ROBOTICS CONTROL USING ACTIVE DISTURBANCE REJECTION CONTROL

OUSAMA SAID KHAIRALLAH

Bachelor of Science in Electrical Engineering

Cleveland State University

May, 2007

submitted in partial fulfillment of requirements for the degree

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

CLEVELAND STATE UNIVERSITY

December, 2009

© Copyright by Ousama Said Khairallah 2009

This thesis has been approved

By the Department of Electrical and Computer Engineering

And the College of Graduate Studies

_____

Thesis Committee Chairperson, Dr. Charles Alexander

_____

Department/Date

_____

Dr. Zhiqiang Gao

_____

Department/Date

_____

Dr. Lili Dong

_____

Department/Date

DEDICATION

I dedicate this thesis and all of my work to my family: my parents, my brothers Johnny and Bassam, my sisters May, Fidaa, and Maha, and especially to my father Mr. Said Hanna Attah Khairallah. I thank you for all the support. You have emphasized the importance of academia and said that an education is a weapon that is sharper than a sword and greater than an atomic bomb; with an education, I can build a future like no other. With that in mind, you are the reason that I was able to finish my masters thesis.

Also I would like to thank every one that supported me mentally and emotionally throughout my journey, especially my brother Bassam who always stood beside me, no matter what, no matter where. I just want you to know that I love you and I will always remember what you have done for me.

Maha, you are the reason I smiled when things got bad, the reason that life was much better when the world was down. I love you, sister, and I thank you for all the support that you have given me throughout the years.

Keeping the best for last, I dedicate this thesis to my mother, the greatest mother in the world; the mother that would never stop caring, the mother that never stops worrying: my world, my every thing, my mother Madelen Khairallah. You are and always will be the best and the greatest mother in the universe. I love you.

ACKNOWLEDGEMENT

Dr. Charles Alexander I would like to thank you for your support and help. It has been a great journey, working with you side-by-side during my college career. You have always supplied me with the mortar to hold my ideas together and helped me to develop the tools I needed to become a stronger, better, more mature human being and a great engineer. I came to this world with a sole purpose: to make a difference. With your help and support, I know that I now have a strong foundation to do so. I thank you again and wish you the best in the future.

Dr. F. Eugenio Villaseca, it has been a privilege knowing you. I have grown to share your passion and love of the electrical engineering field. I know that the knowledge and education that you helped me to obtain is priceless. I thank you again for all your help and support; and I give you the best wishes from my heart.

ROBOTIC'S CONTROL USING ACTIVE DISTURBANCE REJECTION CONTROL

OUSAMA SAID KHAIRALLAH

ABSTRACT

Conventional robotics control has been set in stone since the sixties. The world has been waiting too long for a new age of control to change the world of Robotics. Active Disturbance Rejection Control (ADRC) is a newly reformed Control methodology. It has been used, in very limited applications, as a replacement for PID control. In this thesis, I will cover the different aspects of the kinematics and dynamics of a robotic manipulator. I will also examine the feasibility of using ADRC to control a robotic manipulator.

To explain ADRC, a simple example that demonstrates the concepts and theory of Active Disturbance Rejection Control will be discussed. Using this example, the establishment of relevance to the mathematical module; of a rotary prismatic robotic manipulator will be accomplished. A control system for the module using Matlab software and mathematical computations will be implemented.

TABLE OF CONTENTS

ABSTRACT vi

LIST OF TABLES x

LIST OF FIGURES xi

LIST OF EQUATIONS xiii

CHAPTER I INTRODUCTION AND HISTORICAL BACKGROUND 1

1.1 Historical Overview 1

1.2 Definitions 3

CHAPTER II FORWARD KINEMATICS FOR MULTI-LINK ROBOTS 4

2.1 Forward Kinematics 4

2.2 Skew Matrices 5

2.3 Derivation of the Jacobian 7

2.4 Joint types 8

2.4.1 Type 1: Prismatic Joints 8

2.4.2 Type 2: Revolute Joints 8

2.4.3 Combining the Linear and Angular Velocity Jacobians 9

CHAPTER III DYNAMICS OF MULTI-LINK ROBOTS 10

3.1 Robot Dynamics 10

3.2 Kinetic and Potential Energy 11

3.2.1 Multiple Link Robot Kinetic Energy 12

3.3 Potential Energy for a Multi-Link Robot Manipulator 13

3.4 Possible Design Problem 16

CHAPTER IV ACTIVE DISTURBANCE REJECTION CONTROL 18

4.1 Introduction to ADRC..... 18

4.2 Extended State Observer..... 20

CHAPTER V MATHEMATICAL MODELING..... 23

5.1 Mathematical Model of an RP Manipulator 23

5.2 Forward Kinematics and DH Convention for an RP Manipulator 23

5.3 Jacobian Matrices for RP manipulator…….26

CHAPTER VI IMPLEMENTING ADRC CONCEPT IN THE RP MANIPULATOR .. 31

6.1 State Observer Design..... 33

CHAPTER VII SIMULATIONS..... 36

7.1 Simulations without disturbances or noise 36

7.2 Simulations with Feed Forward 40

7.3 Adding Disturbance to the System 45

7.4 Adding Noise to the System Simulation of Sensor Noise 49

7.5 Robust-Passivity Based Control Simulation…….54

CHAPTER VIII EVALUATION OF THE SIMULATION RESULTS 56

8.1 Evaluation of the Ideal System Simulation:..... 56

8.2 Evaluation of the System after Adding Feed Forward..... 56

8.3 Evaluation of the System after Adding External & Internal Disturbances:. 57

8.4 Evaluation of the System after Introducing Sensor Noise:..... 57

8.5 Evaluation of a Robust-Passivity Based control with Sensor Noise: 60

8.6 Active Disturbance Rejection Control vs. Robust-Passivity Based Control.61

CHAPTER IX CONCLUDING REMARKS 62

BIBLIOGRAPHY 65

APPENDIX 66

A. Derivation of Two-Link Manipulator State 66

LIST OF TABLES

Table I DH Parameter Table 24

LIST OF FIGURES

Figure 1. Joint Link System Drawing 24

Figure 2. System Setup in Simulink 36

Figure 3. Output Tracking at a Frequency of 100 without Feed Forward 37

Figure 4. Output Tracking at a Frequency of 200 No Feed Forward 37

Figure 5. Output Tracking at a Frequency of 300 No Feed Forward 38

Figure 6. Output Tracking at a Frequency of 400 No Feed Forward 38

Figure 7. Output Tracking at a Frequency of 500 No Feed Forward 39

Figure 8. Output Tracking at a Frequency of 1000 No Feed Forward 39

Figure 9. Output Tracking at a Frequency of 10000 No Feed Forward 40

Figure 10. System Setup with Feed Forward 41

Figure 11. Output Tracking at a Frequency of 100 with Feed Forward 41

Figure 12. Output Tracking at a Frequency of 200 with Feed Forward 42

Figure 13. Output Tracking at a Frequency of 300 with Feed Forward 42

Figure 14. Output Tracking at a Frequency of 400 with Feed Forward 43

Figure 15. Output Tracking at a Frequency of 500 with Feed Forward 43

Figure 16. Output Tracking at a Frequency of 1000 with Feed Forward 44

Figure 17. Output Tracking at a Frequency of 10000 with Feed Forward 44

Figure 18. System Setup with Sinusoidal Disturbance 45

Figure 19. Output Tracking at a Frequency of 100 with Disturbance 46

Figure 20. Output Tracking at a Frequency of 200 with Disturbance 46

Figure 21. Output Tracking at a Frequency of 300 with Disturbance 47

Figure 22. Output Tracking at a Frequency of 400 with Disturbance 47

Figure 23. Output Tracking at a Frequency of 500 with Disturbance 48

Figure 24. Output Tracking at a Frequency of 1000 with Disturbance 48

Figure 25. Output Tracking at a Frequency of 100000 with Disturbance 49

Figure 26. System Setup with Additional Noise..... 50

Figure 27. Output Tracking at a Frequency of 100 with Noise 50

Figure 28. Output Tracking at a Frequency of 200 with Noise 51

Figure 29. Output Tracking at a Frequency of 300 with Noise 51

Figure 30. Output Tracking at a Frequency of 400 with Noise 52

Figure 31. Output Tracking at a Frequency of 500 with Noise 52

Figure 32. Output Tracking at a Frequency of 1000 with Noise 53

Figure 33. Output Tracking at a Frequency of 1000 with Noise 53

Figure 34: Robust-Passivity Based Control of an RP manipulator……55

Figure 35: Using Robust-Passivity Based Control with noise……..……56

Figure 36: Robust-Passivity Based control Zoom in with noise……56

Figure 37. Zoom in of the Output Tracking at a Frequency of 1000 with Noise 58

Figure 38. Zoom in of the Output Tracking at a Frequency of 10000 with Noise 59

Figure 39. Output Tracking at a Frequency of 600 with Noise 59

Figure 40. Zoom in of the Output Tracking at a Frequency of 600 with Noise 60

LIST OF EQUATIONS

(2.1) 5

(2.2) 6

(2.3) 6

(2.4) 6

(2.5) 7

(2.6) 7

(2.7) 7

(2.8) 8

(2.9) 8

(2.10) 8

(2.11) 8

(2.12) 8

(2.13) 9

(2.14) 9

(3.15) 10

(3.16) 11

(3.17) 11

(3.18) 11

(3.19) 12

(3.20) 12

(3.21) 12

(3.22) 12
(3.23) 13
(3.24) 13
(3.25) 13
(3.26) 13
(3.27) 14
(3.28) 14
(3.29) 14
(3.30) 14
(3.31) 15
(3.32) 15
(3.33) 15
(3.34) 15
(3.35) 16
(3.36) 16
(3.37) 16
(4.38) 18
(4.39) 19
(4.40) 19
(4.41) 19
(4.42) 20
(4.43) 20
(4.44) 20

(4.45) 21
(4.46) 21
(4.47) 21
(4.48) 21
(4.49) 22
(5.50) 23
(5.51) 25
(5.52) 25
(5.53) 26
(5.54) 26
(5.55) 26
(5.56) 27
(5.57) 27
(5.58) 27
(5.59) 28
(5.60) 28
(5.61) 28
(5.62) 29
(6.63) 31
(6.64) 32
(6.65) 32
(6.66) 33
(6.67) 33

(6.68) 34

(6.69) 35

CHAPTER I

INTRODUCTION AND HISTORICAL BACKGROUND

1.1 Historical Overview

Robots have been in place since the Golden age of Greece when mankind's understanding of the laws of motion first started to evolve. Robots incorporated the structure of these laws and gave these theoretical laws a physical manifestation. In the year 270 BC, a Greek engineer named Ctesibus created organs and water clocks with movable fingers [1]. In 1921 the word "robot" was used in a play called Rossum's Universal Robots written by Czech writer Karel Capek [1]. Twenty years later, in 1941, Isaac Asimov a renowned science fiction writer, used the term "robotics" to describe the technology of robots and expected the rise of the robot industry [2]. One year later Asimov brought the laws of robotics to life in his story "Runaround," in which he defined the "Three Laws of Robotics":

- A robot shall not injure a human or put a human in harms way.

- A robot must obey the orders of humans as long no conflict will occur with law number one.
- A robot must protect its own existence as long no conflict will be done with the First or Second Law [1].

Six years later Norbert Wiener published a book regarding "Cybernetics" that influenced artificial intelligence research [2]. Finally, eight years later in 1956, the first robotics company in the world was established by George Devol and Joseph Engelberger [1].

The robotic age as we know it today began in 1961when the first industrial robot, called UNIMATE [1], was put to work in the General Motors automobile company in New Jersey. The first artificial robotic arm controlled by a computer was implemented in 1963. Called the Rancho Arm [2], this robotic arm had six joints that were used to obtain the flexibility of the human arm.

Robots continued to trickle gradually into the industrial world until the year 1974, when robotics control took a big leap forward. A small robotic arm (the Silver Arm) was designed to performed small-parts assembly using feedback from touch and pressure sensors [1]. This robot marked the beginning age of feedback control in the robotics world.

In 1979 a TV camera was mounted on a cart that crossed a room filled with chairs without human assistance. The cart camera took pictures in the room and fed them back to the computer, which determined the distance between the cart and the chairs and defined the path of motion. This was the beginning of the use of machine vision with robots [1].

Robots continued to evolve through the seventies, taking the shape that we see them in now. As the robot has evolved, so also robotic control systems have evolved to accommodate the changes that industrial demand has brought to this technology.

1.2 Definitions

A robot is a reprogrammable, multifunctional manipulator designed to move objects, parts, tools, or specialized devices through variable programmed motions for the performance of a variety of tasks [3].

A robotic manipulator is a composition of links that are connected to each other by a set of joints. There are two different types of joint configurations: the rotary (Revolute) joint and the liner (Prismatic) joint. The combination of multiple sets of links and joints is what defines the Kinematic chain [3]. A Revolute joint allows travel in a rotary motion relative to a plane between two links; it is known as an (R) type joint. A Prismatic joint allows travel in a linear fashion with respect to a plane between two links; it is known as a (P) type joint.

In this thesis project, a robot manipulator of the type RP will be evaluated and developed as the system to be controlled using the Active Disturbance Rejection Control system. We will start with the development of the mathematical model; then obtain the control system and the simulations. To obtain such a model, we must first evaluate the the forward kinematics of multi-link robotic manipulator.

CHAPTER II

FORWARD KINEMATICS FOR MULTI-LINK ROBOTS

2.1 Forward Kinematics

Forward kinematics for a solid robot can be defined as the relationship between the joints of the robot and the position and orientation of the end effector. The variables in the robot systems are the rotation angles in the case of rotational joint design and the extending of the link in the prismatic joint design.

To develop the forward kinematics of a robot manipulator, we need to set up a set of restriction rules to eliminate any hardship during the analysis of the forward kinematics. This convention or set of rules is called "The Denavit-Hartenberg Convention". The Denavit-Hartenberg Convention, known as the DH convention, is a set of rules that can be used to select the frames of reference in robotics application. Without this set of rules the robot frame of reference could get very complicated. The convention pertains in multiples of transformation A_i shown as the product of four basic transformations:

$$
\begin{aligned}
A_i &= Rot_{z,\theta i} Trans_{z,di} Trans_{x,ai} Rot_{x,\alpha i} \\
&= \begin{bmatrix} c_{\theta i} & -s_{\theta i} & 0 & 0 \\ s_{\theta i} & c_{\theta i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha i} & -s_{\alpha i} & 0 \\ 0 & s_{\alpha i} & c_{\alpha i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} c_{\theta i} & -s_{\theta i} c_{\alpha i} & s_{\theta i} s_{\alpha i} & a_i c_{\theta i} \\ s_{\theta i} & c_{\theta i} c_{\alpha i} - c_{\theta i} s_{\alpha i} & & a_i s_{\theta i} \\ 0 & s_{\alpha i} & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
\tag{2.1}
$$

Where X axis and the Z axis are the only permitted axes of operation, no action would be done on the Y axis. The variables and the constant terms are defined as follows: *θ_i is the joint angle (variable), α_i is the link twist, d_i is the link offset (variable), a_i is the link length. All the terms above are for a specific link and joint; in this case defined as link(i) and joint(i).*

Now that we have introduced the forward kinematics and related the Cartesian position and orientation of the end effector to the position of the joints, it is time to address the relationship between the end effector velocities and the joints velocities. This relationship is defined in a set of matrices that is called the Jacobian.

A Jacobian matrix is a set of first order partial derivatives of a system. This matrix describes the derivatives of any scalar function. In this case, the concentration is on the general form of the Jacobian matrix for a robotic manipulator. This form of Jacobian is defined for an n-link robot configuration for a set of linear and angular velocities.

2.2 Skew Matrices

We also need to take into consideration the Skew Symmetric matrices, which will be useful in future, analysis of the Jacobian matrix. The skew symmetric matrices are

helpful in developing the rotation matrices that would be used in the derivation of the velocity matrix relating the coordinate systems, respectively.

For an n x n matrix, S is defined as the skew matrix if and only if $S^T + S = 0$.

So for a 3x3 skew matrix it will have the following form:

$$S = \begin{bmatrix} 0 & -s_3 & s_2 \\ s_3 & 0 & -s_1 \\ -s_2 & s_1 & 0 \end{bmatrix} \tag{2.2}$$

where all the diagonal values are zeros as shown and the symmetry is consistent and for a unit vector i , j , k , respectively,

$$i = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad j = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad k = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.3}$$

The skew symmetric matrices are shown below, respectively,

$$S(i) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \quad S(j) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad S(k) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.4}$$

Some of the useful skew symmetric matrices properties are listed below:

1. The linear property: $S(ax+by) = aS(x) + bS(y)$

2. For any two vectors a, b the vector cross product: $a \times b = S(a)b$

3. For any $R \in SO(3)$ *and* $a \in R^3$ *where SO(3) is the 3x3 skew symmetric matrix and* R^3 *is a 3x3 rotational matrix*
$RS(a)R^T = S(Ra)$

4. For any n x n skew symmetric matrix S and any vector $Y \in R^n$: $Y^T SY = 0$

2.3 Derivation of the Jacobian

Let us derive the Jacobian for an n-link robot with joint variables $q_1, q_2 \ldots, q_n$

So

$$T_n^0(q) = \begin{bmatrix} R_n^0(q) & o_n^0(q) \\ 0 & 1 \end{bmatrix} \tag{2.5}$$

where T_n^0 *is the transformation matrix,* R_n^0 *is the orientation transformation,* o_n^0 *is the end-effector position; and* q_i *represents the joint variables. All the variables are a function of time as the robot changes position.*

Let ω_n^0 *be the angular velocity of the end effector and* v_n^0 *be the linear velocity of the end effector.*

Let the linear velocity be the derivative of the end effector position.

$$v_n^0 = \dot{o}_n^0 = \sum_{i=1}^{n} \frac{\partial o_n^0}{\partial q_i} \Rightarrow Jv_i = \frac{\partial o_n^0}{\partial q_i} \tag{2.6}$$

Let (p_i *) be a point on the ridged frames dependent on (i) and* z_{i-1}^0 *be the axis of rotation.*

$$\omega_n^0 = p_1 \dot{q}_1 z_1 + p_2 \dot{q}_2 R_1^0 z_2 + \ldots\ldots\ldots\ldots + p_n \dot{q}_n R_{n-1}^0 z_{i-1} = \sum_{i-1}^{n} p_i \dot{q}_i z_{i-1}^0 \tag{2.7}$$

2.4 Joint types

As discussed in the Introduction, there are two types of joints in robotic manipulators: Prismatic joints and Revolute joints. Let us examine these in more detail.

2.4.1 Type 1: Prismatic Joints

In this type of joint the motion is a totally linear translation. This type of motion is always parallel to the axis z_{i-1} and the magnitude is the translation $\dot{d}_i$, where d_i is a variable defined in the DH convention.

So

$$\dot{o}_n^0 = \dot{d}_i \, R_{i-1}^0 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \dot{d}_i \, z_{i-1}^0 \tag{2.8}$$

The case of prismatic joint Jvi is shown below:

$$Jv_i = z_{i-1} \tag{2.9}$$

2.4.2 Type 2: Revolute Joints

For a revolute joint (i) we set one joint to be moving and all the other joints to be fixed; the linear velocity of the end effector is the cross product of the angular velocity and the distance between the two centers of mass or two joints, dependent on the configuration.

$$\omega = \dot{\theta}_i \, z_{i-1} \tag{2.10}$$

$$r = o_n - o_{i-1} \tag{2.11}$$

$$Jv_i = \omega \times r \Rightarrow Jv_i = z_{i-1} \times (o_n - o_{i-1}) \tag{2.12}$$

2.4.3 Combining the Linear and Angular Velocity Jacobians

In this section we can summarize the Jacobian for the linear and angular velocities for each of the revolute and the prismatic joints.

$$\begin{aligned} &\textit{The upper half of the jacobian is shown as follows}: \\ &Jv_i = \begin{cases} z_{i-1} \times (o_n - o_{i-1}) & \textit{for revolute joint } i \\ z_{i-1} & \textit{for prismatic joint } i \end{cases} \end{aligned} \tag{2.13}$$

$$\begin{aligned} &\textit{The lower half of the jacobian is shown as follows}: \\ &J\omega_i = \begin{cases} z_{i-1} & \textit{for revolute joint } i \\ 0 & \textit{for prismatic joint } i \end{cases} \end{aligned} \tag{2.14}$$

After describing the Forward Kinematics of a multi-link manipulator, we can move into a new section defining the robot dynamics. During this analysis the mathematical model will be described in terms of energy characteristics and we will define the connection between the forward kinematics and the robot dynamics. The Jacobian that was formulated in the chapter above will be used as the linking chain in the next chapter.

