People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering

Department of Power and Control

Final Year Project Report Presented in Partial Fulfilment of the Requirements for the Degree of

MASTER

In **Control Engineering**

Option: **Control Engineering**

Title:

Design of an Active Disturbance Rejection Control for the ED-7220C Robot Arm

Presented by:

- **BOUDERHEM Mohamed El Amine**
- **KOUNTAR Amira Nadjah**

Supervisor:

Dr. R. Guernane

Registration Number:……/2017

Abstract

Nowadays, the vast majority of Industrial applications use industrial robot arms which impose the improvement and development of the control techniques that concerns robot manipulators systems. There are many control techniques that can be applied to do the control of manipulators. One of the control strategies is to deal with each joint of the manipulator as a single input/single output (SISO) system, i.e. decoupling the robotic system. This project reports the design of a decoupling control for the ED-7220C 5-axes robot arm position. The biggest challenge in this control problem is the absence of a practical dynamic model to be used in the control due to the lack of information about the physical parameters of this robot arm. Hence model based control techniques will most probably fail in achieving the goal of this project. The solution was to think of a model free control technique. One of the most recent model free control techniques is the extended state observer (ESO) based active disturbance rejection control (ADRC). In this project ESO based ADRC technique has been designed to control the ED-7220C robot manipulator.

Keywords: Robot Arm, ED-7220C, Robot Manipulator, Control, ESO, ADRC.

Dedication

To mom and dad. It is impossible to thank you adequately for everything you have done for me. You the best gift I have ever been given. I love you both.

To my grandfather God's mercy on his soul. Thanks for teaching me the right values and for helping in my education. RIP grandpa.

To my two brothers Walid and Raouf. For supporting me and being with me in my best and my worse.

To my teammate in this project Amira who without her contribution this project could not be realized.

To my teachers from my earliest years at school to now for the effort you made to teach me with a special thanks to Prof. ***HARRICHE*** *a teacher who inspire me to love the field I am studying now.*

To Prof. ***HAWKING*** *who inspired me to love science through his books.*

To all my friends, colleagues, teachers and everyone I got to know at IGEE for helping me improve myself by their ideas, advices and critics with a special thanks to ***Lokmane***, ***Hocine***, ***Khaoula*** *and* ***Younes****. I'm really lucky to have you all in my life.*

Mohamed El Amine BOUDERHEM

Dedication

To the persons that I love most in this world, my beloved parents: Mohamed & Akila. Thank you for supporting me throughout my life, your love & patient and encouraging me to reach my goals.

To my dear brothers: Amir& Adel and sweetest sisters: Ikram & Zina.

To the person who always gives me positive energy whenever I see her: Tata Houria.

To the memory of my grandfather who taught me the meaning of life.

To my lovely teacher who made me realize my real capacities: Mrs. Rouifed.

To my friends who I love. Thank you for all the moments and memories that we shared, specially my Squad who has always believed in me, supported me, inspired me and stood by my side in my hard moments.

To my lovely, sweetie and best friends Khaoula, Hiba & Hadjar.

To my cutest cat: Aqua.

Amira Nadjah KOUNTAR

Acknowledgment

In the name of Allah, the Most Gracious and the Most Merciful Alhamdulillah, all praises to Allah for the strengths and His blessings in completing this dissertation.

We would like to express our most sincere gratitude to our supervisor, Dr. Reda GUERNANE, for the continuous support of our Master project, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped us in all the time of research and writing of this thesis.

Very special thanks to our parents for their love and sympathy. Thanks for being always present for us and helping us to become the persons we are today.

We also wish to thank all the teachers and staff of out institute for their cooperation and help during our five years journey at the IGEE.

Last and not least, special thanks go to all our friends and colleagues for being always helpful with their advice and positive energy with a special thanks to Mr. Tarek BOUAMER a PhD Student at IGEE for devoting his time in helping us, also for his motivation and guidance.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

RRR	Revolute – Revolute – Revolute
RRP	Revolute – Revolute – Prismatic
RPP	Revolute – Prismatic – Prismatic
PPP	Prismatic – Prismatic – Prismatic
FK	Froward Kinematic
IK	Inverse Kinematic
DOF	Degree Of Freedom
Rot	Rotation
Trans	Translation
DH	Denavit Hartenberg
LSPB	Linear Segments with Parabolic Blends
PD	Proportional plus Derivative
PID	Proportional plus Integral plus Derivative
DC	Direct Current
LTI	Linear Time Invariant
ADRC	Active Disturbance Rejection Control
ESO	Extended State Observer
ROM	Range Of Motion

General Introduction

In the last few decades, the use of Robot manipulators has become frequent especially in manufacturing lines, because of their efficiency and ability to execute tasks in a way that human cannot handle. This use of robot arms in industry however, has generated a wide range of control problems that should be solved before the robot manipulator can perform in a right way. In this report, the control problem of the ED-7220C robot arm is addressed. To adjust the behavior and performance of the robot arm and make it convenient for robot applications designed in industry.

The first chapter presents a general introduction to the topic where different types of robot manipulators are presented along with the different applications and use of these robots. In the second chapter, forward and inverse kinematic models are derived, the forward kinematics helps to determine the Cartesian coordinates of the end effector knowing the joint configuration of the robot, while the inverse kinematics does the inverse. That is to determine the joints configuration knowing the Cartesian coordinates of the end effector. The next chapter, introduces how to generate trajectories for the robot arm to follow and mention the different problems encountered when using each technique. Also, it presents the reachable workspace of the robot arm which means the set of points that the robot arm can effectively reach.

The fourth chapter which is the main one. It presents the dynamic model of the ED-7220C robot arm as a function of its unknown physical parameters and shows its nonlinear and time varying nature. Then, theoretical background of the extended state observer based, Active disturbance rejection control is introduced with its application on the robotic system of this project along with the design of a simple PD controller to show how inefficient is the use of classical control for such a system. At the end of the chapter, an extended state observer with ADRC control is designed for the robotic system.

Chapter 1

Overview of Robot Manipulators

1 Introduction

Robotics is a relatively young field of modern technology that crosses traditional engineering boundaries. Understanding the complexity of robots and their applications requires knowledge of electrical engineering, mechanical engineering, systems and industrial engineering, computer science, economics, and mathematics. New disciplines of engineering, such as manufacturing engineering, applications engineering, and knowledge engineering have emerged to deal with the complexity of the field of robotics and factory automation [1].

The word robot was introduced in 1920 be a Czech playwright which mean work. Basically, a robot is an autonomous device that use computer such as teleoperators, underwater vehicles, autonomous land rovers, etc [1].

Figure1.1. ED7220C Robotic Arm [2].

Figure1.1. shows a typical robot that is essentially a mechanical arm operating under computer control. Such devices, though far from the robots of science fiction, are nevertheless extremely complex electro-mechanical systems whose analytical description requires advanced methods, presenting many challenging and interesting research problems.

A robot manipulator is seen as more than just a series of mechanical linkages. Arm mechanism is only one element in a comprehensive automated system, is shown in Figure1.2. which consists of an arm, external power source, end-of-arm tooling, external

and internal sensors, computer interface, and control computer. Even the programmed software is considered as an integral part of the overall system, since the manner in which the robot is programmed and controlled can have a major impact on its performance and subsequent range of applications.

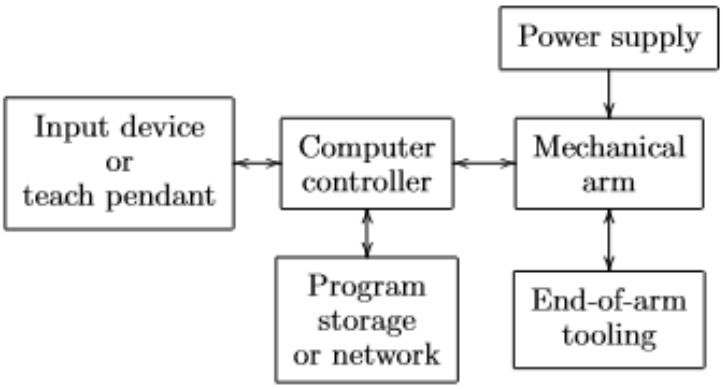


Figure1.2. Components of Robotic System [1].

Although there are many possible ways that use prismatic and revolute joints to construct kinematic chains, in practice only a few of these are commonly used. Here we briefly describe several arrangements that are most typical.

2 Common Kinematic Arrangements of Manipulators

2.1 Articulated Manipulator (RRR)

Figure1.3. shows the (ABB IRB1400) articulated manipulator which called a revolute manipulator [1]. The (RRR) means the type of joint is (Revolute – Revolute – Revolute) and (P) means the type of joint is (Prismatic).

Figure1.3. The ABB IRB1400 Robot [1].

A common revolute joint design is the parallelogram linkage such as the motorman SK16. in both of these arrangements joint axis z_2 is parallel to z_1 and both z_1

and z_2 are perpendicular to z_0. This kind of manipulator is known as an elbow manipulator. The structure and terminology associated with the elbow manipulator are shown in Figure1.4.

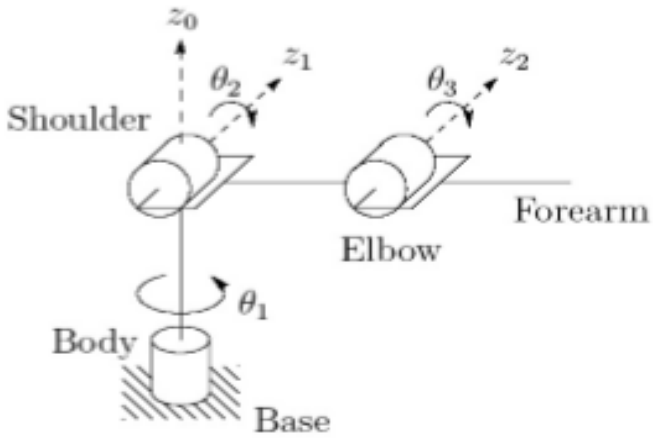


Figure1.4. Structure of The Elbow Manipulator [1].

The revolute manipulator provides relatively large freedom of movement in a compact space; the elbow manipulator has several advantages that make it an attractive and popular design. The parallelogram linkage manipulator is that the actuator for joint 3 is located on link 1. Since the weight of the motor is born by link 1, links 2 and 3 can be made more lightweight and the motors themselves can be less powerful.

2.2 Spherical Manipulator (RRP)

The spherical manipulator can be obtained by replacing the third or elbow joint in the revolute manipulator by a prismatic joint, as shown in Figure1.5. The term spherical manipulator derives from the fact that the spherical coordinates defining the position of the end-effector with respect to a frame whose origin lies at the intersection of the three z-axes are the same as the first three joint variables [1].

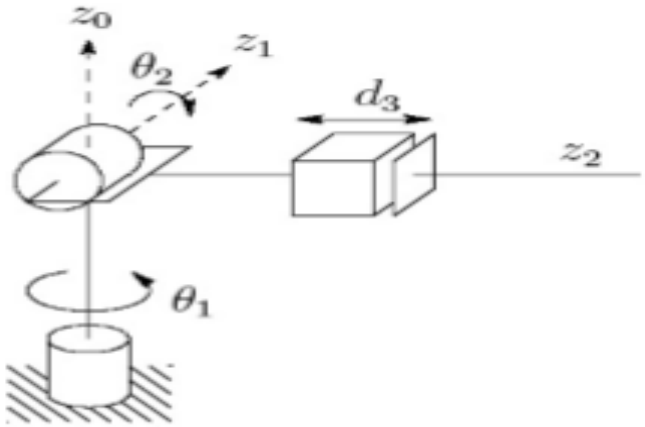


Figure1.5. The Spherical Manipulator [1].

2.2.1 SCARA Manipulator (RRP)

The SCARA arm (for Selective Compliant Articulated Robot for Assembly) shown in Figure1.6. is a popular manipulator [1]. It is a spherical manipulator. The SCARA has an RRP structure; it is quite different from the spherical manipulator in both appearance and in its range of applications. The SCARA has z_0, z_1, and z_2 mutually parallel.

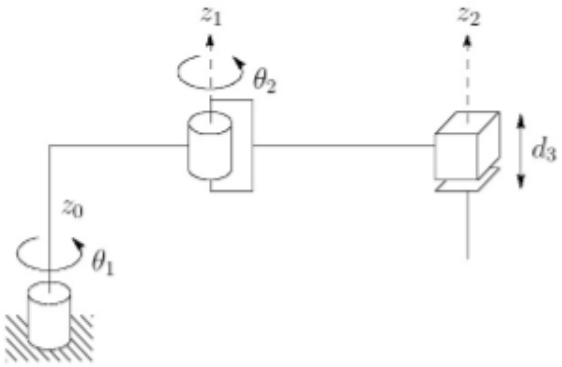


Figure1.6. The SCARA (Selective Compliant Articulated Robot for Assembly) [1].

2.3 Cylindrical Manipulator (RPP)

The cylindrical manipulator is shown in Figure1.7. The first joint is revolute and produces a rotation about the base, while the second and third joints are prismatic. As the name suggests, the joint variables are the cylindrical coordinates of the end-effector with respect to the base.

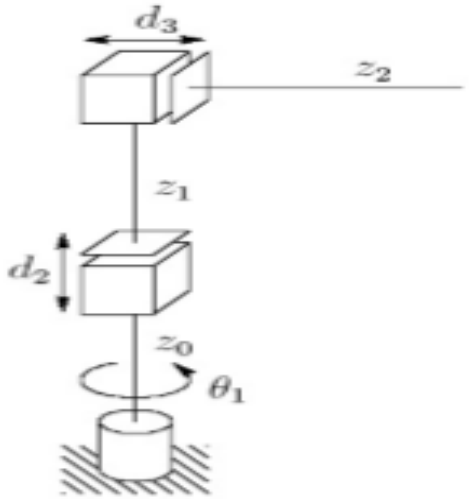


Figure1.7. The Cylindrical Manipulator [1].

2.4 Cartesian Manipulator (PPP)

A manipulator whose first three joints are prismatic is known as a Cartesian manipulator, shown in Figure1.8. For the Cartesian manipulator, the joint variables are the Cartesian coordinates of the end-effector with respect to the base [1].

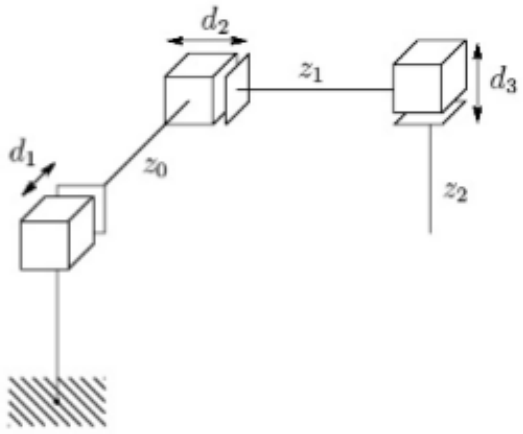


Figure1.8. The Cartesian Manipulator [1].

2.5 Parallel Manipulator

A parallel manipulator has two or more independent kinematic chains connecting the base to the end-effector. Figure1.9. shows the ABB IRB 940 Tricept robot [1]. The kinematic description of parallel robots is fundamentally different from that of serial link robots; therefore, requires different methods of analysis.

Figure1.9. The ABB IRB940 Tricept Parallel Robot [1].

3 Robot Applications

Robots are used in various industrial and related fields, below are some of the main applications of robotics.

3.1 Robotic handling operations

Material handling is the most popular application with 38% of operational stock of industrial robots worldwide. This includes machine tending, palatalizing and various

operations for metal machining and plastic molding. With the introduction of collaborative robots in the last few years, this part of the market is always increasing [3].

3.2 Robotic Welding

This segment mostly includes spot welding and arc welding which is mainly used by the automotive industry. Spot welding is still more popular than arc welding but not for long; as arc welding is becoming very popular in the metal industry. More small workshops are beginning to introduce welding robots into their production. With the price of robots are going down and the various tools now available on the market, it is now easier to automate a welding process [3].

3.3 Robotic Assembly

Assembly operations include: fixing, press-fitting, inserting, disassembling, etc. This category of robotic applications seems to have decreased over the last few years, even while other robotic applications have increased. The reason why the applications are diversified is because of the introduction of different technologies such as force torque sensors and tactile sensors that gives more sensations to the robot [3].

3.4 Robotic Dispensing

Here we are talking about painting, gluing, applying adhesive sealing, spraying, etc. Only 4% of the operational robots are doing dispensing. The smoothest of robot makes a repeatable and accurate process [3].

3.5 Robotic Processing

Processing is not a big segment of industrial robots (only 2%) and this is probably because a lot of automated machines are available on the market to do specifically these applications. The main application areas are mechanical, laser and water jet cutting [3].

4 Conclusion

This chapter covers the different common kinematic arrangements of Robot manipulators where we have seen the Articulated Manipulator (RRR) which is the case of our Robot Arm.

Chapter 2

Robot Kinematics

1 Introduction

Kinematics studies the motion of bodies without consideration of the forces or moments that cause the motion. Robot kinematics refers the analytical study of the motion of a robot manipulator [4]. In other word, the kinematics of a robot manipulator describes the relationship between the motion (position, velocity, acceleration, and higher derivatives of the position variables) of the joints of the manipulator and the resulting motion of the rigid bodies [5], which form the robot, without consideration of the forces and torques causing the motion.

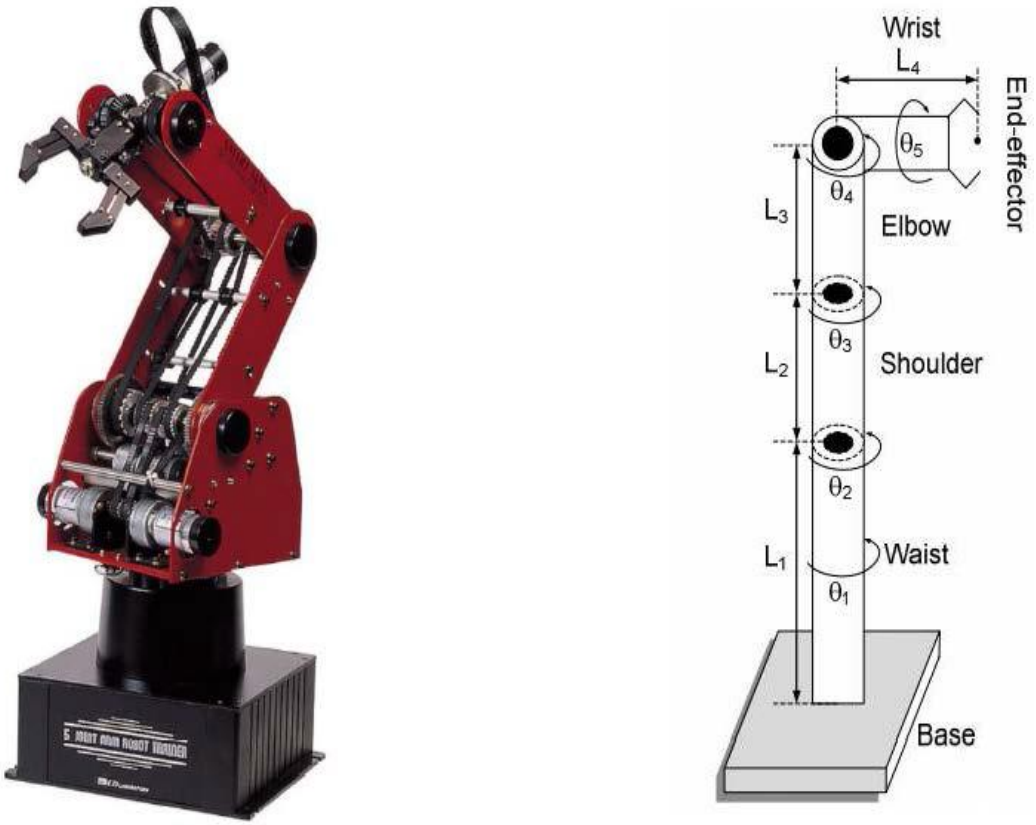


Figure2.1. ED-7220C and its kinematics model [2].

