

Design and control of a 3 axis pick and place scara robots

by

Hasitha Darshana Chandrasekara

A capstone project submitted in partial fulfillment of the requirements for the
degree of Bachelor of Science in Engineering
Mechatronics

Examination Committee: Dr. Manukid Parnichkun (Chairperson)
Dr. Than Lin
Dr. Erik L.J. Bohez

Nationality: Sri Lankan

Asian Institute of Technology
School of Engineering and Technology
Thailand
May 2014

Acknowledgements

First of all I would like to acknowledge my capstone project advisor Dr. Manukid Parnichkun for the immeasurable support and guidance provided throughout the project time-line while also appreciating the constant motivation provided to finish the project on time.

My appreciation also goes to the two committee members Associate Professor Erik L. J. Bohez and Dr. Than Lin for their valuable input that helped me in times of difficulties throughout the project.

I would also like to extend my sincere gratitude to my fellow and senior colleagues who helped me by sharing their knowledge and experience related to both the mechanical and electronics phases of the project.

Last but not least I would like to show my appreciation to my parents for the encouragement and unbounded support provided to me throughout my life.

Abstract

This paper will include the thesis for the development of a 3 degree of freedom Scara robot. The robot is to be developed for a pick and place application which would function in manual mode by taking inputs from a user. If time permits it would be further developed to operate in automatic mode that would allow it to undertake repetitive tasks.

Table of Contents

Chapter	Title	Page
	Title Page	
	Acknowledgements	i
	Abstract	ii
	Table of Contents	iii
	List of Figures	iv
	List of Tables	v
1	Introduction	1
	1.1 Background	1
	1.2 Statement of the problems	2
	1.3 Objective	3
2	Literature Review	4
	2.1 Information	4
	Methodology	5
	3.1 Mechanical Design	5
	3.2 Electronic Design	9
	3.3 Programming	11
4	Results and Discussion	23
	4.1 Testing of Scara-manipulator for position	23
	4.2 Dimensions and work-space of Scara-manipulator	24
5	Conclusion and Recommendations	25
	5.1 Conclusion	25
	References	26
	Appendix	27

List of Figures

Figure	Title	Page
1.1.1	Scara-Robot	1
3.1.1.1	Rough Sketch of Scara Robot	5
3.1.1.2	Detailed design of the Scara Robot on Solid-Works 2010	6
3.1.1.3	SolidWorks design of Link Mount	7
3.1.1.4	SolidWorks design of Base	7
3.1.1.5	Aluminum holder	8
3.1.2.1	DC Motor	9
3.1.2.2	Rotary Potentiometers	9
3.2.1.1	Circuit diagram	10
3.2.2.1	Arduino Mega 256	11
3.2.3.1	Opto-Isolated DC Motor Board	12
3.2.4.1	Pictorial illustration of how PWM works	12
3.3.1.1	PID Code Part1	13
3.3.1.2	PID Code Part2	14
3.3.1.3	PID Code Part3	15
3.3.1.4	PID Code Part4	16
3.3.2.1	Graphical User Interface	17
3.3.4.1	Flow chart for GUI and Arduino code	22
4.1.2	Robot Work space	24

List of Tables

Table	Title	Page
3.1.2.1	Actuator and Sensor specifications	8
3.3.3.1	DH Parameters of the robot	17
4.1.1	Test Results	22
4.2.1	Dimensions and work-space of Scara-manipulator	23

Chapter 1

Introduction

1.1 Background Information

The Scara acronym stands for Selective Compliant Assembly Robot or Selective Compliant Articulate Robot. The Scara robot is a variation of articulated robot-arms that are widely used in industries for applications ranging from welding, painting, assembly and pick and place. The main reason for the popularity of industrial robots is the efficiency in doing repetitive tasks that would be otherwise monotonous and tedious work for a human being to carry out. Robots of this nature can work continuously without any breaks hence allowing industries to achieve a competitive advantage over an alternate human labor force. However a key disadvantage of an industrial robot over a human laborer is its inability to make timely decisions given a particular scenario where quick decisions need to be made.

One of the key features of an Industrial robot is that they consist of a series of links that are supported by motor actuated joints which may extend from a base to an end-effector. Each of the links constitute to a single degree of freedom while an articulate robot should consist a minimum of three degrees of freedom which equates to three links from base to end-effector. These three links are usually termed as “shoulder”, “elbow” and “wrist” so that its functionality mimics that of a human hand. To access all areas of a defined work envelope an articulate robot should typically consist of six degrees of freedom. Industrial robots also have the ability to be reprogrammed hence allowing industries to utilize them for different tasks depending on the requirement.



Figure1.1: Scara Robot

This project intends to implement a three degree of freedom SCARA robot which consists of two rotary joints and a prismatic joint that could be programmed to reach a desired point in an allocated work-space. The primary purpose of the robot would be to pick and place a predefined object from an initial location to a destination point. All areas of work including mechanics, electronics and programming in the field of robotics will be covered with the implementation of this project

1.2 Statement of Problem

The basic problems/obstacles that will have to be overcome throughout the implementation of this project have been summarized below:

Conceptualization- At this stage the basic requirements of the robot are to be analyzed and an initial design is to be finalized. Types of Motors required (power rating), transmission systems, bearings are to be decided upon at this level of the project.

Detailed Design- Here the robot is to be modelled using a CAD package. The dimensions of the robot which constitute mainly of the link lengths have to be finalized for simulation in software such as SolidWorks or Matlab.