CHAPTER III

DYNAMICS OF MULTI-LINK ROBOTS

3.1 Robot Dynamics

In the previous sections, kinematics modeling covered only the motion aspect of the robotic manipulator without taking into consideration any effect of forces or torques on the motion of the robot. Robot dynamics is useful in the analysis of forces and torques on the motion of robot manipulators. A set of dynamic equations will be used to construct the representation of robot dynamics. These dynamic equations are called Euler-Lagrange equations; that is, the Euler-Lagrange equations provide an energy dynamic approach to any mechanical system that could be used to analyze the system for any force or torque effect on the system. For simplicity, the Lagrangian is developed and shown for a simple one degree of freedom system as follows: From Newton's Second Law for a mass m that moves in the vertical direction,

$$m\ddot{y} = f - mg$$

and the left hand side of the equation can be written as follows:

$$m\ddot{y} = \frac{d}{dt}(m\dot{y}) = \frac{d}{dt}\frac{\partial}{\partial \dot{y}}\left(\frac{1}{2}m\dot{y}^2\right) = \frac{d}{dt}\frac{\partial k}{\partial \dot{y}} \qquad \textbf{(3.15)}$$

and $k = \frac{1}{2}m\dot{y}^2$ *in the case of kinetic energy*

$$\text{and also for the potential energy } p$$

$$mg = \frac{\partial}{\partial y}(mgy) = \frac{\partial p}{\partial y} \Rightarrow p = mgy \tag{3.16}$$

To determine the set of Euler-Lagrange equations, we must determine the kinetic and the potential energy of the system. We must then obtain the difference between the two sets of energies, which is called the Lagrangian.

$$So \;\; L = k - p = \frac{1}{2} m \dot{y}^2 - mgy \tag{3.17}$$

To obtain a generalized form for the Lagrangian, we must obtain the system in terms of a general form that attains to all the robotic manipulators. So the equation of motion can be defined as shown below in terms of a differential of the partial derivatives of both kinetic and potential energies.

$$\text{after taking the partial derivative of } L$$

$$\text{with respect to } (\dot{y})$$

$$\frac{\partial L}{\partial \dot{y}} = \frac{\partial k}{\partial \dot{y}}$$

$$\text{and for the derivative of } L \text{ with respect to } y$$

$$\frac{\partial L}{\partial y} = \frac{\partial p}{\partial y} \tag{3.18}$$

$$\text{for both equations we can write } m\ddot{y} = f - mg$$

$$\text{as follows}: \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} - \frac{\partial L}{\partial q_k} = \tau_k;\; k = 1, \ldots\ldots\ldots, n$$

$$\text{as a general form } q_k \text{ is the generalized form of } y$$

$$\text{and } \tau_k \text{ is the generalized force applied to } q_k$$

3.2 Kinetic and Potential Energy

In this section the relationship between the robot manipulator's DH convention and the energy concepts will be developed to determine the dynamics of the robot consistent with the kinematics of the manipulator. Looking at the kinetic energy of an object, we

can see that it is the sum of two different types of kinetic energies. One is caused by the linear motion (translational) and the other is caused by the rotational motion.

So the total kinetic energy is the sum of both energies.

$$k = \frac{1}{2} m v^T v + \frac{1}{2} \omega^T I \omega \tag{3.19}$$

Where (I) is a 3X3 symmetric matrix called the Inertia Tensor, v is the linear velocity Jacobian vector and ω is the angular velocity Jacobian; the angular velocity vector can be found from the skew matrix $S(\omega) = \dot{R} R^T$ where (R) is the orientation transformation of the defined object. To obtain the angular kinetic term the inertia tensor must be defined as follows:

$$I = R\mathrm{I}R^T \;\; where \; \mathrm{I} = \begin{bmatrix} \mathrm{I}_{xx} & \mathrm{I}_{xy} & \mathrm{I}_{xz} \\ \mathrm{I}_{yx} & \mathrm{I}_{yy} & \mathrm{I}_{yz} \\ \mathrm{I}_{zx} & \mathrm{I}_{zy} & \mathrm{I}_{zz} \end{bmatrix} \tag{3.20}$$

3.2.1 Multiple Link Robot Kinetic Energy

From the Jacobian velocities Jv_i , $J\omega_i$ we can analyze the velocities of the link *i*. *In terms of the Jacobians and the derivative joint variable (q)*

$$v_i = J_{vi}(q)\dot{q} \tag{3.21}$$

$$\omega_i = J_{\omega i}(q)\dot{q} \tag{3.22}$$

Now we suppose that the total mass of the manipulator is located at the center of mass of each link, respectively, and we take the origin to be the center of mass itself. As we know, the total kinetic energy is equal to the sum of the linear and angular kinetic

energies. Using (3.19), we can develop the kinetic energy equation of the multi-link manipulator as shown below:

So from this equation

$$k = \frac{1}{2} m v^T v + \frac{1}{2} \omega^T I \omega \tag{3.19}$$

we can develop the kinetic energy of the manipulator as a sum of all the linear and angular kinetic energies of all the links; however, here the velocity Jacobians are used to include the relationship between the motion equations and the kinetic energy of the manipulator as shown below:

$$K = \frac{1}{2} \dot{q}^T \left[\sum_{i=1}^{n} \left\{ m_i J_{vi}(q)^T J_{vi}(q) + J_{\omega i}(q)^T R_i(q) I_i R_i(q)^T J_{\omega i}(q) \right\} \right] \dot{q} \tag{3.23}$$

$$Let\ D(q) = \left[\sum_{i=1}^{n} \left\{ m_i J_{vi}(q)^T J_{vi}(q) + J_{\omega i}(q)^T R_i(q) I_i R_i(q)^T J_{\omega i}(q) \right\} \right] \tag{3.24}$$

$$so\ k = \frac{1}{2} \dot{q}^T D(q) \dot{q} \tag{3.25}$$

3.3 Potential Energy for a Multi-Link Robot Manipulator

After concluding the kinetic energy equations for a multi-link manipulator in the section above, and to obtain the total energy involved with the robot, we must include the potential energy that is stored in the robot. Let us take a robot with an n-link configuration; the only source of potential energy comes from the gravitational potential that the robot holds. (P) is the potential energy symbol, (g) is the gravity vector, and r_{ci} is the vector associated with the coordinates of the center of mass for any robot link i.

$$P_i = m_i g^T r_{ci} \tag{3.26}$$

So the total potential energy of the robot manipulator is the sum of all the potential energies of each link i.

$$P = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} m_i g^T r_{ci} \tag{3.27}$$

To obtain the full system analysis as shown before, we must obtain the Lagrangian equation. To do so, we must show the kinetic energy equation again and develop the full system analysis as shown below. Let the kinetic energy to be a quadratic function of the vector $\dot{q}$ as shown before:

$$k = \frac{1}{2} \dot{q}^T D(q) \dot{q} \tag{3.23}$$

After showing the kinetic energy interim of a matrix, we can split the kinetic energy equation into its sub entities so let d_{ij} be the entries of X^n inertia matrix D(q) $\Rightarrow$

$$k = \frac{1}{2} \sum_{i\,j} d_{ij}(q) \dot{q}_i \dot{q}_j \tag{3.28}$$

and to obtain the Lagrangian (L) we must let the potential energy be independent of the velocity of joint, so (L) is defined as a difference between the kinetic energy and the potential energy.

$$\Rightarrow L = k - P = \frac{1}{2} \sum_{i\,j} d_{ij}(q) \dot{q}_i \dot{q}_j - P(q) \tag{3.29}$$

To obtain the full mathematical model of the robotic manipulator; there must be a derivation of the Euler-Lagrange function where the system is the sum of the dynamics for each of the following: acceleration, the velocity, and the position. This sum would define the relations and the attributes of the dynamics with the kinematics of the system, giving the math model a more full and complete description of the actual system. So we can start by taking the partial derivative of the Lagrangian (L) with respect to the kth position.

$$\frac{\partial L}{\partial q_k} = \frac{1}{2} \sum_{i,j} \frac{\partial d_{i\,j}}{\partial q_k} \dot{q}_i \dot{q}_j - \frac{\partial P}{\partial q_k} \tag{3.30}$$

Furthermore, the partial derivative of the Lagrangian (L) with respect to the kth velocity is just the second partial derivative of the position:

$$\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{kj} \dot{q}_j \tag{3.31}$$

$$\Rightarrow \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{kj} \ddot{q}_j + \sum_{i,j} \frac{\partial d_{kj}}{\partial q_i} \dot{q}_i \dot{q}_j \tag{3.32}$$

So k is changing from 1 to n dependent of the number of degrees of freedom of the robot manipulator; in our case the number of degrees of freedom is k = (2).

Then Euler – Lagrange equations can be shown as follows:

$$\sum_j d_{kj} \ddot{q}_j + \sum_{i,j} \left\{ \frac{\partial d_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial d_{ij}}{\partial q_k} \right\} \dot{q}_i \dot{q}_j + \frac{\partial P}{\partial q_k} = \tau_k \tag{3.33}$$

We can replace the sum of the partial derivatives of (d_{kj}) with respect to (q) by a sum of partial derivatives $\left[\sum_{i,j} c_{ijk} \dot{q}_i \dot{q}_j \right]$

This sum is defined in terms of both the first order derivative of the elements of (q) and the position. All the equations are shown below:

$$but \sum_{i,j} \left\{ \frac{\partial d_{kj}}{\partial q_i} \right\} \dot{q}_i \dot{q}_j = \frac{1}{2} \sum_{i,j} \left\{ \frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ij}}{\partial q_k} \right\} \dot{q}_i \dot{q}_j = \sum_{i,j} c_{ijk} \dot{q}_i \dot{q}_j$$

$$\Rightarrow c_{ijk} = \frac{1}{2} \left\{ \frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k} \right\} \tag{3.34}$$

(c_{ijk}) are known as the "Christoffel symbols".

$$Let\ \ g_k = \frac{\partial P}{\partial q_k} \tag{3.35}$$

As a result the Euler – Lagrange equations can be expressed as a sum of the elements (d_{kj}) the acceleration factors, a sum of the elements (c_{ijk}) the velocity factors, and the potential energy factors (g_k) shown below:

$$\sum_{j=1} d_{kj}(q)\ddot{q}_j + \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ijk}(q)\,\dot{q}_i\,\dot{q}_j + g_k = \tau_k \qquad k = 1,.....,n \tag{3.36}$$

Per the Matrix format, the equation above can be converted into a set of matrices in terms of the acceleration, the velocity, and the acceleration as shown below:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau \tag{3.37}$$

3.4 Possible Design Problem

One issue that we have to consider is the effect of the actuator on the robot manipulator and the effect on the trajectory tracking; this issue would only be considered if the dynamics of the actuator would affect the dynamics of the manipulator. In the industrial world, the output of the robot is the main thing to be considered and nothing else really matters. If the robot can do the job consistently without crashing during the operation, the design would be considered successful.

One way to obtain such a result is to over-size the actuators so that they can handle any torque exerted on the manipulator arm from the load and other links. In this thesis, we are concerned with the accurate tracking of the trajectory or path rather than the dynamics of the actuators. Therefore, we will assume that the actuators are large enough to handle the torque forces for the load and the link-to-link dynamics.

After developing the Euler – Lagrange dynamic equation, we can use it as the basis for the mathematical model and apply the Active Disturbance Rejection control system to this model to obtain a controlled system with the minimum amount of error. This control system will be defined in the next chapter and the relations between the system and ADRC will be developed during this analysis.

CHAPTER IV

ACTIVE DISTURBANCE REJECTION CONTROL

4.1 Introduction to ADRC

The general idea behind ADRC is to use an estimate of the disturbances and the unknown parts of the module and cancel these unknown terms to obtain a controlled system. This control theory was proposed by Professor Han; Dr. Gao [4]then used it to develop the parameterization concept of ADRC. To explore the ADRC concept in greater depth, we will take a second order system as shown below:

$$\ddot{y} = -a_1 \dot{y} - a_2 y + bu + d \tag{4.38}$$

where (y) is the output of the system, (u) is the input of the system, (d) is the external disturbance. Now we will analyze the system using the ADRC concept. Let's consider the following:

Let $f(\dot{y}, y, d)$ the system general unknown terms and disturbances so:

$$
\begin{aligned}
&f(\dot{y}, y, d) = -a_1 \dot{y} - a_2 y + d \Rightarrow \\
&\ddot{y} = f(\dot{y}, y, d) + bu
\end{aligned}
\tag{4.39}
$$

where the control system will generate an estimate of the function (f) denoted ($\hat{f}$) so the control input would be defined as follows:

$$
u = \frac{u_0 - \hat{f}}{b} \tag{4.40}
$$

So if we apply the control to the above equation, we can see that the (b) terms will cancel each other, leaving us with the second integrator of (y) equal to the difference of the error and the error estimate and the initial desired control; the error will be canceled, leaving the output equal to the input as shown in the following:

$$
\begin{aligned}
&\ddot{y} = f + b\frac{u_0 - \hat{f}}{b} \\
&if \quad e = f - \hat{f} \;\; and \;\; e \approx 0 \\
&then \;\; f - \hat{f} \approx 0 \\
&b \, will \;\; cancel \;\; itself \;\; leaving \\
&\ddot{y} = u_0 \quad where \;\; u_0 \, is \; the \; desired \;\; control
\end{aligned}
\tag{4.41}
$$

Using this concept would estimate the unknown and nonlinear parts of the model, stabilize the output, and drive it to the desired input. Furthermore, to obtain this result, Dr. Han has suggested using the Extended State Observer (ESO) to estimate the function shown above. ESO is the essence of the ADCR control theory; in the ESO, the whole concept is to present the function f as an Extended State, giving the second order system an additional state and making it a third order ADRC [4].

4.2 Extended State Observer

The ESO states can be given as follows: where (x_1) is the first state variable, (x_2) is the second state variable, and (x_3) is the extended state variable that would be used as an essential part of the control system to estimate the error (f). This setup of the extended state observer makes the observer a third order observer for a second order system usage.

$$\begin{aligned} x_1 &= x \\ x_2 &= \dot{x} \\ x_3 &= f \end{aligned} \tag{4.42}$$

As shown above, we added the third state equal to the function that we want to eliminate, hinting at the idea of the ESO.

$$\begin{aligned} &\dot{x}_1 = x_2 \\ &\dot{x}_2 = x_3 + bu \\ &But \quad x_3 = f \Rightarrow \dot{x}_2 = f + bu \\ &\dot{x}_3 = \dot{f} \\ &y = x_1 \end{aligned} \tag{4.43}$$

Now we can obtain the state space module accordingly from the equations developed above (4.41) the state space representation is shown below in (4.42).

$$\begin{aligned} &\dot{x} = Ax + Bu + E\dot{f} \\ &y = Cx \\ &where\ A, B, C, E\ are\ matrixes\ shown\ below \\ &A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ b \\ 0 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \quad E = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \end{aligned} \tag{4.44}$$

To obtain the ESO of this system using the parameterization concept of Dr. Gao, we can set up the equations using the estimation of the system and the gain vector (L) as follows:

$$
\begin{aligned}
&\dot{\hat{x}} = A\hat{x} + Bu + L(y\text{-}\hat{y}) \\
&but\ \mathrm{y} = \mathrm{Cx} \Rightarrow \hat{\mathrm{y}} = \mathrm{C}\hat{\mathrm{x}}\ \ then \\
&\dot{\hat{x}} = A\hat{x} + Bu + L(y\text{-}C\hat{x}) \\
&\dot{\hat{x}} = A\hat{x} + Bu + Ly\text{-}LC\hat{x} \\
&\quad = (A - LC)\hat{x} + Bu + Ly \\
&L\ is\ a\ gain\ vector\ can be\ shown\ as\ L = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix}
\end{aligned}
\tag{4.45}
$$

Using the parameterization concept developed by Dr. Gao, we can show the vector (L) interims of ω_0 as follows:

$$
L = \begin{bmatrix} 3\omega_0 \\ 3\omega_0^{\ 2} \\ \omega_0^{\ 3} \end{bmatrix}
\tag{4.46}
$$

Making a state representation of the observer model to be shown in detail below in equations (4.47) and (4.48), the representation is in a state equation format. This representation is confined to a single input single output system.

$$
A - LC = \begin{bmatrix} -3\omega_0 & 1 & 0 \\ -3\omega_0^{\ 2} & 0 & 1 \\ -\omega_0^{\ 3} & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ b \\ 0 \end{bmatrix} \qquad L = \begin{bmatrix} 3\omega_0 \\ 3\omega_0^{\ 2} \\ \omega_0^{\ 3} \end{bmatrix}
\tag{4.47}
$$

$$
\begin{aligned}
&where\ \dot{\hat{x}} = \begin{bmatrix} \dot{\hat{x}}_1 \\ \dot{\hat{x}}_2 \\ \dot{\hat{x}}_3 \end{bmatrix} \qquad and \quad \hat{x} = \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \hat{x}_3 \end{bmatrix} \\
&x_3 = f \Rightarrow \hat{x}_3 = \hat{f}\ \ is\ an\ estimate\ of\ \ x_3 = f
\end{aligned}
\tag{4.48}
$$

so the control law to obtain a pure double integrator system is

$$
\begin{aligned}
&u = \frac{u_0 - \hat{f}}{b} = \frac{u_0 - \hat{x}_3}{b} \\
&and\ u_0 = K_p(r - \hat{x}_1) - K_d \hat{x}_2 \quad Gao\ [4] \\
&as\ shown\ before\ \ddot{y} = u_0 \Rightarrow \\
&\ddot{y} = K_p(r - \hat{x}_1) - K_d \hat{x}_2 \quad where\ r\ is\ the\ setpoint
\end{aligned}
\tag{4.49}
$$

We can select the gains to make a critically damped system by making both controller gains based on a single parameter called the controller bandwidth ω_c, as proposed by Dr. Gao "Let $K_p = \omega_c^2 \ \ and \ \ K_d = 2\omega_c$" [4].

After introducing the Active Disturbance Rejection Control system and the general system dynamic in the form of the Euler – Lagrange equation, we can set up the mathematical module of the RP manipulator. The mathematical analysis in the next section will be conclusive to a step by step solution of a two-degree of freedom robotic manipulator.

CHAPTER V

MATHEMATICAL MODELING

5.1 Mathematical Model of an RP Manipulator

With robotic modeling and ADRC control in mind, we will now show the mathematical analysis of the robotic math model step by step and then obtain the appropriate ADRC control according to the model.