2 Forward and Inverse Kinematics

The kinematics solutions of any robot manipulator are divided into two solutions, the first one is the solution of Forward kinematics, and the second one is the inverse kinematics solution. Forward kinematics will determine where the robot's manipulator hand will be if all joints are known. Where the inverse kinematics will calculate what each joint variable must be if the desired position and orientation of end-effector is determined. Hence, Forward Kinematics (FK) is defined as transformation from joint space to Cartesian space where as Inverse Kinematics (IK) is defined as transformation from Cartesian space to joint space [6].

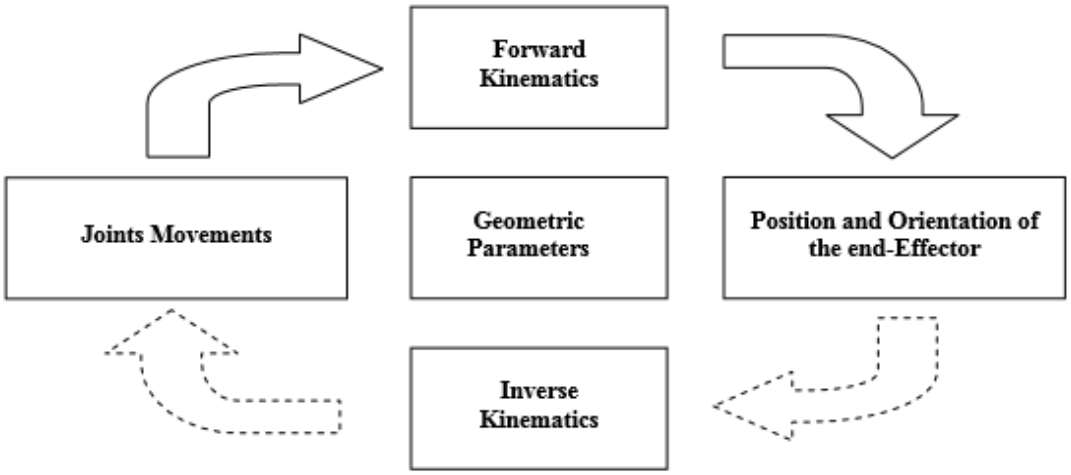


Figure2.2. Kinematics block diagram [6].

2.1 Forward Kinematics

Forward kinematics (FK) can be stated as follows [7]: Given the joint variables (angles θ_i for revolute joints and link-offset d_i for prismatic joints) of the robot, determine the position (x, y, z) and orientation R_d of the end effector with respect to a specified reference frame.

$$F(\theta_1, \theta_2, \dots, \theta_n) = [x, y, z, R_d] \tag{2.1}$$

2.1.1 Kinematics Chain

Considering that, each joint has a single degree-of-freedom (DOF), the action the action of each joint can be described by a single real number: the angle of rotation (θ_1, $\theta_2 \dots \theta_n$) in the case of a revolute joint or the displacement (d_1, $d_2 \dots d_n$) in the case of a prismatic joint [8].

For a robot manipulator that has n joints will have n+1 links. Starting from the base, the joints are numbered from 1 to n, and the links are numbered from 0 to n. thus, the links *i-1* and *i* are connected by the joint *i*. With the i^{th} joint, we associate a joint variable, denoted by q_i. In the case of a revolute joint, q_i is the angle of rotation, and it is the joint displacement in the case of a prismatic joint [8].

The homogeneous matrix A_i expresses the coordinates (position and orientation) of a point from frame *i* to frame *i-1*. The matrix A_i varies as the configuration of the robot is changed. However, the assumption that all joints are either revolute or prismatic means that A_i is a function of only a single joint variable, namely q_i. In other words, [8]:

$$A_i = A_i(q_i) \tag{2.2}$$

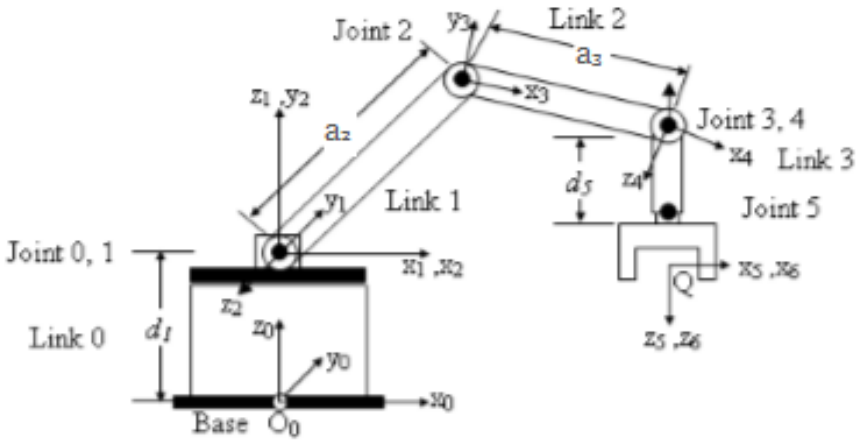


Figure2.3. ED-7220C Robot Arm Frame Assignment.

The homogeneous matrix that transform the coordinates from frame *j* to frame *i* is called the *transformation matrix* denoted by T_j^i(j>i). Denoting the position and orientation of the end-effector with respect to the inertial or the base frame by a three-dimensional vector O_n^0 and a 3x3 rotation matrix R_n^0, respectively, we define the homogenous matrix [8]:

$$T_n^0 = \begin{bmatrix} R_n^0 & O_n^0 \\ 0 & 1 \end{bmatrix} \tag{2.3}$$

Then the position and orientation of the end-effector in the inertial frame are given by:

$$T_n^0 = A_1(q_1)\, A_2(q_2) \ldots A_n(q_n) \tag{2.4}$$

Each homogeneous transformation A_i is of the form [8]:

$$A_i = \begin{bmatrix} R_i^{i-1} & O_i^{i-1} \\ 0 & 1 \end{bmatrix} \tag{2.5}$$

Hence [8]:

$$T_j^i = A_{i+1} \ldots A_j = \begin{bmatrix} R_j^i & O_j^i \\ 0 & 1 \end{bmatrix} \tag{2.6}$$

The matrix R_j^i expresses the orientation of frame *j* relative to frame *i* and is given by the rotational parts of the A-matrices as [8]:

$$R_j^i = R_{i+1}^i \ldots R_j^{j-1} \tag{2.7}$$

The coordinate vectors O_j^i are given recursively by the formula [8]:

$$O_j^i = O_{j-1}^i + R_{j-1}^i d_j^{j-1} \tag{2.8}$$

2.1.2 Denavit Hartenberg Convention

Denavit Hartenberg or DH convention is a commonly used convention for selecting frames of reference in robotic applications. In this convention [8], each homogenous transformation A_i is represented as a product of "four" basic transformations:

$$A_i = \text{Rot}(z, \theta_i)\ \text{Trans}(z, d_i)\ \text{Trans}(x, a_i)\ \text{Rot}(x, \alpha_i) \tag{2.9}$$

Where the notation Rot (x, α_i) stands for rotation about x_i axis by α_i, Trans (x, a_i) is translation along x_i axis by a distance a_i, Rot (z, θ_i) stands for rotation about z_i axis by θ_i, and Trans (z, d_i) is the translation along z_i axis by a distance d_i, (designating $c_i\ as\ cos\theta_i\ and\ s_i\ as\ sin\theta_i$ etc).

$$A_i = \begin{bmatrix} C_{\theta i} & -S_{\theta i} & 0 & 0 \\ S_{\theta i} & C_{\theta i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C_{\alpha i} & -S_{\alpha i} & 0 \\ 0 & S_{\alpha i} & C_{\alpha i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_i = \begin{bmatrix} C_{\theta i} & -S_{\theta i}C_{\alpha i} & S_{\theta i}S_{\alpha i} & a_i C_{\theta i} \\ S_{\theta i} & C_{\theta i}C_{\alpha i} & -C_{\theta i} & a_i S_{\theta i} \\ 0 & S_{\alpha i} & C_{\alpha i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.10}$$

The four quantities θ_i, a_i, d_i, α_i are the parameters of link *i* and joint *i*.

The various parameters in previous equation are given the following names:

a_i (Link length) is the distance between z_{i-1} to z_i measured along x_i.

α_i (Link twist) is the angle from z_{i-1} to z_i measured along x_i.

d_i (Link offset) is the distance between x_{i-1} to x_i measured along z_{i-1}.

θ_i (Joint angle) is the angle from x_{i-1} to x_i measured along z_{i-1}.

In the usual case of a revolute joint θ_i, is called the joint variable, the other three quantities are the fixed link parameters.

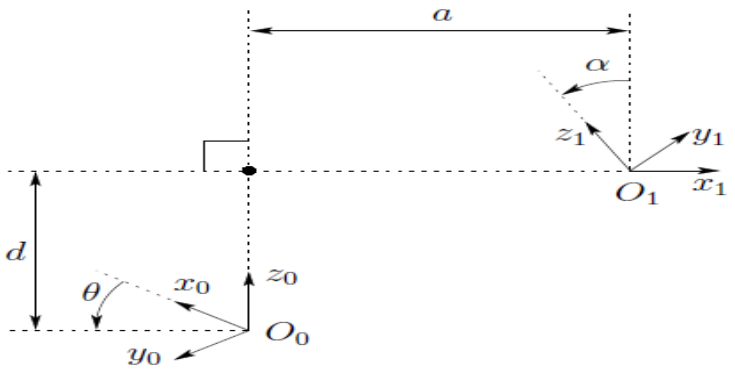


Figure2.4. DH frame assignment [8].

2.1.3 ED-7220C DH Parameters [2]

Many methods can be used in the direct kinematics calculation. The Denavit Hartenberg (DH) analysis is one of the most used, in this method the direct kinematics is determined from some parameters that have to be defined, depending on each mechanism [9]. However, it was chosen to use the homogeneous transformation matrix. In this, analysis, once it is easily defined one coordinate transformation between two frames, where the position and orientation are fixed one with respect to the other it is possible to work with elementary homogeneous transformation operations [6]. DH parameters for ED-7220C defined for the assigned frames in Table2.1.

Joint (i)	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	d_1	θ_1
2	-90°	0	0	θ_2
3	0	a_2	0	θ_3
4	0	a_3	0	θ_4
5	-90°	0	d_5	θ_5
6	0	0	0	Θ_6

Table2.1. DH parameter for ED-7220C robot arm.

Joint	Waist	Shoulder	Elbow	Wrist
Symbol	d_1	a_2	a3	d_5
Link length (mm)	385	220	220	155

Table2.2. The link lengths.

Using the parameter of Table2.1. and Table2.2. the transformation matrices A_1 to A_5 can be obtained as shown below. For example, A_1 shows the transformation between frames 0 and 1.

$$A_1 = \begin{bmatrix} C_{\theta 1} & -S_{\theta 1} & 0 & 0 \\ S_{\theta 1} & C_{\theta 2} & -C_{\theta 1} & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

$$A_2 = \begin{bmatrix} C_{\theta 2} & 0 & -S_{\theta 2} & a_2 C_{\theta 2} \\ S_{\theta 2} & C_{\theta 2} & -C_{\theta 2} & a_2 S_{\theta 2} \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

$$A_3 = \begin{bmatrix} C_{\theta 3} & -S_{\theta 3} & 0 & a_3 C_{\theta 3} \\ S_{\theta 3} & C_{\theta 3} & -C_{\theta 3} & a_3 S_{\theta 3} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

$$A_4 = \begin{bmatrix} C_{\theta 4} & -S_{\theta 4} & 0 & 0 \\ S_{\theta 4} & C_{\theta 4} & -C_{\theta 4} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

$$A_5 = \begin{bmatrix} C_{\theta 5} & 0 & -S_{\theta 5} & 0 \\ S_{\theta 5} & 0 & -C_{\theta 5} & 0 \\ 0 & -1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.15}$$

Using the above values of the transformation matrices; the link transformations can be concatenated (multiplied together) to find the single transformation that relates frame (5) to frame (0):

$$T_5^0 = A_1 A_2 A_3 A_4 A_5 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.16}$$

The transformation given by equation (2.16) is a function of all 5 joint variables. From the robots' joint position, the Cartesian position and orientation of the last link may be computed using above equation (2.16).

The first three columns in the matrices represent the orientation of the end effectors, whereas the last column represents the position of the end effectors. The orientation and position of the end effectors can be calculated in terms of joint angles using:

$$\begin{cases} n_x = c_1 s_{234} c_5 + s_1 s_5 \\ n_y = -s_1 c_{234} c_5 - c_1 s_5 \\ n_z = c_{234} c_5 \end{cases} \tag{2.17}$$

$$\begin{cases} o_x = -c_1 s_{234} s_5 + s_1 c_5 \\ o_y = s_1 c_{234} c_5 + c_1 c_5 \\ o_z = -c_{234} s_5 \end{cases} \tag{2.18}$$

$$\begin{cases} a_x = c_1 c_{234} \\ a_y = s_1 c_{234} \\ a_z = -s_{234} \end{cases} \tag{2.19}$$

$$\begin{cases} p_x = c_1(c_{234}d_5 + a_3c_{23} + a_2c_2) \\ p_y = s_1(c_{234}d_5 + a_3c_{23} + a_2c_2) \\ p_z = -s_{234}d_5 + a_3s_{23} + a_2s_2 + d_1 \end{cases} \quad (2.20)$$

2.2 Inverse Kinematics

Inverse kinematics (IK) can be stated as follows: Given the position and orientation of the end-effector frame relative to a specified reference frame, determine the joint variables in Cartesian space [7]. That means if the final link configuration is known, what is the possible configuration of the robot manipulator to move the end-effectors of the robot arm to desired position and orientation in space [10]. Inverse kinematic problem may express mathematically as follows:

$$F(x, y, z, R_d) = [\theta_1, \theta_2, \dots, \theta_n] \quad (2.21)$$

The solution of inverse kinematic is more complex than direct kinematics and there is not any global analytical solution method. Each manipulator needs a particular method considering the system structure and restrictions. However, there are two useful solutions approaches; geometric and algebraic used for deriving the inverse kinematics solution.

2.2.1 Geometric Approach

Using inverse kinematics Cartesian mode, by specifying the desired target position of the gripper in Cartesian space as (x_d, y_d, z_d) where z_d is the height, and the angle of the gripper relative to ground, ψ (Figure2.6.), is held constant. This constant ψ allows users to move objects without changing the object's orientation. In addition, by either keeping ψ fixed in position mode or keeping the wrist fixed relative to the rest of the arm, the inverse kinematic equations can be solved in closed form as we now show for the case of a fixed ψ [6].

The lengths d_1, a_2, a_3 and d_5 correspond to the base height, Shoulder length, Elbow length and gripper length, respectively are constant. The angles θ_1, θ_2, θ_3, θ_4 and θ_5 correspond to Base rotation, Shoulder, Elbow, wrist, and end effector, respectively. These angles are updated as the specified position in space changes. We solve for the joint angles of the arm, $\theta_{1:4}$ given desired position (x_d, y_d, z_d) and ψ which are inserted by the user [6].

From Figure2.5. we clearly see that θ_1= atan2(y_d, x_d) and the specified radial distance from the base d are related to x_d and y_d by:

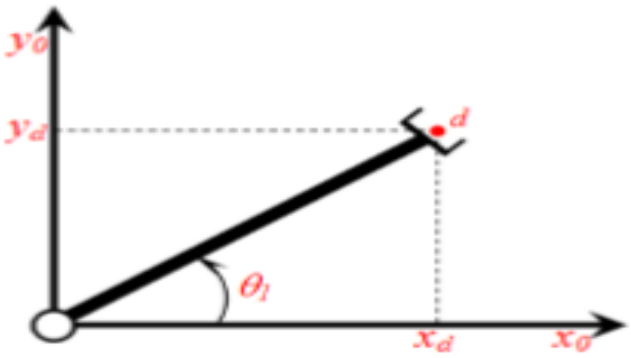


Figure2.5. Top view of the ED-7220C [6].

$$\begin{cases} d = \sqrt{x_d^2 + y_d^2} \\ x_d = d\cos\theta_1 \\ y_d = d\sin\theta_1 \end{cases} \tag{2.22}$$

From the planar view in Figure2.6., we find a relationship between joint angles θ_2, θ_3 and θ_4 and ψ the angle of the gripper relative to the ground as follows:

$$\psi = \theta_2 + \theta_3 + \theta_4 \tag{2.23}$$

Hence, we can obtain radial distance and height as follows:

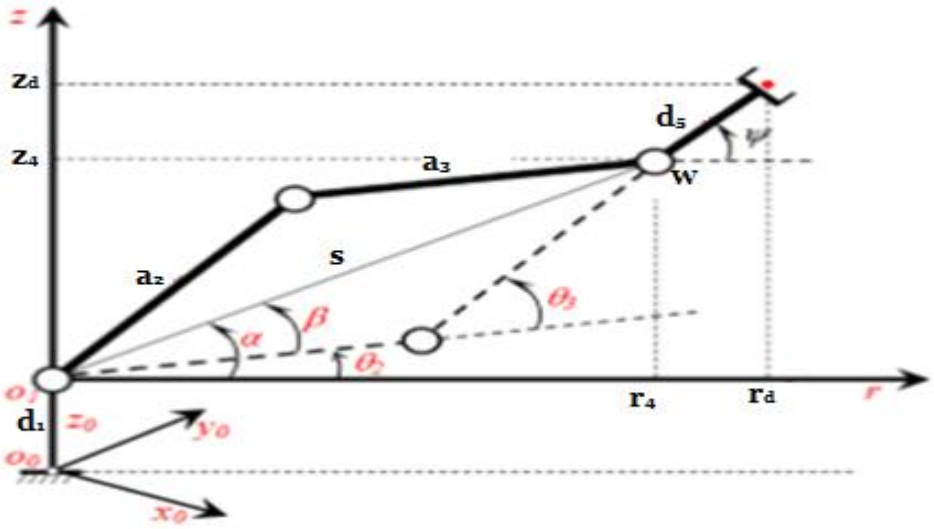


Figure2.6. Side view of ED-7220C [6].

$$\begin{cases} r_4 = r_d - d_5\cos\psi \\ z_4 = z_d - d_5\sin\psi \end{cases} \tag{2.24}$$

To obtain the joint angles θ2, θ3 and θ4, we need solve the intermediate values α, s and β:

$$\begin{cases} \alpha = atan2(z_4 - d_1, r_4) \\ s = \sqrt{(z_4 - d_1)^2 + (r_4)^2} \end{cases} \tag{2.25}$$

Using trigonometric and cosine laws to find β:

$$a_3^2 = s^2 + a_2^2 - 2sa_2^2 \cos\beta \rightarrow \beta = \cos^{-1}\left(\frac{s^2+a_2^2-a_3^2}{2sa_2^2}\right) \quad (2.26)$$

With these intermediate values, we can now find the remaining angle values as:

$$\begin{cases} \theta_2 = \alpha \mp \beta \\ \theta_3 = atan2(s^2 - a_2^2 - a_3^2, 2a_2a_3) \\ \theta_4 = \psi - \theta_2 - \theta_3 \end{cases} \quad (2.27)$$

2.2.2 Analytical (algebraic) Approach

The algebraic solution of the IK exists only for restricted class of cases. The joint angles could be expressed using the end effector position. The number of nonlinear equations increases with the DOFs [11].

