CINVESTAV prototye

Appendix D

Simulink controls

The Simulink controls used in simulation for Chapter 3 (neural control), Chapter 4 (fuzzy control) and Chapter 5 (neuro-fuzzy control) are shown in Figures D.1, D.2 and D.3 respectively.

Figures D.4 and D.5 show the Dynamixel embedded code used in Simulink in order to set the speed, read the speed and read the position of the four Dynamixel motors used in the five degrees of freedom redundant robot in real time implementations.

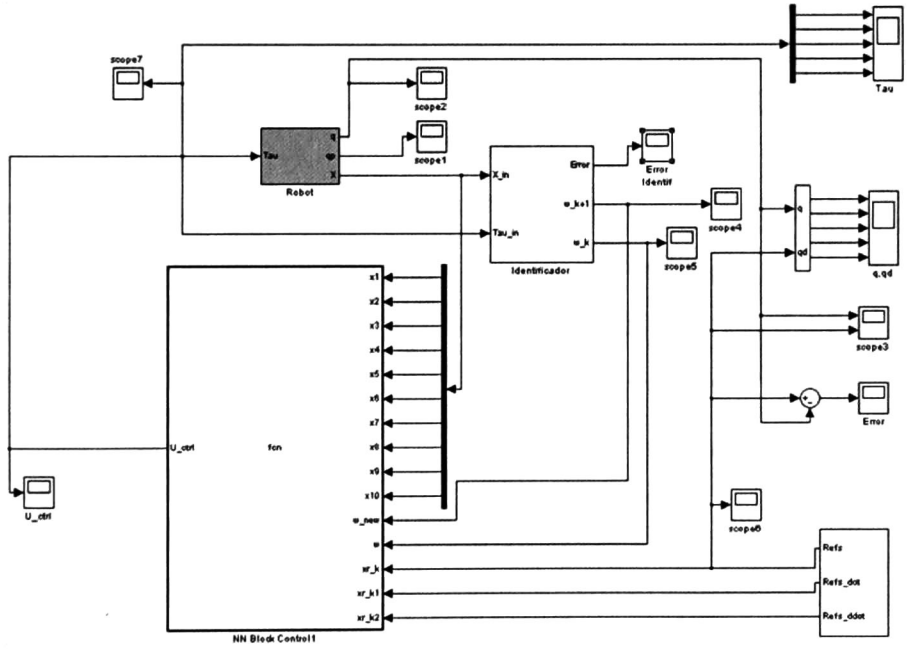


Figure D.1: Neural control

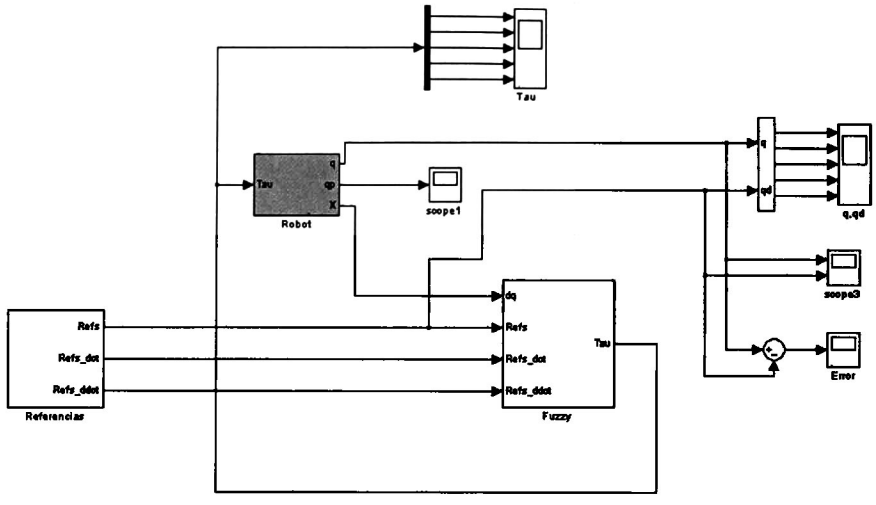


Figure D.2: Fuzzy control

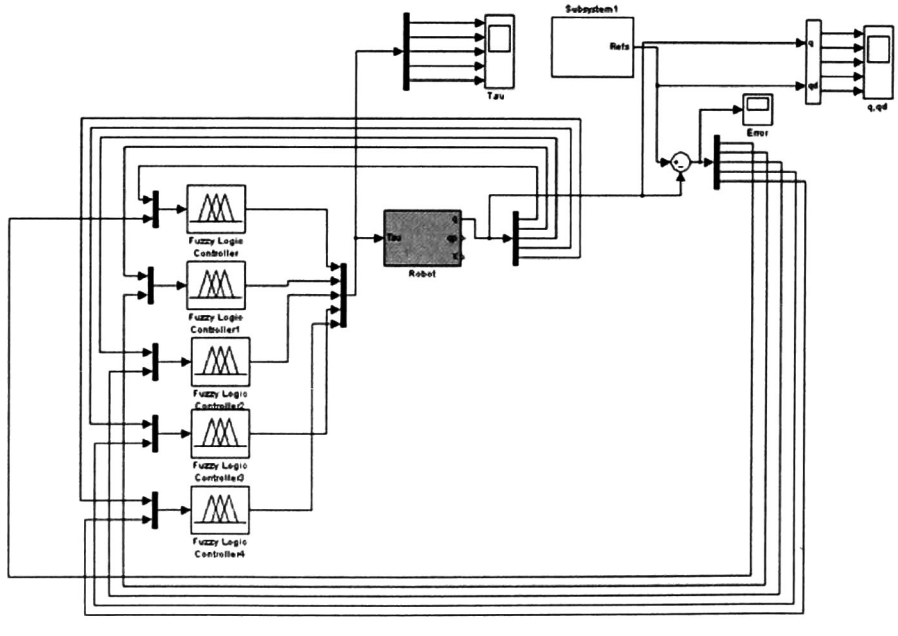


Figure D.3: Neuro-fuzzy control

```
function y = fcn(u)
  This block supports the Embedded MATLAB subset.

% Load dinamixel libraries:
feval('loadlibrary','dynamixel','dynamixel.h');
feval('libfunctions','dynamixel');
feval('calllib','dynamixel','dxl_initialize');

  Set the turn end-less mode:
feval('calllib','dynamixel','dxl_write_word',1,8,0);
feval('calllib','dynamixel','dxl_write_word',2,8,0);
feval('calllib','dynamixel','dxl_write_word',3,8,0);
feval('calllib','dynamixel','dxl_write_word',4,8,0);

  Set goal speed:
GoalSpeed1 = u(1);
GoalSpeed2 = u(2);
GoalSpeed3 = u(3);
GoalSpeed4 = u(4);

if (GoalSpeed1<0)
    GoalSpeed1=abs(GoalSpeed1)+(4*256);
end

if (GoalSpeed2<0)
    GoalSpeed2=abs(GoalSpeed2)+(4*256);
end

if (GoalSpeed3<0)
    GoalSpeed3=abs(GoalSpeed3)+(4*256);
end

if (GoalSpeed4<0)
    GoalSpeed4=abs(GoalSpeed4)+(4*256);
end

feval('calllib','dynamixel','dxl_write_word',1,32,GoalSpeed1);
feval('calllib','dynamixel','dxl_write_word',2,32,GoalSpeed2);
feval('calllib','dynamixel','dxl_write_word',3,32,GoalSpeed3);
feval('calllib','dynamixel','dxl_write_word',4,32,GoalSpeed4);

% Read current position:
PresentPos1 = feval('calllib','dynamixel','dxl_read_word',1,36);
PresentPos2   feval('calllib','dynamixel','dxl_read_word',2,36);
PresentPos3   feval('calllib','dynamixel','dxl_read_word',3,36);
PresentPos4 = feval('calllib','dynamixel','dxl_read_word',4,36);

