



YILDIZ TECHNICAL UNIVERSITY
ELECTRICAL AND ELECTRONICS FACULTY
DEPARTMENT OF COMPUTER ENGINEERING

SENIOR PROJECT

KINEMATIC ANALYSIS FOR ROBOT ARM

Project Manager : Assist.Prof.Sırma Ç. Yavuz

Project Group

04011503 COŞKUN YETİM

İstanbul, 2009

CONTENTS

| | |
|---|------|
| List of Abbreviations | iv |
| List of Figures | v |
| List of Tables..... | vii |
| Acknowledgements | viii |
| Abstract | ix |
| Özet | x |
| 1. Introduction | 1 |
| 2. Feasibility of The Project | 3 |
| 2.1. Software Feasibility | 3 |
| 2.2. Hardware Feasibility | 3 |
| 2.3. Technical Feasibility | 4 |
| 2.4. Economical Feasibility | 4 |
| 2.5. Legal Feasibility | 5 |
| 3. Basic Manipulator Geometries | 6 |
| 3.1. Open Chain Manipulator Kinematics..... | 7 |
| 3.2. Closed Chain Manipulator Kinematics..... | 7 |
| 4. Homogeneous Transformations..... | 8 |
| 4.1. Right Handed Coordinate Systems | 8 |
| 5. Forward and Inverse Kinematics | 12 |
| 5.1. Forward Kinematics | 12 |
| 5.2. Inverse Kinematics | 14 |
| 5.2.1. Solving The Inverse Kinematics | 15 |
| 5.2.1.1. Analytic Method | 15 |
| 5.2.1.1. Inverse Jacobian Method | 16 |
| 6. Implementation | 18 |
| 6.1. Software Implementation | 18 |

| | |
|------------------------------------|----|
| 6.2. Hardware Implementation | 20 |
| 7. Conclusion | 21 |
| References | 22 |
| CV | 23 |

LIST OF ABBREVIATIONS

| | |
|-----|-----------------------------|
| FK | Forward Kinematics |
| IK | Inverse Kinematics |
| IPK | Inverse Position Kinematics |

LIST OF FIGURES

| | |
|--|----|
| Figure 1.1 Basic robot arm. 1. Base, 2. joint, 3. link and the last part, grapper..... | 2 |
| Figure-2.1 Microcontroller..... | 3 |
| Figure-2.2 Servo machines..... | 3 |
| Figure-2.3 A sheet of plastic..... | 4 |
| Figure-3.1 Open chain serial robot arm..... | 7 |
| Figure-3.2 Stewart platform..... | 7 |
| Figure-4.1 Homogeneous Transformation matrix..... | 8 |
| Figure-4.2 Using the right hand rule to compute the direction of the z axis..... | 9 |
| Figure-4.3 Using the right hand rule to compute the direction of any axis given the directions of the other two..... | 9 |
| Figure-4.4 The right hand rule to determine the direction of positive angles. Point your right thumb along the positive direction of the axis you wish to rotate around. Curl your fingers. The direction that your fingers curl is the direction of positive rotation.... | 9 |
| Figure-4.5 Two coordinate frames that differ by only a translation..... | 10 |
| Figure-4.6 A simple arm..... | 11 |
| Figure-4.7 The robot arm from figure 3.6 with the joint rotated by ψ degrees..... | 11 |
| Figure-5.1 Right angle triangle..... | 13 |
| Figure-5.2 A simple forward kinematics | 13 |
| Figure-5.3 Forward kinematics by composing transformations..... | 14 |
| Figure-5.4 Inverse kinematics | 14 |
| Figure-5.5 Analytic Method to solve inverse kinematics. | 15 |

| | |
|--|----|
| Figure-5.6 Cosine Law. | 15 |
| Figure-5.7 Iteratively solution of inverse kinematics..... | 17 |
| Figure-6.1 User interface..... | 18 |
| Figure-6.2 Serial port configuration..... | 19 |
| Figure-6.3 Proteus view..... | 20 |

LIST OF TABLES

| | |
|---|----|
| Table 2.1 Minimum requirements for the project..... | 4 |
| Table 2.2 Software cost and hardware cost..... | 5 |
| Table 3.1. Manipulator kinematic..... | 6 |
| Table 6.1 Packet contents..... | 19 |

ACKNOWLEDGEMENTS

I would like to thank to my project supervisor Assistant Proffessor Sırma Ç. Yavuz (Yıldız Technical University). During the project development progress, she was very patient and helpful. Always she directed me correctly.

I would like to thank to Erkan Uslu who is a research member in the university. I appreciate that he shared his knowledge and experiences. He has been ready for my questions although sometimes they were boring.

I would like to thank Alperen Bal for the workspace paints.

ABSTRACT

In this project, I researched the kinematic analysis of robot arm. The kinematic analysis is the relationships between the positions, velocities, and accelerations of the links of a manipulator. The kinematics separate in two types, direct kinematics and inverse kinematics. In forward kinematics, the length of each link and the angle of each joint is given and we have to calculate the position of any point in the work volume of the robot. In inverse kinematics, the length of each link and position of the point in work volume is given and we have to calculate the angle of each joint.

The forward kinematic analysis is not difficult to solve. It is solved by using simple homogeneous matrices. On the other hand, the inverse kinematics is so hard to solve and it will be harder if we increase the freedom degrees. There are different methods to solve the inverse kinematics. The analytic method and Jacobian method are well-known. In the project I used the analytic method.

In the thesis application, I designed a prototype robot arm with 3 freedom degrees. User interface application was created in the personal computer and the data was sent to the hardware application board by using serial communication cable. The program that runs over the application board receives the data and operates. So the gripper can be moved to the position we want to go.

ÖZET

Bu projede robot kolunun kinematic analizi üzerinde çalışılmıştır. Kinematic, harekete bağlı olarak robot kolundaki eklem ve hareket parçaları arasındaki ilişkiyi ifade eder. İleri yönlü kinematic ve geri yönlü kinematic olmak üzere iki çeşittir. İleri yönlü kinematic analizde ana bağlantı noktasının konumu, hareket parçalarının uzunlukları ve eklem açıları verilir. Uç elemanın konumu bulunmak istenir. Geri yönlü kinetic analizde ise uç noktanın konumu verilir ve bu noktaya gitmek için gerekli eklem açı değerleri bulunmaya çalışılır.