Construction/Physical Implementation- The construction material for the robot-arm links have to be decided while this decision will be influenced by factors like strength, stiffness and cost of material. The links, transmission system, bearings and belts have to be put in place.

Electronics and Programming- The electronics section of this project would involve developing a number of motor-driver circuits which would be used to run the three DC motors in each of the actuator joints. A type of micro-controller to be used for motor control is to be finalized. The programming part involves developing a GUI that would allow the user to control the theta values of each joint to move the robot in manual mode. Algorithms that allow the robot to function in automatic mode to undertake repetitive tasks will also need to be studied for implementation.

Testing- At this point the Scara robot is to be tested for its functionality by allowing it to reach a desired point in a predefined work-space by inputting theta values through a GUI. Tests will be carried for various positions while the Scara-robot should be able to function in this manner automatically for predefined parameters which may be useful as an industrial application.

1.3 Objectives

This section consists of the solutions that could be considered most suitable for the above stated problems in the statement of problem section.

Design and construct the 3 DOF Scara robot. The robot is to be built using Aluminum box bars mainly due to the fact that this material is strong, light, corrosion resistant and easy to cut, drill and shape

Develop a controller for the Scara robot to follow a predefined path. The basic idea is to control each of the three DC gear motors in the actuator joints through a micro-controller that would receive commands through a PC.

Develop a PID algorithm and computer GUI that allows a user to manually operate the robot by input of theta values to each of the actuator joints. Algorithms for the robot to operate in automatic mode will also be implemented later on if time permits.

Operate the Scara robot in such a way that it could be easily implemented to operate on an assembly line in an industry.

Chapter2

Literature Review

Based on (Mahdi, Florin, Riad, 2009) simulation and kinematic modeling of robots is becoming increasingly popular due to the fact that this can be used for layout evaluation and feasibility studies. Before implementing the robot it is important to come up with a complete mathematical model with servo motor dynamics and dynamic simulation. Similarly for the 3 DOF Scara robot the equations of kinematics will be derived using Denavit-Hartenberg notation. Each of the DC motors used in the actuator joints will be studied and modeled. It is also expected that the performance of the robot-actuator system is to be analyzed and verified using Matlab/Simulink as detailed in this research paper.

According to (Taylan, Canan, 2004) the DC motors of a Scara robot should be studied with PD controller action. The equations of motion should be derived using Lagrangian mechanics. This research paper further emphasizes on how the robot-actuator control system should be examined with numerical simulation and experimentally verified.

Based on (Manjunath, 2007) direct kinematics will give the position and orientation of the robot in 3D space while inverse kinematics will give the joint variables that define the same manipulator orientation and position. The paper further identifies that direct and inverse kinematics can be utilized for planning trajectories in the tool-configuration space. Trajectory planning usually interpolates/approximates a desired path by a class of polynomial functions and generates a sequence of time based control set points for the control of the manipulator from the starting point to the destination.

Chapter3

Methodology

This project involved the design and construction of a 3 DOF Scara robot that included the mechanical, electronics and programming aspects of the robotics-engineering field. Below is a detailed description of how the robot was built in each phase thus achieving the targets mentioned in the objectives section.

3.1 Mechanical Design

3.1.1 Mechanical Construction

This Scara robot consists of 3 degrees of freedom that include two rotary joints with a prismatic joint. It has a RRP configuration where the first two joints are rotary followed by a third prismatic joint. Three DC gear motors were utilized to control the actuator joints of the robot. The first motor controls the joint between the base and link 1 of the robot along the Z axis while the second motor will carry out the same task for the joint between link 1 and link 2. The third motor controls the joint between link 2 and 3 through a ‘rack and pinion’ mechanism where the rotary motion of the motor will be transformed to linear motion with the help of the rack.

After initially coming up with a rough design with pen and paper I went on to design the Scara-Robot in SolidWorks 2010 where the dimensions of the links and base of the robot were finalized.