5.2 Forward Kinematics and DH Convention for an RP Manipulator

Starting with the Forward kinematics of the robot as proposed in the introduction, we need to obtain the DH convention of the RP robot manipulator.

$$
\begin{aligned}
A_i &= Rot_{z,\theta i} Trans_{z,di} Trans_{x,ai} Rot_{x,\alpha i} \\
&= \begin{bmatrix} c_{\theta i} & -s_{\theta i} & 0 & 0 \\ s_{\theta i} & c_{\theta i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha i} & -s_{\alpha i} & 0 \\ 0 & s_{\alpha i} & c_{\alpha i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} c_{\theta i} & -s_{\theta i} c_{\alpha i} & s_{\theta i} s_{\alpha i} & a_i c_{\theta i} \\ s_{\theta i} & c_{\theta i} c_{\alpha i} & -c_{\theta i} s_{\alpha i} & a_i s_{\theta i} \\ 0 & s_{\alpha i} & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
\tag{5.50}
$$

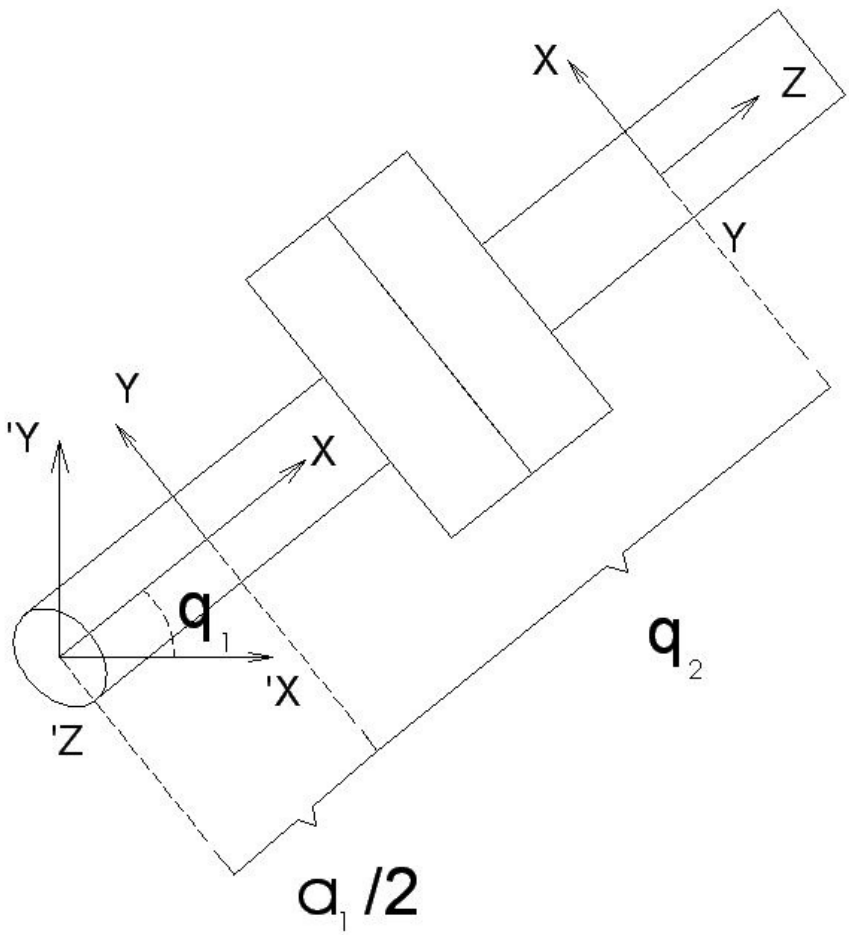

Figure 1. Joint Link System Drawing

We start the robotic manipulator setup by drawing the joint link system as shown above. We then set up the DH parameter table for a two-link RP manipulator as shown in the table below:

Table I DH Parameter Table

Link	α_i	θ_i	d_i	a_i
1	0	q_1^*	0	$\frac{a_1}{2}$
2	90	90	0	q_2^*

To obtain the matrix configuration of the DH convention, we will do the following: The robotic manipulator will make the first transformation; which will contain a rotation across the Z-axis of a variable angle q_1; the center of mass then will be translated from the joint position to the middle of the link a_1. The transformation is shown in equation (5.51).

$$T_0^1 = A_1 A_2 = \begin{bmatrix} C_{q_1} & -S_{q_1} & 0 & 0 \\ S_{q_1} & C_{q_1} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & \frac{a_1}{2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_{q_1} & -S_{q_1} & 0 & C_{q_1}\frac{a_1}{2} \\ S_{q_1} & C_{q_1} & 0 & S_{q_1}\frac{a_1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(5.51)}$$

$$rotation\ on\ Z\ of\ q_1 \quad Translation \frac{a_1}{2} on\ X$$

The second transformation will be obtained by rotating across the Z-axis at an angle of 90 degrees, followed by a rotation across the X-axis at an angle of 90 degrees as well, and then followed by a translation on the X-axis with a variable length q_2. The total transformation is the multiplication of both the first transformation and the second transformation.

$$T_1^2 = A_4 A_5 A_3 = \begin{bmatrix} \cos(90) & -\sin(90) & 0 & 0 \\ \sin(90) & \cos(90) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(90) & -\sin(90) & 0 \\ 0 & \sin(90) & \cos(90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Rightarrow$$

$$T_1^2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & q_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(5.52)}$$

$$T_{final} = T_0^1 T_1^2 = \begin{bmatrix} -\sin q_1 & 0 & \cos q_1 & \cos(q_1)q_2 + \frac{1}{2}a_1 \sin(q_1) \\ \cos q_1 & 0 & \sin q_1 & \sin(q_1)q_2 + \frac{1}{2}a_1 \sin(q_1) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

These mathematical calculations were done using Matlab using the values defined in Table 1: so T_{1Zero} as shown in (5.53) is the first transformation and T_{2Zero} as shown in (5.54) is the total transformation.

$$T_{1Zero} = \begin{bmatrix} \cos(q_1) & -\sin(q_1) & 0 & \frac{1}{2}\cos(q_1) * a_1 \\ \sin(q_1) & \cos(q_1) & 0 & \frac{1}{2}\sin(q_1) * a_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.53}$$

$$T_{2Zero} = \begin{bmatrix} -\sin(q_1) & 0 & \cos(q_1) & \cos(q_1)q_2 + \frac{1}{2}\cos(q_1) * a_1 \\ \cos(q_1) & 0 & \sin(q_1) & \sin(q_1)q_2 + \frac{1}{2}\sin(q_1) * a_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.54}$$

5.3 Jacobian Matrices for RP manipulator

After obtaining the transformation matrixes of the system, we can obtain the Jacobians of the system using Matlab as well. J_1 is defined as the total Jacobian of the first joint including all the linear velocity Jacobian terms and the angular velocity Jacobian.

$$J_1 = \begin{bmatrix} -\frac{1}{2}a_1 \sin(q_1) & 0 \\ \frac{1}{2}a_1 \cos(q_1) & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \tag{5.55}$$

Similarly, we can find J_2 where (J_2) is the total Jacobian of the second joint including the linear and angular velocity Jacobian terms; all the rest of the calculations for the Jacobians will be attached as a Matlab file.

$$J_2 = \begin{bmatrix} -\sin(q_1)q_2 - \frac{1}{2}\sin(q_1)a_1 & \cos(q_1) \\ \cos(q_1)q_2 + \frac{1}{2}\cos(q_1)a_1 & \sin(q_1) \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \tag{5.56}$$

Now we will set up the inertia matrix addressed in the dynamics portion of the mathematical module. Also known as the Inertia Tensor, these matrices are useful in the analysis of the D Matrix to obtain the inertial frame of the system.

$$I_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & I_{1z} \end{bmatrix}$$
$$I_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & I_{2y} & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.57}$$

To obtain the Euler-Lagrange equation we must calculate the D matrix where D is a composition of the Jacobian Matrices, the Inertia Tensor Matrices, the masses, and the transformation matrices. The combination is shown in (5.58) below:

$$D = \left[m_1 * J_{V1}{}^T * J_{V1}\right] + \left[m_2 * J_{V2}{}^T * J_{V2}\right] + \left[J_{W1}{}^T * T_1(1:3,1:3) * I_1 * T_1(1:3,1:3)^T * J_{W1}\right] + \left[J_{W2}{}^T * T_{2zero}(1:3,1:3) * I_2 * T_{2zero}(1:3,1:3)^T * J_{W2}\right] \tag{5.58}$$

After Solving the D Matrix for the supposed parameters using Matlab software and then canceling all the opposite but equal terms, we will obtain the final form of the D matrix as shown in (5.59) below:

$$D = \begin{bmatrix} \frac{1}{4} * m_1 * a_1^{\,2} + m_2 * q_2^{\,2} + m_2 * q_2 * a_1 + \frac{1}{4} * m_2 * a_1^{\,2} + I_{1z} + I_{2y} & 0 \\ 0 & m_2 \end{bmatrix} \tag{5.59}$$

Now we can calculate the C matrix, also known as the coupling matrix. The C matrix is the relation between the two joint dynamics, giving us the nonlinear terms that would make the system as a nonlinear system in nature.

$$\begin{aligned} C &= \begin{bmatrix} C_{11} & C_{21} \\ C_{12} & C_{22} \end{bmatrix} \\ C_{11} &= C_{111}\,\dot{q}_1 + C_{211}\,\dot{q}_2 \\ C_{12} &= C_{121}\,\dot{q}_1 + C_{221}\,\dot{q}_2 \\ C_{21} &= C_{112}\,\dot{q}_1 + C_{212}\,\dot{q}_2 \\ C_{22} &= C_{122}\,\dot{q}_1 + C_{222}\,\dot{q}_2 \end{aligned} \tag{5.60}$$

After calculating the C matrix using Matlab, the result is as follows:

$$C = \begin{bmatrix} (m_2 * q_2 + \frac{1}{2} m_2 * a_1)\,\dot{q}_2 & (m_2 * q_2 + \frac{1}{2} m_2 * a_1)\,\dot{q}_1 \\ (-m_2 * q_2 - \frac{1}{2} m_2 * a_1)\,\dot{q}_1 & 0 \end{bmatrix} \tag{5.61}$$

Using the potential energy, we will now calculate the g(q) matrix.

$$P_1 = \frac{1}{2} m_1 * g * a_1 * \sin(q_1)$$

$$P_2 = m_2 * g * \left[\frac{a_1}{2} + q_2 \right] * \sin(q_1)$$

$$P = P_1 + P_2$$

$$P = \frac{1}{2} m_1 * g * a_1 * \sin(q_1) + m_2 * g * \left[\frac{a_1}{2} + q_2 \right] * \sin(q_1)$$

$$g_1 = diff\,(P, q_1) \qquad \textbf{(5.62)}$$

$$g_2 = diff\,(P, q_2)$$

$$G = [g_1; g_2]$$

$$G = \begin{bmatrix} \frac{1}{2} m_1 * g * a_1 \cos(q_1) + m_2 * g * \left[\frac{a_1}{2} + q_2 \right] * \cos(q_{1)} \\ m_2 * g * \sin(q_1) \end{bmatrix}$$

After obtaining all the related matrices implicitly including the D matrix, the C matrix, and the G matrix, we can assemble the total system into the form shown below in equation (5.62); after using Matlab to calculate the math model, it is totally confined in this equation.

$$D(q)\ddot{q}+C(q,\dot{q})\dot{q}+g(q)=\tau$$

$$\begin{bmatrix} \left[\frac{1}{4}m_1*a_1^{\,2}+m_2*q_2^{\,2}+m_2*q_2*a_1+\frac{1}{4}m_2*a_1^{\,2}+I_{1z}+I_{2y}\right]*\ddot{q}_1 \\ +2*\left[m_2*q_2+\frac{1}{2}m_2*a_1\right]*\dot{q}_2\,\dot{q}_1+\frac{1}{2}m_1*g*a_1\cos(q_1)+m_2*g*\left[\frac{1}{2}a_1+q_2\right]*\cos(q_1) \\ \\ m_2*\ddot{q}_2+\left[-m_2*q_2-\frac{1}{2}m_2*a_1\right]*\dot{q}_1^{\,2}+m_2*g*\sin(q_1) \end{bmatrix}=\tau \qquad \textbf{(5.62) cont.}$$

Looking at the mathematical model above (5.62) we can see that the system is nonlinear in nature due to the nonlinear terms such as $\dot{q}_2\dot{q}_1$, $\sin(q_1)$, *etc* ., which makes it difficult to analyze; however, the beauty of Active Disturbance Rejection Control System is that it can be used as a linear controller to control nonlinear systems.

In the next chapter, the control system will be analyzed using the work of the mathematical model developed in this chapter and using ADRC as a control methodology. To understand the system better, the math model will be processed as a link per link approach, dividing the two-link manipulator into two separate systems linked with the feedback form the encoder system of each arm.

CHAPTER VI

IMPLEMENTING ADRC CONCEPT IN THE RP MANIPULATOR

Let us evaluate the D matrix, which is the matrix factor of the acceleration, or in other terms, the double integrator factor of the position. We can see that it is a symmetric matrix, and that would help us to perform a substitution of the complicated equations with single terms such as (a, b, etc). Let the first term be (a) and the second term be (b) as shown below:

$$D = \begin{bmatrix} \frac{1}{4} * m_1 * a_1^{\ 2} + m_2 * q_2^{\ 2} + m_2 * q_2 * a_1 + \frac{1}{4} * m_2 * a_1^{\ 2} + I_{1z} + I_{2y} & 0 \\ 0 & m_2 \end{bmatrix}$$

$$a = \frac{1}{4} * m_1 * a_1^{\ 2} + m_2 * q_2^{\ 2} + m_2 * q_2 * a_1 + \frac{1}{4} * m_2 * a_1^{\ 2} + I_{1z} + I_{2y} \qquad \textbf{(6.63)}$$

$$b = m_2$$

$$so$$

$$D = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

But the term (a) is dependent on the position of the second link; i.e., a = a(t), and q_2 is what (a) depends upon.

Now we can evaluate the dynamic function as shown below where we can replace the (q) terms with (y) terms to obtain a more uniform setup in the control's world where (y) terms are known more widely.

$$
\begin{aligned}
&D(q)\ddot{q}+C(q,\dot{q})\dot{q}+g(q)=\tau \\
&let \\
&\ddot{q}=\ddot{y}=\begin{bmatrix}\ddot{y}_1\\ \ddot{y}_2\end{bmatrix} and \begin{bmatrix}\ddot{q}_1\\ \ddot{q}_2\end{bmatrix}=\ddot{q} \qquad also\ \ let\ \ \dot{y}=\dot{q} \qquad , \qquad y=q\ \ and\ \ \tau=u
\end{aligned}
\tag{6.64}
$$

All notations are matrix-based and therefore the system is just a second order system and, as with a normal second order system, we can analyze the system even further as shown in (6.65). We will move the D matrix using its inverse into the right-hand side and leaving the double integrator on the left-hand side; let the error function be equal to the rest of the system. This would make it possible to control the system using ADRC.

$$
\begin{aligned}
&\ddot{y}=f(\dot{y},y,d)+bu \\
&so\ in\ \ general: \\
&\ddot{q}=\left[D^{-1}\right]*f(\dot{q},q,d)+\left[D^{-1}\right]*\tau \\
&f(\dot{q},q,d)=-C(\dot{q})-g(q)+d
\end{aligned}
\tag{6.65}
$$

For a clearer understanding of the system, the process will split the system from a multi input multi output system (MIMO) into two systems. These systems will be controlled in an individual fashion to make the control easier to observe and analyze. The two systems will be linked with feed back from the position of the second arm to obtain a fully dynamic control system. We can split the previous equation into two equations, as shown below in (6.66); also, the control output is defined in the same equation to eliminate any error from the system.

$$\ddot{q}_1 = \frac{1}{a} * f(\dot{q}_1, q_1, d) + \frac{1}{a}\tau_1 \quad where\ a = a(t)\, dependent\ on\ q_2$$

$$\ddot{q}_2 = \frac{1}{b} * f(\dot{q}_2, q_2, d) + \frac{1}{b}\tau_2$$

$$let\ \tau_1 = a(\tau_{0-1} - \hat{f}_1) \quad and \quad \tau_2 = b(\tau_{0-2} - \hat{f}_2) \quad where\ \tau = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \ and\ \tau_0 = \begin{bmatrix} \tau_{0-1} \\ \tau_{0-2} \end{bmatrix}$$

$$\Rightarrow \ddot{q}_1 = \tau_{0-1} \quad and \quad \ddot{q}_2 = \tau_{0-2} \Leftrightarrow \ddot{q} = \tau_0 \quad the\ control$$

$$where\ \hat{f} \approx f$$

$$let\ x_1 = q\ ,\ x_2 = \dot{q}\ ,\ and\ \ x_3 = f.$$

$$let\ h = \dot{f}$$

then

the model is represented as shown below:

$$\begin{cases} \dot{x} = Ax + B\tau + Eh \\ y = Cx \end{cases}$$

$$where \tag{6.66}$$

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ b \\ 0 \end{bmatrix} \qquad E = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad and \qquad C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

The total purpose of the control system is to drive the system to the desired control pattern; in this case, we want to eliminate all the disturbances and all the undefined variables and states to obtain a controlled system.

6.1 State Observer Design

Now we can set up the ESO for the model; the ESO plant can be defined as follows:

$$\begin{cases} \dot{\hat{x}} = A\hat{x} + B\tau + L(x_1 - \hat{x}_1) \\ \hat{y} = C\hat{x} \end{cases} \tag{6.67}$$

L is the gain vector defined such that the extended state observer eigenvector is in line with $-\omega_0$ *as shown below.*

$$L = \begin{bmatrix} 3\omega_0 \\ 3\omega_0^2 \\ \omega_0^3 \end{bmatrix} \qquad \textbf{(6.67) cont.}$$

This is the special power of ADRC: to make the tuning parameter associated with only one tuning parameter, which makes it easy and simple to tune and to control. The control law shown before can be defined as $\tau = (\tau_0 - \hat{x}_3)/b$, making the system a double integrator system ($\ddot{y} = \tau_0$). This system can easily be controlled using a PD controller. $\tau_0 = k_p(r - \hat{x}_1) - k_d \hat{x}_2$ Where (k_p) and (k_d) are chosen to be (ω_c^2 , $2\omega_c$), respectively, to place all the closed loop poles at $-\omega_c$. By doing so, we can reduce the tuning parameters to one.

To setup the ESO for the two-link manipulator, we will treat the system as two sets of systems controlled separately using two ESOs and using the output of the second position from the encoder as a relation factor into calculating the term (a).

The ESO for the first arm could be represented as follows:

$$\begin{bmatrix} \dot{\hat{q}}_1 \\ \ddot{\hat{q}}_1 \\ \dot{\hat{f}}_1 \end{bmatrix} = \begin{bmatrix} -3\omega_o & 1 & 0 \\ -3\omega_o^2 & 0 & 1 \\ -\omega_o^3 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{q}_1 \\ \dot{\hat{q}}_1 \\ \hat{f}_1 \end{bmatrix} + \begin{bmatrix} 0 & 3\omega_o \\ \frac{1}{a} & 3\omega_o^2 \\ 0 & \omega_o^3 \end{bmatrix} \begin{bmatrix} \tau_1 \\ q_1 \end{bmatrix}$$

where State Space representation is **(6.68)**

$$A = \begin{bmatrix} -3\omega_o & 1 & 0 \\ -3\omega_o^2 & 0 & 1 \\ -\omega_o^3 & 0 & 0 \end{bmatrix} \quad and \quad B = \begin{bmatrix} 0 & 3\omega_o \\ \frac{1}{a} & 3\omega_o^2 \\ 0 & \omega_o^3 \end{bmatrix} \quad and \quad C = eye(3)$$

And the D matrix is zero.

For the second Arm, the ESO is represented in the same fashion as shown below:

$$\begin{bmatrix} \dot{\hat{q}}_2 \\ \ddot{\hat{q}}_2 \\ \dot{f}_2 \end{bmatrix} = \begin{bmatrix} -3\omega_o & 1 & 0 \\ -3\omega_o^2 & 0 & 1 \\ -\omega_o^3 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{q}_2 \\ \dot{\hat{q}}_2 \\ \hat{f}_2 \end{bmatrix} + \begin{bmatrix} 0 & 3\omega_o \\ \frac{1}{b} & 3\omega_o^2 \\ 0 & \omega_o^3 \end{bmatrix} \begin{bmatrix} \tau_1 \\ q_1 \end{bmatrix}$$

where State Space representation is **(6.69)**

$$A = \begin{bmatrix} -3\omega_o & 1 & 0 \\ -3\omega_o^2 & 0 & 1 \\ -\omega_o^3 & 0 & 0 \end{bmatrix} \quad and \quad B = \begin{bmatrix} 0 & 3\omega_o \\ \frac{1}{b} & 3\omega_o^2 \\ 0 & \omega_o^3 \end{bmatrix} \quad and \quad C = eye(3)$$

Again, the D matrix is zero. Using the PD controller for both Arms, we setup the ADRC control to control the RP manipulator to obtain the desired pattern. We chose $\omega_c = \frac{1}{2}\omega_0$ so we can tune only one parameter.

After defining the Active Disturbance Rejection Control system and the Mathematical Model in the form of the Euler – Lagrange equation, we can setup a simulation analysis where the system will be tested under various simulated situations. This will demonstrate how robust and accurate the ADRC technology is in keeping the system under control. The simulations throughout the next chapter shall be done with the same bandwidth for each section for consistency.