İleri yönlü kinematic analiz basit dönüşüm matrisleri oluşturarak çözülebilir. Fakat geri yönlü kinematic analizin çözümü oldukça zordur ve serbestlik derecesi arttıkça bu zorluk artmaktadır. Inverse kinematic analizin çözümü için değişik yöntemler kullanılmaktadır. Analitik metod ve Jacobian metod bunların en çok bilinenleridir. Bu projede çözüm yöntemi olarak analitik metod kullanılmıştır.

Projede 3 serbestlik dereceli ve iki hareket elemanına sahip bir yapı tasarlanmıştır. Kullanıcı arayüzü normal kişisel bilgisayarda oluşturulmuş ve gerekli bilgi seri port ile uygulama kartına aktarılmıştır. Uygulama kartı üzerindeki program gelen bilgiyi işleyip eklem noktalarındaki elemanları uygun açılarda döndürmektedir.. Böylece uç elemanın istenilen konuma ulaşması sağlanmaktadır.

1. INTRODUCTION

Robot is a machine that collects the information about the environment using some sensors and makes a decision automatically. People prefer it to use different fields, such as industry, some dangerous jobs including radioactive effects. In this point, robots are regarded as a server. They can be managed easily and provides many advantages.

Robot kinematics is the study of the motion (kinematics) of robots. In a kinematic analysis the position, velocity and acceleration of all the links are calculated without considering the forces that cause this motion. The relationship between motion, and the associated forces and torques is studied in robot dynamics[1].

Robot kinematics deals with aspects of redundancy, collision avoidance and singularity avoidance. While dealing with the kinematics used in the robots we deal each part of the robot by assigning a frame of reference to it and hence a robot with many parts may have many individual frames assigned to each movable part. For simplicity we deal with the single manipulator arm of the robot. Each frame is named systematically with numbers, for example the immovable base part of the manipulator is numbered 0, and the first link joined to the base is numbered 1, and the next link 2 and similarly till n for the last n th link[1].

In the kinematic analysis of manipulator position, there are two separate problems to solve: direct kinematics, and inverse kinematics. Direct kinematics involves solving the forward transformation equation to find the location of the hand in terms of the angles and displacements between the links. Inverse kinematics involves solving the inverse transformation equation to find the relationships between the links of the manipulator from the location of the hand in space. In the next chapters, inverse and forward kinematics will be represented in detail[2].

A robot arm is known as a manipulator. It is composed of a set of joints separated in space by the arm links. The joints are where the motion in the arm occurs. In basic, a robot arm consists of the parts: base, joints, links, and a gripper. The base is the basic part over the arm, it may be fixed or active. The joint is flexible and joins two separated links. The link is fixed and supports the gripper. The last part is a gripper. The gripper is used to hold

and move the objects. Figure-1 shows these parts. In the report, the manipulator types are defined in details.

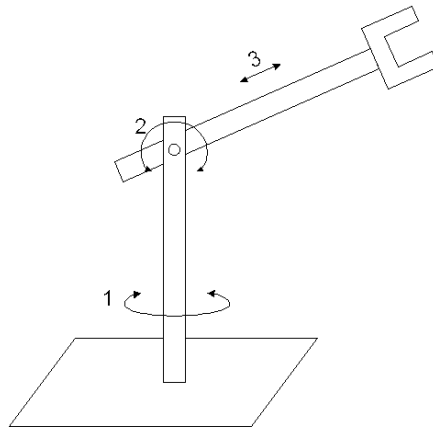


Figure 1.1 Basic robot arm. 1. Base, 2. joint, 3. link and the last part, grapper.

Homogeneous transformation is used to solve kinematic problems. This transformation specifies the location (position and orientation) of the hand in space with respect to the base of the robot, but it does not tell us which configuration of the arm is required to achieve this location. It is often possible to achieve the same hand position with many arm configurations[2]. In the next chapters, this transformation is explained in details with simple examples.

2. FEASIBILITY OF THE PROJECT

During the development of the project, I have been researched the feasibility in the different field, especially software and hardware. The feasibility study is below in details.

2.1. Software Feasibility

In the software feasibility, I tried to choose the best program that solves my needs. I preferred to use JAVA programming language. Because it is known that it run over any operating system with java virtual machine. I created the user interface by using the java-swings. On the other side, I writed the hardware codes by using PIC C program language. MICRO C can be used , too. PICFLASH provides us to load the program onto development kit. You can use the PROTEUS to desing your chip devices.

2.2. Hardware Feasibility

On the hardware side, we should have a development kit with serial commication port to send data and usb port to program the chip.



Figure-2.1 Microcontroller.

In addition to this, we can use servos to rotate robot arm. The servo rotate different angles. Sometimes it is between -90 and 90 degrees. Some of theme can rotate about 90 to 180 degrees.



Figure-2.2 Servo machines.

The last one, we can use a sheet of plastic to cut the links. You can see it following picture.



Figure-2.3 A sheet of plastic.

2.3. Technical Feasibility

Minimum requirements for the project are given in the table 2.1

Table 2.1 Minimum requirements for the project.

| | |
|----------------------------------|--|
| Processor | 600 MHz processor Recommended: 1 gigahertz (GHz) processor |
| RAM | 512 MB Recommended: 1.5 GB |
| Available Hard Disk Space | 1 GB of available space required on system drive |
| Video | 800 X 600, 256 colors. Recommended: 1024 X 768, High Color 16-bit |

2.4. Economical Feasibility

Economic cost of the project can be separated in two groups. First of them is software cost. Another one is hardware cost. Table 2.2 shows software cost and hardware cost.

Table 2.2 Software cost and hardware cost.

| Hardware | Price(\$) | Software | Price(\$) |
|---|-----------|--------------|-----------|
| Computer included recommendation devices. | 1000 | Windows XP | 150 |
| | | Linux | free |
| Development Kit | 2000 | NetBeans IDE | free |
| Servo (for each one) | 20 | PICFLASH | free |
| Plastic sheet | 10 | PIC C | free |
| | | Proteus | 50 |
| TOTAL | 3030 | | 200 |