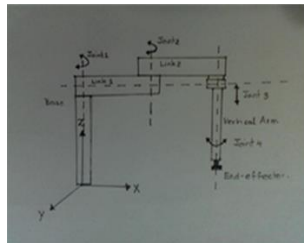


Figure 3.1.1.1: Rough Sketch of Scara Robot

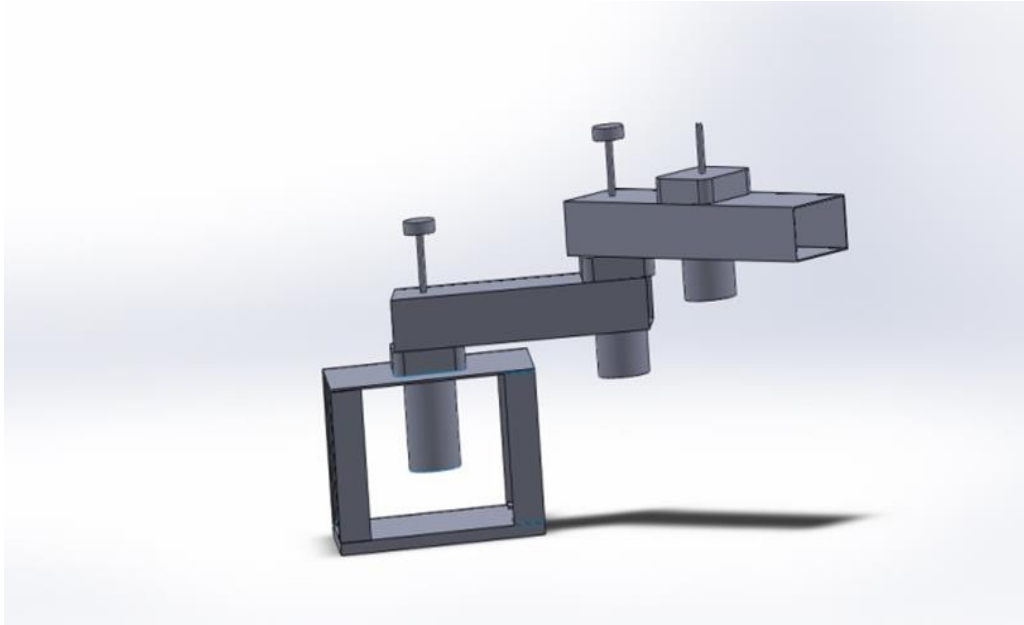


Figure 3.1.1.2: Detailed design of the Scara Robot on Solid-Works 2010

The links were manufactured using Aluminum box bars due to their favorable qualities. The drilling and cutting of the box bars were done by me but however I consulted a metal-worker when complex cuts were required. The mounts to connect the links to the motor shafts were produced by using 5mm thick aluminum sheets and motor couplings. For each link two square aluminum sheets were drilled to which motor couplings were inserted for interference fitting. This unit was then attached to the link using screws. The extended motor shaft from the base motor was passed through each of the 2 units fitted on to the link and using screws the motor-couplings were tightened to the shaft to prevent slip. The second link was similarly mounted on to the shaft of the second motor using these units.

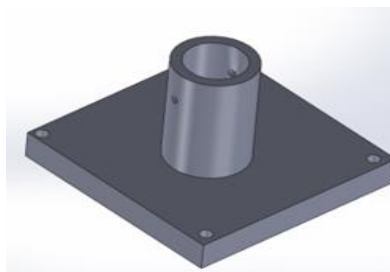


Figure 3.1.1.3: SolidWorks design of Link Mount

When designing the base of the Scara-Robot special emphasis had to be directed towards making the structure heavy hence resulting in improved stability when the robot-arm is in operation. Hence the base was produced using steel as its properties matched the requirements mentioned above.

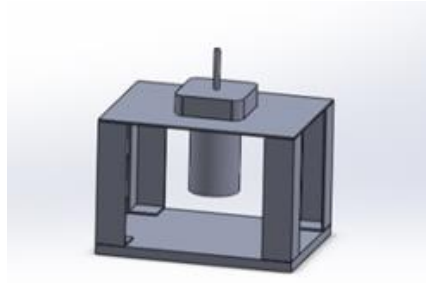


Figure 3.1.1.4: SolidWorks design of base

Once the base was constructed and the motors mounted on each of the links using the newly produced mounts the rack and pinion mechanism had to be installed on the Scara robot which allows for the vertical motion along the Z axis. This unit was purchased and then installed on to the second link successfully using screws which allowed for easy dismantling when required.

After completion of the main structure I had to design two holders using aluminum sheets in order to hold the potentiometers in a fixed position when the motor shafts were in motion. In order to do this drilling, cutting and bending of the aluminum sheets were required.

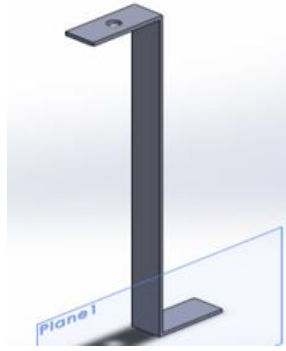


Figure 3.1.1.5: Aluminum holder

3.1.2 Actuator and Sensor specifications

Once all the construction related work was done I had to decide upon the specifications of the actuators and sensors to be used for the Scara-manipulator. The below table summarizes the types of actuators and sensors that were used with their specifications.

Table 3.1.2.1: Actuator and Sensor specifications

	Actuators(DC Gear Motors)		Sensors(Rotary-Potentiometer)	
	Power	RPM	Resistance	Range
Link1	15 W	40	5K Ω	270 ⁰
Link2	15W	60	5K Ω	270 ⁰
Rack &Pinion	15W	120	5K Ω	270 ⁰



Figure 3.1.2.1: DC Motor



Figure 3.1.2.2: Rotary Potentiometers

3.2 Electronic design

3.2.1 Electronic circuit

Once the physical implementation of the robot was completed, the electronics were put in place. The basic idea was to utilize three motor-driver circuits which would be used to run the three DC gear motors in each of the actuator joints. Opto-Isolated DC Motor-driver boards were connected to a micro-controller which was used to control the DC motors. The micro-controller chosen for this project was the Arduino Mega 256 board that could receive commands through a PC which is connected by means of a USB port. This board was particularly chosen as it allowed the required number of inputs and outputs by the Scara-Robot. In order to achieve position control, PID was then implemented where the position of the motors is to be obtained through rotary potentiometers. Below is a complete circuit diagram depicting on how the inputs and out puts were connected between the micro-controller, potentiometers, motor-driver boards.