The first joint movement, defined by θ_1, can be calculated using geometric parameters only:

$$\theta_1 = \text{atan2}(y_d, x_d) \quad (2.28)$$

By using the position vector of the end-effector obtained from the forward kinematics, and specifying the desired position of the gripper (x_d, y_d, z_d), this results in the following equations:

$$\begin{cases} x_d = c_1(c_{234}d_5 + a_3c_{23} + a_2c_2) \\ y_d = s_1(c_{234}d_5 + a_3c_{23} + a_2c_2) \\ z_d = -s_{234}d_5 + a_3s_{23} + a_2s_2 + d_1 \end{cases} \quad (2.29)$$

From $x_d^2 + y_d^2$ we get:

$$a_3c_{23} + a_2c_2 = \pm\sqrt{(x_d)^2 + (y_d)^2} - c_{234}d_5 \quad (2.30)$$

From z_d, we can get the value of the joint θ_3:

$$a_3s_{23} + a_2s_2 = z_d + d_5s_{234} - d_1 \quad (2.31)$$

From the sum of the squared equations (2.30) and (2.31), we have:

$$c_3 = \frac{(z_d + d_5s_{234} - d_1)^2 + (\pm\sqrt{(x_d)^2+(y_d)^2} - c_{234}d_5)^2 - a_2^2 - a_3^2}{2a_2a_3} \quad (2.32)$$

Putting:

$$s_\psi = s_{234} = c_1a_x + s_1a_y \quad (2.33)$$

$$c_\psi = c_{234} = a_z \tag{2.34}$$

Then equation (2.32) becomes:

$$c_3 = \frac{(z_d + d_5 s_\psi - d_1)^2 + (\pm\sqrt{(x_d)^2 + (y_d)^2} - c_\psi d_5)^2 - a_2^2 - a_3^2}{2a_2 a_3} \tag{2.35}$$

$$s_3 = \pm\sqrt{1 - c_3^2} \tag{2.36}$$

From (2.35) and (2.36) we get:

$$\theta_3 = atan2(s_3, c_3) \tag{2.37}$$

After calculate $\theta_3$we can get θ_2 by using the equations of the position. From z_d and x_d we get:

$$s_2 = \frac{[c_1(z_d + d_5 s_\psi - d_1) - a_3 s_3 (x_d - c_\psi d_5)](a_3 c_3 + a_2)}{c_1[(c_2 a_3 + a_2)^2 + a_3^2 s_3^2]} \tag{2.38}$$

$$c_2 = \pm\sqrt{1 - s_2^2} \tag{2.39}$$

From (2.38) and (2.39) we get:

$$\theta_2 = atan2(s_2, c_2) \tag{2.40}$$

Then, $\theta_4$is obtained as follows:

$$\theta_4 = \psi - \theta_2 - \theta_3 \tag{2.41}$$

We can find θ_5 by using total transformation matrix:

$$\begin{cases} s_5 = s_1 r_{11} - c_1 r_{21} \\ c_5 = s_1 r_{12} - c_1 r_{22} \end{cases} \tag{2.42}$$

Then:

$$\theta_5 = atan2(s_5, c_5) \tag{2.43}$$

3 Conclusion

This chapter represents the description of the Kinematics model of ED-7220C Robot Arm and its DH parameters. we have seen that Forward Kinematics (FK) is defined as transformation from joint space to Cartesian space where as Inverse Kinematics (IK) is defined as transformation from Cartesian space to joint space.

Chapter 3

Trajectory Planning

1 Introduction

The target of the trajectory planning is to generate the reference inputs for the manipulator control system that ensures the implementation of the desired movement (to enable a robot end-effector to move from an initial position to a desired position). This transition forms the basis for many high-level tasks like reaching objects or avoiding obstacles [12].

It can be assumed that a trajectory planning algorithm takes as inputs the geometric path, the kinematic and dynamic constraints of the manipulator; the output is the trajectory of the joints, or of the end-effector, expressed as a sequence of values of position, velocity and acceleration [13]. A trajectory is specified by its starting pose (position and orientation of the end effector) and the ending pose of the end effector [14].

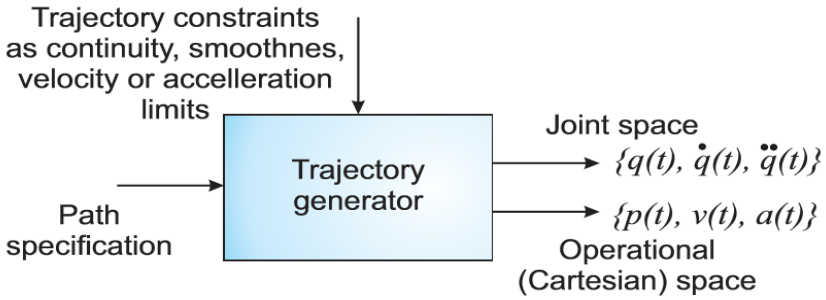


Figure3.1. Trajectory Planning Block Diagram [15].

From the block diagram of the figure3.1., there are mainly two types of trajectories, joint space trajectories and Cartesian space trajectories. Joint space trajectories are generated by defining smooth functions between the starting angle (distance for prismatic joints) and the ending angle for each degree of freedom, the trajectory of the end effector between the starting pose and the ending pose is not considered [16]. For Cartesian trajectories, the trajectory of the end effector is specified in Cartesian coordinates and corresponding joint space configurations are computed for each point of the Cartesian trajectory using inverse kinematics, the problem of singularities and no existence of exact solution for the inverse kinematics rises for this method [14].

2 Joint Space Trajectories

A trajectory planned in the joint space consists in the acquisition of the values, for each joint of the manipulator, corresponding to the via-points set by the user [13]. These via-points are then converted to a set of desired joint angles using inverse kinematics. A smooth function is then found for each of the n joints that starts from the starting set of joints angle, passes through the via points and end at the goal point [16].

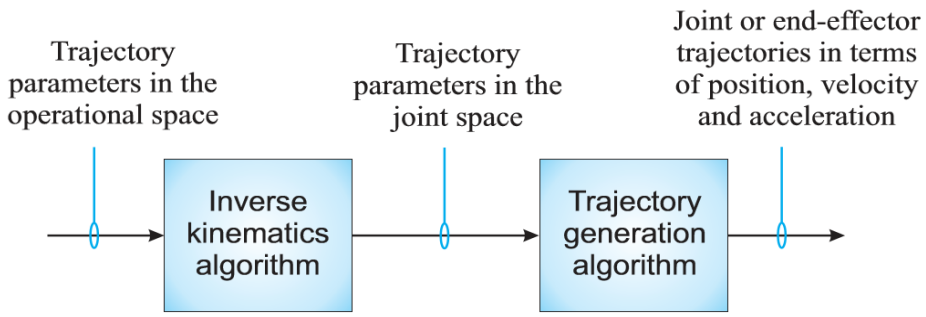


Figure3.2. Trajectory from the operational space to the joints space [15].

The main problem then is to find a trajectory that connects an initial to a final configuration while satisfying other specified constraints at the endpoints (e.g., velocity and/or acceleration constraints). Without loss of generality, we will consider planning the trajectory for a single joint, since the trajectories for the remaining joints will be created independently and in exactly the same way. Thus, we will concern ourselves with the problem of determining q(t), where q(t) is a scalar joint variable [8].

We suppose that at time t_0 the joint variable satisfies [8]:

$$q(t_0) = q_0 \tag{3.1}$$

$$\dot{q}(t_0) = v_0 \tag{3.2}$$

And we wish to attain the values at t_f [8]:

$$q(t_f) = q_f \tag{3.3}$$

$$\dot{q}(t_f) = v_f \tag{3.4}$$

Figure3.3. shows a suitable trajectory for this motion. In addition, we may wish to specify the constraints on initial and final accelerations. In this case, we have two additional equations [8]:

$$\ddot{q}(t_0) = \alpha_0 \tag{3.5}$$

$$\ddot{q}(t_f) = \alpha_f \tag{3.6}$$

The trajectories defined in the joint space are generated by means of interpolation functions which respect the limits imposed [13]. However, there many are trajectory planning methods require polynomial functions with independent variables those methods are explained briefly as follow:

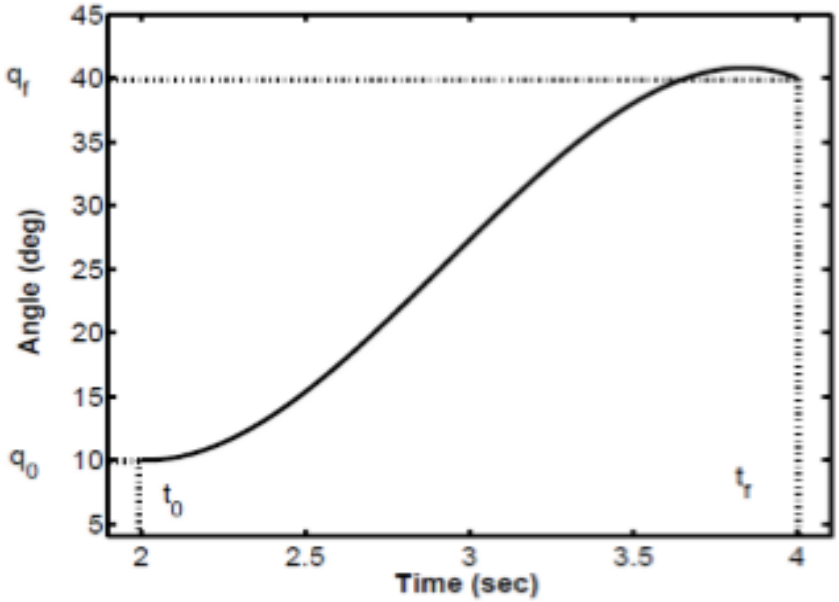


Figure3.3. Typical Joint Space Trajectory [8].

2.1 Cubic Polynomial Trajectories

Suppose that we wish to generate a trajectory between two configurations, and that we wish to specify the start and end velocities for the trajectory. One way to generate a smooth curve such as that shown in Figure3.3. is by a polynomial function of t. If we have four constraints to satisfy, such as (3.1) -(3.3), we require a polynomial with four independent coefficients that can be chosen to satisfy these constraints. Thus, we consider a cubic trajectory of the form [8]:

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \qquad \text{For Distance} \qquad (3.7)$$

Then the desired velocity and acceleration are automatically given as [8]:

$$\dot{q}(t) = a_1 + 2a_2 t + 3a_3 t^2 \qquad \text{For Velocity} \qquad (3.8)$$

$$\ddot{q}(t) = 2a_2 + 6a_3 t \qquad \text{For Acceleration} \qquad (3.9)$$

2.2 Quantic Polynomial Trajectories

A cubic trajectory gives continuous positions and velocities at the start and finish points times but discontinuities in the acceleration. The derivative of acceleration is called the jerk. A discontinuity in acceleration leads to an impulsive jerk, which may excite vibration modes in the manipulator and reduce tracking accuracy [8].

For this reason, one may wish to specify constraints on the acceleration as well as on the position and velocity. In this case, we have six constraints (one each for initial and

final configurations, initial and final velocities, and initial and final accelerations). Therefore, we require a fifth order polynomial [8]:

$$\begin{aligned} q(t) &= a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 && \text{For Distance} \\ \dot{q}(t) &= a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4 && \text{For Velocity} \\ \ddot{q}(t) &= 2a_2 + 6a_3 t + 12a_4 t^2 + 20a_5 t^3 && \text{For Acceleration} \end{aligned} \tag{3.10}$$

2.3 Linear Segments with Parabolic Blends (LSPB)

Another way to generate suitable joint space trajectories is by so-called Linear Segments with Parabolic Blends or (LSPB) for short. This type of trajectory is appropriate when a constant velocity is desired along a portion of the path. The LSPB trajectory is such that the velocity is initially "ramped up" to its desired value and then "ramped down" when it approaches the goal position. To achieve this, we specify the desired trajectory in three parts. The first part from time t_0 to time t_b is a quadratic polynomial. This results in a linear "ramp" velocity. At time t_b, called the blend time, the trajectory switches to a linear function. This corresponds to a constant velocity. Finally, at time $t_f - t_b$ the trajectory switches once again, this time to a quadratic polynomial so that the velocity is linear [8].

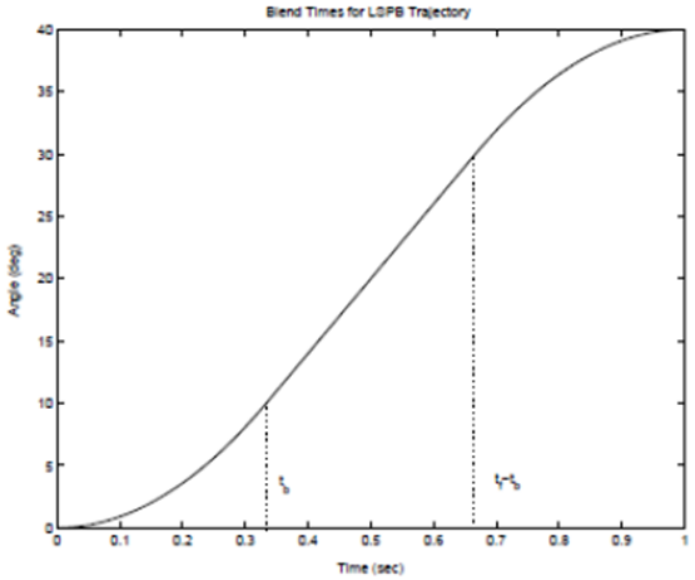


Figure3.4. Blend times for LSPB trajectory [8].

We choose the blend time t_b so that the position curve is symmetric as shown in Figure3.4. for convenience suppose that $t_0 = 0$ and $\dot{q}(t_f) = 0 = \dot{q}(0)$. Then between times 0 and t_b we have:

$$q(t) = a_0 + a_1 t + a_2 t^2 \tag{3.11}$$

So that the velocity is:

$$\dot{q}(t) = a_1 + 2a_2 t \tag{3.12}$$

The constraints q(0) = q_0 and $\dot{q}(0) = 0$ imply that:

$$a_0 = q_0 \tag{3.13}$$

$$a_1 = 0 \tag{3.14}$$

At time t_b we want the velocity to equal a given constant, say V. Thus, we have:

$$\dot{q}(t_b) = 2a_2 t_b = V \tag{3.15}$$

This implies that:

$$a_2 = \frac{V}{2t_b} \tag{3.16}$$

Therefore, the required trajectory between 0 and t_b with $\alpha = \frac{V}{t_b}$ is given as:

$$q(t) = q_0 + \frac{V}{2t_b} t^2 = q_0 + \frac{\alpha}{2} t^2 \tag{3.17}$$

$$\dot{q}(t) = \frac{V}{t_b} t = \alpha t \tag{3.18}$$

$$\ddot{q}(t) = \frac{V}{t_b} = \alpha \tag{3.19}$$

Where α denotes the acceleration.

Now, between time t_f and t_f - t_b, the trajectory is a linear segment (corresponding to a constant velocity V):

$$q(t) = a_0 + a_1 t = a_0 + Vt \tag{3.20}$$

Since, by symmetry:

$$q(t_f) = \frac{q_0 + q_f}{2} \tag{3.21}$$

We have:

$$\frac{q_0 + q_f}{2} = a_0 + V\frac{t_f}{2} \tag{3.22}$$

Which yields:

$$a_0 = \frac{q_0+q_f-Vt_f}{2} \tag{3.23}$$

Since the two segments must "blend" at time t_b we require:

$$q_0 + \frac{V}{2}t_b = \frac{q_0+q_f-Vt_f}{2} + Vt_b \tag{3.24}$$

Which gives upon solving for the blend time t_b:

$$t_b = \frac{q_0-q_f+Vt_f}{V} \tag{3.25}$$

Note that we have the constraint $0 < t_b \leq \frac{t_f}{2}$. This leads to inequality:

$$\frac{q_f-q_0}{V} < t_f \leq \frac{2(q_f-q_0)}{V} \tag{3.26}$$

The inequality can be written in another way:

$$\frac{q_f-q_0}{t_f} < V \leq \frac{2(q_f-q_0)}{t_f} \tag{3.27}$$

Thus, the specified velocity must be between these limits or the motion is not possible. The portion of the trajectory between t_f-t_b and t_f is now found by symmetry considerations. The complete LSPB trajectory is given by [8]:

$$q(t) = \begin{cases} q_0 + \frac{\alpha}{2}t^2 & 0 \leq t \leq t_b \quad with\ \alpha > 0 \\ \frac{q_0+q_f-Vt_f}{2} + Vt & t_b < t \leq t_f - t_b \quad with\ a > 0 \\ q_f - \frac{at_f^2}{2} + at_f t - at^2 & t_f - t_b < t \leq t_f \quad with\ a > 0 \end{cases} \tag{3.28}$$

3 Cartesian Space Trajectories

In Cartesian space trajectory planning, the position, orientation, velocity and acceleration of terminal gripper can be expressed by analytic function, and then the joints' information can be obtained by inverse kinematics. The method has the disadvantage of complicated calculations and may accompany with singular configurations. However, it is more intuitive and can improve the capability of obstacle avoidance for manipulator [17].

3.1 Cartesian Straight-Line Motion

In planning and generating Cartesian straight-line paths, a spline of linear functions with parabolic blends is appropriate. During the linear portion of each segment, all three components of position change in a linear fashion, and the end-effector will move along

a linear path in space. However, if we are specifying the orientation as a rotation matrix at each via point, we cannot linearly interpolate its elements, because doing so would not necessarily result in a valid rotation matrix at all times. A rotation matrix must be composed of orthonormal columns, and this condition would not be guaranteed if it were constructed by linear interpolation of matrix elements between two valid matrices. Instead, angle-axis representation can be used to specify an orientation with three numbers. If this representation of orientation is combined with the 3 by 1 Cartesian-position representation, a 6 by 1 representation of Cartesian position and orientation. After converting to angle-axis representation, the problem reduces to finding spline functions that smoothly vary the six quantities from path point to path point [16].

3.2 Geometric Problems with Cartesian Paths

Because a continuous correspondence is made between a path shape described in Cartesian space and joint positions, Cartesian paths are prone to various problems relating to workspace and singularities [16].

3.2.1 Intermediate Points Unreachable

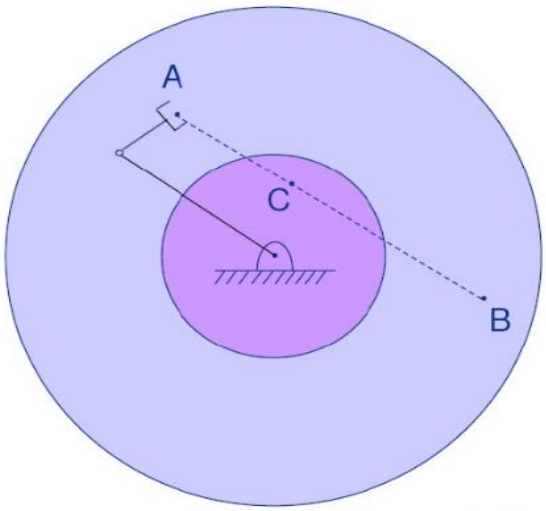


Figure3.5. Cartesian-path problem of type 1: Initial (A) and Goal (B) Points are reachable, but intermediate points (C) unreachable [16].

Although the initial location of the manipulator and the final goal point are both within the manipulator workspace, it is quite possible that not all points lying on a straight line connecting these two points are in the workspace [16]. As an example, consider the planar two-link robot shown in Figure3.5. and its associated workspace. In this case, link 2 is shorter than link 1, so the workspace contains a hole in the middle whose radius is the difference between link lengths. Drawn on the workspace is a start point A and a goal

point B. Moving from A to B would be no problem in joint space, but if a Cartesian straight-line motion were attempted, intermediate points along the path would not be reachable. This is an example of a situation in which a joint-space path could easily be executed, but a Cartesian straight-line path would fail.