2.5. Legal Feasibility

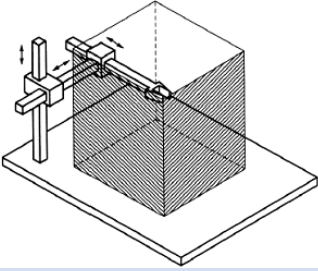
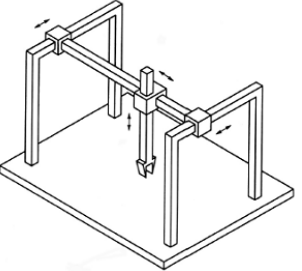
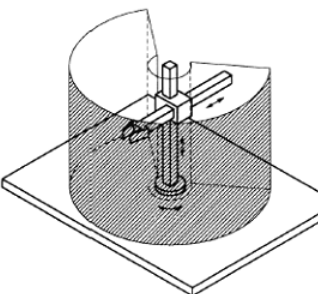
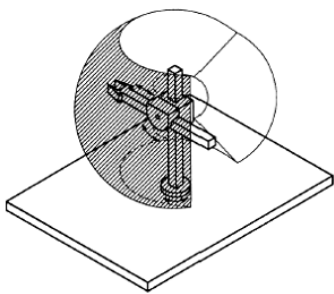
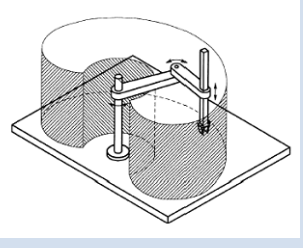
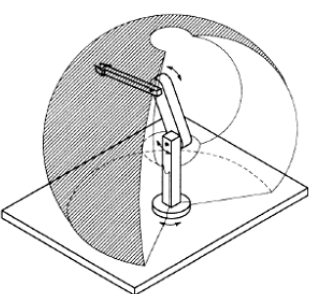
Software needs is usually free. If we have licenses for software, there is no problem. Moreover the part of other articles and researches are referenced at the end of the project.

3. BASIC MANIPULATOR GEOMETRIES

In this section, I look at some basic arm geometries. As I said before, a robot arm or manipulator is composed of a set of joints, links, grippers and base part.

The joints are where the motion in the arms occurs, while the links are of fixed construction. Thus the links maintain a fixed relationship between the joints. The joints may be actuated by motors or hydraulic actuators. There are two sorts of robot joints, involving two sorts of motion. A revolute joint is one that allows rotary motion about an axis of rotation. An example is the human elbow. A prismatic joint is one that allows extensions or telescopic motion. An example is a telescoping automobile antenna. There are some types of manipulator kinematic below.

Table 3.1. Manipulator kinematic

| Name | Figure | Name | Figure |
|--------------------|---|----------------|---|
| Cartesian |  | Gantry |  |
| Cylindrical |  | Sphre |  |
| Scara |  | Anthropomorhic |  |

3.1. Open Chain Manipulator Kinematics

In this types of the arm, mechanics of a manipulator can be represented as a kinematic chain of rigid bodies (links) connected by revolute or prismatic joints. One end of the chain is constrained to a base, while an end effector is mounted to the other end of the chain. Figure-3.1 shows an open chain serial robot arm[4].

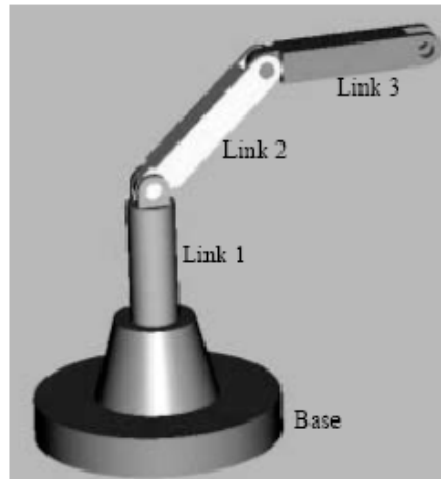


Figure-3.1 Open chain serial robot arm.

In the open chain robot arm, The resulting motion is obtained by composition of the elementary motions of each link with respect to the previous one. The joints must be controlled individually.

3.2. Closed Chain Manipulator Kinematics

Closed Chain Manipulator is much more difficult than open chain manipulator. Even analysis has to take into account statics, constraints from other links, etc. Parallel robot is a closed chain. For this type of robots, the best example is the Stewart platform. Figure-3.2 shows Stewart platform[4].

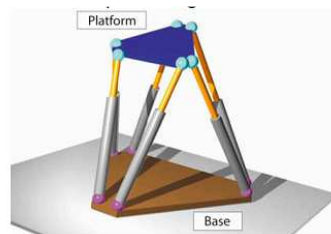


Figure-3.2 Stewart platform.

4. HOMOGENEOUS TRANSFORMATIONS

Homogeneous transformation is used to calculate the new coordinate values for a robot part. Transformation matrix must be in square form. Figure-4.1 shows the transformation matrix.

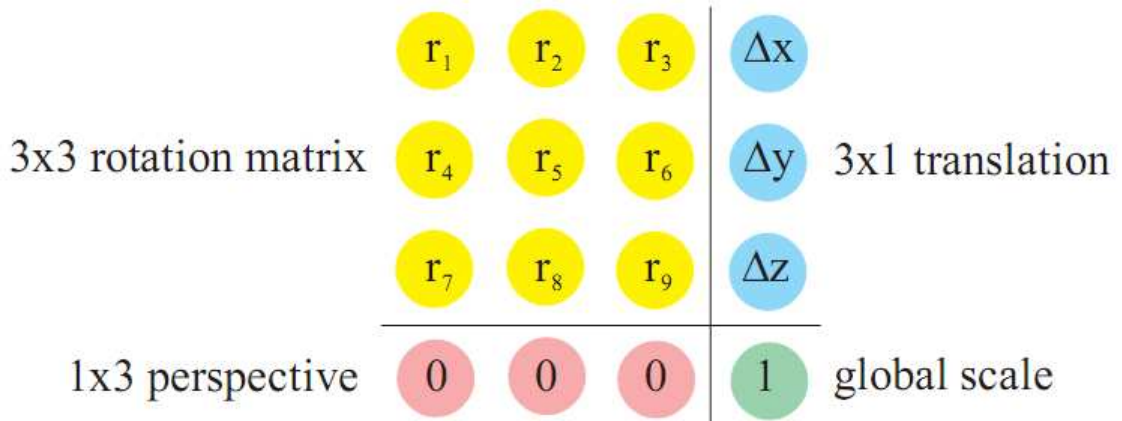


Figure-4.1 Homogeneous Transformation matrix.

3x3 rotation matrix may change with respect to rotation value. 3x1 translation matrix shows the changing value between the coordinate systems. Global scale value is fix and 1. Also 1x3 perspective matrix is fix.

4.1. Right Handed Coordinate Systems

In a right handed coordinate system, if you know the directions of two out of the three axes, you can figure out the direction of the third. Let's suppose that you know the directions of the x and y axes. For example, suppose that x points to the left, and y points out of the paper. We want to determine the direction of the z axis. To do so, take your right hand, and hold it so that your fingers point in the direction of the x axis in such a way that you can curl your fingers towards the y axis. When you do this, your thumb will point in the direction of the z axis. This process is illustrated in Figure-4.2. The chart in figure 4.2 details how to compute the direction of any axis given the directions of the other two[3].