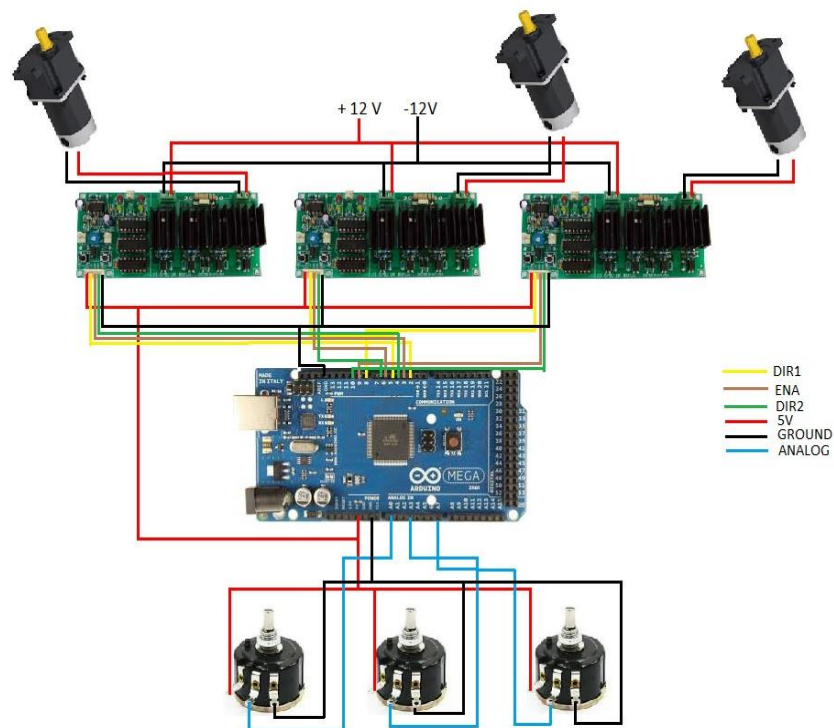


Figure 3.2.1.1: Circuit diagram of the connections between the Arduino mega 256, Opto-Isolated Motor-board and Potentiometer

3.2.2 Arduino micro-controller

As mentioned above the Arduino Mega 256 was specifically chosen as the micro-controller for the Scara-manipulator due to the large number of input and output pins it provides but the reason for choosing Arduino over other types of micro-controllers available in the market have been listed below:

1. Cost-effective
2. Extensible and open source software
3. Clear Programming environment similar to C++

The arduino board consists of a simple open hardware design with an ATMel AVR processor and on-board I/O support.

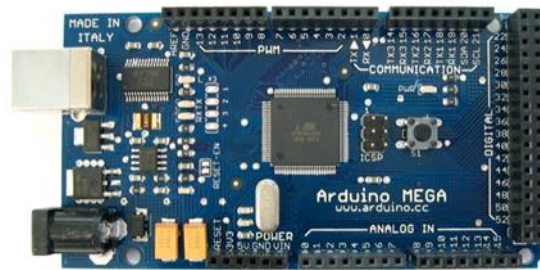


Figure 3.2.2.1: Arduino Mega 256

3.2.3 Opto-Isolated DC Motor Board

This board features direction and speed control on-board or via a standard TTL input to a microcontroller or control board. The board features high powered MOSFET output drivers for DC Motors from 6-24Vdc and with a current handling capacity of up to 5A. This board is ideal for use as a motor speed controller in DC motor systems.

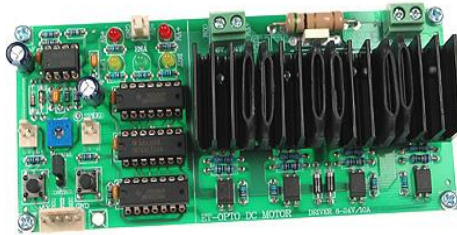


Figure 3.2.3.1: Opto-Isolated DC Motor Board

3.2.4 Pulse Width Modulation

The Arduino Mega can control the output voltage to the three DC motors by means of Pulse-Width-Modulation which is a technique used to get analog results by a digital means. What actually happens here is that the micro-controller outputs a digital signal which inherently has two states namely “full-on” which is equivalent to 5V and “off” which is equivalent to 0V. By varying the time-portion the signal is on with respect to the time the signal is off we can simulate voltages in between the full 5V and 0V. The “on-time” of the signal is usually referred to as the pulse-width and by modulating or changing the “width” or “time-period” of this signal we can output any voltage within the full-on and off state.

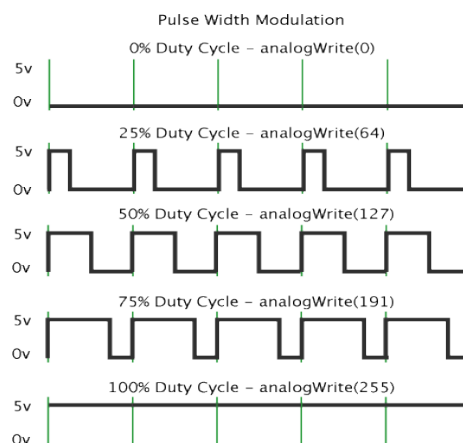


Figure 3.2.2.1: Pictorial illustration of how PWM works

3.3 Programming

Programming the robot was the final phase of the project before the testing could be comprehensively done. Firstly the PID algorithm for the Scara-manipulator was implemented through the Arduino IDE and once the robot was successfully tuned with appropriate PID constants I went on to design the GUI in Visual Basic 2010 so that the robot-arm would go to a position when a user manually enters a theta value or a specified point within the work-space.