3.2.2 High Joint Rates Near Singularity

There are locations in the manipulator's workspace where it is impossible to choose finite joint rates that yield the desired velocity of the end effector in Cartesian space. It should not be surprising, therefore, that there are certain paths (described in Cartesian terms) which are impossible for the manipulator to perform. If, for example, a manipulator is following a Cartesian straight-line path and approaches a singular configuration of the mechanism, one or more joint velocities might increase toward infinity. Because velocities of the mechanism are upper bounded, this situation usually results in the manipulator's deviating from the desired path [16].

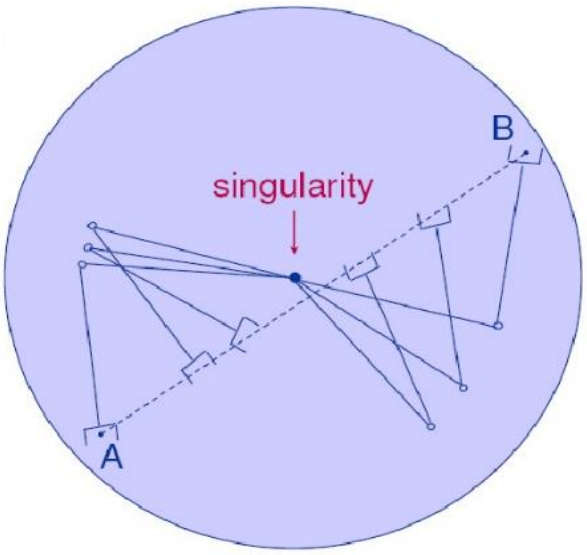


Figure3.6. Cartesian-path problem of type 2: Approaching singularities some joint velocities go to ∞ causing deviation from the path [16].

As an example, Figure3.6. shows a planar two-link (with equal link lengths) moving along a path from point A to point B. The desired trajectory is to move the end tip of the manipulator at constant linear velocity along the straight-line path. In the figure, several intermediate positions of the manipulator have been drawn to help visualize its motion. All points along the path are reachable, but as the robot goes past the middle portion of the path, the velocity of joint one is very high. The closer the path comes to the joint-one axis, the faster this rate Will be. One approach is to scale down the overall

velocity of the path to a speed where all joints stay within their velocity capabilities. In this way, the desired temporal attributes of the path might be lost, but at least the spatial aspect of the trajectory definition is adhered to [16].

3.2.3 Start and Goal Reachable in Different Solutions

A third kind of problem that could arise is shown in Figure3.7. Here, a planar two-link with equal link lengths has joint limits that restrict the number of solutions with which it can reach a given point in space. In particular, a problem will arise if the goal point cannot be reached in the same physical solution as the robot is in at the start point. In Figure3.7. the manipulator can reach all points of the path in some solution, but not in any one solution. In this situation, the manipulator trajectory planning system can detect this problem without ever attempting to move the robot along the path and can signal an error to the user [16].

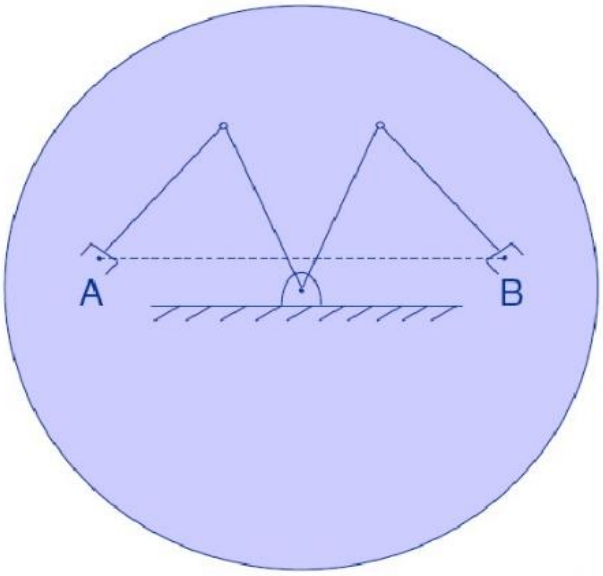


Figure3.7. Cartesian-path problem of type 3: Start point (A) and goal point (B) are reachable in different joint space solutions [16].

To handle these problems with paths specified in Cartesian space, most industrial manipulator-control systems support both joint-space and Cartesian-space path generation. The user quickly learns that, because of the difficulties with Cartesian paths, joint-space paths should be used as the default, and Cartesian-space paths should be used only when actually needed by the application [16].

4 Conclusion

This chapter covers the two main types of the trajectory planning: Joint Space Trajectories and Cartesian Space Trajectories.

Chapter 4

Control Design

1 Introduction

High performance Robot manipulators systems are often difficult to control. This is due to the nonlinear and time varying nature of these systems. This fact has caused that most of the model based control techniques has shown a lack of efficiency and robustness especially because of the coupling existing between the different joint. In the case of the ED-7220C robot manipulator, the experiment has shown that the use of classical linear compensator like the PD controller for instance will not adjust neither the transient performance. Also, a steady state error exists despite that the system is assumed to be a type one system. Plus, when the joint of the robot arm moves with a high speed, the joint system may become unstable. Especially, if there are more than one joint moving simultaneously.

In this project, we have realized a control of a robot manipulator ED-7220C. where we did not have any information about the physical parameters of the robot arm. This resulted in an additional challenge in this realization.

2 Modeling the System (ED-7220C Robot Arm):

The dynamic model of the ED-7220C robot arm is derived using Lagrange-Euler technique whose law is [18]:

$$\frac{d}{dt}\left(\frac{dL}{d\dot{q}}\right) - \frac{dL}{dq} = \mathcal{T} \tag{4.1}$$

With:

$$L = K - U \tag{4.2}$$

Where:

K: The kinetic energy of the robot system.

U: The potential energy of the robot system.

q: an n x 1 vector of the joints angles.

$\dot{q}$: an n x 1 vector of the joints angular speed.

$\mathcal{T}$: an n x 1 vector representing external torques acting on each joint.

The Kinetic energy is found as [18]:

$$K = \frac{1}{2}(A(q)\dot{q_1^2} + B(q)\dot{q_2^2} + C(q)\dot{q_3^2} + D(q)\dot{q_4^2}) \tag{4.3}$$

(See in the Appendix C the functions: A(q), B(q), C(q) and D(q))

And the potential energy is found as:

$$U = E(q)g \tag{4.4}$$

In matrix form the dynamic model of the ED-7220C robot is given as:

$$\mathcal{T} = \begin{bmatrix} A & 0 & 0 & 0 \\ 0 & B & 0 & 0 \\ 0 & 0 & C & 0 \\ 0 & 0 & 0 & D \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \\ \ddot{q}_4 \end{bmatrix} + \begin{bmatrix} \frac{\partial A}{\partial t} & 0 & 0 & 0 \\ 0 & \frac{\partial B}{\partial t} & 0 & 0 \\ 0 & 0 & \frac{\partial C}{\partial t} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} - \frac{1}{2} [\dot{q}_1 \quad \dot{q}_2 \quad \dot{q}_3 \quad \dot{q}_4] \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{\partial A}{\partial q_2} & 0 & 0 & 0 \\ \frac{\partial A}{\partial q_3} & \frac{\partial B}{\partial q_3} & 0 & 0 \\ \frac{\partial A}{\partial q_4} & \frac{\partial B}{\partial q_4} & \frac{\partial C}{\partial q_4} & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{\partial E}{\partial q_2} \\ \frac{\partial E}{\partial q_3} \\ \frac{\partial E}{\partial q_4} \end{bmatrix} g \tag{4.5}$$

$$M(q)\ddot{q} + \left(\dot{M}(q) - \frac{1}{2}\dot{q}^T C(q,\dot{q})\right)\dot{q} + \nabla_q U = \mathcal{T}_{ext} \tag{4.6}$$

2.1 Joint Dynamic Model

The model in Equation (4.6) could be decomposed to a system of equation, where every equation is a model for a separate joint.

For joint 1:

$$A\ddot{q}_1 + \frac{\partial A}{\partial t}\dot{q}_1 = \mathcal{T}_1 \tag{4.7}$$

For joint 2:

$$B\ddot{q}_2 + \frac{\partial B}{\partial t}\dot{q}_2 - 0.5\frac{\partial A}{\partial q_2}{\dot{q}_1}^2 + \frac{\partial E}{\partial q_2}g = \mathcal{T}_2 \tag{4.8}$$

For joint 3:

$$C\ddot{q}_3 + \frac{\partial C}{\partial t}\dot{q}_3 - 0.5\left(\frac{\partial A}{\partial q_3}{\dot{q}_1}^2 + \frac{\partial B}{\partial q_3}{\dot{q}_2}^2\right) + \frac{\partial E}{\partial q_3}g = \mathcal{T}_3 \tag{4.9}$$

For joint 4:

$$D\ddot{q}_4 + \frac{\partial D}{\partial t}\dot{q}_4 - 0.5\left(\frac{\partial A}{\partial q_4}{\dot{q}_1}^2 + \frac{\partial B}{\partial q_4}{\dot{q}_2}^2 + \frac{\partial C}{\partial q_4}{\dot{q}_3}^2\right) + \frac{\partial E}{\partial q_4}g = \mathcal{T}_4 \tag{4.10}$$

The four models could be rearranged so that we could write them as:

$$\ddot{q}_i = f_i(q,\dot{q},u_i,\omega) + u_i \tag{4.11}$$

Where:

U: the control input (generally directly proportional to the torque).

ω: The disturbance applied on the system including the gravity term and the noise.

2.2 Assumptions Concerning the Function *f ():*

From the equations shown above we can notice that all the functions f_i *()* are composed from sine and cosine functions plus some finite constants (see in Appendix C). From this we can assume that the functions f_i *()* and their derivatives – say h_i *()* - with respect to time are bounded functions.

Also, since we don't have any accurate information about neither the mass of the joints, nor their length and center of mass location the functions f_i *()*, these functions can't be exactly computed. So, the only information we have about the same functions is the information presented in the previous paragraph.

3 Control using Classical Techniques

This control is done using PID design technique for each joint. In position control problems where the actuator is actually a DC motor, the preferred type of controller is the PD. This is because the model of the system generally includes a pole at the origin and thus the system is type one. Which means that there is a guarantee (theoretically and assuming system is Linear) that the steady-state error will be null. Besides, the addition of the integrator will affect the stability of the system and slow the system response down. So, when possible the integrator should preferably be avoided in this kind of control.

There is a problem in using this technique which consists of the nonexistence of an accurate model for any of the robot arm joints. Besides, even if we find a model which seems to be accurate, it will not be valid along the total motion of the robot arm since many parameters changes by the variation in the joint angles and angular speeds.

3.1 PD Control Design

A digital PD compensator was designed to control each joint of the robot arm. And since we didn't have an accurate model for the joints we've used "Ziegler-Nichols method".

3.1.1 Ziegler-Nichols Method

It is a heuristic method for tuning a PID controller generally developed by J.G. Ziegler and N.B. Nichols. In this method two values are found experimentally which are K_u and T_u. The procedure is as follow [19]:

- Set the proportional, derivative and integral gain to "0".

- The proportional gain is then increased until a critical gain K_u is reached where the output of the loop has a stable consistent oscillation.
- The values K_u and T_u (the oscillation period) are then collected and the gains of the PD could be set as:

Ziegler–Nichols method			
Control Type	**K_p**	**T_i**	**T_d**
PD	$0.8K_u$	-	$T_u/8$

Table4.1. Ziegler-Nichols tuning chart for PD Compensator **[19]**.

The Transfer function of the PD is then:

$$C(z) = K_p(1 + T_d(1 - z^{-1})) \tag{4.12}$$

Remark: The discrete version of the PD compensator is obtained using the Euler backward transformation because it's the transformation that suits the Derivative term.

Joints	K_p	T_d
Joint 1	12	1.5
Joint 2	8	1.2
Joint 3	9	0.9
Joint 4	5	0.3

Table4.2. PD Compensator Experiment's Coefficients for the Joints.

3.2 Results

After trying the PD compensated system and analyzing its performance (analysis is presented in chapter 5) it was clear that the PD isn't the most appropriate controller to be used for this kind of systems. Because the nonlinearities and time variation of the system were too large to be neglected.

4 Active Disturbance Rejection Control [20]

As it was shown previously classical control fails when applied on a system with high nonlinearities and time varying nature (when system isn't LTI). Hence, a solution to control the robot arm which as an example of that kind of systems is to find a model independent control technique where having an appropriate model for the system isn't a necessary condition for the control. One technique which belongs to this category is "Active disturbance rejection control" (ADRC) technique [20].

4.1 Generalities and Problem Formulation

Most physical plants are highly nonlinear and time varying (including the robot manipulator ED-7220C). That's why in this case most of control techniques which depends on the model of the system don't give an accurate and guaranteed control since they require an accurate model for the system. However, the ADRC technique gives a very important and a viable alternative, since it needs really few information about the system.

In this approach, physical systems are modeled using the following general function:

$$y^{(n)} = f\left(y^{(n-1)}, \dots\dots, y, u, w, t\right) + b.u \tag{4.13}$$

Where:

y: is the output of the system.

u: is the input of the system.

w: is the disturbance signal.

n: is the order of the system.

We put:

$$\begin{array}{c} x_1(t)=y \\ x_2(t)=\dot{y} \\ \vdots \\ x_n=y^{(n-1)} \end{array} \tag{4.14}$$

Then, a state space equivalent model could be written from the equation (4.13):

$$\begin{array}{c} \dot{x}_1(t)=x_2(t) \\ \dot{x}_2(t)=x_3(t) \\ \vdots \\ \dot{x}_n(t)=f(t)+b.u \end{array} \tag{4.15}$$

Where $[x_1\ x_2\ \dots . x_n]^{\mathrm{T}} \in \mathrm{R}^{n+1}$ represent the sate vector of the system.

In this method, the function $f(t)$ is considered as an extended state $x_{n+1} = f(t)$. Hence, the extended state space equation will be:

$$\begin{array}{c} \dot{x}_1(t)=x_2(t) \\ \dot{x}_2(t)=x_3(t) \\ \vdots \\ \dot{x}_n(t)=x_{n+1}(t)+b.u \\ \dot{x}_{n+1}(t)=h(t) \end{array} \tag{4.16}$$

Where:

$h(t)$ is the time derivative of $f(t)$ (i.e. $h(t) = \frac{\partial f}{\partial t}$).

Those states are then estimated using an extended state observer that includes the extended state which is used to compensate for the function $f(t)$ by using:

$$u = u_0 - f(t) \tag{4.17}$$

Then, we can design a simple classical controller (PD for example) to meet the transient and steady state performance requirements of the system.

The ADRC technique could be summarized in this block diagram:

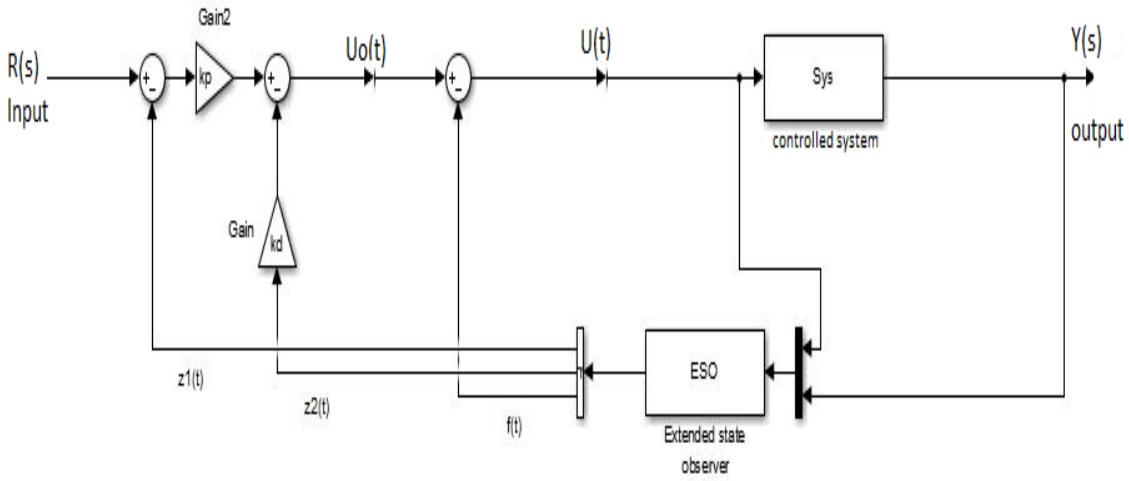


Figure4.9. Conceptual Diagram of ADRC

4.2 Extended State Observer (ESO)

Like it was mentioned previously, ADRC requires the design of an observer to estimate the states of the system. The observer that is used in this technique is called "Extended State observer" [20]. The ESO is described using the following state space form.

$$\begin{aligned} \dot{z}_1 &= z_2 - \beta_1.(z_1 - y) \\ \dot{z}_2 &= z_3 - \beta_2.(z_2 - y) \\ &\vdots \\ \dot{z}_n &= z_{n+1} - \beta_n.(z_1 - y) + b.u \\ \dot{z}_{n+1} &= \beta_{n+1}.(z_1 - y) \end{aligned} \tag{4.18}$$

Where:

z_i: represent the estimate of the state x_i.

$[\beta_1\ \beta_2\ \ldots\ldots \beta_{n+1}]^{\mathrm{T}}$: represents the observer gain vector.

y: represents the output signal.

u: represents the control signal.

Hence, with this technique, only two parameters are to be tuned:

- **The observer bandwidth** [20]**:** which is the key in finding the components of the observer gain vector:

$$\Delta(s) = s^{n+1} + \beta_1.s^{n-1} + \cdots + \beta_n.s + \beta_{n+1} = (s + \omega_0)^{n+1} \tag{4.19}$$

- **The controller gain ω_c** [20]**:** which determines how fast is the system response. Also, the gains of the controller (the PD controller is used) K_p and K_d are tuned according to the value of ω_c

4.3 Convergence of the ESO Observer [20]

The error between the estimated state and the real state need to converge to zero as quickly as possible for a good control. In this part, we discuss the convergence of the ESO for a system with model of the form given in the equation (4.13) [20].

An ESO is represented using system of equations (4.18), While the system states are modeled using the system of equations (4.16).

Let's define a new state: $\tilde{x}_i = x_i - z_i$

Then from the equations (4.16) and (4.18) we get [20]:

$$\begin{gathered}
\dot{\tilde{x}}_1 = \tilde{x}_2 - \beta_1\tilde{x}_1 = \tilde{x}_2 - \omega_0^1 \propto_1 \tilde{x}_1 \\
\dot{\tilde{x}}_2 = \tilde{x}_3 - \beta_2\tilde{x}_1 = \tilde{x}_3 - \omega_0^2 \propto_2 \tilde{x}_1 \\
\vdots \\
\dot{\tilde{x}}_n = \tilde{x}_{n-1} - \beta_n\tilde{x}_1 = \tilde{x}_{n-1} - \omega_0^n \propto_n \tilde{x}_1 \\
\dot{\tilde{x}}_{n+1} = h - \beta_{n+1}\tilde{x}_1 = h - \omega_0^{n+1} \propto_{n+1} \tilde{x}_1
\end{gathered} \tag{4.20}$$

Then, we define the error as: $\epsilon_i = \frac{\tilde{x}_i}{\omega_0^{i-1}}$. Hence, the error equation is written:

$$\dot{\epsilon} = \omega_0.A.\epsilon + B.\frac{h}{\omega_0^n} \tag{4.21}$$

Where:

$$A = \begin{bmatrix} -\propto_1 & 1 & 0 & \cdots & 0 \\ -\propto_2 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -\propto_n & 0 & \cdots & 0 & 1 \\ -\propto_{n+1} & 0 & \cdots & 0 & 0 \end{bmatrix} \text{ and: } B = \begin{bmatrix} 0 \\ 0 \\ \cdots \\ 1 \end{bmatrix}$$

If we assume that h $(\omega, X, \dot{X}, t, u)$ is bounded (which is the case for the ED-7220C robotic system) then we can prove that the error will converge very close to zero in case ω_0 is high enough (Proof in Appendix D).