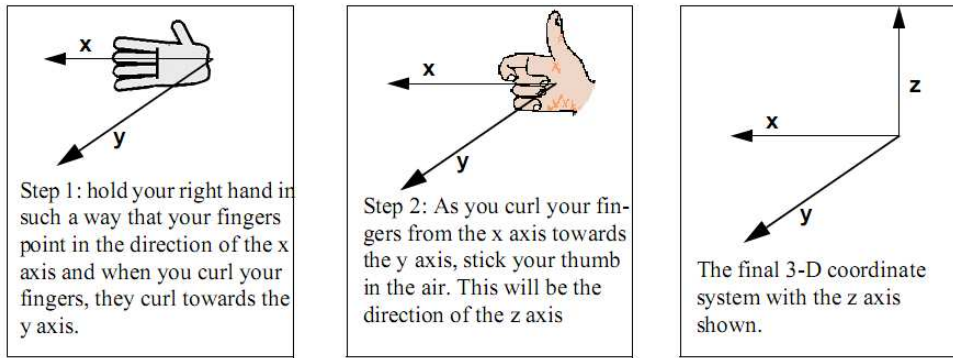


Figure-4.2 Using the right hand rule to compute the direction of the z axis.

Sometimes we want to talk about rotating around one of the axes of a coordinate frame by some angle. Of course, if you are looking down an axis and want to spin it, you need to know whether you should spin it clockwise or counter-clockwise. We are going to use another right hand rule to determine the direction of positive rotation[3].

| If you know the direction of these axes. | Point the fingers of your right hand in the direction of this axis. | Curl you right fingers towards the direction of this axis. | Your thumb will point in the direction of this axis. |
|--|---|--|--|
| x & y | x | y | z |
| y & z | y | z | x |
| x & z | z | x | y |

Figure-4.3 Using the right hand rule to compute the direction of any axis given the directions of the other two.

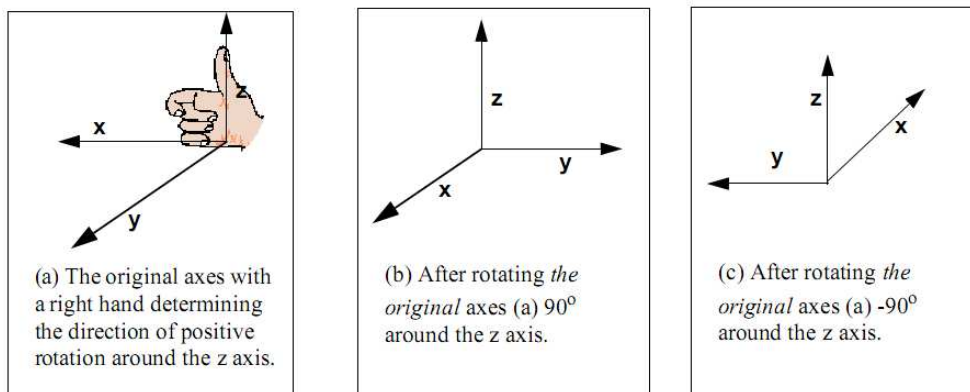


Figure-4.4 The right hand rule to determine the direction of positive angles. Point your right thumb along the positive direction of the axis you wish to rotate around. Curl your fingers. The direction that your fingers curl is the direction of positive rotation.

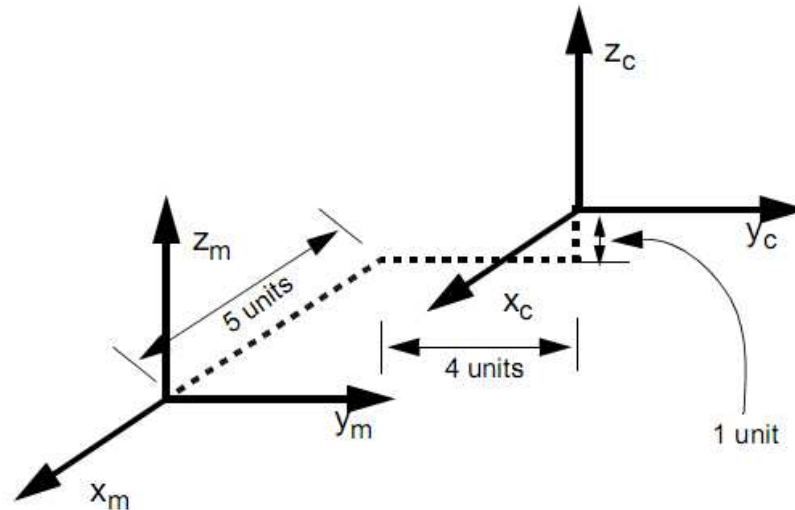


Figure-4.5 Two coordinate frames that differ by only a translation.

For the figure 4.5, the rotation matrix,

$$\text{Rotation matrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ and changing for x, y, z axis,}$$

$$\Delta X = X_m - X_c = 5, \Delta Y = Y_m - Y_c = -4, \Delta Z = Z_m - Z_c = -1$$

Thus, The transformation matrix,

$$T_c^m = \begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 1 & 0 & -4 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T_j^k = The transformation that you use to take a point in j-coordinates and compute its location in k-coordinates.

If there is a rotation around the x, y or z axis, The rotation matrix reforms below,

$$\text{Rot x}(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.1)$$

$$\text{Rot } y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.2) \quad \text{Rot } z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

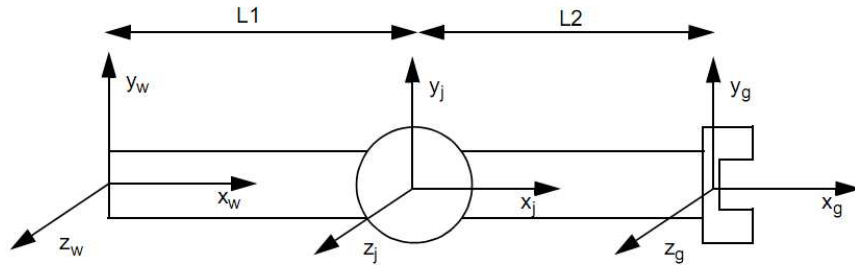


Figure-4.6 A simple arm.