3.3.1 PID code implementation

When implementing the PID code I used the arduino PID library and edited the code accordingly to meet the requirements of my system. Once the code was successfully edited it was uploaded to the microcontroller and the tuning parameters were manually configured by trial and error. Below I have explained in detail the most important parts of the code while the full code is made available in the appendix.

```
double Setpoint1, Input1, Output1;
double Output2;
double Setpoint2, Input2, Output3;
double Output4;
double Setpoint3, Input3, Output5;
double Output6;

PID myPID1(&Input1, &Output1, &Setpoint1, 4, 0.1 , 0.01, DIRECT);
PID myPID2(&Input1, &Output2, &Setpoint1, 4, 0.1, 0.01, REVERSE);
PID myPID3(&Input2, &Output3, &Setpoint2, 4, 0.1 , 0.01, DIRECT);
PID myPID4(&Input2, &Output4, &Setpoint2, 4, 0.1, 0.01, REVERSE);
PID myPID5(&Input3, &Output5, &Setpoint3, 4, 0.1 , 0.01, DIRECT);
PID myPID6(&Input3, &Output6, &Setpoint3, 4, 0.1, 0.01, REVERSE);
```

Figure 3.3.1.1: PID Code Part1

As illustrated in figure 9 above the first part of the code has defined the input, output and set point parameters in double precision floating- point format. These parameters will be called in the PID algorithm based on the sample time that has been pre-set. In the second part of the code

six separate instances of the PID library have been defined where two PID's which can be identified as 'DIRECT' and 'REVERSE' are utilized for the two directions of the motor. Therefore 'myPID1' and 'myPID2' has been defined for the first base-motor of the Scara-manipulator, 'myPID2' and 'myPID3' for the second rotary motor and finally 'myPID5' and 'myPID6' for the third motor in the prismatic joint. Also the Kp, Ki and Kd values have been defined for each of the PID's.

```
void setup()
{
    pinMode(A0, INPUT);
    pinMode(A3, INPUT);
    pinMode(A7, INPUT);
    pinMode(2, OUTPUT);
    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(5, OUTPUT);
    pinMode(6, OUTPUT);
    pinMode(7, OUTPUT);
    pinMode(8, OUTPUT);
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    Serial.begin(9600);

    myPID1.SetMode(AUTOMATIC);
    myPID2.SetMode(AUTOMATIC);
    myPID3.SetMode(AUTOMATIC);
    myPID4.SetMode(AUTOMATIC);
    myPID5.SetMode(AUTOMATIC);
    myPID6.SetMode(AUTOMATIC);
}
```

Figure 3.3.1.2: PID Code Part2

In the above figure 10 the 'void setup()' section of the code has been illustrated and in this section the pins of the micro-controller have been configured as inputs or outputs depending on the particular task they have been assigned to carry out. Pins A0, A3 and A7 have been configured as inputs to receive the analog signals from the potentiometers while pins 2-10 have been configured as outputs where the PWM signal as well as the Direction signals have to be

transmitted to each of the three motors. In the second part of the code the mode of the PID has been set to automatic.

```
void loop()
{
  if (Serial.available())
  {
    byte ch = Serial.read();
    switch(ch)
    {
      case '0'...'9':
        value = value * 10 + ch - '0'; break;

      case 'C' :
        if ((value >= 0) && (value <= 180))
        {
          s1 = value;
        }
        value = 0;
        break;

      case 'R' :
        if ((value >= 0) && (value <= 180))
        {
          s2 = value;
        }
        value = 0;
        break;

      case 'L' :
        if ((value >= 0) && (value <= 50))
        {
          s3 = value;
        }
        value = 0;
        break;
    }
  }
}
```

Figure 3.3.1.3: PID Code Part 3

Figure 11 displays the part of the program that receives the serial input from the computer to the micro-controller. According to the code any integer value between 0 to 180 could be received by

the program to be defined as the set point values for the first two motors while any integer between 0 to 50 could be received for the third motor in the prismatic joint. Since characters can only be received by the Arduino micro-controller one byte at a time I have used the code: (value = value * 10 + ch - '0'; break ;) in order to convert the ASCII values and add up the values allowing us to form three digit numbers. Also a switch case statement has been used to uniquely identify which value should be sent to one set-point out of the three available ones. This is done by sending a unique letter following each three digit sequence so the program knows which set-point the value received should be sent to.

```
Setpoint1=s1;
sensorValue1=map(analogRead(A0),171,853,0,180);
Input1 = sensorValue1;
if ((Input1-Setpoint1)>0)
  { digitalWrite(2, LOW);
    digitalWrite(4, HIGH);
    myPID2.Compute();
    analogWrite(3,Output2);
  }

if ((Input1-Setpoint1)<0)
  { digitalWrite(4, LOW);
    digitalWrite(2, HIGH);
    myPID1.Compute();
    analogWrite(3,Output1);
  }
```

Figure 3.3.1.4: PID code Part 4

Figure 12 displays the last part of the PID code where analog readings ranging from 0 to 1024 from the potentiometer are mapped onto appropriate values ranging from 0-180. This input value is then compared to the set-point value input by the user within the if condition statement. If the error between the two values is a positive value instructions will be sent to change the direction of the motor through the digitalWrite function by setting one direction pin high and the other low and at the same time the computed PWM value will be sent through another pin through the analogWrite function. If the error yields a negative value the same procedure will take place except that the pins to be set high and low for direction change will be opposite to that when the

error is a positive value. However a separate PWM value is also calculated when the error is a negative value with respect to a positive error.

3.3.2 Graphical User Interface

The graphical user interface for the Scara manipulator was designed using Visual Basic 2010 Express. Visual Basic allowed me to easily build up the GUI as it provides a good blend of graphical and programming tools that matched my requirements. Initially the GUI components such as the text boxes and buttons were put in place and then they were programmed one by one depending on each task each component has to carry out. For example the “connect’ button was coded to establish a serial transmission link between the arduino micro-controller and the computer.