5 Control of ED-7220C Robot Arm by ADRC Technique

5.1 Problem Formulation

As it was mentioned previously, the robot arm is actually a time varying system and highly nonlinear. Plus, by increasing the speed of operation the nonlinearities increases and it becomes a very difficult task to control it. Also, when decoupling the equations of the different joints, each joint dynamic can be modeled as:

$$\ddot{q}_i = f_i(q, \dot{q}, u_i, \omega_i) + u_i \tag{4.22}$$

Furthermore, we have $f_i(q, \dot{q}, u_i, \omega_i)$ and its time derivative (let's call it $h_i(q, \dot{q}, u_i, \omega_i)$) are both bounded since they consist of a composition of sines and cosines functions.

For these reasons, it's possible to apply the ADRC to control the joints of the robot arm.

5.2 The Control Procedure

First of all, we write the equation (4.22) in state space form with $\begin{bmatrix} x_{i,1} & x_{i,2} \end{bmatrix}^T = \begin{bmatrix} q_i & \dot{q}_i \end{bmatrix}^T$ for each joint:

$$\begin{aligned} \dot{x}_{i,1}(t) &= x_{i,2}(t) \\ \dot{x}_{i,2}(t) &= f_i(q, \dot{q}, u_i, \omega_i) + u_i \end{aligned} \tag{4.23}$$

Then, a third state is added $x_{i,3}(t) = f_i(q, \dot{q}, u_i, \omega_i)$ hence the extended state space form will be as follow:

$$\begin{aligned} \dot{x}_{i,1}(t) &= x_{i,2}(t) \\ \dot{x}_{i,2}(t) &= x_{i,3} + u_i \\ \dot{x}_{i,3}(t) &= h_i(q, \dot{q}, u_i, \omega_i) \end{aligned} \tag{4.24}$$

Next, the extended state observer is designed for each joint following this equation:

$$\begin{aligned} \dot{z}_{i,1}(t) &= z_{i,2}(t) + \beta_1 . (x_{i,1} - z_{i,1}) \\ \dot{z}_{i,2}(t) &= z_{i,3} + u_i + \beta_2 . (x_{i,1} - z_{i,1}) \\ \dot{z}_{i,3}(t) &= \beta_3 . (x_{i,1} - z_{i,1}) \end{aligned} \tag{4.25}$$

With $[\beta_1 \ \beta_2 \ \beta_3]^{\mathrm{T}}$ is the gain vector of the observer which is obtained using the following equation:

$$\Delta(s) = s^3 + \beta_1 . s^2 + \beta_2 . s + \beta_3 = (s + \omega_0)^3 \tag{4.26}$$

And $\omega_0 = 150$

Hence, we find $[\beta_1 \quad \beta_2 \quad \beta_3]^{\mathrm{T}} = [450 \quad 67500 \quad 3375000]^{\mathrm{T}}$

Notice that:

- ω_0 was tuned experimentally and not theoretically, because of the absence of information about physical parameters of the robot arm ED-7220C.
- ω_0 was chosen to be high enough to get a good tracking of the real states of the system and respecting the limitation of the controller (The frequency limitations of the Arduino Due).

Since the controller and the observer is designed on a digital control board (Arduino Due), the ESO should then be discretized. The discretization was done using Tustin transformation to have:

$$\begin{aligned} Z_{i,1}(k+1) &= -0.4623\ Z_{i,1}(k) + 0.05646\, Z_{i,2}(k) + 1.4623\ x_{i,1}(k) \\ Z_{i,2}(k+1) &= -5.085\ Z_{i,1}(k) + 0.4241 Z_{i,2}(k) + 0.01495\ Z_{i,3}(k) + 0.01495\, u_i(k) + 54.85 x_{i,1}(k) \\ Z_{i,3}(k+1) &= -807.9\ Z_{i,1}(k) - 8.483\ Z_{i,2}(k) + 0.9109\, Z_{i,3}(k) + 807.9\ x_{i,1}(k) \end{aligned} \quad (4.27)$$

Here is a block diagram representing the ESO:

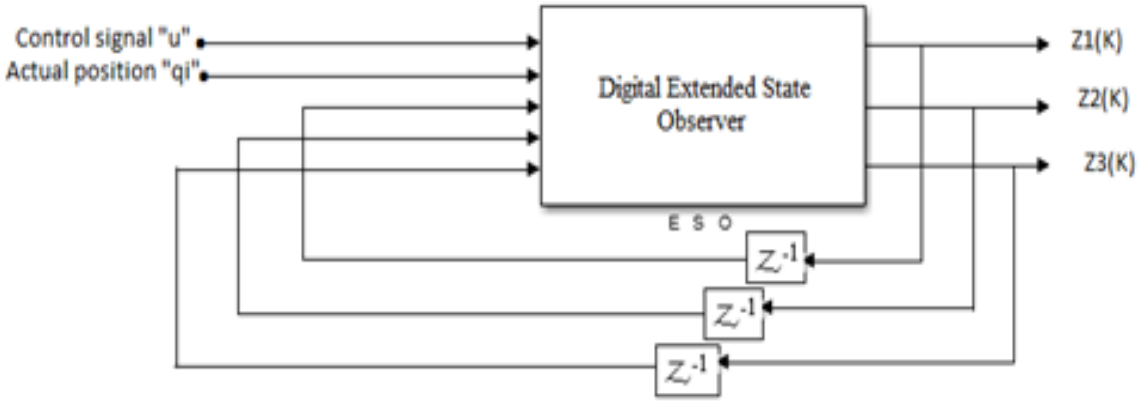


Figure4.10. The ESO Block Diagram

Now that we have the estimates of the three states; we can use $z_{i,1}$ and $z_{i,2}$ which are the estimates of the position and the angular speed respectively, and $z_{i,3}$the estimate of the $f_i()$ of the i^{th} joint to apply the following control law [21]:

$$u_i = k_p\left(ref_i - z_{i,1}\right) - k_d\, z_{i,2} - z_{i,3} \quad (4.28)$$

With a good tracking of the ESO (i.e. $z_{i,1} \cong q_i\ , z_{i,2} \cong \dot{q}_i\ and\ z_{i,3} \cong f_i()$) "$u_i$" could be approximated to:

$$u_i = k_p\ (ref_i - q_i) - k_d\ \dot{q}_i - f_i \quad (4.29)$$

Finally, the new system could be described in this state space form:

$$\begin{aligned} \dot{x}_{i,1}(t) &= x_{i,2}(t) \\ \dot{x}_{i,2}(t) &= k_p\left(ref_i - x_{i,1}\right) - k_d x_{i,2} \end{aligned} \quad (4.30)$$

In matrix form:

$$\begin{bmatrix}\dot{x}_{i,1}\\ \dot{x}_{i,2}\end{bmatrix} = \begin{pmatrix}0 & 1\\ -k_p & -k_d\end{pmatrix}\begin{bmatrix}x_{i,1}\\ x_{i,2}\end{bmatrix} + \begin{bmatrix}0\\ k_p\end{bmatrix}.ref_i$$
$$q_i = [1 \quad 0]\begin{bmatrix}x_{i,1}\\ x_{i,2}\end{bmatrix} \tag{4.31}$$

The equivalent transfer function of the system is:

$$T(s) = \frac{q_i(s)}{R_i(s)} = \frac{k_p}{s^2+k_d s+k_p} \tag{4.32}$$

By tuning k_p and k_v such that:

$$k_p = \omega_c^2 \qquad and \qquad k_d = 2\ \omega_c$$

The total transfer function could be rewritten as:

$$T(s) = \frac{q_i(s)}{R_i(s)} = \frac{\omega_c^2}{s^2+2\ \omega_c s+\omega_c^2} \tag{4.33}$$

Which is a conventional second order system with:

- Natural frequency: $\omega_n = \omega_c$
- Damping ratio: $\xi = 1$

6 Conclusion

This chapter represents and explains the ESO based ADRC technique from the problem formulation of the technique and how to design the ESO to what conditions should be satisfied for the ESO to converge to the real values of the states. After, The ESO was designed specifically to our robotic system. Finally, the control law acting on the system was designed based on the estimated states by the ESO.

Chapter 5

Results & Discussion

1 Introduction

After we have design ESO based ADRC control, it should be simulated then tested on the real system to be evaluated based on its results. For this purpose, a circuit was designed and implemented with the control algorithm including the extended state observer programmed on an Arduino Due board. In the following chapter, the implemented circuit along with the simulation and the results are presented before evaluating the performance of the robotic system under the active disturbance rejection control.

2 The Implemented Circuit

To do the control of the robot arm we needed to establish the system shown in the figure below:

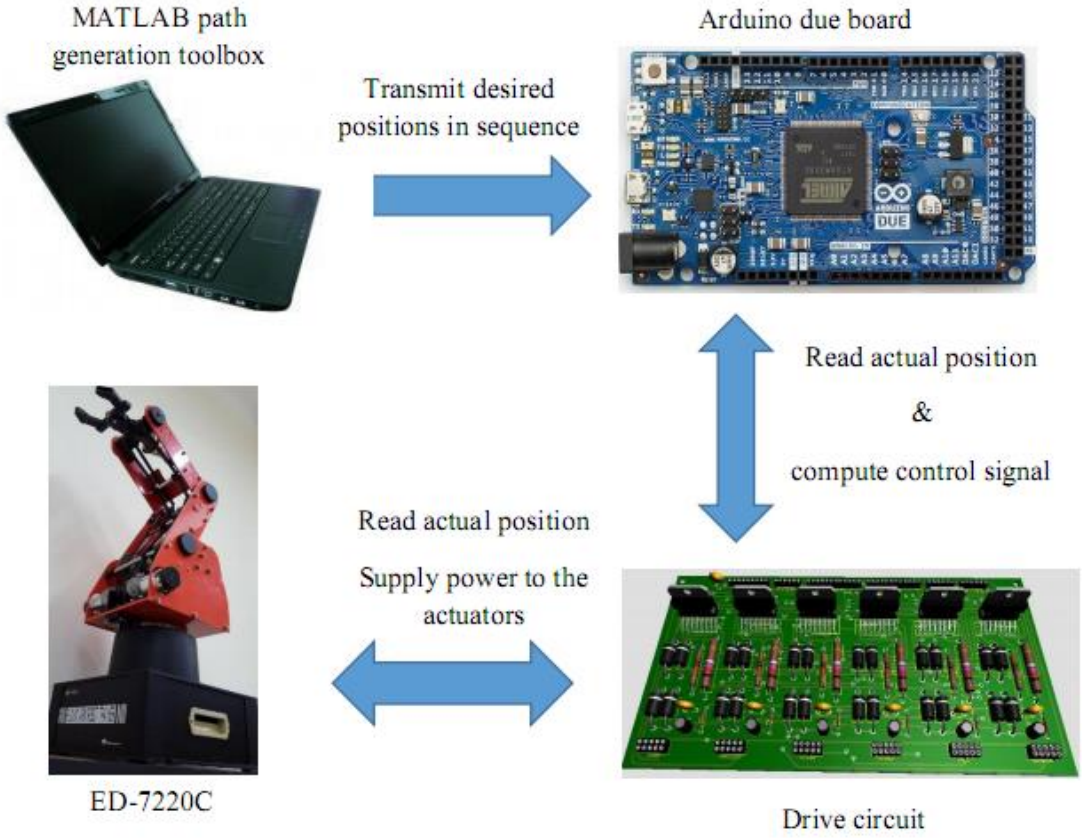


Figure5.1. Overall system schematic [14].

Where the path generation is done in computer where the path points coordinates are transformed to revolute variables for each joint. Next, the Arduino receives the variables from the computer and transform it into reference encoder count for each joint. Also, it executes a control algorithm, sends commands to the Drive circuit which by itself drives the motors of the joints according to the received commands. Also, it receives feedback

information (actual encoder count to know the angle of rotation) from the robot arm to send it back to the Arduino in a loop fashion.

3 Simulation

Before applied on the physical system (the robot arm). ESO based ADRC technique was simulated on Simulink using the same block diagram shown in Figure4.8. with the controlled system having a model with varying coefficients to simulate the time varying nature of the robot arm joints models, and the ESO along with the k_p and k_d chosen exactly equal to the values applied on the real system control.

The coefficients kp and kv where tuned to be equal to:

$$k_p = 225 \quad \text{And} \quad k_d = 30$$

Which corresponds to $\omega_c = 15$ which guarantees a good transient performance and $\omega_o$is still larger enough for a good tracking of the states as it was mentioned previously.

The response of the system to a reference of 1000 counts of the encoder of the joint's motor (which corresponds to an angle of 20.7 ° in case of base joint) is recorded via the Scope of Simulink and presented in the following figure which shows a good transient response with a steady-state error very close to zero.

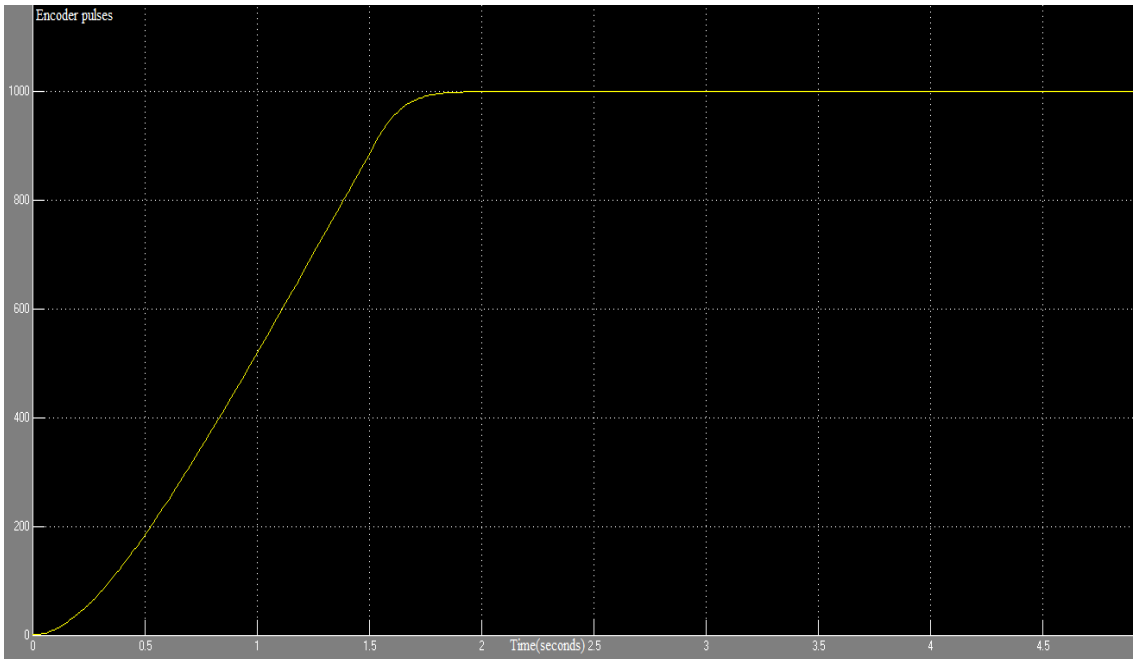


Figure5.2. The response of the system to a reference of 1000 counts.

4 Results of the Control

Now that the control systems were designed, It was necessary to test them on the robot arm system. In this par both PD and ADRC controls results are presented and here we will see how ADRC is more efficient in improving both the transient and the steady state performance.

4.1 PD control Results and Discussion

After designing the PD controllers for the joints of the robot arm, the new closed loop systems were put under experiment to see their responses for a step input. For this, two types of motions were examined: single joint motion for all the joints and simultaneous motion of all the joint.

For the Single joint motion control the experiment has shown that the error is large (around 5% or greater) while the settling time is large compared to the desired value for all the joints. The figures below show the step response to the robot arm joints systems after using a PD controller.

The reason for these results is the existence of nonlinearities which consists of the gravity effect (especially for the shoulder and the elbow joints) and the inertia which increases by increasing the rotational speed. These nonlinearities are large enough that they can't be compensated using a PD compensator.

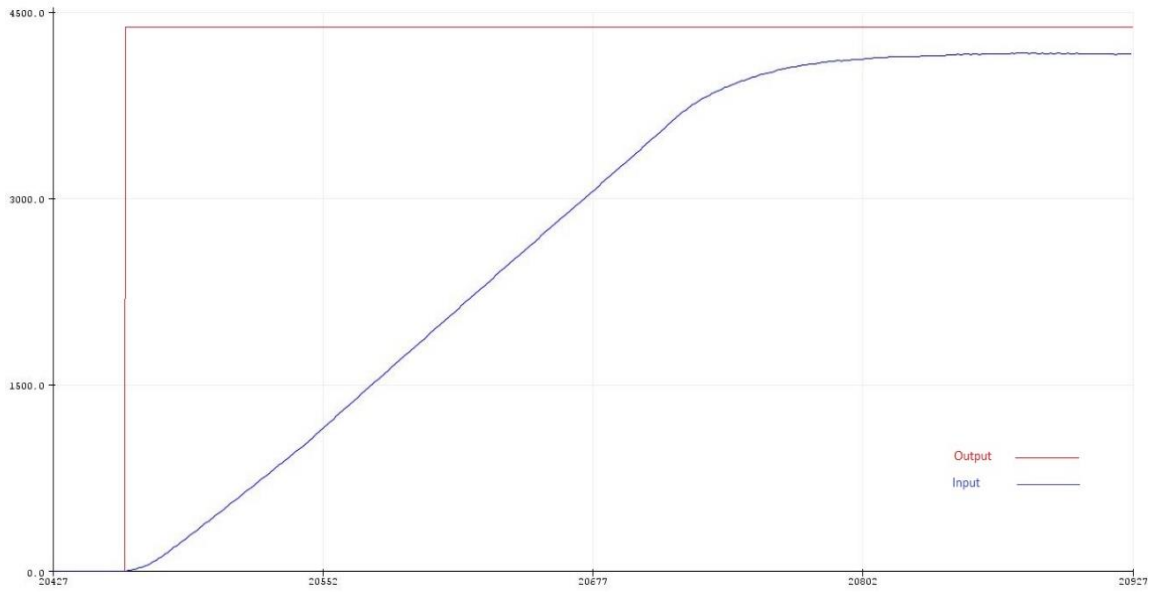


Figure5.3. step response of the base joint during single motion.

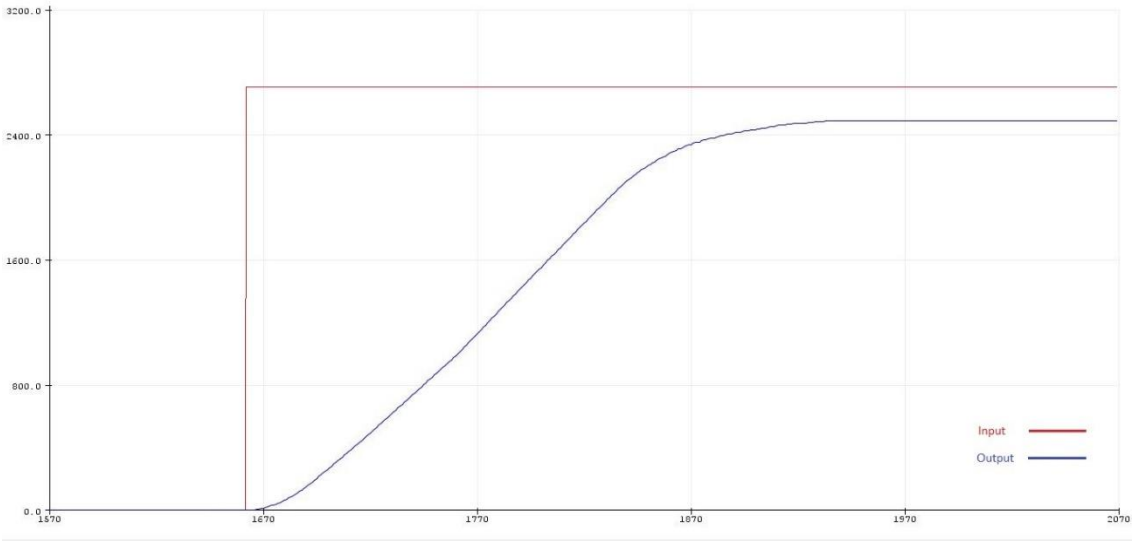


Figure5.4. step response of the base joint during single motion.