u(5)=u(5)*PresentPos1;
u(6)=u(6)*PresentPos2;
u(7)=u(7)*PresentPos3;
u(8)=u(8)*PresentPos4;

  Read modified speed:
PresentSpeed1 = feval('calllib','dynamixel','dxl_read_word',1,38);
PresentSpeed2 = feval('calllib','dynamixel','dxl_read_word',2,38);
PresentSpeed3 = feval('calllib','dynamixel','dxl_read_word',3,38);
PresentSpeed4 = feval('calllib','dynamixel','dxl_read_word',4,38);

u(9)=u(9)*PresentSpeed1;
u(10)=u(10)*PresentSpeed2;
u(11)=u(11)*PresentSpeed3;
u(12)=u(12)*PresentSpeed4;

y = u;
```

Figure D.4: Simulink Embedded code

```
function y = fcn(u)
  This block supports the Embedded MATLAB subset.

  Read current speed (from modified speed):
PresentSpeed1=u(1);
PresentSpeed2=u(2);
PresentSpeed3=u(3);
PresentSpeed4=u(4);

if (PresentSpeed1 >= 1024)
    PresentSpeedOut1=-(PresentSpeed1-(4*256));
else
    PresentSpeedOut1=PresentSpeed1;
end

if (PresentSpeed2 >= 1024)
    PresentSpeedOut2=-(PresentSpeed2-(4*256));
else
    PresentSpeedOut2=PresentSpeed2;
end

if (PresentSpeed3 >= 1024)
    PresentSpeedOut3=-(PresentSpeed3-(4*256));
else
    PresentSpeedOut3=PresentSpeed3;
end

if (PresentSpeed4 >= 1024)
    PresentSpeedOut4=-(PresentSpeed4-(4*256));
else
    PresentSpeedOut4=PresentSpeed4;
end

u(1)=PresentSpeedOut1;
u(2)=PresentSpeedOut2;
u(3)=PresentSpeedOut3;
u(4)=PresentSpeedOut4;

y = u;
```

Figure D.5: Simulink Embedded code

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.P.N.
UNIDAD GUADALAJARA

"2011, Año del Turismo en México"

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional aprobó la tesis

Construcción y Control de un Robot Redundante de Cinco Grados de Libertad - Construction and Control of a Five Degrees of Freedom Redundant Robot

del (la) C.

Roberto LOERA DÍAZ

el día 07 de Noviembre de 2011.

Dr. Eduardo José Bayro Corrochano
Investigador CINVESTAV 3D
CINVESTAV Unidad Guadalajara

Dr. Bernardino Castillo Toledo
Investigador CINVESTAV 3C
CINVESTAV Unidad Guadalajara

Dra. Ofelia Begovich Mendoza
Investigador CINVESTAV 3B
CINVESTAV Unidad Guadalajara