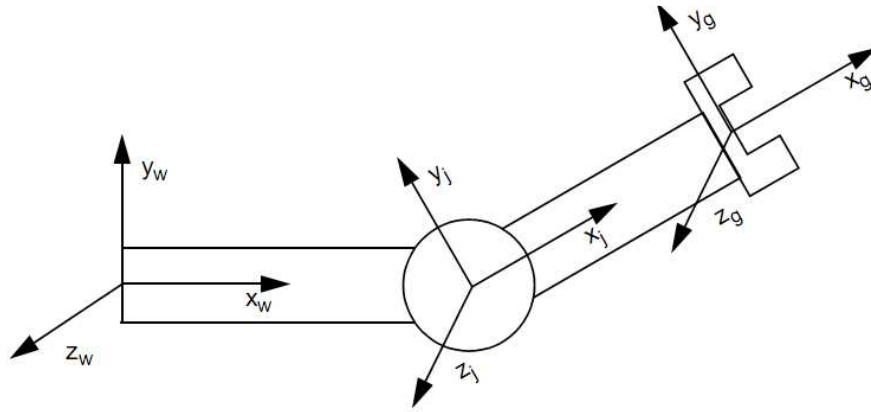


Figure-4.7 The robot arm from figure 4.6 with the joint rotated by ψ degrees.

For the figure 4.7, Transformation matrix,

$$\text{Rot } z(\theta) = \begin{bmatrix} 1 & 0 & 0 & L1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos\psi & -\sin\psi & 0 & 0 \\ \sin\psi & \cos\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & L2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 & L2\cos\psi + L1 \\ \sin\psi & \cos\psi & 0 & L2\sin\psi \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.3)$$

5. FORWARD AND INVERSE KINEMATICS

Robot kinematics are mainly of the following two types: forward kinematics and inverse kinematics. Forward kinematics is also known as direct kinematics. In forward kinematics, the length of each link and the angle of each joint is given and we have to calculate the position of any point in the work volume of the robot. In inverse kinematics, the length of each link and position of the point in work volume is given and we have to calculate the angle of each joint. They are detailed below.

5.1. Forward Kinematics(FK)

Forward kinematics is the method for determining the orientation and position of the end effector, given the joint angles and link lengths of the robot arm[5]. The forward position kinematics (FPK) solves the following problem: "Given the joint positions, what is the corresponding end effector's pose?"[1].

In the serial chains, the solution is always unique: one given joint position vector always corresponds to only one single end effector pose. The FK problem is not difficult to solve, even for a completely arbitrary kinematic structure.

Methods for a forward kinematic analysis:

- using straightforward geometry
- using transformation matrices

In the parallel chains (Stewart Gough Manipulators, it is shown in the figure-3.2), the solution is not unique: one set of joint coordinates has more different endeffector poses. In case of a Stewart Platform there are 40 poses possible which can be real for some design examples. Computation is intensive but solved in closed form with the help of algebraic geometry[1].

The relationships between angles and sides can be found using the right angle triangle in the figure-5.1.

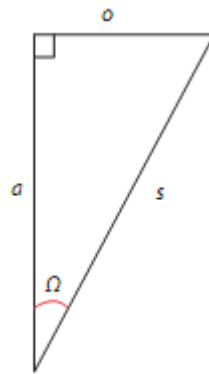


Figure-5.1 Right angle triangle.

$$s \cdot \sin(\Omega) = o, \quad (5.1)$$

$$s \cdot \cos(\Omega) = a \text{ and } = + -2 \cdot a \cdot s \cdot \cos(\Omega) \text{ (Cosine Theory)} \quad (5.2)$$

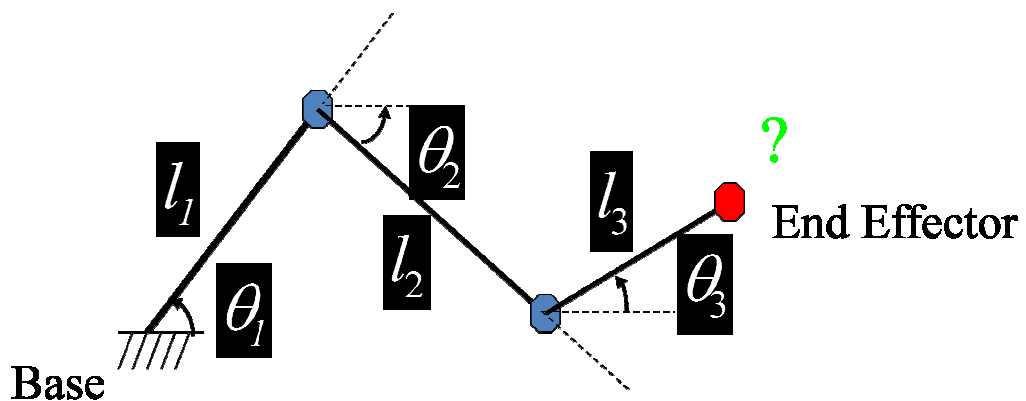


Figure-5.2 A simple forward kinematics.

In the above figure, we want to find out what the coordinates of end effector are. Using a simple geometric method,

$$x = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) + l_3 \cos(\theta_3) \quad (5.3)$$

$$y = l_1 \sin(\theta_1) - l_2 \cos(\theta_2) + l_3 \cos(\theta_3) \quad (5.4)$$

Also, forward kinematics by composing transformations,

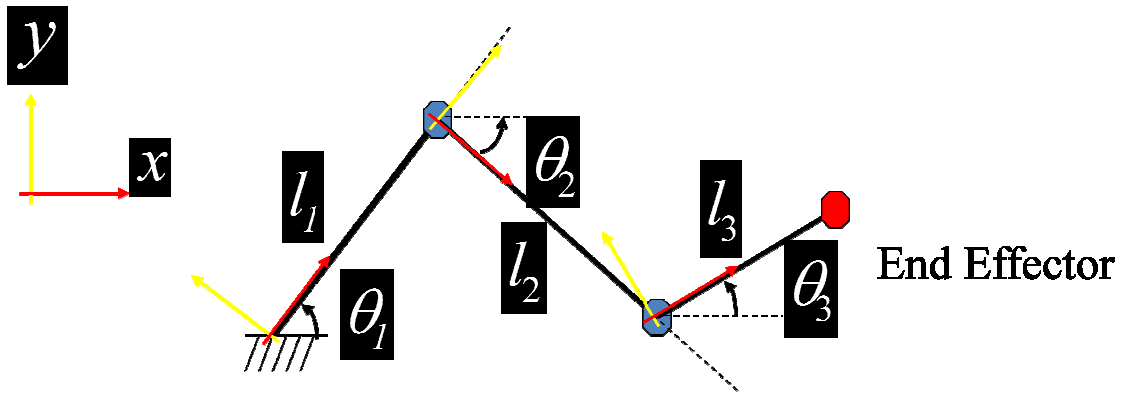


Figure-5.3 Forward kinematics by composing transformations.