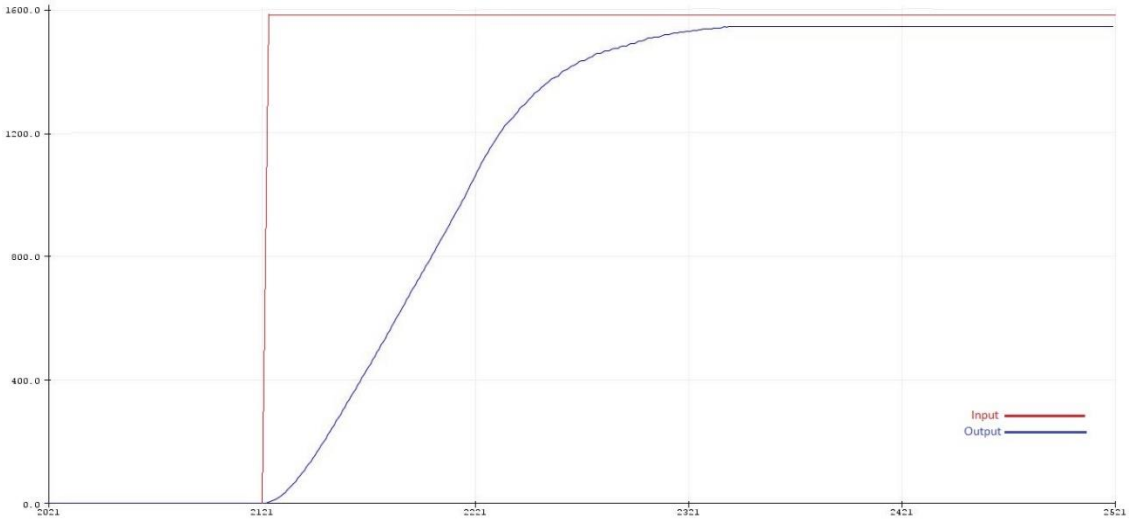


Figure5.5. step response of the shoulder joint during single motion.

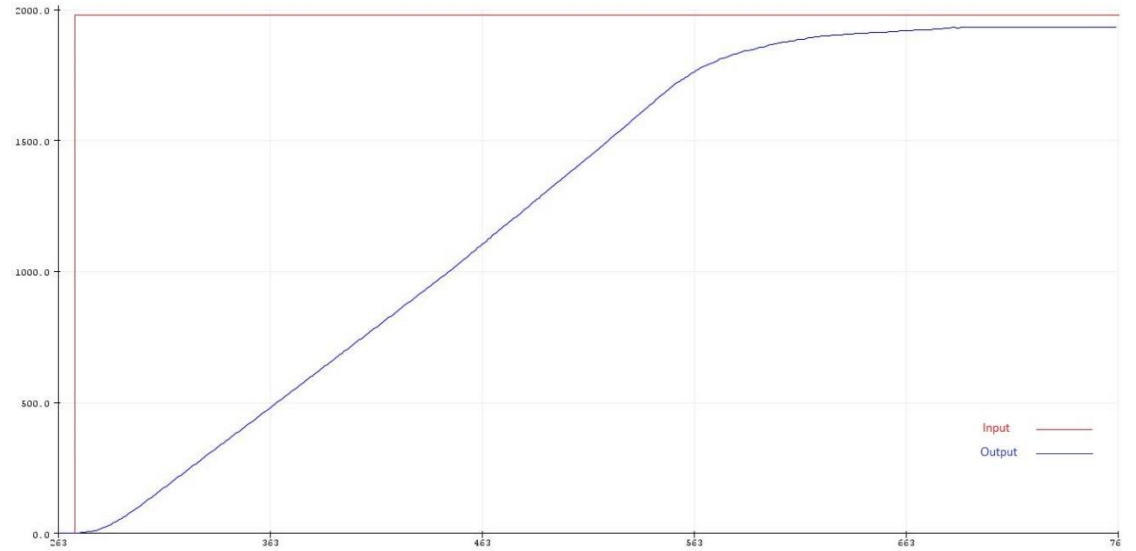


Figure5.6. step response of the wrist joint during single motion.

In other hand, when all the joints (or at least two from the three joints: base-shoulder-elbow) moves together the steady state error along with the settling and rise times increase. The next four figures show the step responses of the joint when moving simultaneously.

This significant increase was predictable because when the joints moves simultaneously the coupling effect becomes tangible. Also, if the speed of rotation is high enough (when the H-bridges are sourced with high voltage) the Coriolis term in the dynamics model becomes very large which affects the performance of the system.

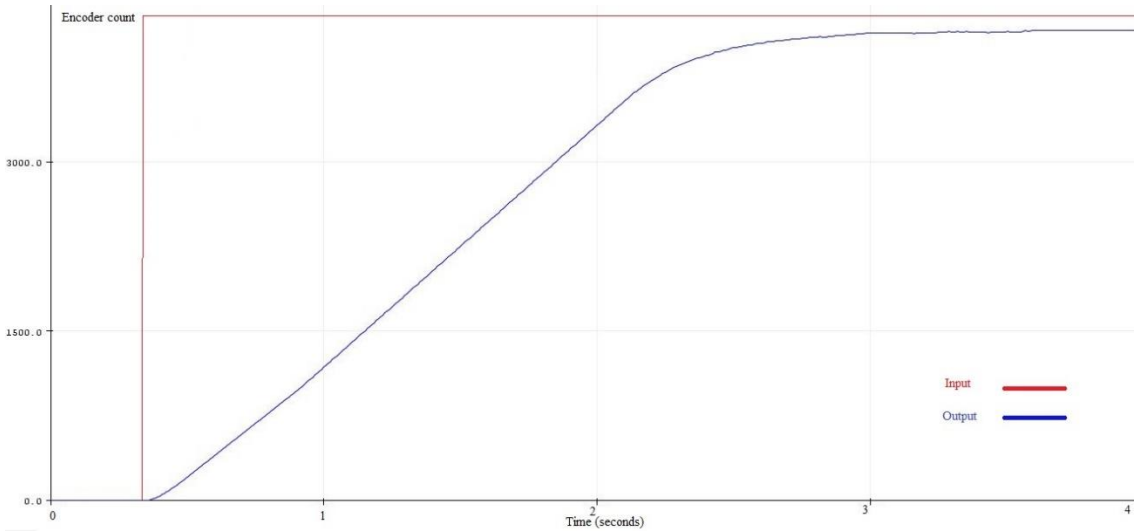


Figure5.7. step response of the base joint during full motion.

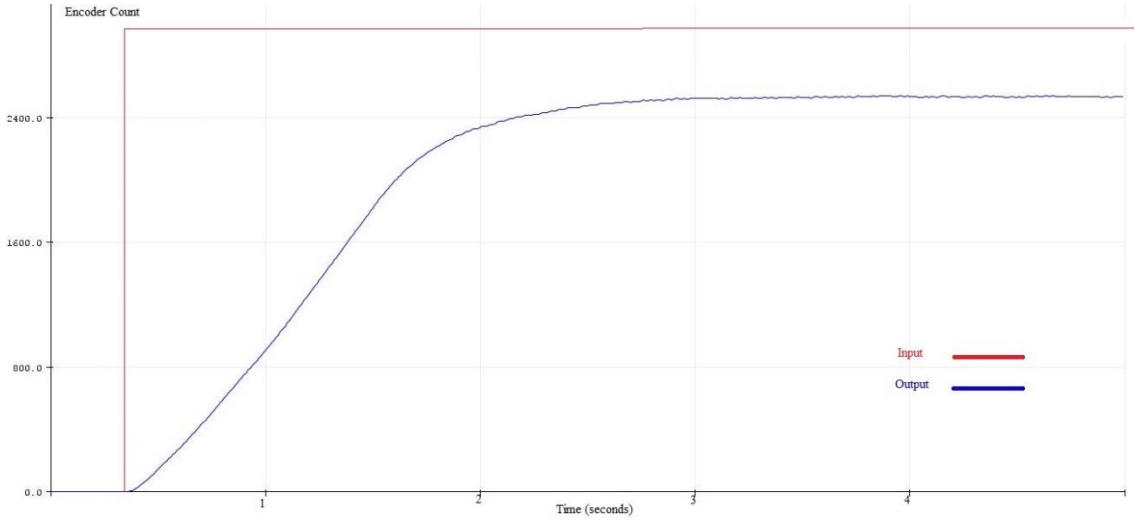


Figure5.8. step response of the elbow joint during full motion.

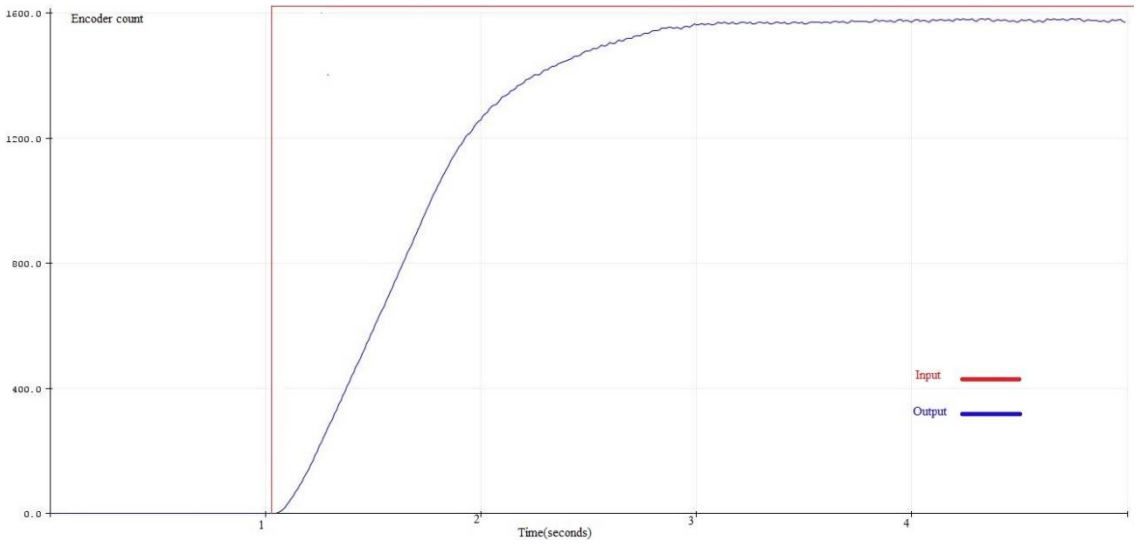


Figure5.9. step response of the shoulder joint during full motion.

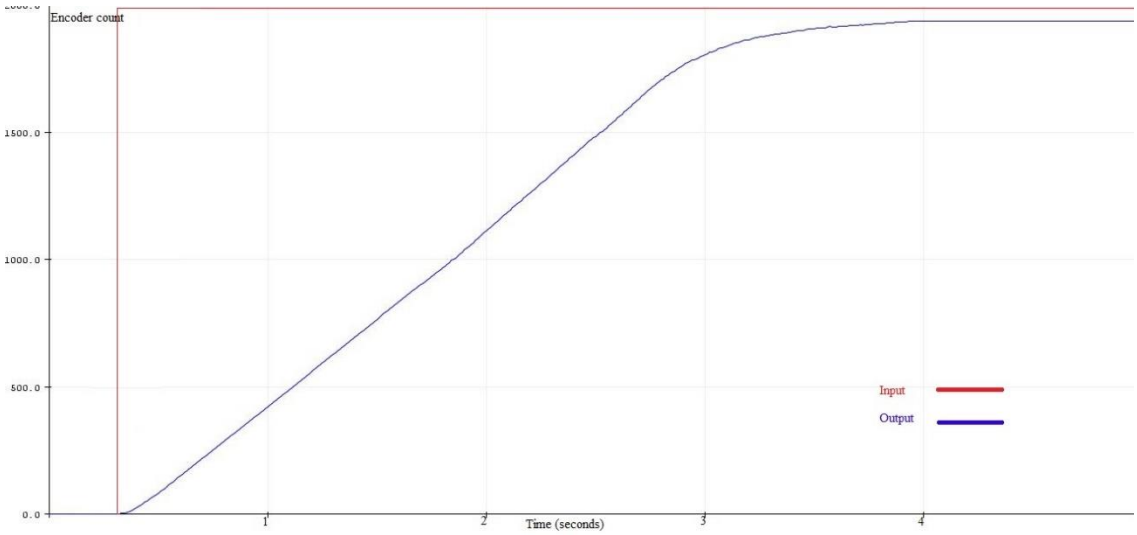


Figure5.10. step response of the wrist joint during full motion.

4.2 ADRC Control Results and Discussion

When applied on the robot arm joints; the ADRC control has given very good results on all the joints. Using the same values of ω_o , ω_c , k_p and k_d used in the simulation, the ESO was designed on the Arduino along with the Control law "u_i".

Despite that the ESO estimate the function $f()$. It could not solve the problem of saturation in the control signal, and that's because the saturation exists in the controller and consists of one of its physical characteristics. The existence of this nonlinearity has affected the transient performance of the robot joints (increased the settling time with the rise time). And caused some little intangible overshoot sometimes.

The effect of the saturation on the transient response could be attenuated by increasing the reference voltage source feeding the H-bridges the higher possible. As it gives more initial acceleration which increased the speed largely, and that will result in a smaller settling and rise times.

The following flowchart explains the algorithm implemented on the controller board:

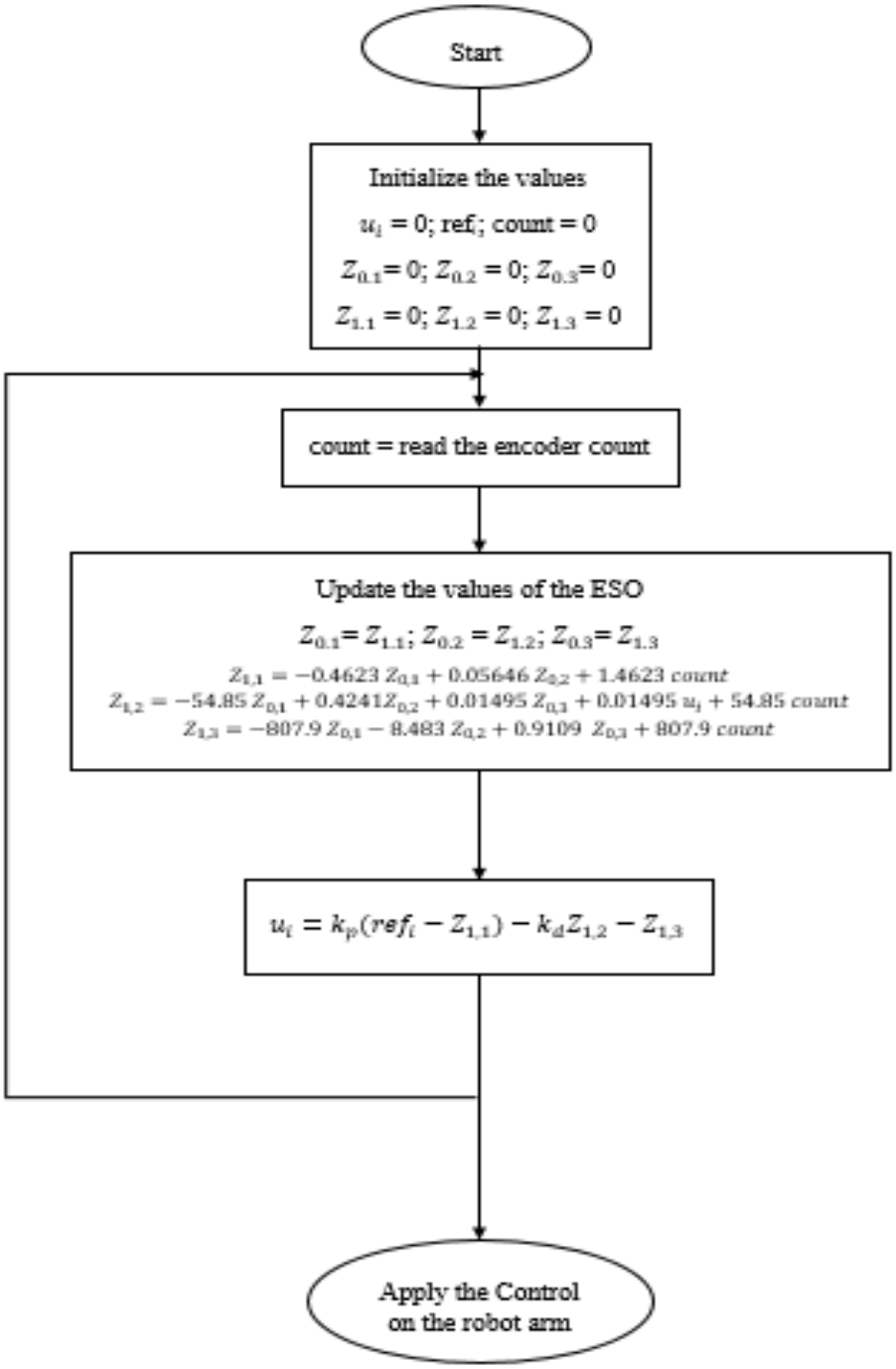


Figure5.11. Flowchart of the control algorithm implemented.

The response for a step input is presented in the following figures:

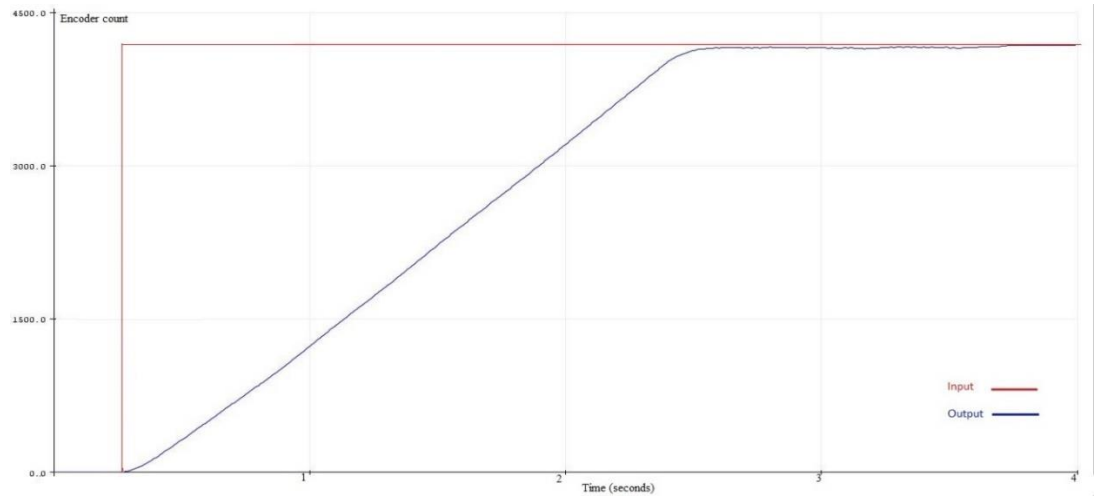


Figure5.12. step response of the base joint.