CHAPTER VII

SIMULATIONS

7.1 Simulation without disturbances or noise

We set up the system in Simulink model as shown below:

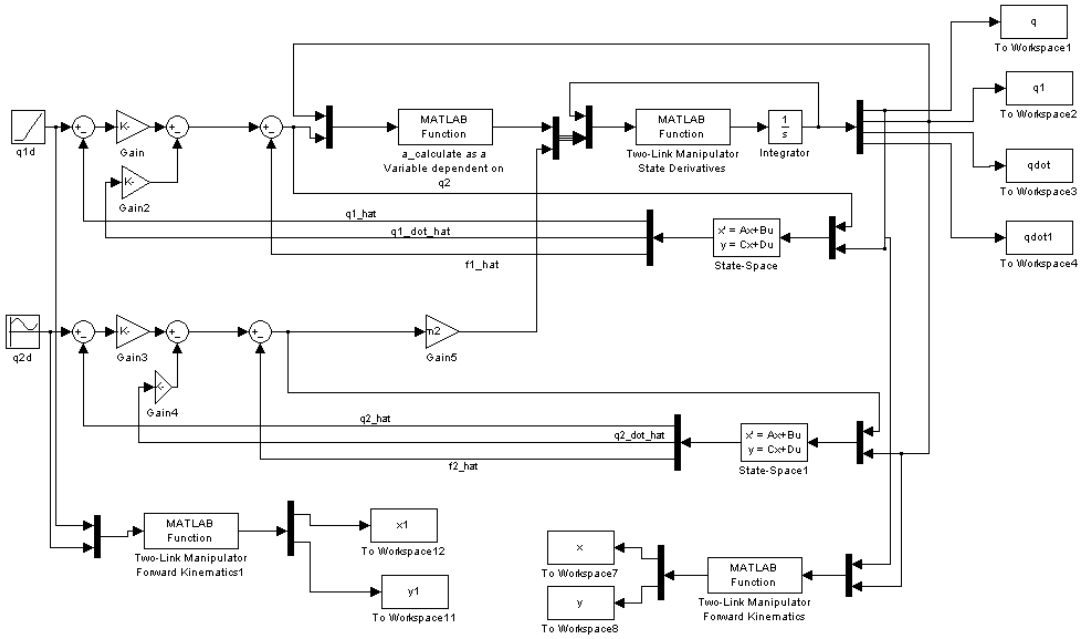

Figure 2. System Setup in Simulink

We then run the system using different parameters for tuning purposes at different ω_o values as shown in the following graphs.

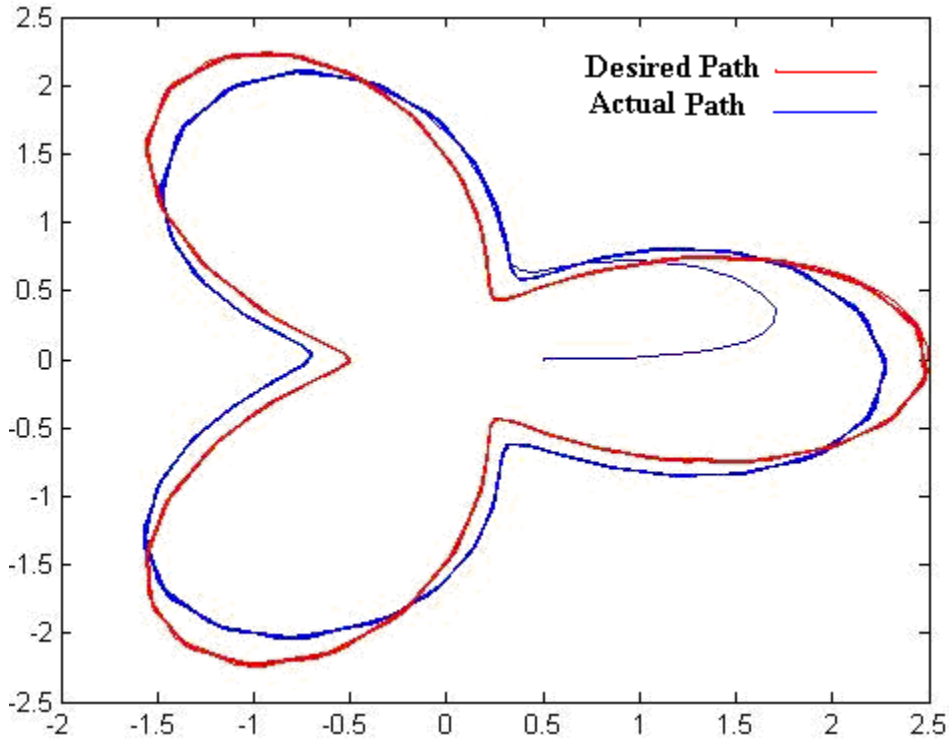

Figure 3. Output Tracking at a Frequency of 100 without Feed Forward

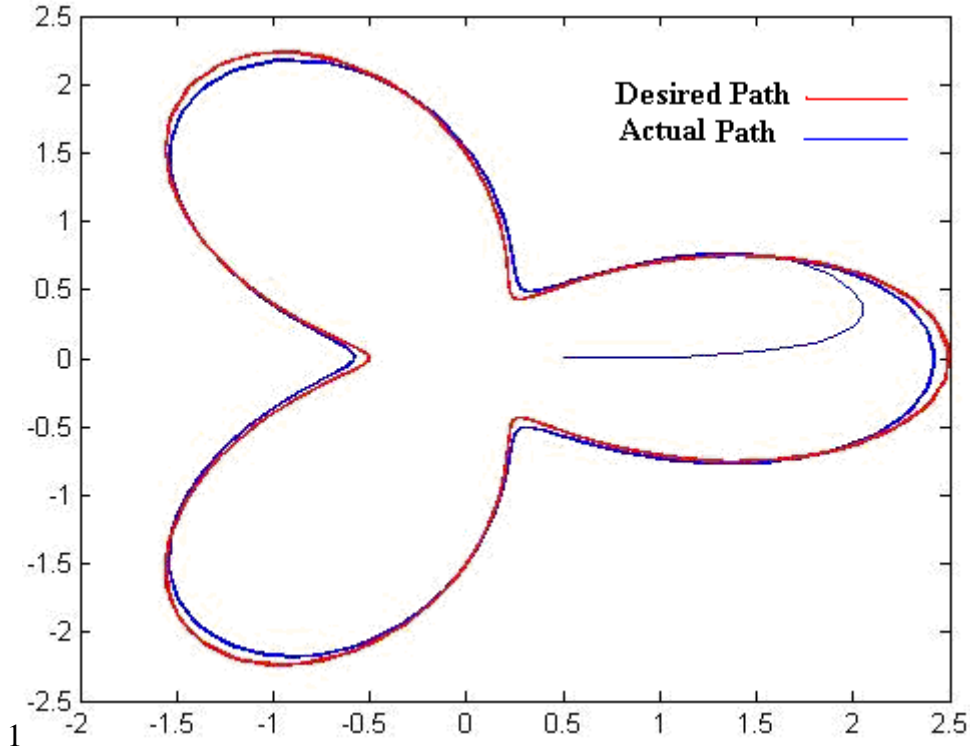

Figure 4. Output Tracking at a Frequency of 200 No Feed Forward

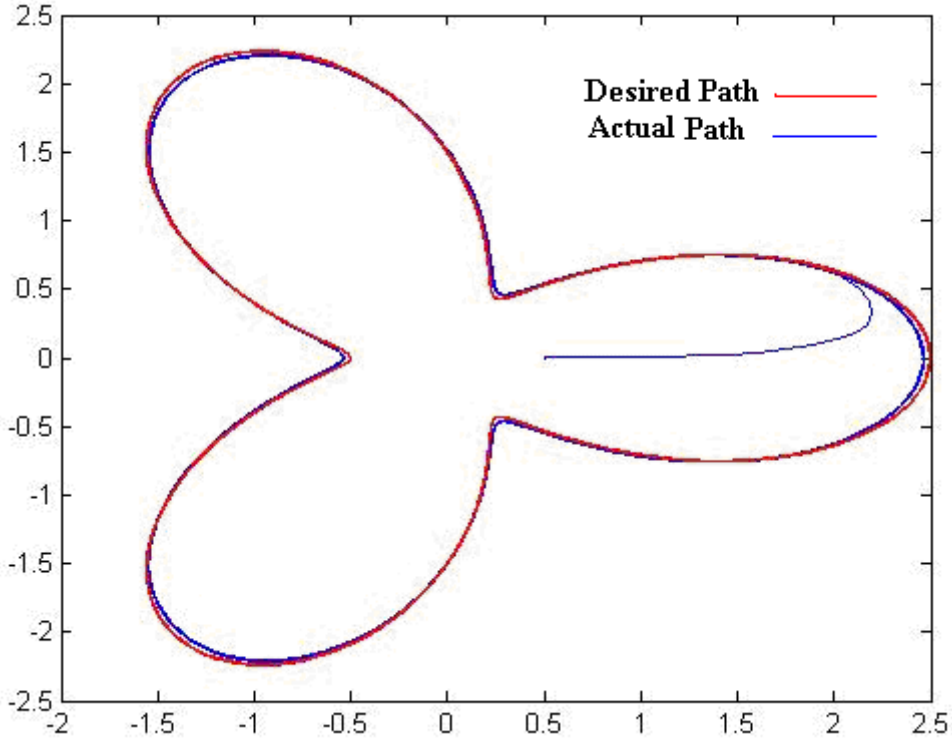

Figure 5. Output Tracking at a Frequency of 300 No Feed Forward

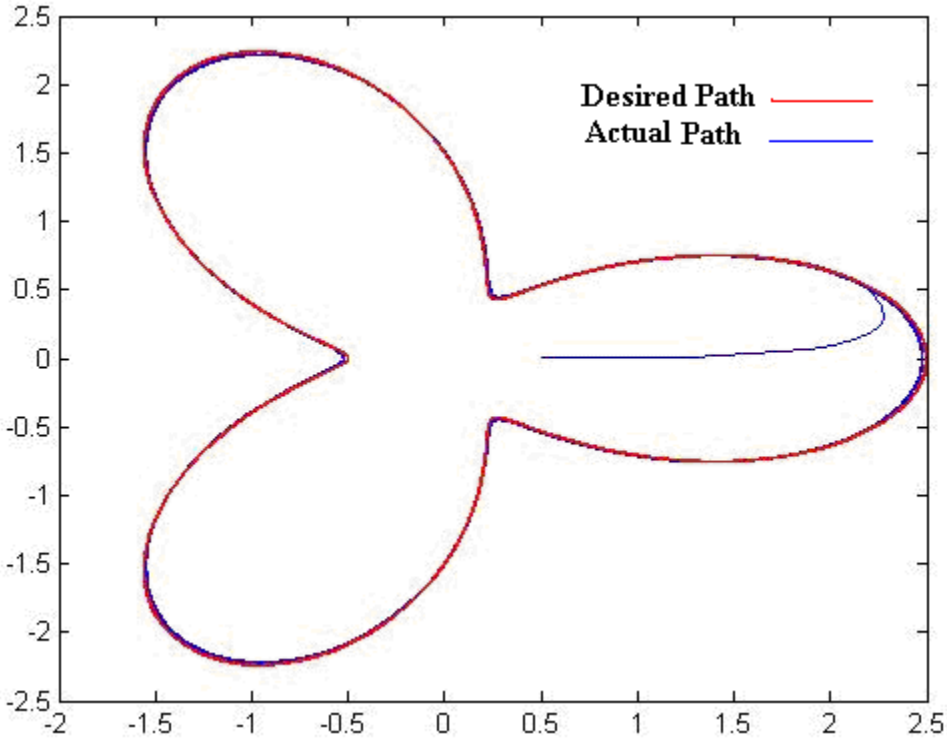

Figure 6. Output Tracking at a Frequency of 400 No Feed Forward

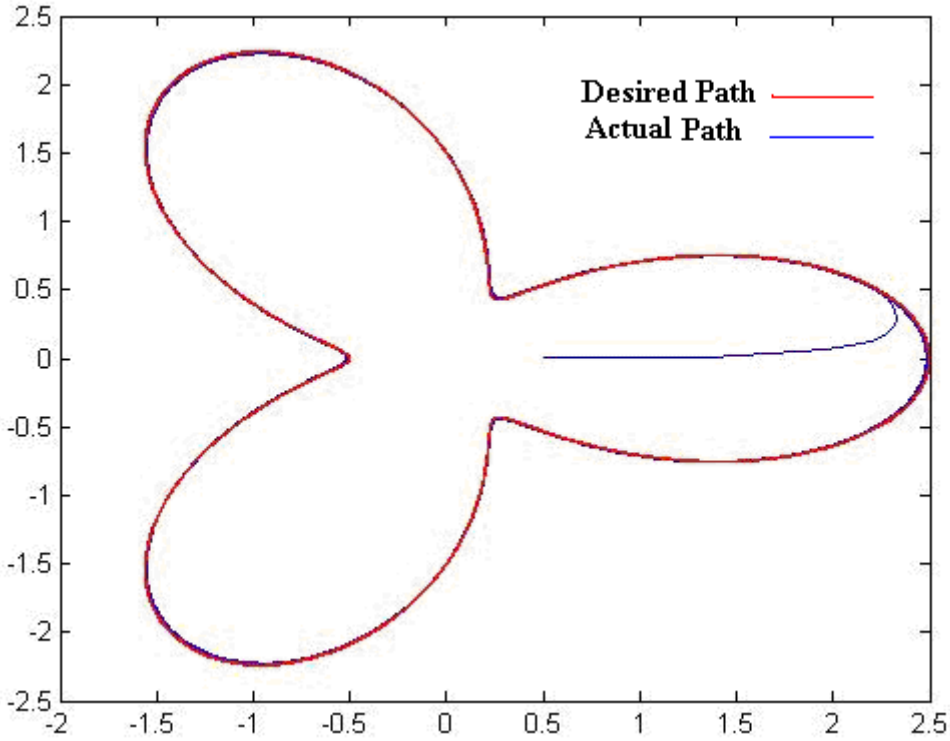

Figure 7. Output Tracking at a Frequency of 500 No Feed Forward

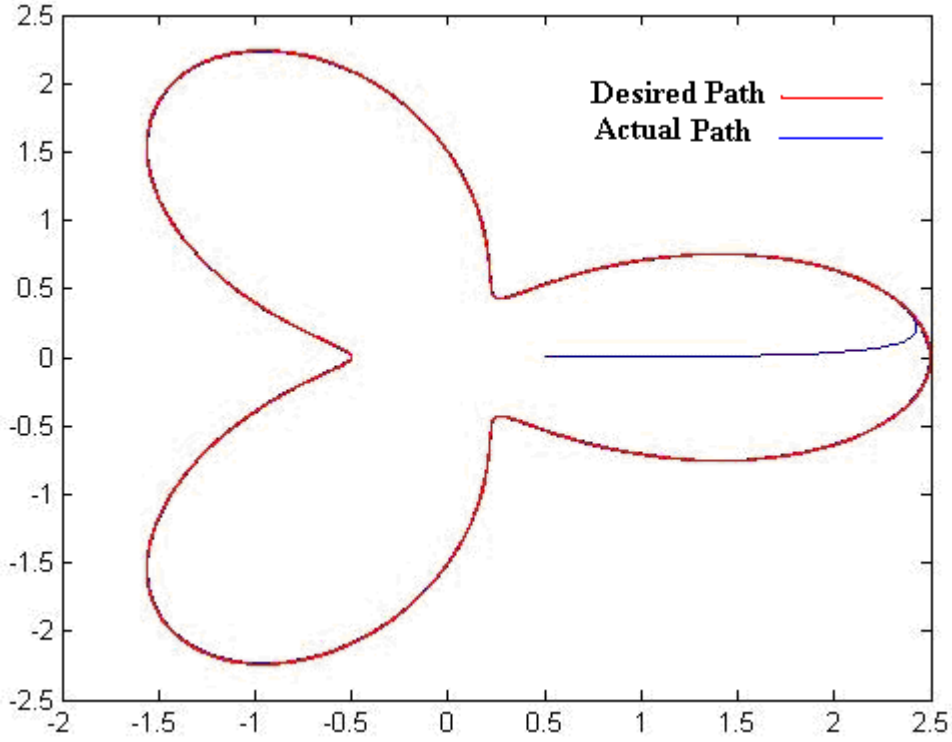

Figure 8. Output Tracking at a Frequency of 1000 No Feed Forward

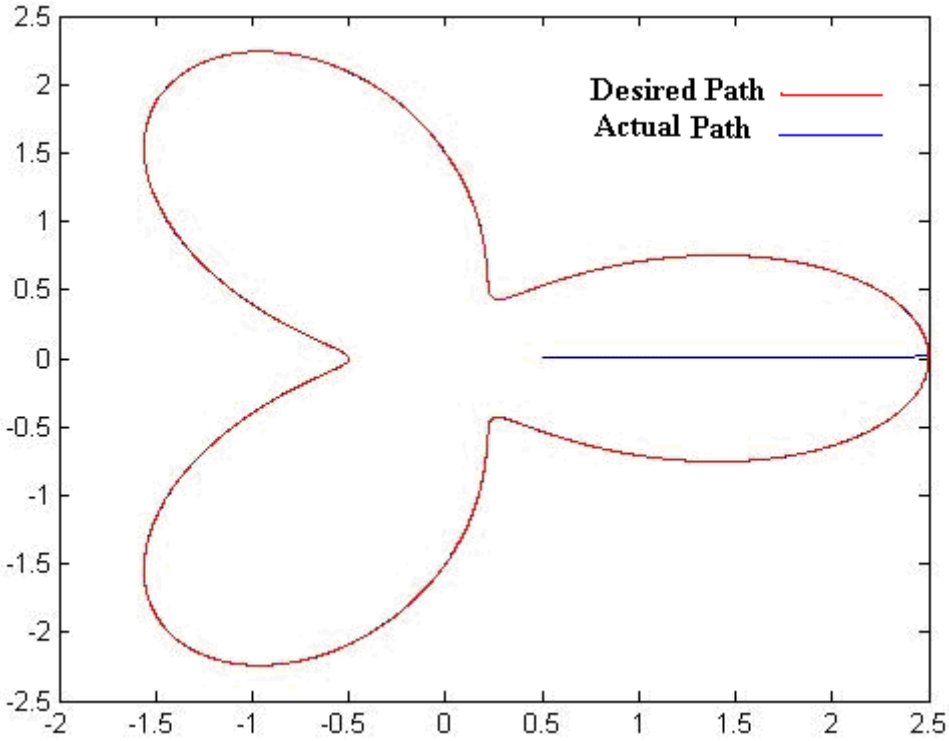

Figure 9. Output Tracking at a Frequency of 10000 No Feed Forward

7.2 Simulations with Feed Forward

After running the simulation using Simulink without any type of feed forward, we can see that the system works well but could be tuned with less control if we used the feed forward. In any system, we can add the desired output according to our knowledge about the system. In the second part of this simulation we will add the feed forward to the system and observe the differences between the control with feed forward and the control without feed forward.