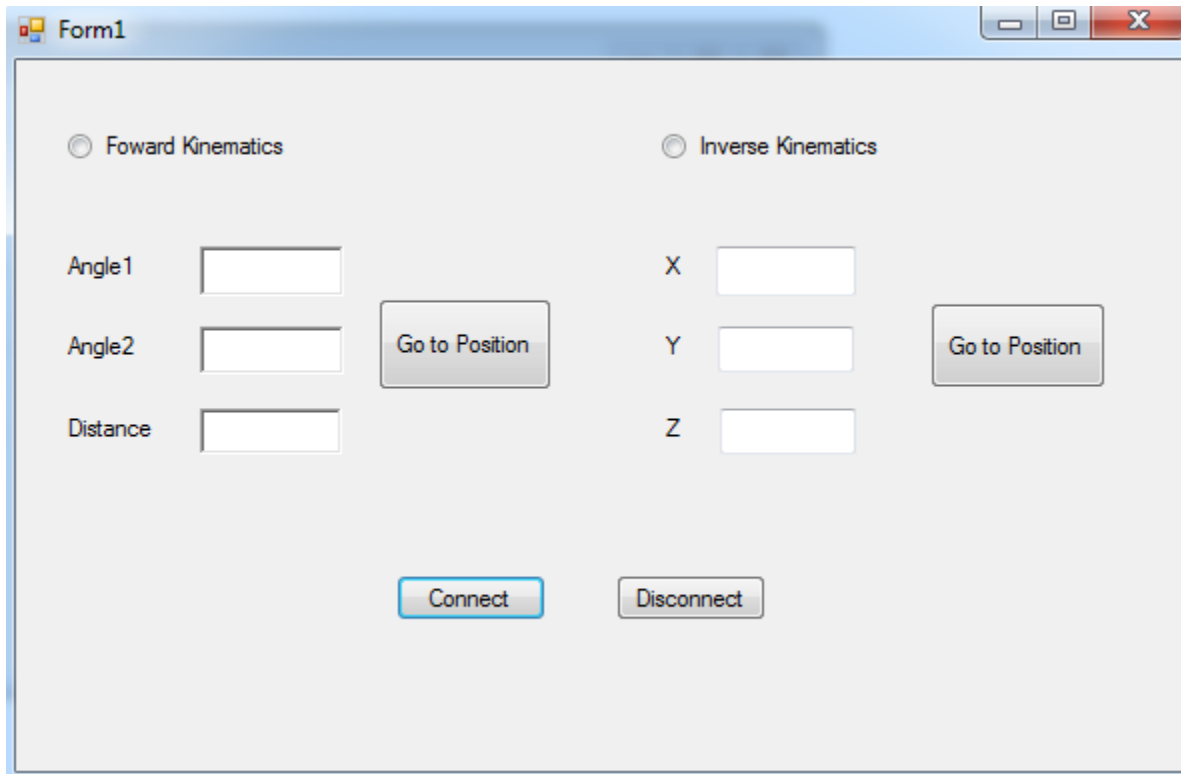


Figure 3.3.3.1: Graphical User Interface

3.3.3 Inverse kinematic analysis

Table3.3.3.1 DH Parameters of the robot

i	θ_i	d_i	a_i	α_i
1	θ_1	0	l_1	0
2	θ_2	0	l_2	0
3	0	$-d_3$	0	0

$$T_{01} = \begin{pmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & l_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & 0 & l_1\sin\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1)$$

$$T_{12} = \begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & l_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & l_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2)$$

$$T_{23} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (3)$$

$$T_{03} = \begin{pmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) & 0 & \cos(\theta_1 + \theta_2)l_2 + \cos(\theta_1)l_1 \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) & 0 & \sin(\theta_1 + \theta_2)l_2 + l_1\sin(\theta_1) \\ 0 & 0 & 1 & -d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (4)$$

End effector position can be given by following coordinates

$$P_x = l_1 \cos \theta_1 + l_2 \cos(\theta_1 + \theta_2) \quad (5)$$

$$P_y = l_1 \sin \theta_1 + l_2 \sin \beta_2 \quad (6)$$

$$P_z = -d_3 \quad (7)$$

Solve for θ_2

$$\begin{aligned} P_x^2 + P_y^2 &= (l_1 \cos \theta_1 + l_2 \cos(\beta_2))^2 + (l_1 \sin \theta_1 + l_2 \sin \beta_2)^2 \\ &= l_1^2 + l_2^2 + 2l_1 l_2 \cos \beta_2 \end{aligned} \quad (8)$$

$$\cos \theta_2 = (P_x^2 + P_y^2 - (l_1^2 + l_2^2)) / 2l_1 l_2 \quad (9)$$

$$\theta_2 = \text{Atan2} (P_x^2 + P_y^2 - (l_1^2 + l_2^2) / 2l_1 l_2) \quad (10)$$

Solve for θ_1

Rearranging above (6) and (7) equations.

$$P_x = (l_1 + l_2 \cos \theta_2) \cos \theta_1 - l_2 \sin \theta_2 \sin \theta_1 \quad (11)$$

$$P_y = (l_1 + l_2 \cos \theta_2) \sin \theta_1 + l_2 \sin \theta_2 \cos \theta_1 \quad (12)$$

Let's take as

$$a_1 = (l_1 + l_2 \cos \theta_2) \text{ and } a_2 = l_2 \sin \theta_2 \quad (13)$$