$$= \begin{pmatrix} \cos(\theta_1) & -\sin(\theta_1) & l_1 \\ \sin(\theta_1) & \cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\theta_2) & -\sin(\theta_2) & l_2 \\ \sin(\theta_2) & \cos(\theta_2) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\theta_3) & -\sin(\theta_3) & l_3 \\ \sin(\theta_3) & \cos(\theta_3) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (5.5)$$

5.2. Inverse Kinematics(IK)

Inverse kinematics is the opposite of forward kinematics. This is when you have a desired end effector position, but need to know the joint angles required to achieve it[5]. The inverse position kinematics (IPK) solves the following problem: "Given the actual end effector pose, what are the corresponding joint positions?" In contrast to the forward problem, the solution of the inverse problem is not always unique: the same end effector pose can be reached in several configurations, corresponding to distinct joint position vectors[1]. Although way more useful than forward kinematics, this calculation is much more complicated too.

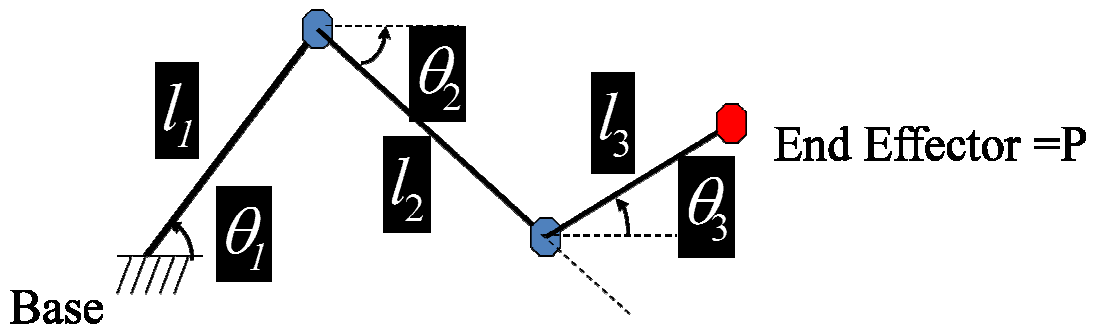


Figure-5.4 Inverse kinematics

$$\theta_1, \theta_2, \theta_3 = f^{-1}(P), \quad (5.6)$$

In the figure-5.4, there are 3 unknown values. But we have 2 equations,

$$x = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) + l_3 \cos(\theta_3) \quad (5.7)$$

$$y = l_1 \sin(\theta_1) - l_2 \cos(\theta_2) + l_3 \cos(\theta_3) \quad (5.8)$$

The problems in IK :

- There may be multiple solutions,
- For some situations, no solutions,
- Redundancy problem.

5.2.1. Solving The Inverse Kinematics

Although way more useful than forward kinematics, this calculation is much more complicated. There are several methods to solve the inverse kinematics.

5.2.1.1. Analytic Method

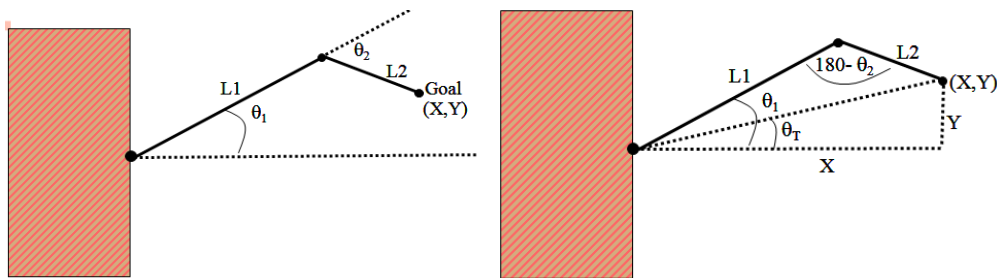


Figure-5.5 Analytic Method to solve inverse kinematics.

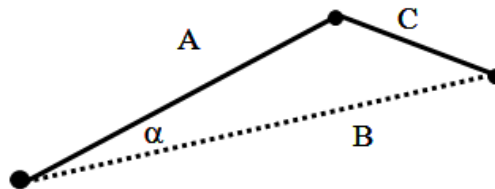


Figure-5.6 Cosine Law.

$$\cos(a) = \frac{A^2 + B^2 - C^2}{2AB} \quad (\text{Cosine Law.}) \quad (5.9)$$

In the figure-5.5, using the cosine law angles are found.

$$\cos(\theta_T) = \frac{X}{\sqrt{X^2 + Y^2}} \quad (5.10) \quad \theta_T = \cos^{-1}\left(\frac{X}{\sqrt{X^2 + Y^2}}\right) \quad (5.11)$$

$$\cos(\theta_1 - \theta_T) = \frac{L_1^2 + X^2 + Y^2 - L_2^2}{2L_1\sqrt{X^2 + Y^2}} \quad (5.12) \quad \theta_1 = \cos^{-1}\left(\frac{L_1^2 + X^2 + Y^2 - L_2^2}{2L_1\sqrt{X^2 + Y^2}}\right) + \theta_T \quad (5.13)$$

$$\cos(180 - \theta_2) = \frac{L_1^2 + L_2^2 - (X^2 + Y^2)}{2L_1L_2} \quad (5.14)$$

$$\theta_2 = 180 - \cos^{-1}\left(\frac{L_1^2 + L_2^2 - (X^2 + Y^2)}{2L_1L_2}\right) \quad (5.15)$$

5.2.1.1. Inverse Jacobian Method

It is used when linkage is complicated. Iteratively the joint angles change to approach the goal position and orientation.

Jacobian is the n by m matrix relating differential changes of q to differential changes of P (dP).