We set up the system as follows:

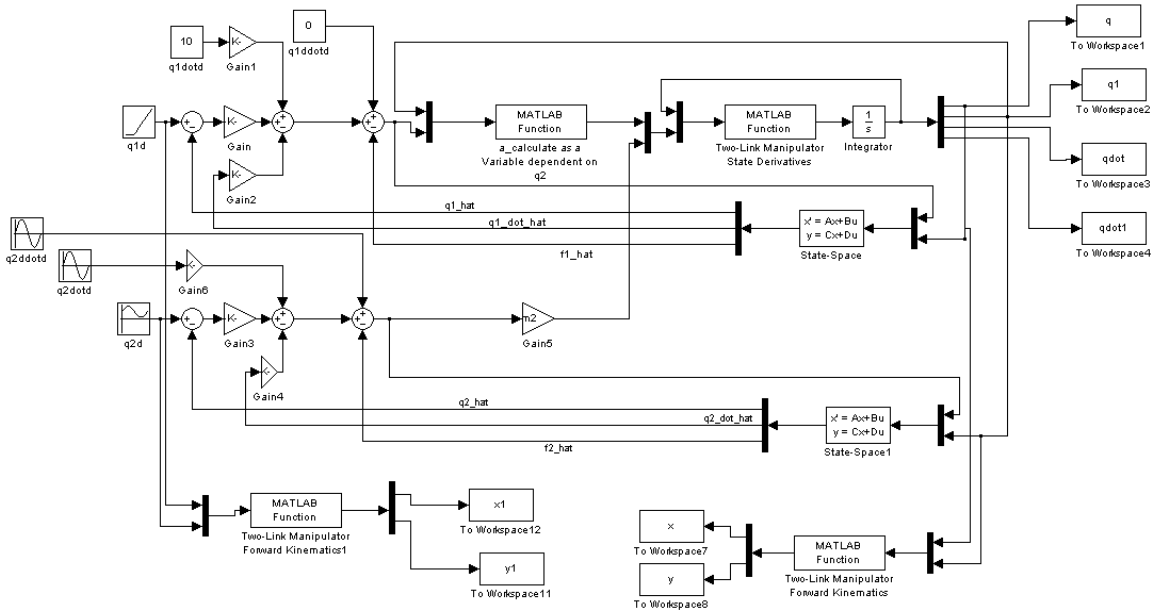

Figure 10. System Setup with Feed Forward

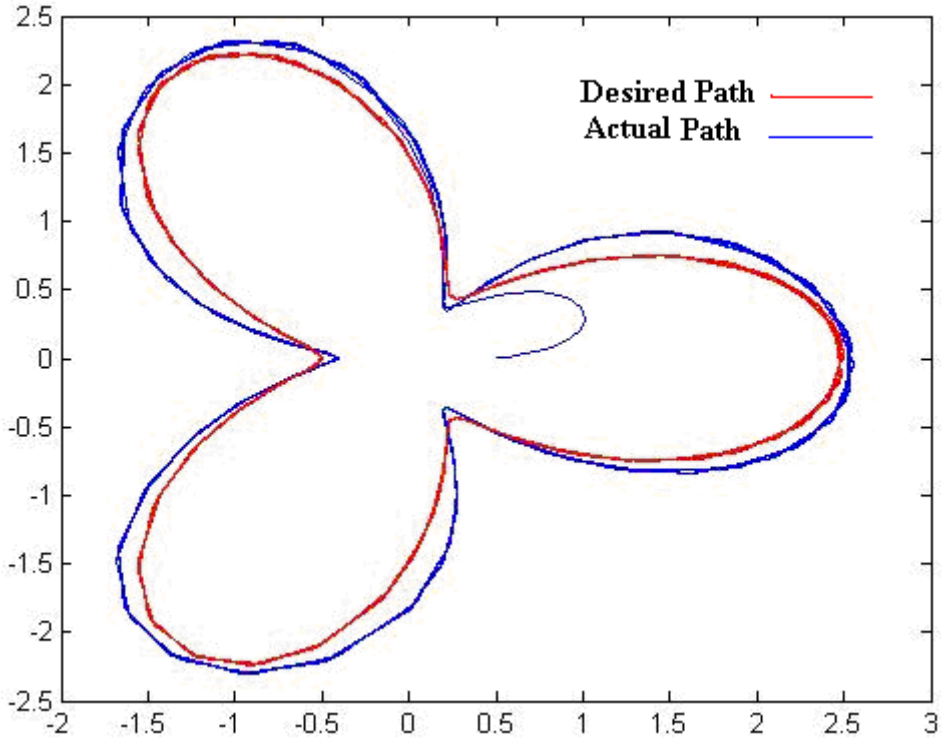

Figure 11. Output Tracking at a Frequency of 100 with Feed Forward

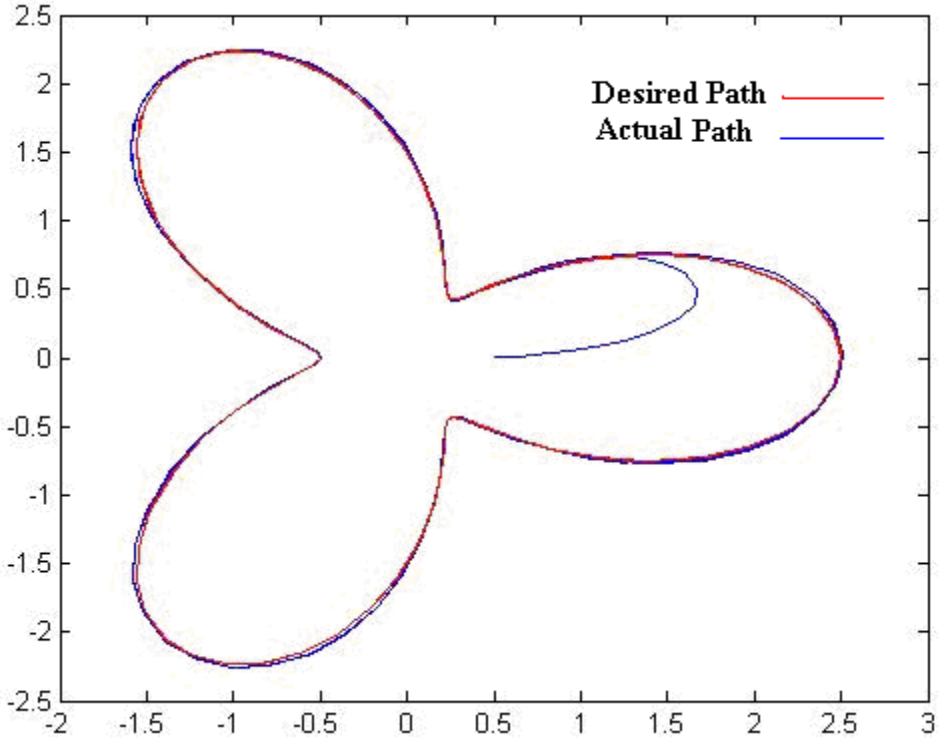

Figure 12. Output Tracking at a Frequency of 200 with Feed Forward

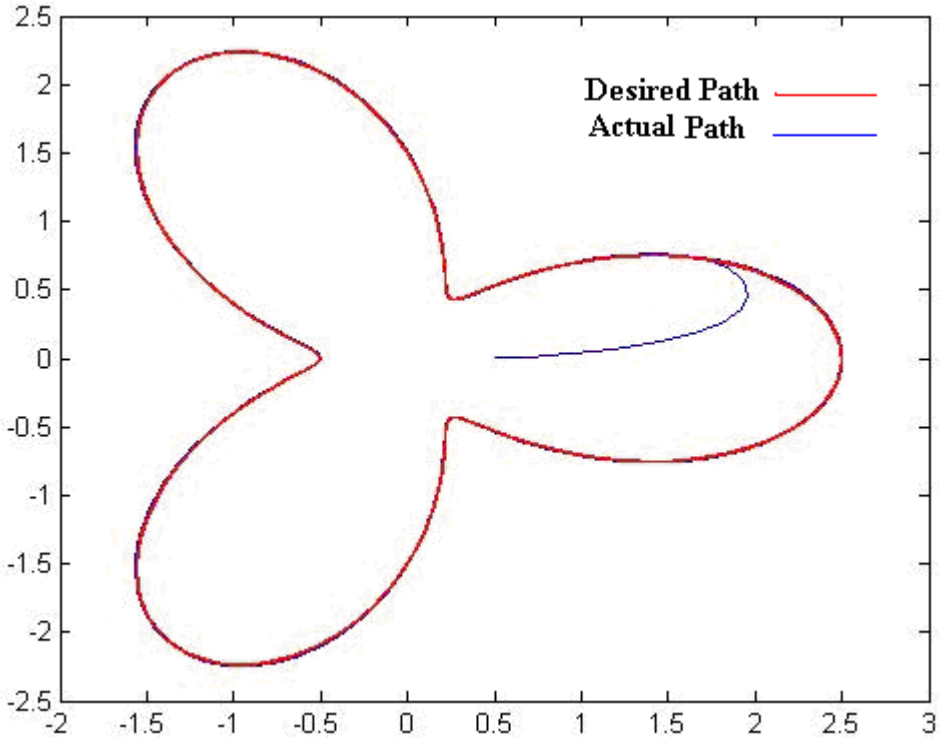

Figure 13. Output Tracking at a Frequency of 300 with Feed Forward

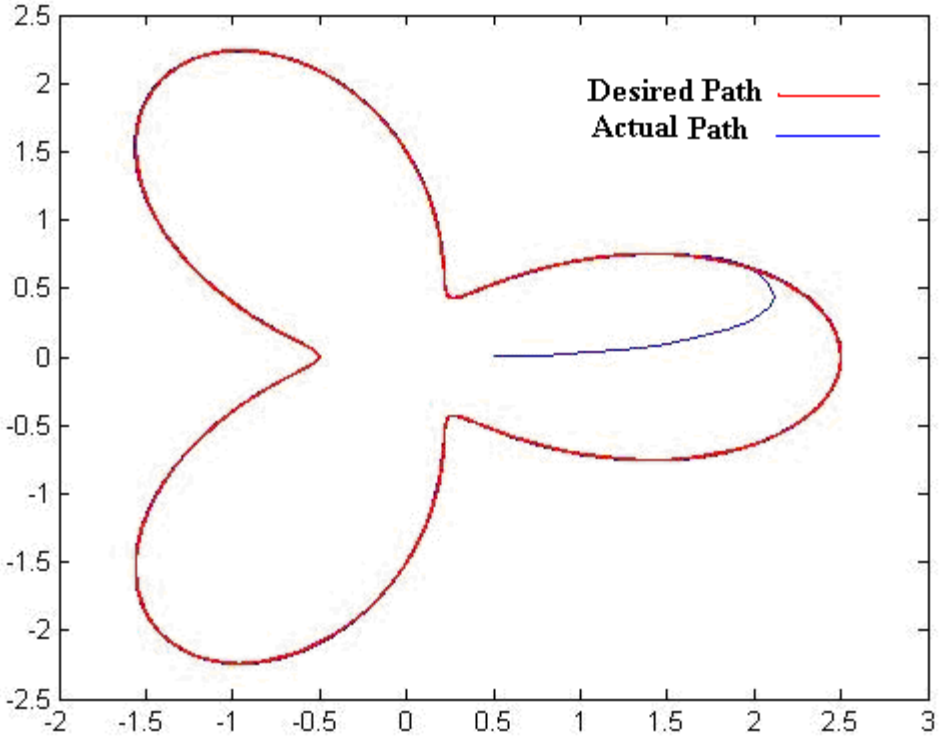

Figure 14. Output Tracking at a Frequency of 400 with Feed Forward

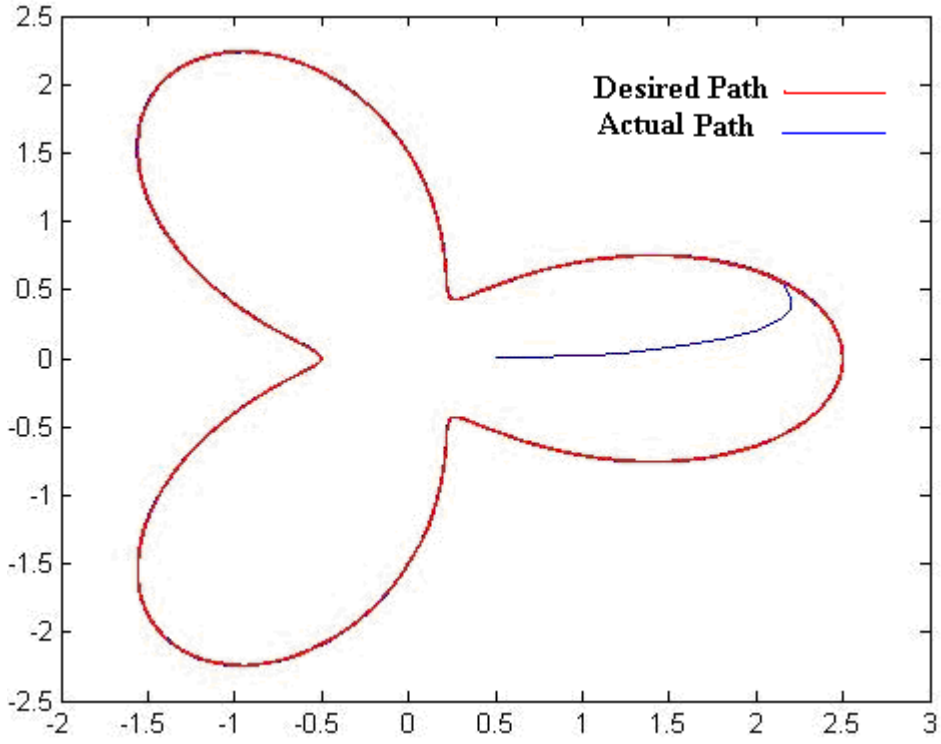

Figure 15. Output Tracking at a Frequency of 500 with Feed Forward

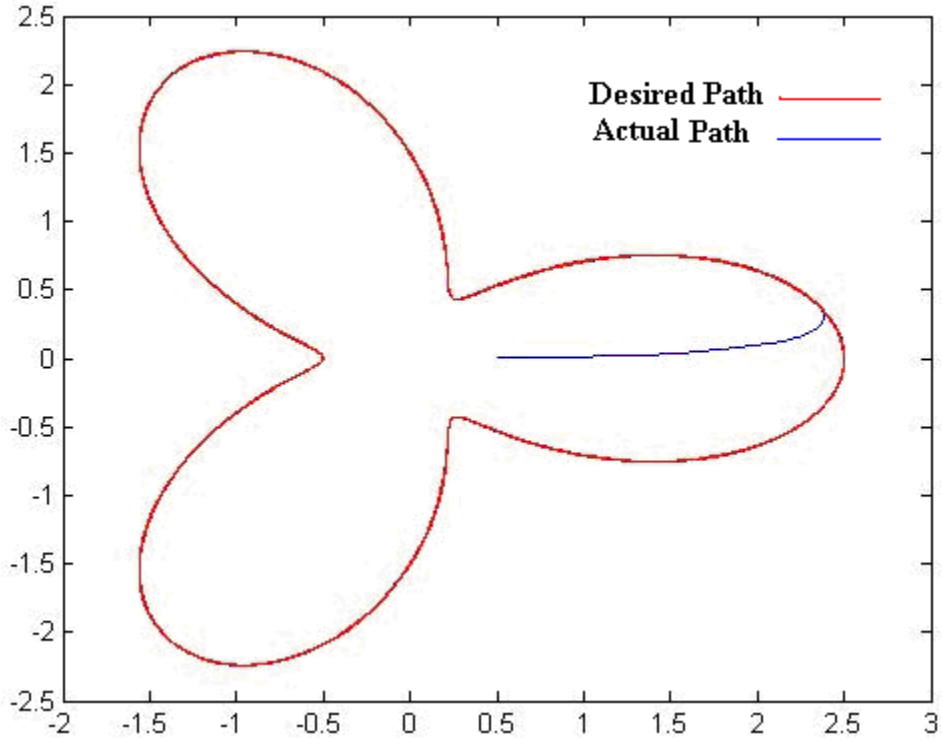

Figure 16. Output Tracking at a Frequency of 1000 with Feed Forward

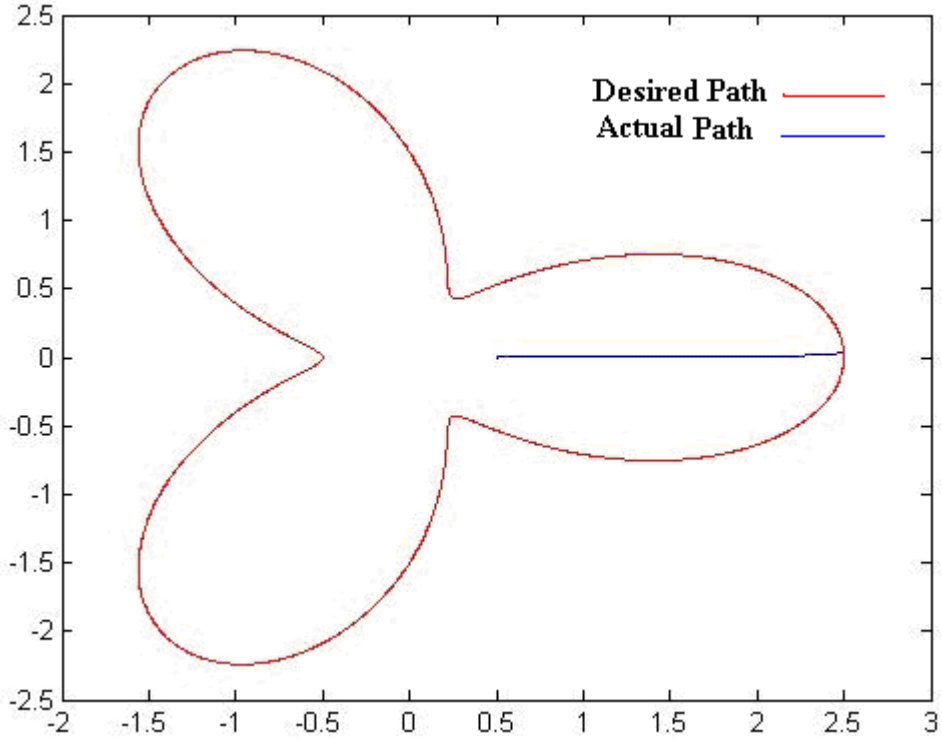

Figure 17. Output Tracking at a Frequency of 10000 with Feed Forward

After adding the feed forward to the system, we can see that the output tracks the desired pattern much faster, as we expected, thus giving the system more robustness and accuracy.

7.3 Adding Disturbance to the System

To simulate any type of external disturbance that the system could encounter, in this section we will add a sinusoidal disturbance to the system to throw off the balance of the ideal system and help prove the robustness of the ADRC control.

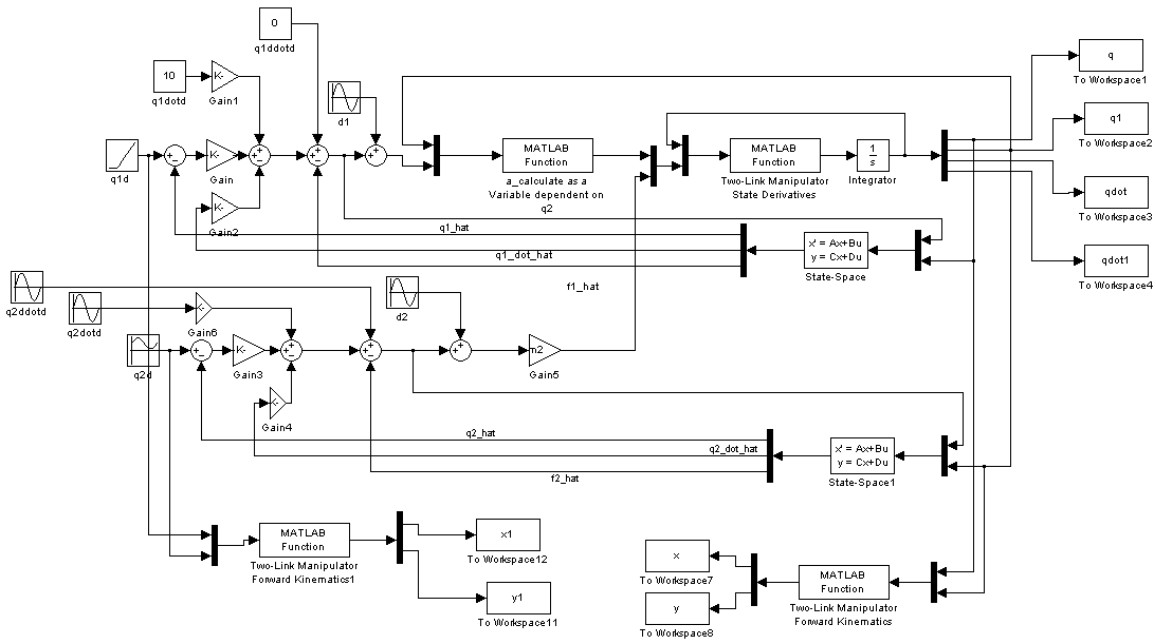

Figure 18. System Setup with Sinusoidal Disturbance

We now run the system under the same tuning parameters used in both prior experiments.