Rearranging above equation

$$P_x = a_1 \cos \theta_1 - a_2 \sin \theta_1 \quad (14)$$

$$P_y = a_1 \sin \theta_1 + a_2 \cos \theta_1 \quad (15)$$

$$\begin{aligned}\frac{Px}{\sqrt{a_1^2+a_2^2}} &= \frac{a_1}{\sqrt{a_1^2+a_2^2}} \text{Cos}\theta_1 - \frac{a_2}{\sqrt{a_1^2+a_2^2}} \text{Sin}\theta_1 \\ &= \text{Cos}(\gamma + \theta_1) \quad ; \text{ where } \gamma = \text{Atan2}\left(\frac{l_2 \text{Sin}\theta_2}{(l_1 + l_2 \text{Cos}\theta_2)}\right)\end{aligned}\quad (16)$$

$$\begin{aligned}\frac{Py}{\sqrt{a_1^2+a_2^2}} &= \frac{a_1}{\sqrt{a_1^2+a_2^2}} \text{Sin}\theta_1 + \frac{a_2}{\sqrt{a_1^2+a_2^2}} \text{Cos}\theta_1 \\ &= \text{S}(\gamma + \theta_1) \quad ; \text{ where } \gamma = \text{Atan2}\left(\frac{l_2 \text{Sin}\theta_2}{(l_1 + l_2 \text{Cos}\theta_2)}\right)\end{aligned}\quad (17)$$

$$\frac{D}{c} = \frac{Py}{Px} = \text{Tan}(\gamma + \theta_1) \quad \text{where } \gamma = \text{Tan}\left(\frac{l_2 \text{Sin}\theta_2}{(l_1 + l_2 \text{Cos}\theta_2)}\right)\quad (18)$$

$$\theta_1 = \text{Atan2}\left(\frac{Py}{Px}\right) - \text{Atan2}\left(\frac{l_2 \text{Sin}\theta_2}{(l_1 + l_2 \text{Cos}\theta_2)}\right)\quad (19)$$

$$\therefore \theta_1 = \text{Atan2}\left(\frac{Py}{Px}\right) - \text{Atan2}\left(\frac{l_2 \text{Sin}\theta_2}{(l_1 + l_2 \text{Cos}\theta_2)}\right)\quad (20)$$

$$\theta_2 = \text{Atan2}\left(\frac{Px^2 + Py^2 - (l_1^2 + l_2^2)}{2l_1 l_2}\right)\quad (21)$$

Link1 total angle $\beta_2 = \theta_1 + \theta_2$

Gripper Horizontal position can be written as (P_x, P_y)

$$P_x = l_1 \text{Cos}\theta_1 + l_2 \text{Cos}\beta_2\quad (22)$$

$$P_y = l_1 \text{Sin}\theta_1 + l_2 \text{Cos}\beta_2\quad (23)$$

$$\begin{aligned}
J_{11} &= \partial P_x / \partial \theta_1 \\
&= - [l_1 \sin \theta_1 + l_2 \sin \beta_2]
\end{aligned} \tag{24}$$

$$\begin{aligned}
J_{12} &= \partial P_x / \partial \theta_2 \\
&= - [l_1 \sin \beta_2]
\end{aligned} \tag{25}$$

$$\begin{aligned}
J_{13} &= \partial P_x / \partial d_4 \\
&= 0
\end{aligned} \tag{26}$$

$$\begin{aligned}
J_{21} &= \partial P_y / \partial \theta_1 \\
&= [l_1 \cos \theta_1 + l_2 \cos \beta_2]
\end{aligned} \tag{27}$$

$$\begin{aligned}
J_{22} &= \partial P_y / \partial \theta_2 \\
&= l_2 \cos \beta_2
\end{aligned} \tag{28}$$

$$\begin{aligned}
J_{23} &= \partial P_y / \partial d_4 \\
&= 0
\end{aligned} \tag{29}$$

$$J_a = \begin{pmatrix} -[l_1 \sin \theta_1 + l_2 \sin \beta_2] & -l_2 \sin \beta_2 & 0 \\ [l_1 \cos \theta_1 + l_2 \cos \beta_2] & l_2 \cos \beta_2 & 0 \end{pmatrix} \tag{30}$$

$$\dot{P}_x = -[l_1 \sin \theta_1 + l_2 \sin \beta_2] \dot{\theta}_1 - [l_2 \sin \beta_2] \dot{\theta}_2 \tag{31}$$

$$\dot{P}_y = [l_1 \cos \theta_1 + l_2 \cos \beta_2] \dot{\theta}_1 + [l_2 \cos \beta_2] \dot{\theta}_2 \tag{32}$$