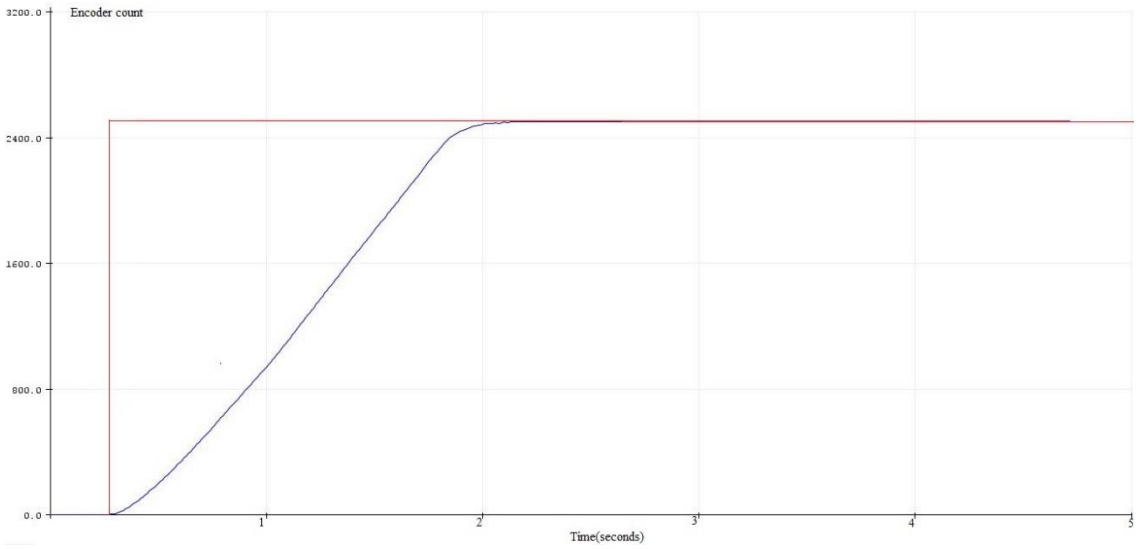


Figure5.13. step response of the elbow joint.

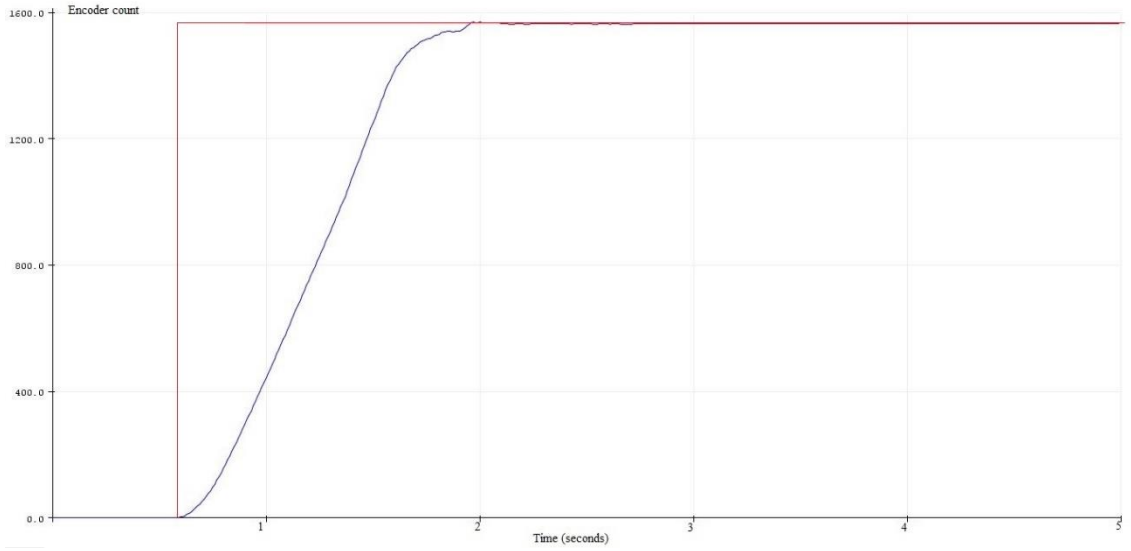


Figure5.14. step response of the shoulder joint.

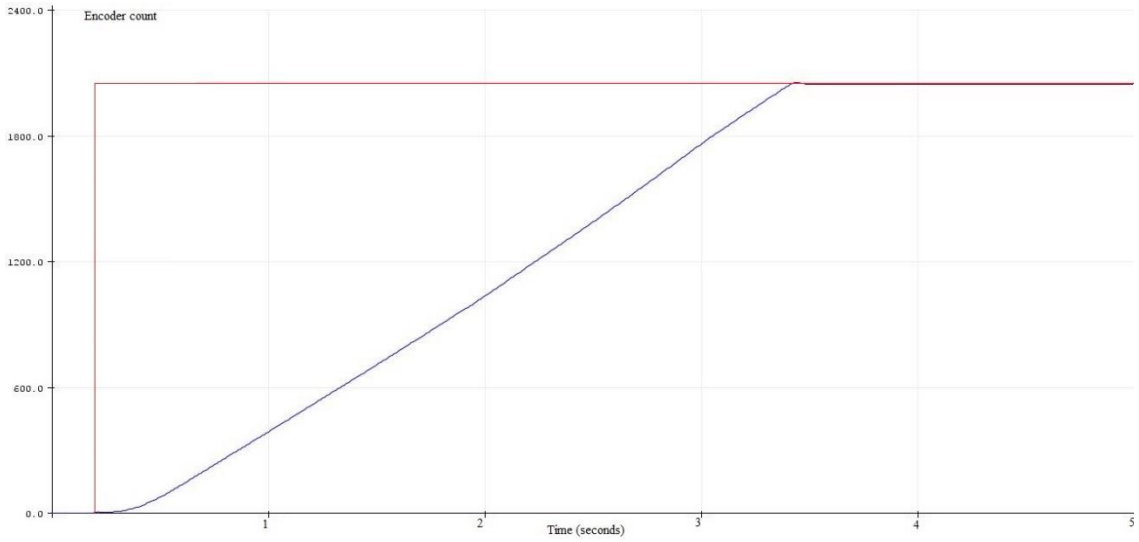


Figure5.15. step response of the wrist joint.

Form the four previous figures we can see that the four joints transient responses have improved comparing to PD control. Also, the Error for the four joints is almost zero which consists of a very large improvement again comparing to the PD control.

A small oscillation behavior was noticed at the steady-state, which is due to the existence of small noise not compensated by the ESO. This noise is amplified because of the existence of the derivative term ($k_v\, z_{i,2}$) which amplifies the effect of the noise. Solution is to implement a dead band strategy, where the control signal is forced to zero whenever the absolute value of the error between the reference angle and the actual angle is less than a tolerated value of error ξ_0:

$$
\begin{aligned}
&if\ error < \xi_0 \\
&\quad u_i = 0 \\
&else \\
&\quad u_i = kp.\left(ref_i - z_{i,1}\right) - kv.z_{i,2} - z_{i,3}
\end{aligned}
$$

5 Conclusion

In this chapter, the circuit used to realize the control of the robot arm was presented at first. Then the results of applying the ADRC control on the robot arm were shown after ADRC was simulated using Simulink. Next, the response of the 4 major joints to a step input was presented. The simulation with the step response figures approved the efficiency of ADRC technique especially in eliminating the error or at least make it very close to zero.

General Conclusion

In this project, Kinematics study of the ED-7220C robot arm was realized by deriving the forward and inverse kinematics of the arm using Denavit-Hartenberg parameters. Then, trajectory planning of the robot was studied via different techniques.

A position control of the robot arm using an active disturbance rejection control based (ADRC) on the extended state observer (ESO) was realized in continuous time then discretized to be programmed on a digital controller board. But before an attempt to realize the control of the robot using classical PD controller was done to compare its efficiency with ADRC's one.

After designing the controllers, both have been evaluated from their step responses which has shown clearly that the PD has a very poor efficiency and inadequate performance. While, ADRC control was successful in eliminating the error and improving the transient response.

The execution of a planned path by the ED-7220C robot arm under the ADRC control was achieved successfully with both the presence and the absence of obstacles.

A future work can include the use of ESO base ADRC on a more complex robotic system or the use of artificial intelligence based control techniques like Fuzzy logic and Neural networks for automatic tuning of the robot controller. And pass to the high-level programming of the robot manipulator using advanced algorithms for navigation, object recognition and picking-placing items.

References

References

[1] Seth Hutchinson, M. Vidysagar and Mark W. Spong, Robot Modeling and Control, New York / Chichester / Weinheim / Brisbane / Singapore / Toronto: JOHN WILEY & SONS, INC., 2005.

[2] Raza Ul Islam, Hamza Khan and Jamshed Iqbal, "Modeling and Analysis of a 6 DOF Robotic Arm Manipulator," *Canadian Journal on Electrical and Electronics Engineering* , vol. 3, no. 6, pp. 300 - 306, July 2012.

[3] S. M. Hammond, "Unit 1 The Engineered World," University Technical College Norfolk, Norwich, England, 2015.

[4] Sedar Kucuk, Zafer and Bingul, "Robot Kinematics: Forward and Inverse Kinematics," in *Industrial Robotics: Theory, Modelling and Control*, Germany, Pro Literatur Verlag, Germany / ARS, Austria, December 2006, pp. 117 - 148.

[5] Zexiang Li, S. Shankar Sastry and Richard M. Murray, A Mathematical Introduction to Robotic Manipulation, CRC Press, 1994.

[6] M. R. AbuQassem, "Simulation and Interfacing of 5 DOF Educational - Master thises," Islamic University of Gaza, Deanery of Graduate Studies, Faculty of Engineering, Electrical Engineering Department , Gaza , June 2010.

[7] E. Westervel, "Robot Modeling and Control," *IEEE CONTROL SYSTEMS MAGAZINE*, no. 12, pp. 113-115, 2006.

[8] Seth Hutchinson, M. Vidysagar and Mark W. Spong, Robot Dynamics and Control, New York: JOHN WILEY & SONS, INC, January 28, 2004.

[9] Bouamer Tarek and El Gheribi Abdallah Nabih, "Design Of A Digital Control System And Path Planning Module For The ED-7220C Robot Arm," IGEE_ex-INELEC, UMBB, Boumerdes, Algeria, June 2015.

[10] Ali Amin Rashidifar, Darvish Ahmadi and Mohammad Amin Rashidifar, "Modeling and Control of 5DOF Robot Arm Using Fuzzy Logic Supervisory Control," *International Journal of Robotics and Automation (IJRA)* , vol. 2, no. 2, pp. 56 - 68, 2013.

[11] R. B. Lukàs Barinka, "Inverse Kinematics - Basics Methods," Dept. of Computer Science & Engineering, Czech Technical University, Prague, Czech Republic.

[12] D. Mitrovic, "Learning Motor Control for Simulated Robot Arms," University Of Edinburgh, Edinburgh, Scotland, 2006.

References

[13] Boscariol, Lunzutti, R? Vidoni and A. Gasparetto, "Trajectory Planning in Robotics," *Mathematics in Computer Science*, vol. 30, no. 08, 2012.

[14] O. Chaoua and O. BELHADJ, "Design of a decoupling position control for the ED-7220C 5-axes robot," IGEE_ex-INELEC, UMBB, Boumerdes, Algeria, May 2016.

[15] V. Hlaváč, "Robot trajectory generation," Czech Technical University, Prague.

[16] J. J. Craig, "Trajectory generation," in *Introduction to Robotics, Mechanics and Control*, United States of America, Pearson Education International, 2005, pp. 201 - 229.

[17] Benguang Zhang, Min Wang and Diansheng Chen, "Cartesian Space Trajectory Planning on 7-DOF Manipulator," in *IEEE Conference on Robotics and Biomimetics*, Zhuhai, China, 2015.

[18] V. Santibanez, A. Loria and R. Kelly, Control of Robot Manipulators in Joint Space, London: Springer-Verlag, 2005, pp. 95-111.

[19] J. G. Zeigler and N. B. Nichols, "Optimum settings for automatic controllers," *Trans. ASME*, vol. 65, p. 433–444, 1943.

[20] Q. ZHENG, "ON ACTIVE DISTURBANCE REJECTION CONTROL: STABILITY ANALYSIS AND APPLICATIONS IN DISTURBANCE DECOUPLING CONTROL," CLEVELAND STATE UNIVERSITY, CLEVELAND, USA, July 2009.

[21] T. Yau, Z.Gao and D. YOO, "Optimal fast tracking observer bandwidth of the linear," *International Journal of Control*, vol. 80, no. 1, pp. 102-111, January 2007.

[22] Ed.co.kr., "Welcome to the education equipment website"(":::: ED □□□□ □□□□□ □□□□ □□□□□. ::::,") ED Corporation, [Online]. Available: http://www.ed.co.kr/eng/02_product/view.asp?topNum=2&pageNum=1&code=A05&idx=207&sproduct= . [Accessed 2 June 2017].

[23] "Arduino.cc," Arduino, [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardDue. [Accessed 2 June 2017].

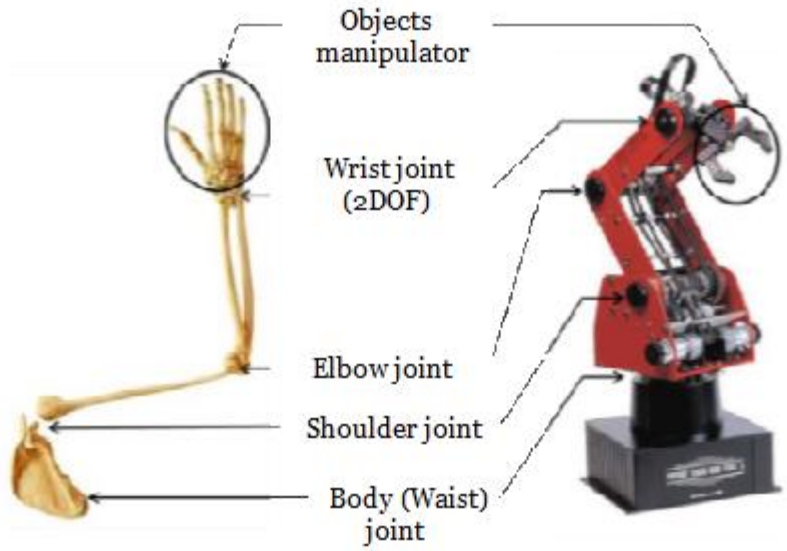
[24] A. Williams, Microcontroller projects using the Basic Stamp, Lawrence, Kan: CMP Books, 2002.

Appendices

Appendix A: Robot Description and Specification

1 ED-7220C Robot Arm

ED7220C Robot Arm developed by ED Corporation, Korea. The robotic arm has been extensively used in research, development and teaching. It is basically a serial [22]manipulator having all joints as revolute. The arm geometrical configuration is made up of waist, shoulder, elbow and wrist in correspondence with the human arm joints, FigureA.1. Each of these joints except the wrist has a single DOF. Wrist can move in two planes (roll and pitch), thus making the end-effector more flexible in terms of object manipulation [2].



FigureA.1. Joints configurations in ED-7220C [2].

Features	Description
Number of Joints	5 Joint ＋ Gripper
Construction	Vertical Articulated Arm
Position precision	±0.5mm
Movement Speed	100mm/s Max
Load Capacity	1 Kg
Opening Gripper	55 mm
Weight	33 Kg

TableA.1 Salient Features of ED-7220C [22].

Appendices

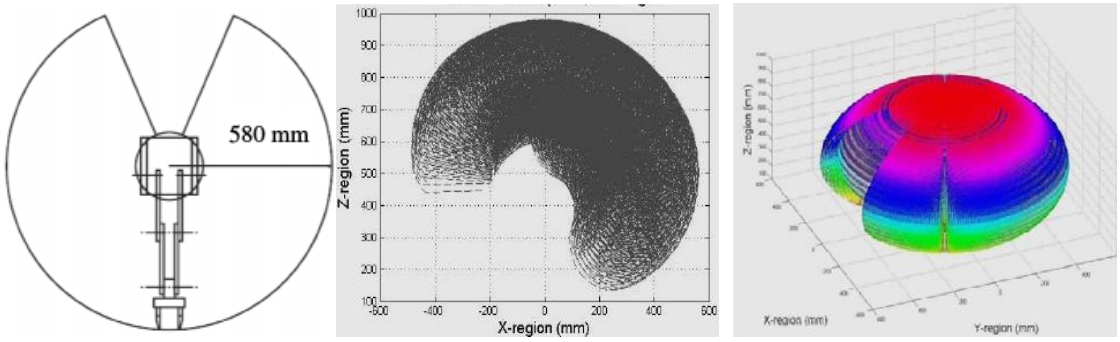
ED-7220C is fully actuated with each DOF achieved by a precise servo motor (DME 38B50G) rated at 24V and equipped with an optical encoder. The end-effector is a two-state gripper having rubber pads. The built in mechanical safety limits restrict the joint motion in case something in the control algorithm goes wrong [2].

Joint	Base	Shoulder	Elbow	Wrist	Gripper
Enc. Counts	0-15000	0-3700	0-3400	0-4500	0-1500

TableA.2 Reading of joint's encoder [22].

2 ED-7220C Workspace

The workspace of ED-7220C arm expresses its ability to reach a specific area. So, it is defined by the space that the robot can reach. Given the link lengths and the information about the range of motion (ROM) for each joint of the robot the workspace can be determined [2].



a. Workspace in XY **b.** Workspace in XZ **c.** 3D Workspace

FigureA.2. ED-7220C Workspace [2].

Base	Shoulder	Elbow	Wrist Pitch	Wrist Roll
310°	+130°/-35°	±130°	±130°	360°

TableA.3. ED-7220C Range of motion (ROM) **[22]**.

Appendix B : Trigonometric Identities

- Difference and sum of angles:

$$\cos(\theta_1 + \theta_2) = \cos\theta_1 \cos\theta_2 - \sin\theta_1 \sin\theta_2 = C_{12}$$

$$\cos(\theta_1 - \theta_2) = \cos\theta_1 \cos\theta_2 + \sin\theta_1 \sin\theta_2$$

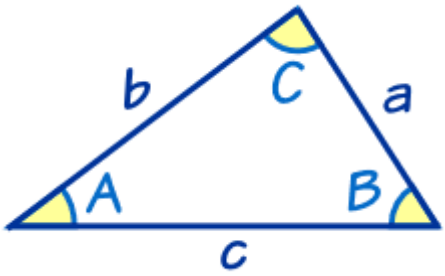
$$\cos(\theta_1 + \theta_2 + \theta_3) = C_{123}$$

$$\sin(\theta_1 + \theta_2) = \cos\theta_1 \sin\theta_2 + \sin\theta_1 \cos\theta_2 = S_{12}$$

$$\sin(\theta_1 - \theta_2) = \cos\theta_1 \sin\theta_2 - \sin\theta_1 \cos\theta_2$$

$$\sin(\theta_1 + \theta_2 + \theta_3) = S_{123}$$

- Law of cosines:



FigureB.1. Triangle

$$c^2 = a^2 + b^2 - 2ab\cos C$$

Which can also be re-arranged to:

$$\cos C = \frac{a^2 + b^2 - c^2}{2ab}$$

Appendix C: Components of the Inertia Matrix M(q) and Potential Energy Term E(q)

$$A(q) = m_2 d_2^2 \sin^2(q_2) + m_3(l_2 \sin(q_2) + d_3 \sin(q_2 + q_3))^2 + m_4(l_2 \sin(q_2) + l_3 \sin(q_2 + q_3) + d_4 \sin(q_2 + q_3 + q_4))^2$$

$$B(q) = m_2 d_2^2 + m_3[(l_2 + d_3 \cos(q_3))^2 + (d_3 \sin(q_3))^2] + m_4[(l_2 + l_3 \cos(q_3) + d_4 \cos(q_3 + q_4))^2 + (l_3 \sin(q_3) + d_4 \sin(q_3 + q_4))^2]$$

$$C(q) = m_3 d_3^2 + m_4[(l_3 + d_4 \cos(q_5))^2 + (d_4 \sin(q_4))^2]$$

$$D(q) = m_4 d_4^2$$

$$E(q) = m_1 d_1 + m_2(l_1 + d_2 \cos(q_2)) + m_3(l_1 + l_2 \cos(q_2) + d_3 \cos(q_2 + q_3)) + m_4(l_1 + l_2 \cos(q_2) + l_3 \cos(q_2 + q_3) + d_4 \cos(q_2 + q_3 + q_4))$$

Where:

m_i : is the i[th] joint mass (along with its corresponding link).

l_i : is the i[th] link length.

d_i : is the i[th] joint distance between the starting of the link and the center of mass.

q_i : is the i[th] joint angle of rotation.