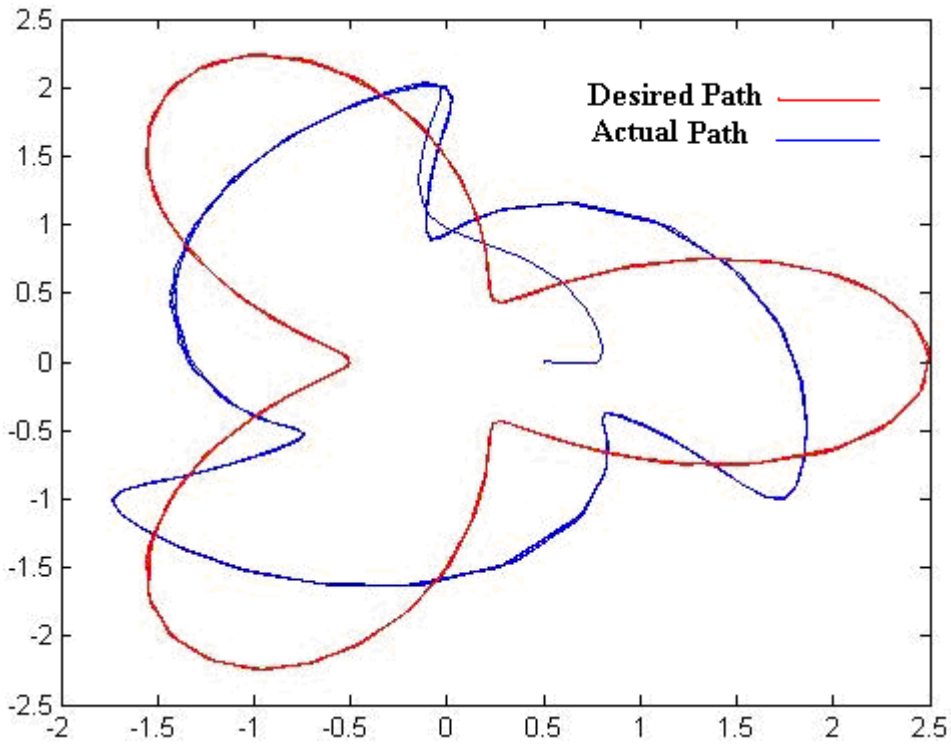

Figure 19. Output Tracking at a Frequency of 100 with Disturbance

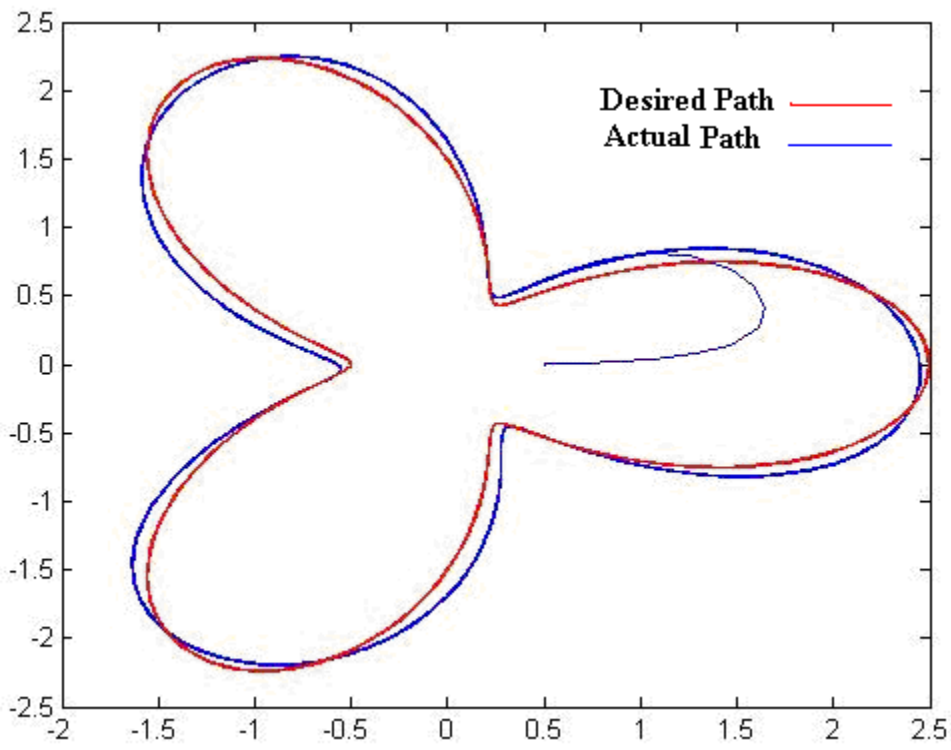

Figure 20. Output Tracking at a Frequency of 200 with Disturbance

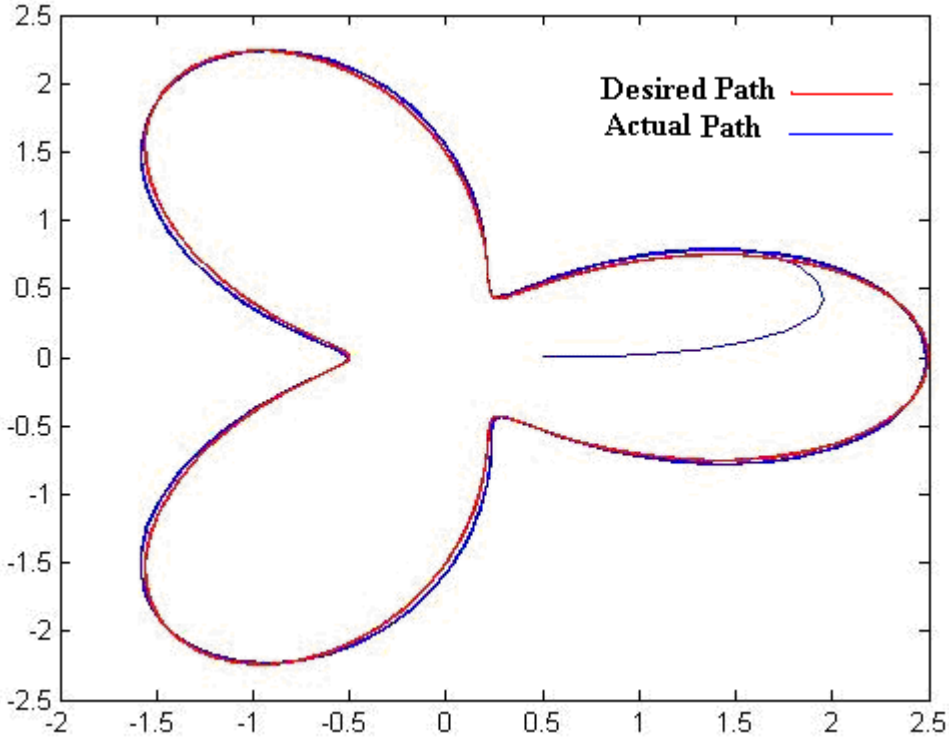

Figure 21. Output Tracking at a Frequency of 300 with Disturbance

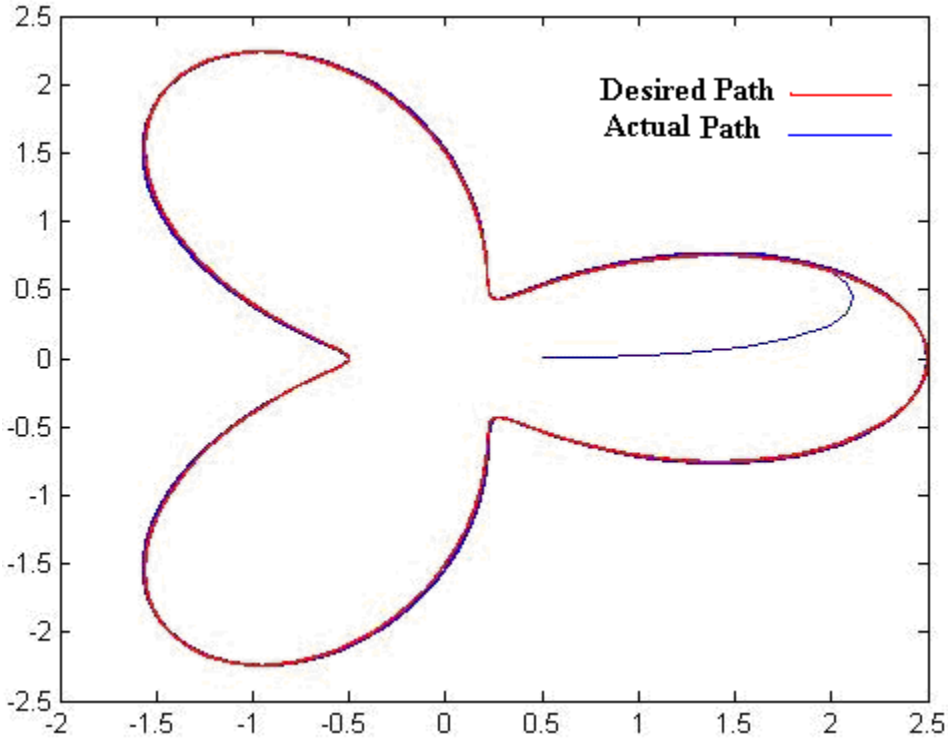

Figure 22. Output Tracking at a Frequency of 400 with Disturbance

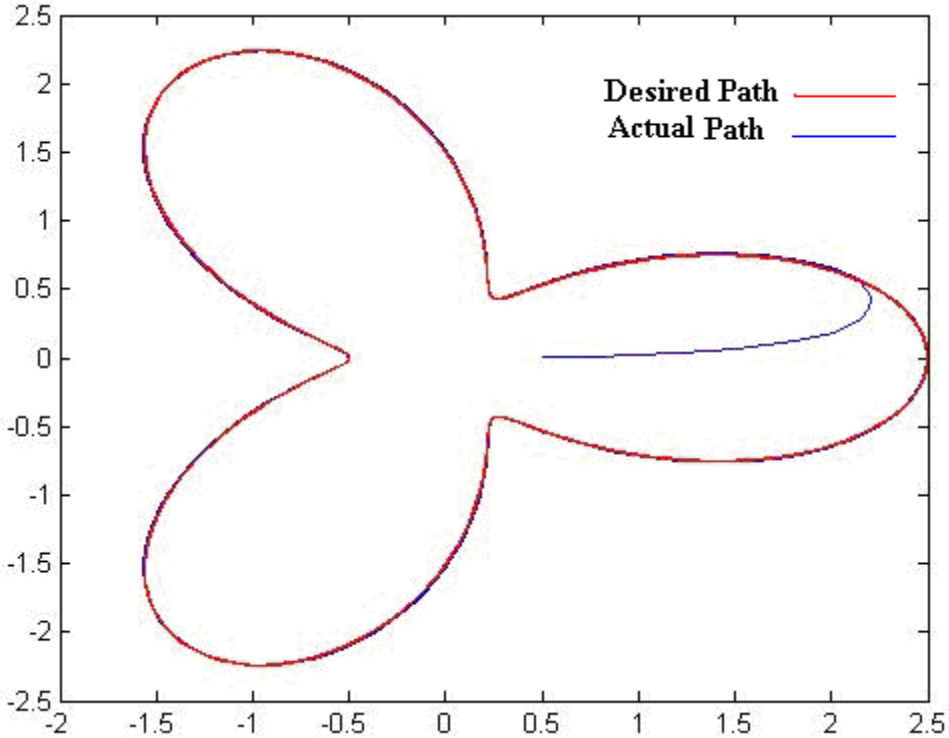

Figure 23. Output Tracking at a Frequency of 500 with Disturbance

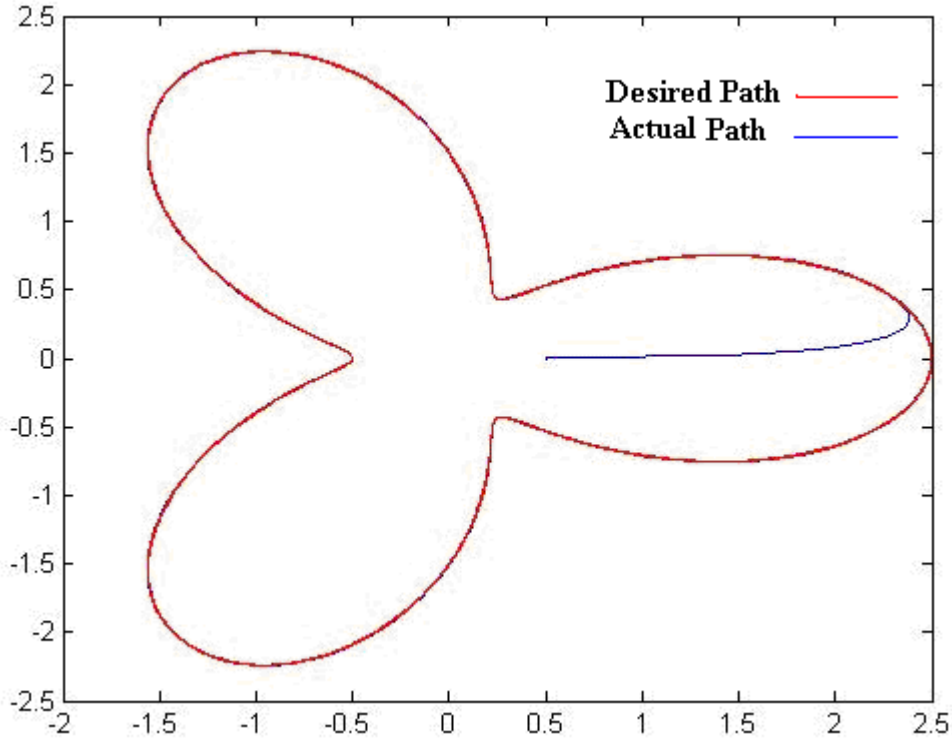

Figure 24. Output Tracking at a Frequency of 1000 with Disturbance

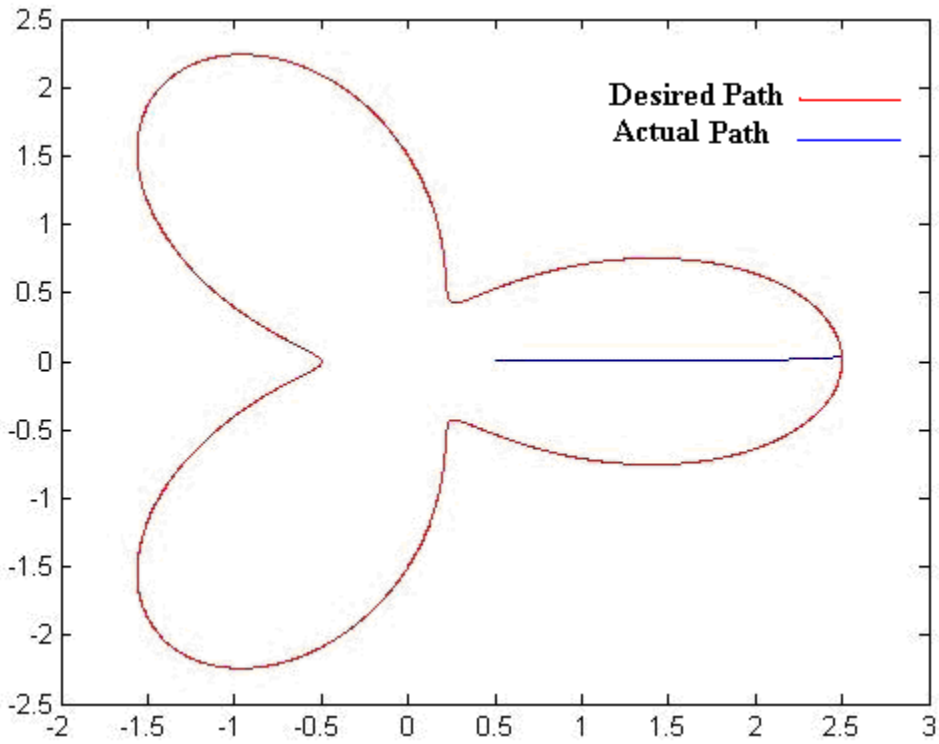

Figure 25. Output Tracking at a Frequency of 100000 with Disturbance

7.4 Adding Noise to the System Simulation of Sensor Noise

These different simulations do not take into consideration any of the sensor noise that could be inserted from the encoder sensor. To account for extraneous noise, we will add noise to the system and simulate it again at all the same frequencies, and any other frequency needed to obtain the best result as shown below:

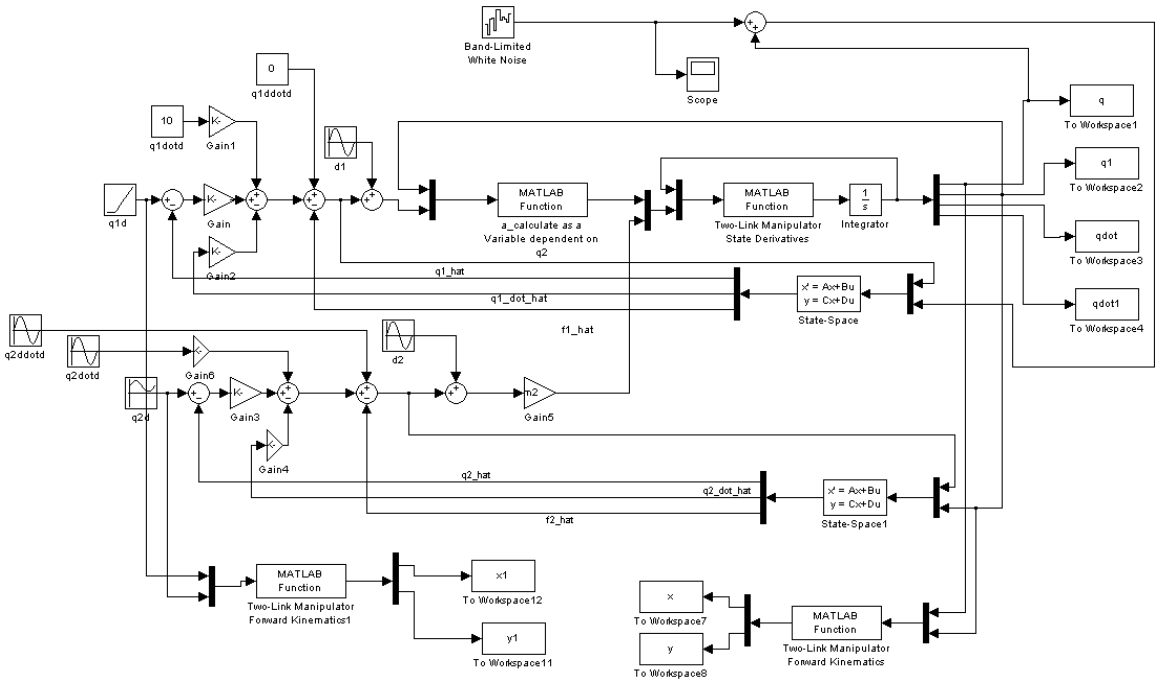

Figure 26. System Setup with Additional Noise

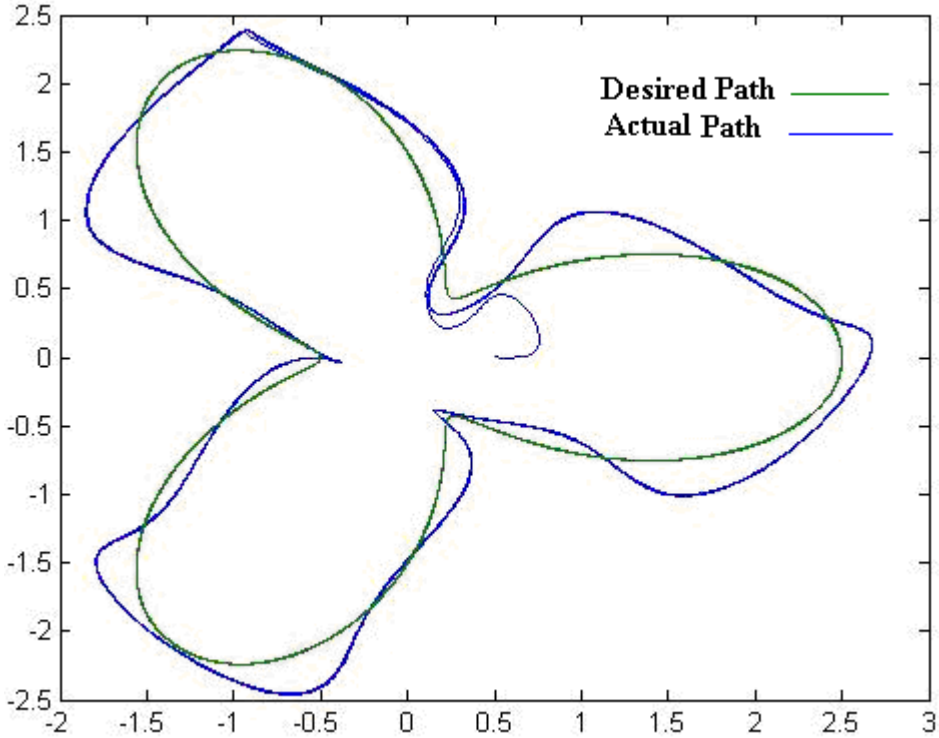

Figure 27. Output Tracking at a Frequency of 100 with Noise

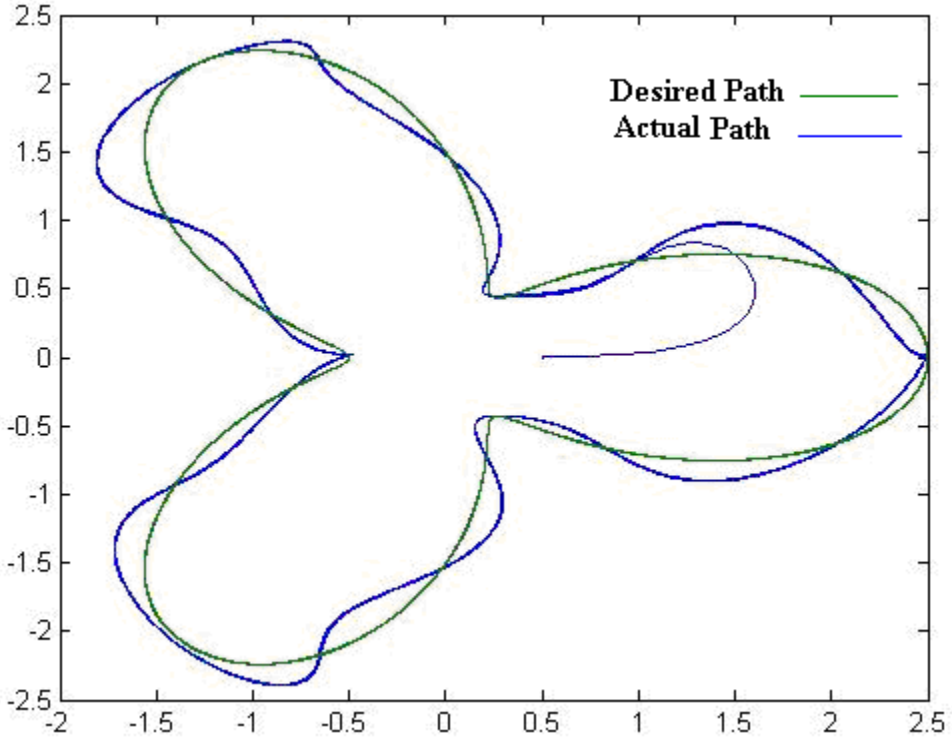

Figure 28. Output Tracking at a Frequency of 200 with Noise

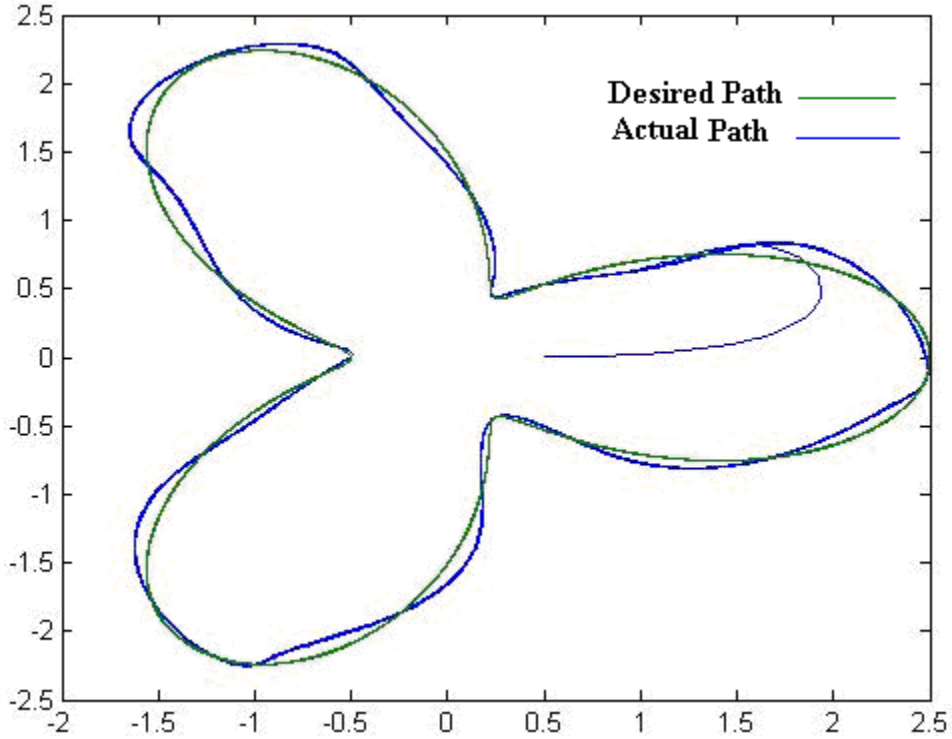

Figure 29. Output Tracking at a Frequency of 300 with Noise

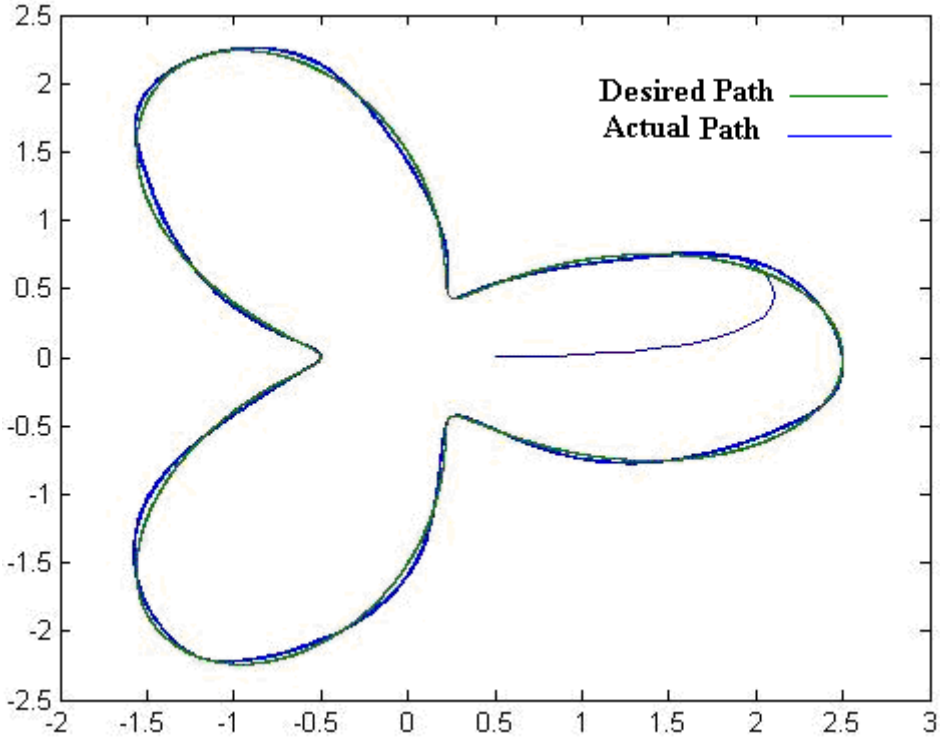

Figure 30. Output Tracking at a Frequency of 400 with Noise

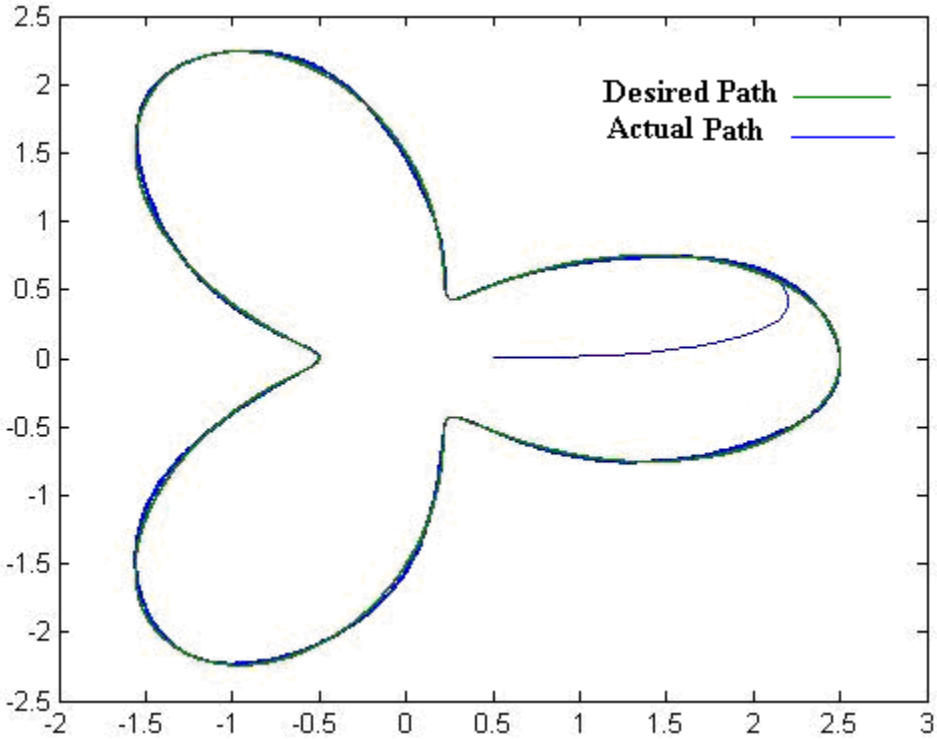

Figure 31. Output Tracking at a Frequency of 500 with Noise

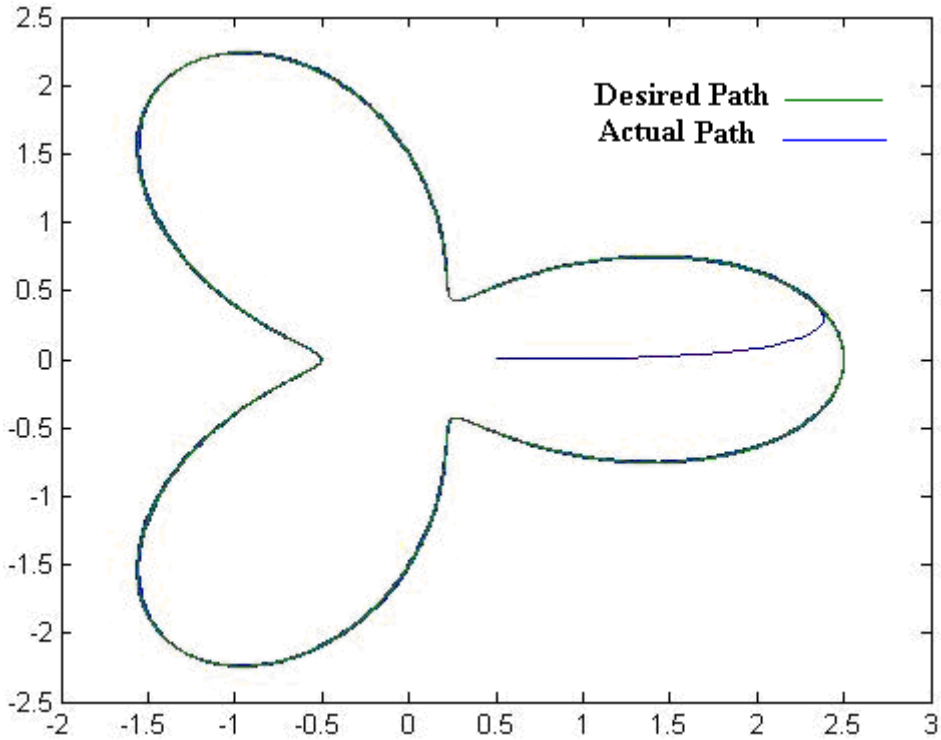

Figure 32. Output Tracking at a Frequency of 1000 with Noise

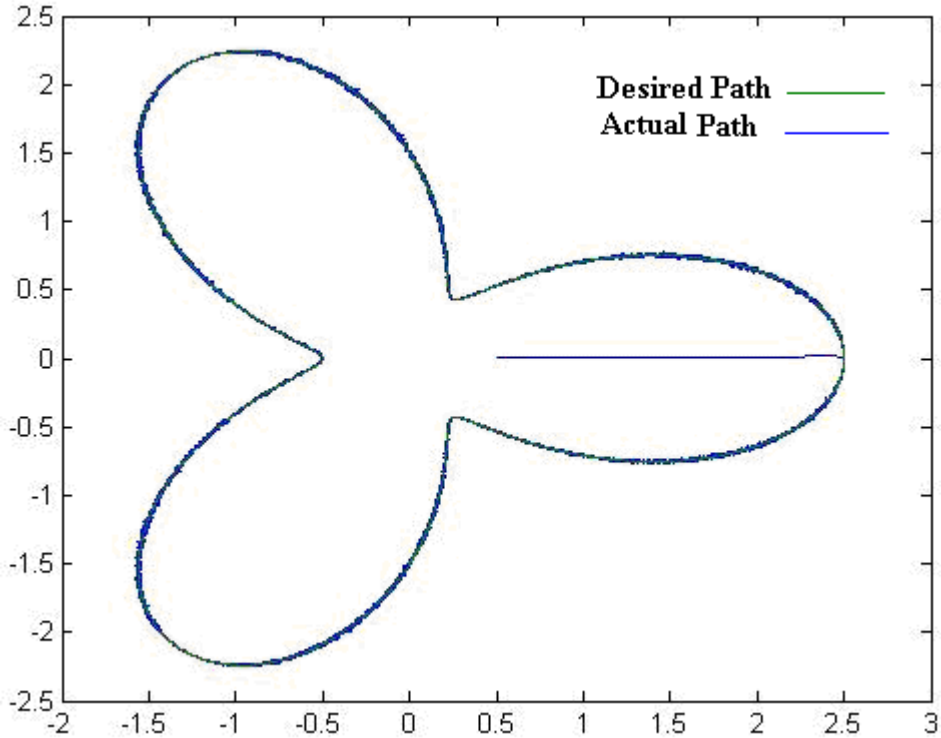

Figure 33. Output Tracking at a Frequency of 1000 with Noise

7.5 Robust-Passivity Based Control Simulation

In this section of the simulation, the RP manipulator was controlled using a previously studied control system. A robust-passivity based control of an RP system was studied and simulated in previous studies of robotic control. The simulation Simulink file is demonstrated below (Figure 34).

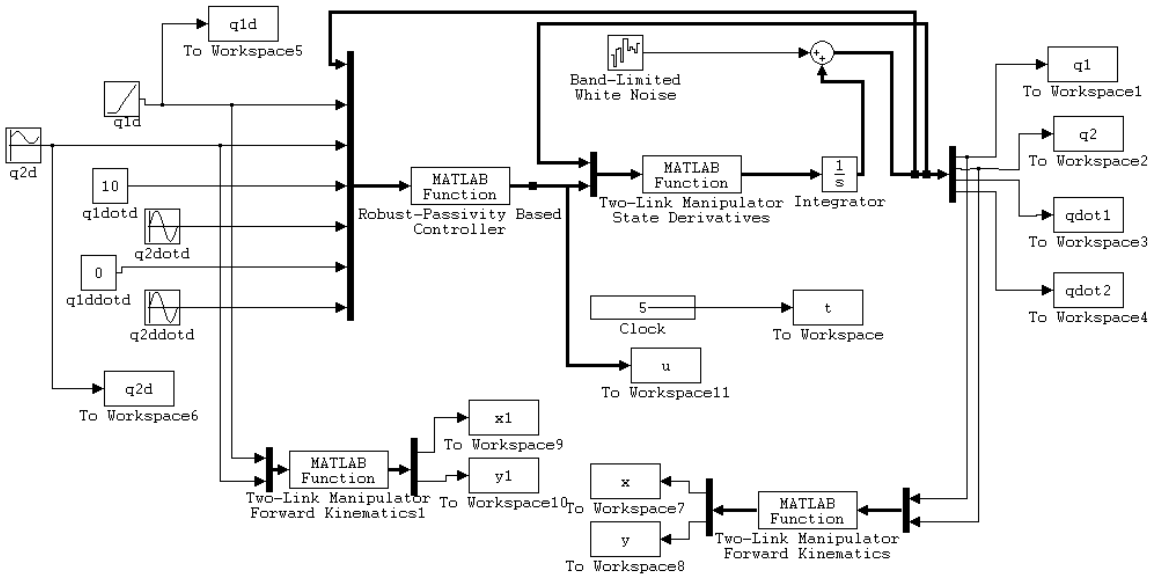

Figure 34: Robust-Passivity Based Control of an RP manipulator

The results of the simulation shown below represent the end-effector position tracking (Figure 35). The system was reproduced at an optimized gain of 600 rad/sec. It is worth noting that the same sensor noise that was used in section 7.4 is used in this in this simulation. The end-effector position tracking is shown in full scale (Figure 35); nevertheless, a closer look was obtained using the zoom feature of the graph (Figure 36).

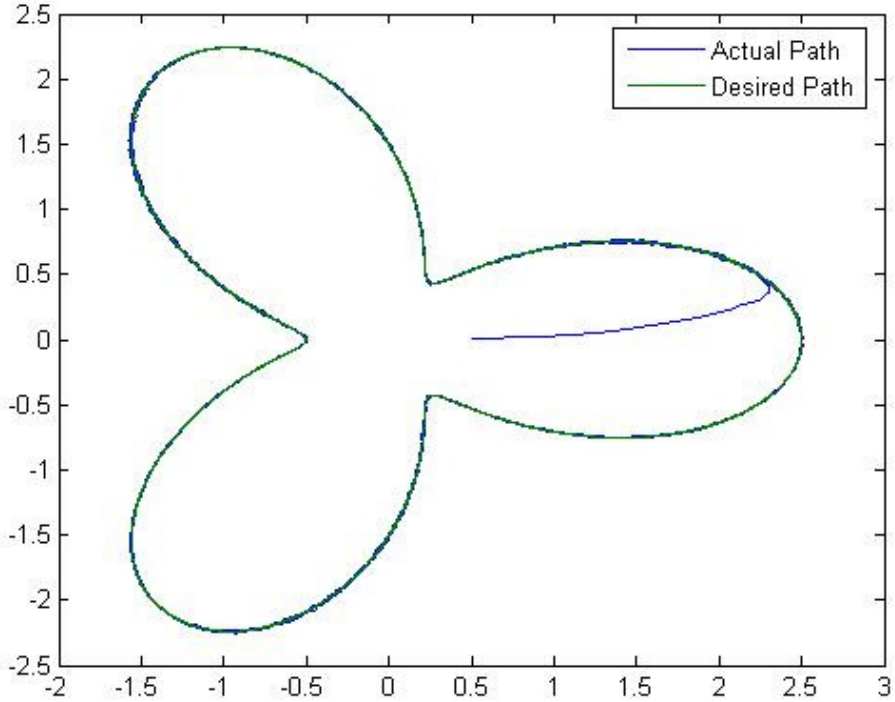

Figure 35: Using Robust-Passivity Based Control with noise

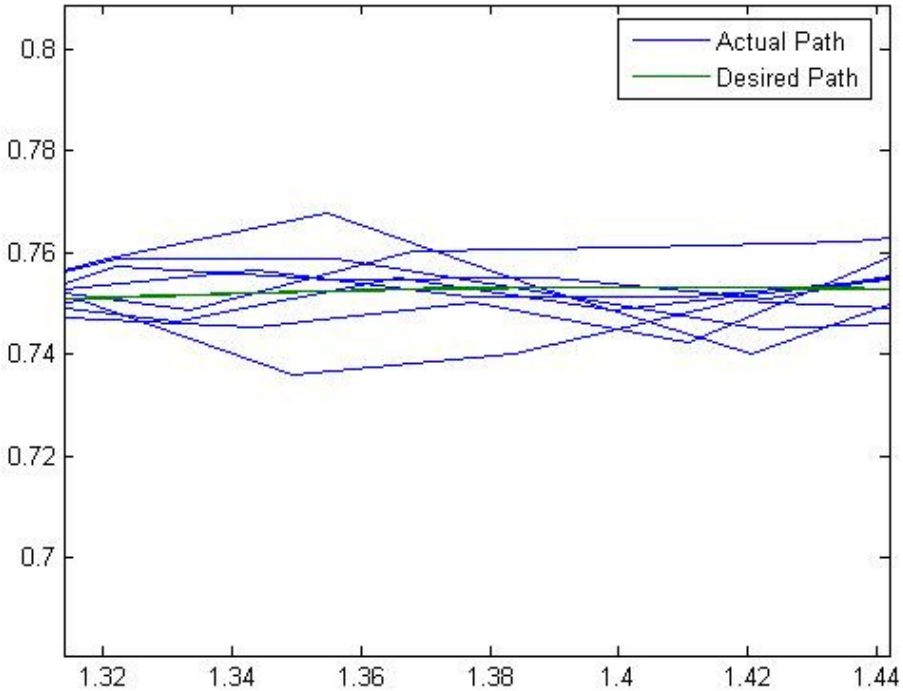

Figure 36: Robust-Passivity Based control Zoom in with noise

CHAPTER VIII

EVALUATION OF THE SIMULATION RESULTS

8.1 Evaluation of the Ideal System Simulation

After running the ideal system under different bandwidth frequencies, we can analyze the end-effector at both low and high frequencies. If we take a look at the system ran under 100 rad/sec frequency (Figure2), we can see that the output profile does not follow desired input profile very well. As the frequency is increased, the performance of the tracking improves dramatically and it is perfect at a frequency of 10000 rad/sec (Figure 9) where the tracking profile of the end-effector position tracked very well with no error.