Jacobian maps velocities in joint space to velocities in cartesian space

$$J(\theta)\dot{\theta} = V \quad (5.16)$$

$$f(\Theta) = P, \quad J(\Theta)d\Theta = dP, \quad J_{ij} = \frac{\partial f_i}{\partial \theta_j} \quad (5.17)$$

An example of Jacobian Matrix,

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f_1(\boldsymbol{\theta}) \\ f_2(\boldsymbol{\theta}) \end{bmatrix} = \begin{bmatrix} l_1 \cos \theta_1 + l_2 \cos \theta_2 + l_3 \cos \theta_3 \\ l_1 \sin \theta_1 - l_2 \sin \theta_2 + l_3 \sin \theta_3 \end{bmatrix} \quad (5.18) \quad \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \mathbf{J} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \quad (5.19)$$

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{\theta})}{\partial \theta_1} & \frac{\partial f_1(\boldsymbol{\theta})}{\partial \theta_2} & \frac{\partial f_1(\boldsymbol{\theta})}{\partial \theta_3} \\ \frac{\partial f_2(\boldsymbol{\theta})}{\partial \theta_1} & \frac{\partial f_2(\boldsymbol{\theta})}{\partial \theta_2} & \frac{\partial f_2(\boldsymbol{\theta})}{\partial \theta_3} \end{bmatrix} = \begin{bmatrix} -l_1 \sin \theta_1 & -l_2 \sin \theta_2 & -l_3 \sin \theta_3 \\ l_1 \cos \theta_1 & -l_2 \cos \theta_2 & l_3 \cos \theta_3 \end{bmatrix} \quad (5.20)$$

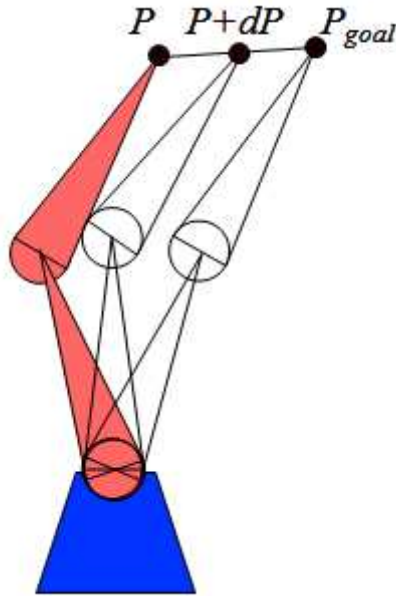


Figure-5.7 Iteratively solution of inverse kinematics.

$$\boldsymbol{\theta} = f^{-1}(P), \quad V = J(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}, \quad \dot{\boldsymbol{\theta}} = J^{-1}(\boldsymbol{\theta})V \quad (5.21)$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \Delta t J^{-1}(\boldsymbol{\theta}_k)V \quad (5.22)$$

In the Jacobian method, the solving can be linearizable about $\boldsymbol{\theta}_k$ locally using small increments.

6. IMPLEMENTATION

The project can be implemented in two parts as software implementation and hardware implementation.

6.1. Software Implementation

Software implementation includes user interface. User controls the robot arm and simulates in space with 3 dimensions. Figure-6.1 shows user interface.

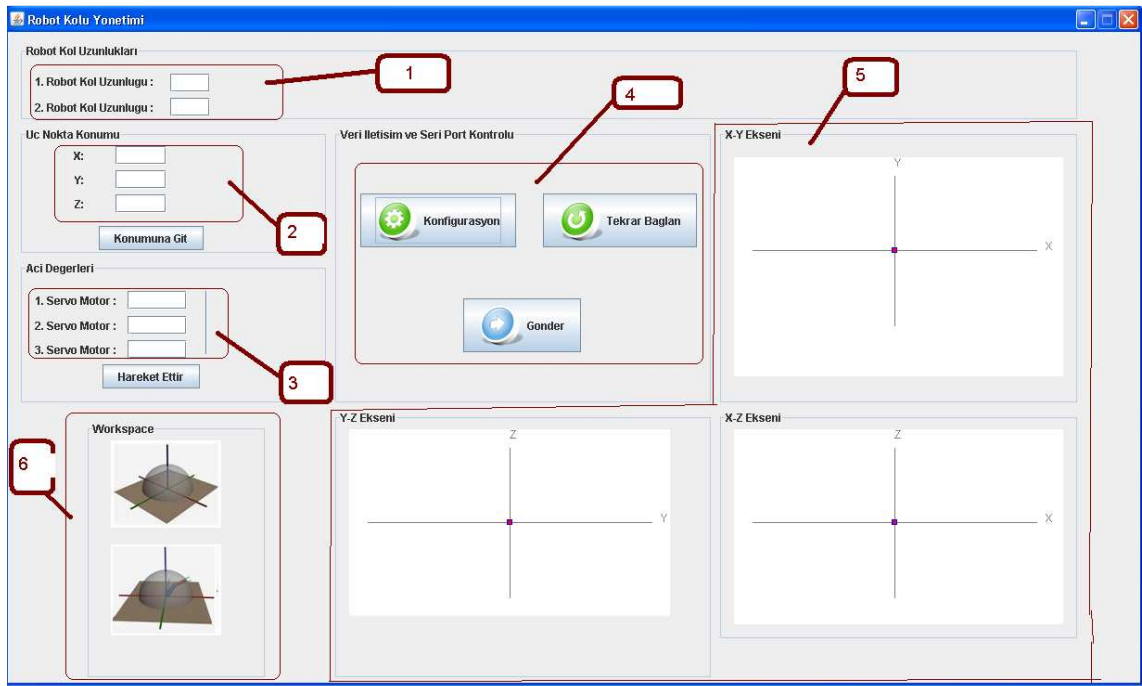


Figure-6.1 User interface.

on the figure-6.1,

- Part 1 shows input area for robot arm length values. User must be enter numeric value.
- Part 2 shows x, y, z position values in the space with 3 dimension. It is grapper position where we want to go. These values must be numeric values too. If user do not enter robot arm lengths, the program will specify.
- Part 3 shows the angles of servos. Servo rotates through the angles. It is accepted that servos rotates about -90 to 90 degrees. If the value is bigger or

smaller , the program will notify the user. Moreover the program notifies the user if the grapper touch the ground, too.

- Part 4 shows serial port configuration and communication. User can determine the communication parameters. Figure-6.2 shows the configuration panel.