Appendix D: Proof of the convergence of the ESO [20]

We have the system represented by its state space:

$$\begin{gathered}\dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \vdots \\ \dot{x}_n = x_{n+1} + bu \\ \dot{x}_{n+1} = h\end{gathered}$$

And the ESO is modeled by this state space:

$$\begin{gathered}\dot{z}_1 = z_2 - \beta_1(z_1 - y) = z_2 - \omega_0\alpha_1(z_1 - y) \\ \dot{z}_2 = z_3 - \omega_0^2\alpha_2(z_1 - y) \\ \vdots \\ \dot{z}_n = z_{n+1} - \omega_0^n\alpha_n(z_1 - y) + bu \\ \dot{z}_{n+1} = -\omega_0^{n+1}{}_0\alpha_{n+1}(z_1 - y)\end{gathered}$$

Putting $\hat{x}_i = x_i - z_i$ we will have:

$$\begin{gathered}\dot{\hat{x}}_1 = \hat{x}_2 - \omega_0\alpha_1\hat{x}_1 \\ \dot{\hat{x}}_2 = \hat{x}_3 - \omega_0^2\alpha_2\hat{x}_1 \\ \vdots \\ \dot{\hat{x}}_n = \hat{x}_{n+1} - \omega_0^n\alpha_n\hat{x}_1 + bu \\ \dot{\hat{x}}_{n+1} = h - \omega_0^{n+1}{}_0\alpha_{n+1}\hat{x}_1\end{gathered}$$

Then putting $\varepsilon_i = \frac{\hat{x}_i}{\omega_0^{i-1}}$

$$\begin{gathered}\varepsilon_1 = \omega_0(\varepsilon_2 - \alpha_1\varepsilon_1) \\ \varepsilon_2 = \omega_0(\varepsilon_3 - \alpha_2\varepsilon_1) \\ \vdots \\ \varepsilon_n = \omega_0(\varepsilon_{n+1} - \alpha_n\varepsilon_1) \\ \varepsilon_{n+1} = \frac{h}{\omega_0^n} - \alpha_n\varepsilon_1\end{gathered}$$

$$\dot{\varepsilon} = \begin{bmatrix}\dot{\varepsilon}_1 \\ \dot{\varepsilon}_2 \\ \vdots \\ \dot{\varepsilon}_{n+1}\end{bmatrix} = \omega_0 A\varepsilon + B\frac{h}{\omega_0^n}$$

Which has a solution:

$$\varepsilon(t) = e^{\omega_0 At}\varepsilon(0) + N(t)$$

Where:

$$N(t) = \int_0^t [e^{\omega_0 A(t-\tau)}B\frac{h(\tau)}{\omega_0^n}]d\tau$$

Appendices

Now, if $h(t, \omega, x, \dot{x}, u)$ is bounded (i.e. $h < \delta$) then:

$$|N_i(t)| \leq \delta \int_0^t \frac{[e^{\omega_0 A(t-\tau)} B]_i}{\omega_0^n} d\tau$$

$$|N_i(t)| \leq \frac{\delta}{\omega_0^n} [|A^{-1}B|_i + |(A^{-1} e^{\omega_0 A t} B)_i|]$$

$$A^{-1} = \begin{bmatrix} 0 & 0 & 0 & \cdots & -\frac{1}{\alpha_{n+1}} \\ 1 & 0 & 0 & \cdots & -\frac{\alpha_1}{\alpha_{n+1}} \\ 0 & 1 & 0 & \cdots & -\frac{\alpha_2}{\alpha_{n+1}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & -\frac{\alpha_n}{\alpha_{n+1}} \end{bmatrix} \; ; \; B = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

So:

$$|A^{-1}B_i| = \begin{cases} \frac{1}{\alpha_{n+1}} & i = 1 \\ \frac{\alpha_{i-1}}{\alpha_{n+1}} & i = \overline{2, n+1} \end{cases} \leq v$$

$$v = max\left\{\frac{1}{\alpha_{n+1}}, \frac{\alpha_{i-1}}{\alpha_{n+1}}\right\}$$

Since A is Hurwitz ($\omega_0$is strictly positive) $\exists\, T_1 > 0$ such that $\left|[e^{\omega_0 A t}]_{ij}\right|$ for all $t \geq T$.

Assuming:

$$A^{-1} = \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1,n+1} \\ S_{21} & S_{22} & & S_{2,n+1} \\ \vdots & & \ddots & \vdots \\ S_{n+1,1} & S_{n+1,2} & \cdots & S_{n+1,n+1} \end{bmatrix}$$

$$e^{\omega_0 A t} = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1,n+1} \\ d_{21} & d_{22} & & d_{2,n+1} \\ \vdots & & \ddots & \vdots \\ d_{n+1,1} & d_{n+1,2} & \cdots & d_{n+1,n+1} \end{bmatrix}$$

Then:

$$|(A^{-1} e^{\omega_0 A t} B)_i| = \left|S_{i,1} d_{1,n+1} + \cdots + S_{i,n+1} d_{n+1,n+1}\right|$$

$$|(A^{-1} e^{\omega_0 A t} B)_i| \leq \frac{\left|S_{i,1}\right| + \left|S_{i,2}\right| + \cdots + \left|S_{i,n+1}\right|}{\omega_0^{n+1}}$$

Appendices

$$|S_{i,1}| + |S_{i,2}| + \cdots + |S_{i,n+1}| = \begin{cases} \dfrac{1}{\alpha_{n+1}} & i = 1 \\ 1 + \dfrac{\alpha_{i-1}}{\alpha_{n+1}} & i = \overline{2, n+1} \end{cases}$$

So:

$$|(A^{-1}e^{\omega_0 At}B)_i| \leq \frac{u}{\omega_0^{n+1}}$$

$$u = max\left\{\frac{1}{\alpha_{n+1}}, 1 + \frac{\alpha_{i-1}}{\alpha_{n+1}}\right\}$$

Finally, we obtain:

$$|P(t)| = \frac{\delta v}{\omega_0^{n+1}} + \frac{\delta u}{(\omega_0^{n+1})^2}$$

In other hand:

$$|[e^{\omega_0 At}\varepsilon(0)]_i| = \left|d_{i,1}\varepsilon_1(0) + d_{i,1}\varepsilon_2(0) + \cdots + d_{i,n+1}\varepsilon_{n+1}(0)\right|$$

$$|[e^{\omega_0 At}\varepsilon(0)]_i| \leq \frac{|\varepsilon_1(0)| + |\varepsilon_2(0)| + \cdots + |\varepsilon_{n+1}(0)|}{\omega_0^{n+1}}$$

$$|[e^{\omega_0 At}\varepsilon(0)]_i| \leq \frac{\varepsilon_{sum}(0)}{\omega_0^{n+1}}$$

Hence:

$$|\varepsilon_i(t)| \leq \frac{\varepsilon_{sum}(0) + \delta v + \dfrac{\delta u}{\omega_0^{n+1}}}{\omega_0^{n+1}}$$

$$|\hat{x}_i(t)| = |\varepsilon_i(t)|\omega_0^{i-1}$$

So:

$$|\hat{x}_i(t)| \leq \left|\frac{x_{sum}(0)}{\omega_0^{n+1}}\right| + \frac{\delta v}{\omega_0^{n-i+2}} + \frac{\delta u}{\omega_0^{2n-i+3}}$$

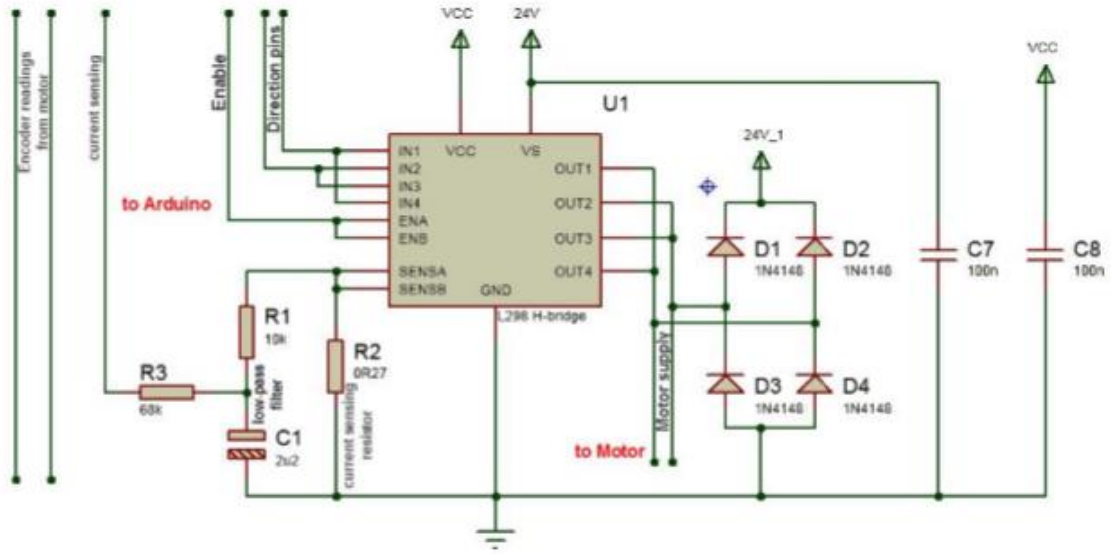
Hence:

$$|x_i(t)| \leq \sigma_i$$

Remark: The larger is ω_0; the smaller is σ_i; the more accurate will be the ESO estimation.

Appendix E: Drive Circuit Design

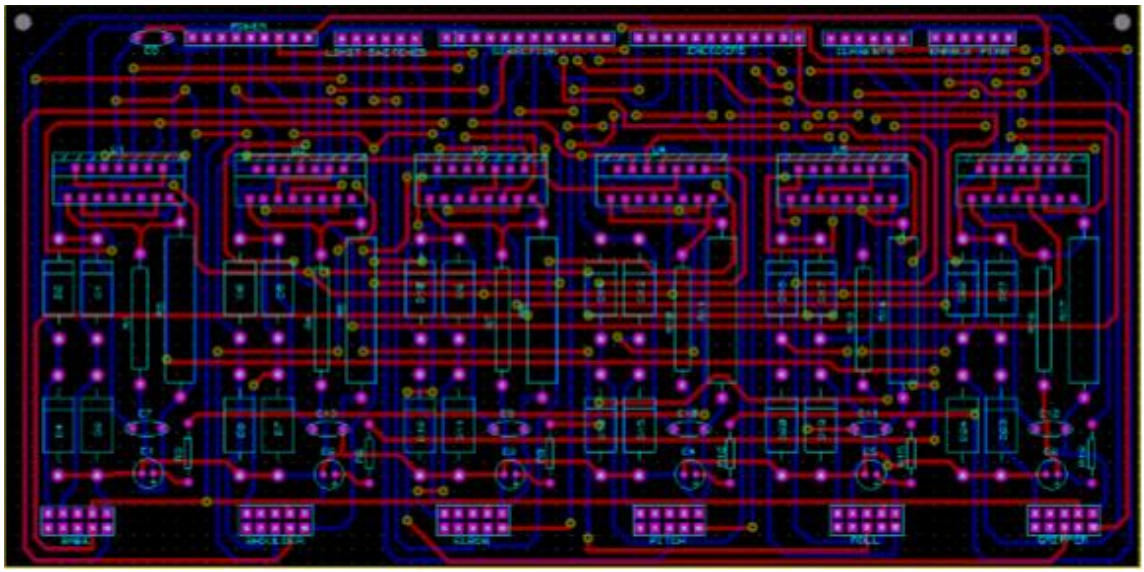
The drive circuit is designed to supply power for 6 actuators; this circuit is composed of 6 identical sub-circuits, each one of them supplies power one actuator separately [14].



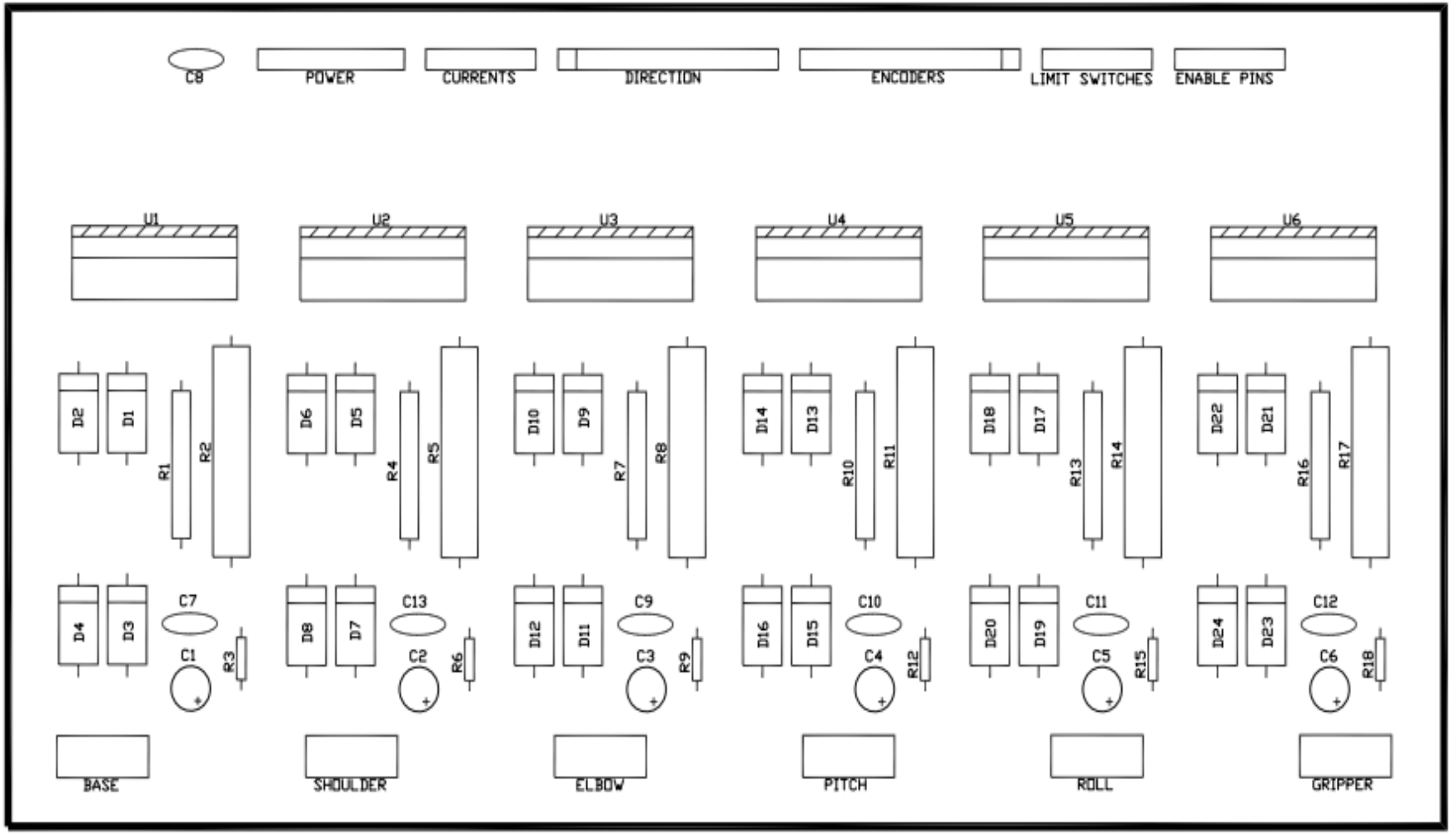
FigureE.1. Drive circuit for one actuator [14].

A small resistor is used to sense the current that is supplied to the motor, the voltage across this resistor is filtered and a high resistor is connected after the filter in order to reduce the current amplitude fed to the Arduino [14].

This design was made using Proteus software, it is used to design and simulate the circuit and also to generate a Printed Circuit Board layout file used to create the drive circuit [14]:



FigureE.2. Printed Circuit Board layout [14].



FigureE.3. Drive Circuit Schematic.

Appendix F: The Main Equipments

1 Arduino DUE Microcontroller

The Arduino Due is a microcontroller board based on the Atmel SAM3X8E ARM Cortex-M3 CPU. It is the first Arduino board based on a 32-bit ARM core microcontroller that made programmable through the familiar Arduino IDE [23].

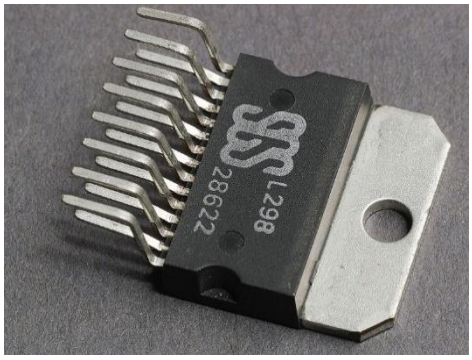
FigureF.1. Arduino DUE Microcontroller [23].

Arduino DUE Microcontroller's Features [23]:

Microcontroller	AT91SAM3X8E
Operating Voltage	3.3V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-16V
Digital I/O Pins	54 (of which 12 provide PWM output)
Analog Input Pins	12
Analog Outputs Pins	2 (DAC)
Total DC Output Current on all I/O lines	130 mA
DC Current for 3.3V Pin	800 mA
DC Current for 5V Pin	800 mA
Flash Memory	512 KB all available for the user applications
SRAM	96 KB (two banks: 64KB and 32KB)
Clock Speed	84 MHz
Length	101.52 mm
Width	53.3 mm
Weight	36 g

2 The L298 H-Bridge

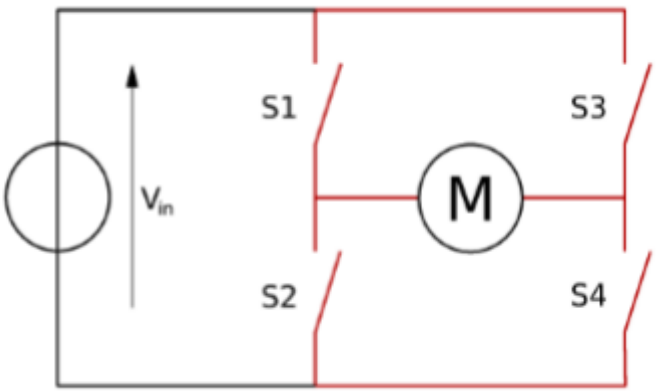
L298 H-bridge is an Integrated circuit used in electronic control of high current load around 4 amperes. These circuits are often used in robotics and other applications to allow DC motors to run forwards and backwards [24].



FigureF.2. L298 H-Bridge IC [24].

L298 H-bridge can control two motors its arrangement is generally used to reverse the polarity/direction of the motor, but can also be used to 'brake' the motor, where the motor comes to a sudden stop, as the motor's terminals are shorted, or to let the motor 'free run' to a stop, as the motor is effectively disconnected from the circuit. The following figure summarizes operation, with S1-S4 corresponding to the diagram [24].

S1	S2	S3	S4	Result
1	0	0	1	Motor moves right
0	1	1	0	Motor moves left
0	0	0	0	Motor free runs
0	1	0	1	Motor brakes
1	0	1	0	Motor brakes
1	1	0	0	Short Power Supply
0	0	1	1	Short Power Supply
1	1	1	1	Short Power Supply



FigureF.3. H-bridge structure and its working principle [24].

As we have seen in Appendix E, we have used six H-Bridges to control the six DC motors of the ED-7220C Robot Arm.