8.2 Evaluation of the System after Adding Feed Forward

To increase the accuracy of tracking the profile, we could add the feed forward feature into the ideal system to improve tracking at low frequencies and to reduce the bandwidth needed to obtain a perfect system. After running the simulation with feed forward at a frequency of 100 rad/sec (Figure 11), the system was much improved with the feed forward and the system became perfect at the frequency of 500 rad/sec (Figure 15). By adding the feed forward to the system, the tracking became much more accurate at lower frequencies and reduced the need of large bandwidth to control the system.

8.3 Evaluation of the System after Adding External & Internal Disturbances

After the analysis of the system in the ideal world where disturbances don't exist, the system must be evaluated in the real world applications; in such an environment it is unavoidable to have external and internal disturbances. A sinusoidal external disturbance was added to the system as well as a few changes to the internal parameters of the mathematical model and to the internal geometry of the system. After the addition of the disturbances to the model the simulation was run again with the same bandwidth that we ran in the ideal system analysis; as a result the system was out of control at the frequency of 100 rad/sec (Figure 19). However, as we gradually increased the frequency, the system regained control and at the frequency of 500 the system was fully under control with a minimal amount of error.

This simulation showed the abilities of ADRC to control the system and to keep it in control under large changes to the system; furthermore, it showed immunity to any external disturbance. One of the features of ADRC is its capability of rejecting any kind of disturbance, internal or external. The changes of link weight, link height, and any changes to the function were compensated for without any problem.

8.4 Evaluation of the System after Introducing Sensor Noise

To complete the analysis of the real world system, we must add sensor noise to show the effects of output noise on the control system. To accomplish this, a white noise was added to the previous system simulation and the system was run with the same bandwidth used before. At the lower frequency levels the system lost control again, showing the effects of real world analysis on the system. At the frequency of 100 rad/sec the end-effector position couldn't follow the desired input tracking.

As the bandwidth was increased the system showed much improvements, but the system went out of control again at a large bandwidth. This means that there is a cutoff frequency that was developed during this simulation. If we take a closer look at the 1000 frequency, we can see that the noise in the encoder has distorted the output, which tells us that we need to find the correct tuning parameter below the cutoff frequency of 1000. This simulation gives us a better understanding of the real world system where too much control could cause the system to go out of control due to the noise factor in the sensors as shown in the following figures:

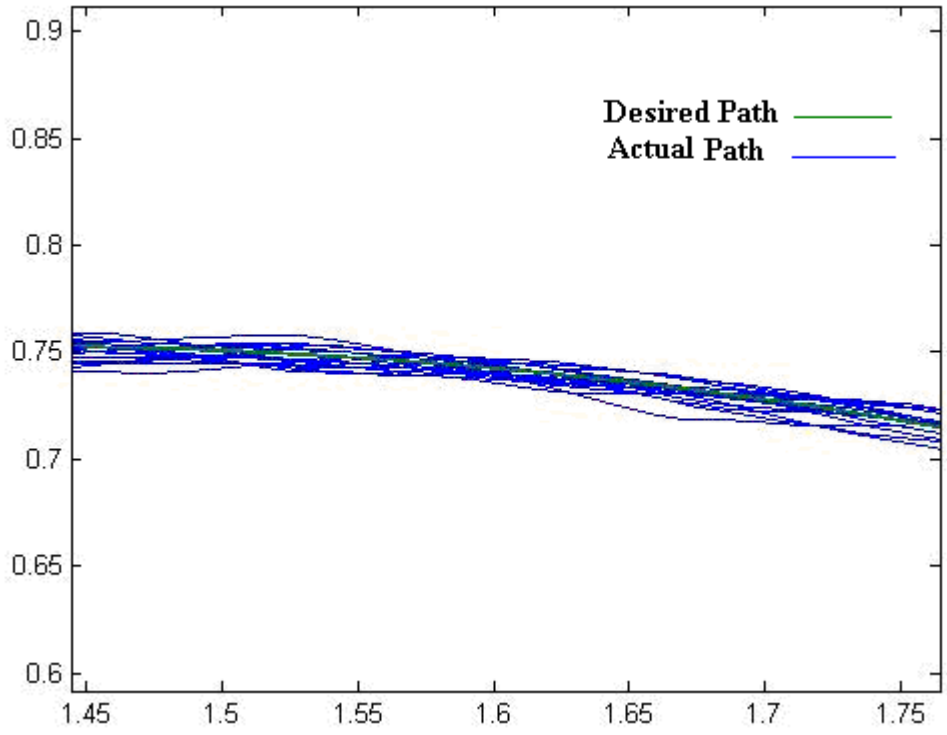

Figure 37. Zoom in of the Output Tracking at a Frequency of 1000 with Noise

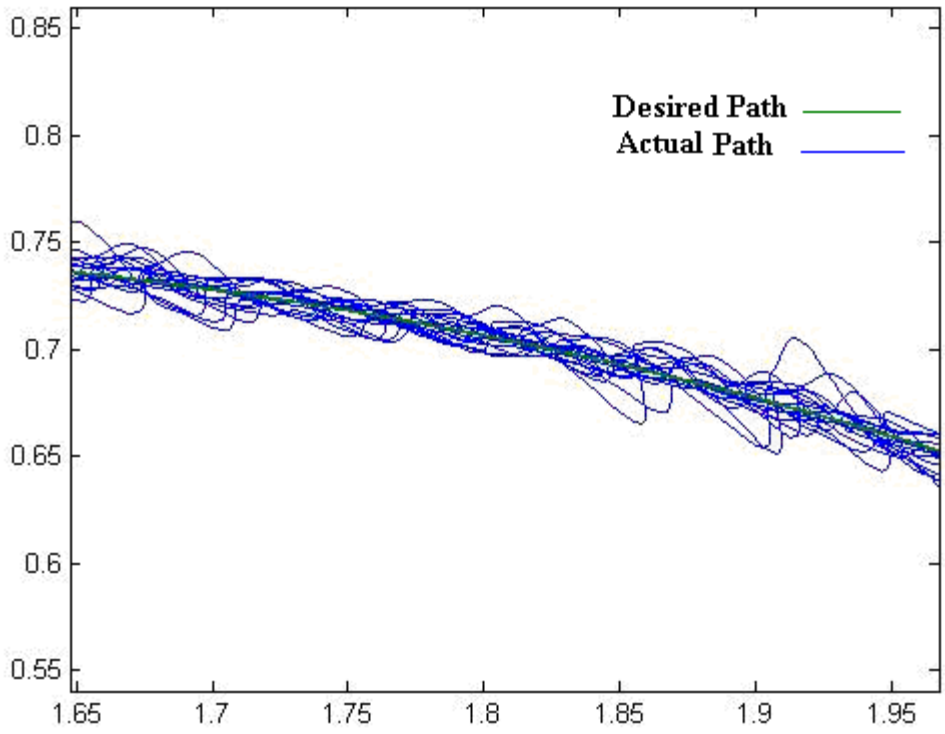

Figure 38. Zoom in of the Output Tracking at a Frequency of 10000 with Noise

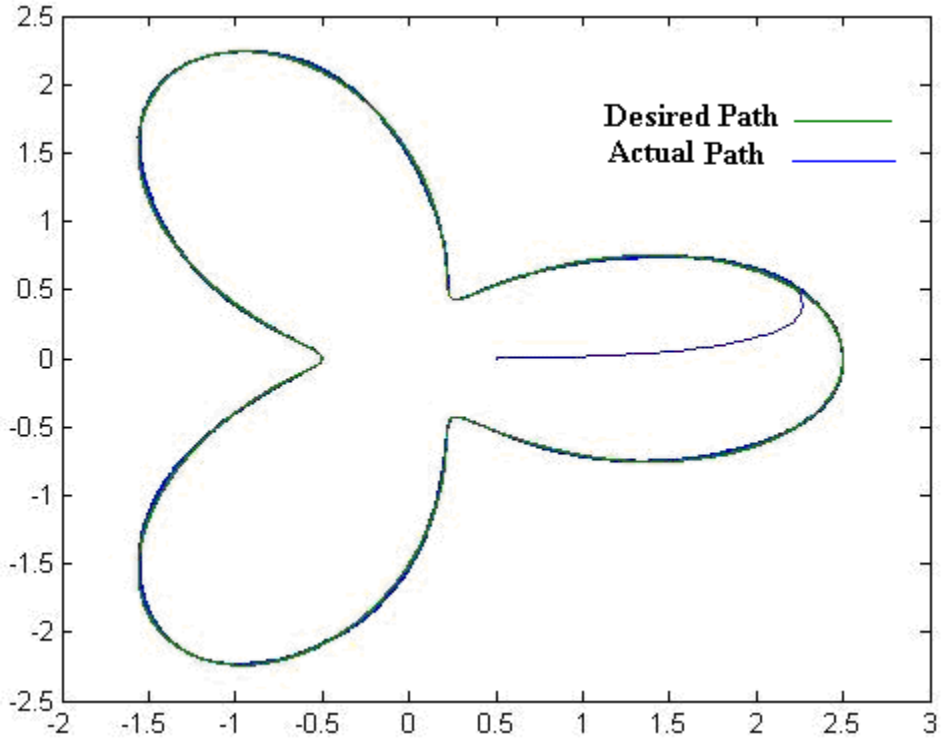

Figure 39. Output Tracking at a Frequency of 600 with Noise

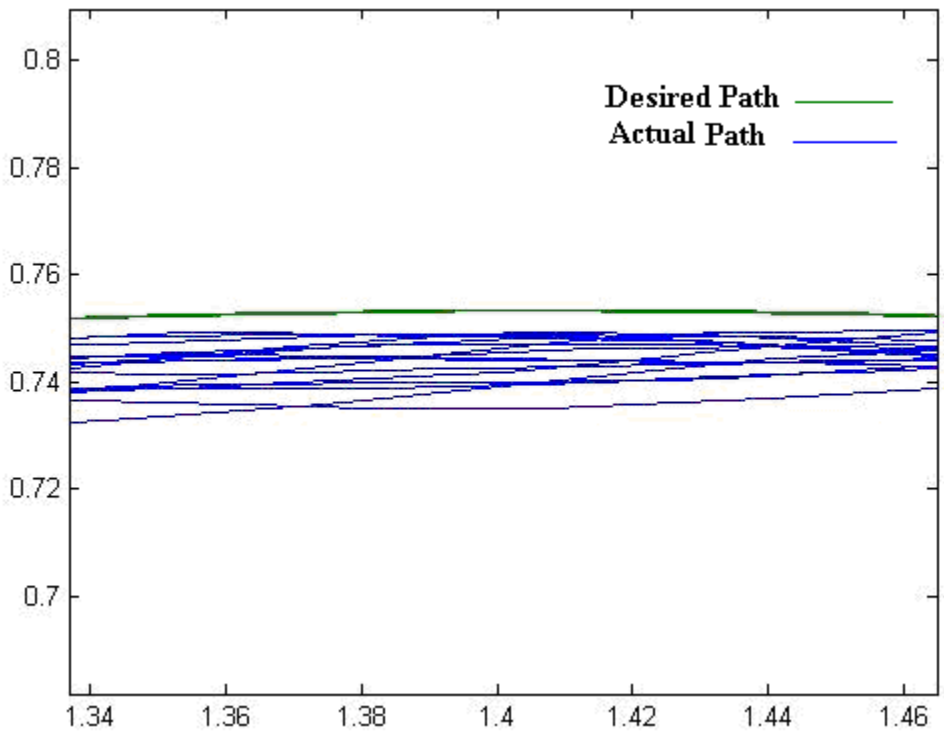

Figure 40. Zoom in of the Output Tracking at a Frequency of 600 with Noise

As we see in the graphs above, the optimal control frequency is below 1000 and above 500. 600 is an appropriate frequency to compensate between the disturbance and the noise. Applying too little control would not give an accurate path; applying too much control would cause distortion to the path. This gives us a control window to obtain the best control for the system in the situation. We should note that this case of extreme noise normally would not occur unless the sensor was going bad.

8.5 Evaluation of a Robust-Passivity Based control with Sensor Noise

After simulating the Robust-Passivity Based control with sensor noise using the same noise ratio as in the previous system; the end-effector position was plotted (Figures 35). A closer look was obtained using the zoom-in feature in Matlab (Figure 36); in which the actual path is colored in blue and the desired path is in green. The actual path

is not very accurate and the tracing was all over the place. This indicates that the system was not in total control due to the sensor noise. Thus, the Robust-Passivity Based control was not able to reject all the sensor noise and obtain a controlled system.

8.6 Active Disturbance Rejection Control vs. Robust-Passivity Based Control

When comparing Figure 36 and 40 -in which the amount of noise and the bandwidth are the same- one can see that Figure 40 is under control and is well tracing the desired path, whereas in Figure 36 the system is not totally under control and the tracking is all over the place. Figure 36 represents the Robust-Passivity Based Control, while Figure 40 represents Active Disturbance Rejection Control. Based on this analysis; one can demonstrate that ADRC is a superior technology to Robust-Passivity Based Control. This suggests that ADRC is a new and revolutionary control system that could be superior over the Robust-Passivity Based Control.

CHAPTER IX

CONCLUDING REMARKS

ADRC is a new, upcoming technology that is very promising. It has replaced PID control in many applications, but in the industrial world it is very difficult to break free from old habits and to replace a decent working control system with a new and nonconventional control solution. However, in the robust world of robotics it is always possible to change a control system because accuracy and repeatability are the golden rules of the system; the smallest malfunction could cause significant damage in material as well as humans nearby.

In this thesis and in my simulations I have proven that ADRC is capable of providing both accuracy and repeatability with a small marginal error that could be tuned to perfection. During my evaluations of the different possibilities of disturbances and noise, Active Disturbance Rejection Control showed very good capabilities of ignoring any noise and disturbances and keeping the system under control at all times in the correct tuning bandwidth.

The control system was able to maintain control under internal system changes as well; where ADRC does not care about the geometry of the system or the accuracy of the mathematical model, thus giving the world of control flexibility and the ability to obtain a controlled system of a model that is missing some of its model features. This capability could be very useful in a situation where we want to use a robotic arm in a different planet and the surrounding matter is not known exactly or is wrongly estimated.

During this research, the mathematical model and the simulation of the system were developed; nevertheless no real demonstration was done on an actual robotic manipulator. Since no studies were done on an actual physical system; only the end-effector tracking was studied and the stability of the system was not tested. Therefore, these topics could be part of a further study or a future research project. However, such a study could be very costly in terms of equipments and tools.

During the simulation, a comparison was developed between Active Disturbance Rejection Control (ADRC) and Robust-Passivity Based Control System. This study concluded that ADRC can be more reliable system that could revolutionize the future and the world as we know it today. ADRC is uniquely designed in such a way that no system has yet been proven more powerful. Nevertheless, the traditional system as we know is very hard to replace due to its proven effectiveness and low cost.

In a simple cylindrical robotic arm the geometry is well known and easy to make a control system for such a robot. Nevertheless, the robotic world has advanced to a much more complicated geometries that make it harder to develop a sufficient mathematical model for the control system; therefore making it difficult to obtain a good control system that would have high levels of accuracy and reliability. ADRC has proven the ability to

disregard the geometry factor of the robot and even to disregard any disturbance; therefore, ADRC was successfully shown to be useful in making a control system that can be very accurate and reliable regardless of what the math model looks like. The RP manipulator study was a demonstration of the power of ADRC. The capabilities of such a system do not stop at a two-link manipulator but could be expanded into multi-link manipulators that could be studied in further doctorial level.

BIBLIOGRAPHY

[1] Smith, Stephanie. *"A Short History of Robots*". Available: http://prime.jsc.nasa.gov/ROV/history.html

[2] *Robot Worx, A Brief History of Robots*. Available: http://www.robots.com/robotics.php?page=history

[3] Spong, Mark W., Seth Hutchinson, and M. Vidyasagar. *"Robot Modeling and Control"* Hoboken: John Wiley & Sons Inc., pp. 5-150, 2006.

[4] Gao, Zhiqiang. *"Scaling and Bandwidth Parameterization-based Controller Tuning"* Cleveland: American Control Conference, Dept. of Electrical and Computer Engineering, Cleveland State U, pp. 2-8, 2003.

[5] Tian, Gang and Zhiqiang Gao. *"Frequency Response Analysis of Active Disturbance Rejection Based Control System"* Singapore: 16th IEEE International Conference on Control Applications, Part of IEEE Multi-conference on Systems and Control, pp. 2-4, October 2007.

[6] Zheng, Qing (Student Member IEEE), Lili Dong, and Zhiqiang Gao (Members IEEE). *"Control and Rotation Rate Estimation of Vibrational MEMS Gyroscopes"* Singapore: 16th IEEE International Conference on Control Applications, Part of IEEE Multi-conference on Systems and Control, pp. 2-5, October 2007.

[7] Zhou, Wankun and Zhiqiang Gao. *"An Active Disturbance Rejection Approach to Tension and Velocity Regulations in Web Processing Lines"* Cleveland: Department of Electrical and Computer Engineering, Cleveland State U. pp. 2-6, October 2007.

[8] Chen, Zhongzhou, Qing Zheng (Student Member IEEE), and Zhiqiang Gao (Member IEEE). *"Active Disturbance Rejection Control of Chemical Processes"* Singapore: 16th IEEE International Conference on Control Applications, Part of IEEE Multi-Conference on Systems and Control, pp. 3-6, October 2007.

[9] Gao, Zhiqiang. *"Active Disturbance Rejection Control: A Paradigm Shift in Feedback Control System Design"* Cleveland: Center for Advanced Control Technologies, Cleveland State U, pp. 2-6, June 2006.

[10] Alexander, Baixi Su-. *"Control of Active Magnetic Bearings for Flywheel Energy Storage"* Cleveland State U, pp. 17-69, July 2006.

APPENDIX

A. Derivation of Two-Link Manipulator State

The two-link manipulator State Derivative is defined as shown below:

$$function\ \dot{z} = stateder(t, z, u)$$
$$m_1 = 10; m_2 = 8; a_1 = 0.6; a_2 = 0.4; I_{1z} = 26; I_{2y} = 12; g = 9.8$$
$$q_1 = z(1); q_2 = z(2); \dot{q}_1 = z(3); \dot{q}_2 = z(4);$$

Finding numerical values for matrices M, C and calculating state derivatives,

$$D = \left[\frac{1}{4} m_1 * a_1^2 + m_2 * q_2^2 + m_2 * q_2 * a_1 + \frac{1}{4} m_2 * a_1^2 + I_{1z} + I_{2y}, 0; 0, m_2\right];$$
$$C = \left[\left(m_2 * q_2 + \frac{1}{2} m_2 * a_1\right)\dot{q}_2, \left(m_2 * q_2 + \frac{1}{2} m_2 * a_1\right)\dot{q}_1 ; \left(-m_2 * q_2 - \frac{1}{2} m_2 * a_1\right)\dot{q}_1, 0\right];$$
$$G = \left[\frac{1}{2} m_1 * g * a_1 * \cos(q_1) + m_2 * g * \left(\frac{1}{2} * a_1 + q_2\right)\cos(q_1); m_2 * g * \sin(q_1)\right];$$

$$\dot{z} = \left[[\dot{q}_1; \dot{q}_2]; inv(D) * (u - C * [\dot{q}_1; \dot{q}_2] - G)\right];$$

The (a) term calculation is defined as shown below:

```
function u=a_cal(q2,u0)

m1=11.436; m2=2.9613; a1=1.0244; a2=0.49416; g=9.8; I1z=12.972; I2y=21.411;

u=(1/4*m1*a1^2+m2*q2^2+m2*q2*a2+1/4*m2*a1^2+I1z+I2y)*u0;
```

The forward kinematics of the two-link manipulator is shown below:

```
function end_eff_pos=forward_kin_2link(q)

Two-link planar manipulator:

Calculate the position of the origin of the end-effector

a1=0.6; a2=0.4; %using nominal quantities

x=((a1/2)+q(2)+(a2/2))*cos(q(1));
```

```
y=((a1/2)+q(2)+(a2/2))*sin(q(1));

end_eff_pos=[x;y];
```