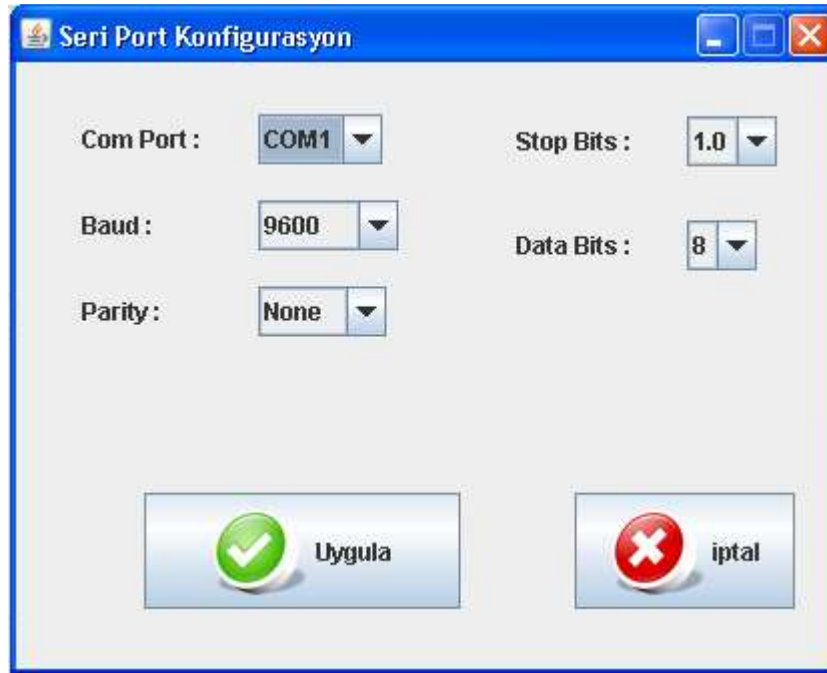


Figure-6.2 Serial port configuration.

In this part, also user can recommunication if the communication is over. When user click send button, profram makes a packet background. The paket is here.

Table 6.1 Packet contents.

| Fist servo angle | Second servo angle | Third servo angle | Error control value |
|------------------|--------------------|-------------------|---------------------|
|------------------|--------------------|-------------------|---------------------|

The error control value calculate by summing the bit value of each angle. It is controled in the hardware side. If there is a problem, program will notify user. In this situaiton, user can resend the data.

- Part 5 shows sights of the robot arms in the space with 3 dimension. It is a simple simulation.

- Part 6 shows workspace. Robot arms can work in the workspace and reach every position in this area.

6.2. Hardware Implementation

In the hardware implementation, I used a development kit and simulation program. The development kit includes a serial port. So we can send data by using serial communication. I have been wrote code for the chip on the board by using PIC C program. Before the code test over the board, I simulated on Proteus. Figure-6.3 shows a design on proteus.

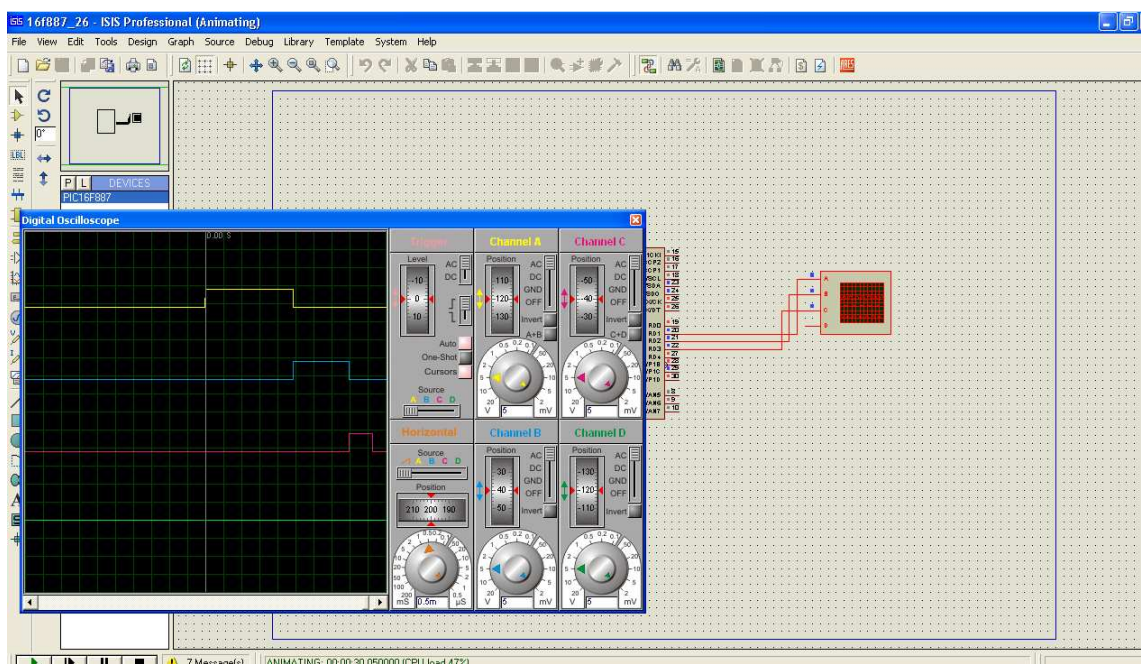


Figure-6.3 Proteus view.

I used the servos to moved the robot arm. Servo rotate angles are sended on the serial port. The servos that I used can be rotate about -90 to 90 degrees. Servos work in 20ms period. It can rotate between about 1ms and 2ms. But sometimes it may be changeable. In the project, I found out that the work frequency is between 0.6ms and 2.4ms. For the frequency with 0.6ms, servo rotates -90 degrees. For 2.4ms, servo roates 90 degrees and lastly for 1.5ms servo is in center.

7. CONCLUSION

In the robot kinematics, the gripper can be moved where is wanted using rotation of links and joints. For this purpose, links and joints are accepted as a coordinate system individually, as using homogeneous transformations.

Robot kinematic is divided in two types: forward kinematic and inverse kinematic. Direct(forward) kinematics involves solving the forward transformation equation to find the location of the hand in terms of the angles and displacements between the links. Inverse kinematics involves solving the inverse transformation equation to find the relationships between the links of the manipulator from the location of the hand in space.

By using user interface program, data is sended as a packet. This packet include servo angles and error check value. This value is controled both by user side program and harfware side program. If there is a problem, the program will notify the user.

Servos are used to move the robot arms. Usually servo works between 1ms and 2ms. But sometimes it may be changeable. In this project I used the frequency with 0.6ms and 2.4ms.

REFERENCES

- [1] Robot Kinematics, Wikipedia Web Site. <http://www.wikipedia.com>
- [2] Crowder, R.M, Automation And Robotics
- [3] Kay, J. , Introduction to Homogeneous Transformations & Robot Kinematics, Rowan University Computer Science Department
- [4] V'aclav Hlav'ač, ROBOT KINEMATICS, Faculty of Electrical Engineering Department of Cybernetics, Czech Technical University.
- [5] Society of Robot Website, <http://www.societyofrobots.com>

CV

Name Surname : Coşkun YETİM
Birth Date : 13th October, 1985
Birth City : Cemişgezek / Tunceli
High School : Elazığ Atatürk Lisesi
Internship : ETCBASE Software Company