3.3.4 Flow chart for GUI and arduino code

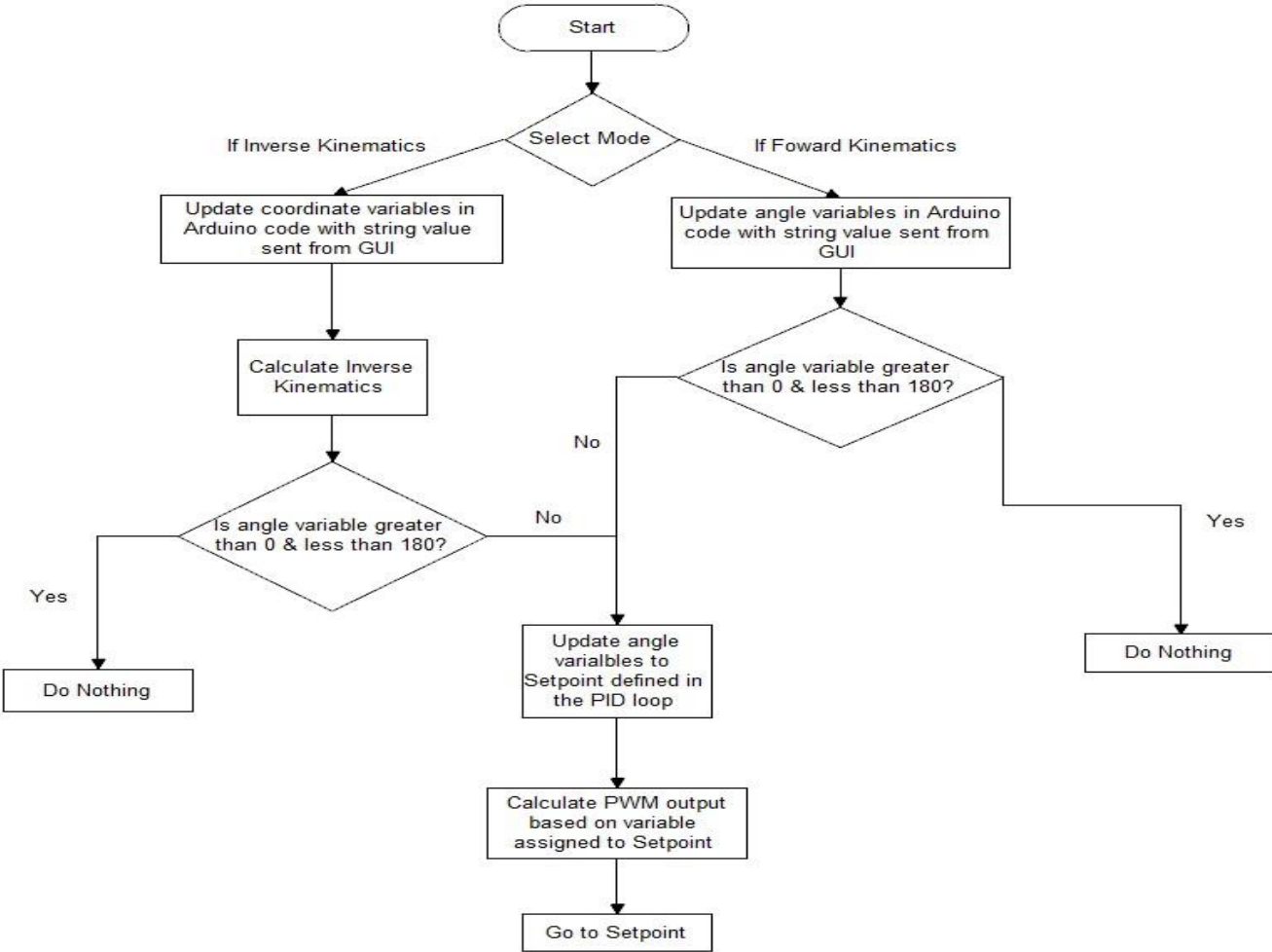


Figure 3.3.4.1 Flow chart for the GUI and arduino program

Chapter4

Results and Discussion

4.1 Testing Scara-manipulator for position

After the Scara-manipulator was successfully tuned with the appropriate PID constants K_p , K_i and K_d it was tested thoroughly for position-control. This was achieved by sending two theta values θ_1 and θ_2 and a distance value d_3 for the three links through the arduino serial console. The arduino code was also modified so that once the links reach their final position the current position of each of the three links was printed on the screen. In the below table the values input by the user have been compared to the current position values printed on the serial console after the links have achieved their position. In the final column of Table3 the maximum tolerance level of the Scara-manipulator once it has reached its position has been given separately for $(\theta_1+\theta_2)$ and d_3 .

Table 4.1.1: Test Results

Trial No.	Equivalent Angle and Distance sent to each Link			Actual Potentiometer Reading after position has been reached by each link			Tolerance level for $(\theta_1+\theta_2)$ and d_3 in degrees and millimeters	
	Link 1 $(\theta_1)_{\text{degrees}}$	Link 2 $(\theta_2)_{\text{degrees}}$	Distance $(d_3)_{\text{mm}}$	Link 1 $(\theta_1)_{\text{degrees}}$	Link 2 $(\theta_2)_{\text{degrees}}$	Distance $(d_3)_{\text{mm}}$	$(\theta_1+\theta_2)_{\text{degrees}}$	$(d_3)_{\text{mm}}$
-	Link 1 $(\theta_1)_{\text{degrees}}$	Link 2 $(\theta_2)_{\text{degrees}}$	Distance $(d_3)_{\text{mm}}$	Link 1 $(\theta_1)_{\text{degrees}}$	Link 2 $(\theta_2)_{\text{degrees}}$	Distance $(d_3)_{\text{mm}}$	$(\theta_1+\theta_2)_{\text{degrees}}$	$(d_3)_{\text{mm}}$
1	10	10	10	12	8	13	4	3
2	25	30	15	27	31	13	3	2
3	45	60	22	47	62	24	4	2
4	60	45	34	60	44	34	1	0
5	82	110	5	80	111	7	3	2
6	110	0	28	110	0	28	0	0
7	132	130	45	132	128	46	2	1
8	150	120	50	149	118	50	3	0
9	170	90	11	168	90	9	2	2
10	180	180	39	180	179	41	1	2

Maximum Tolerance level for $(\theta_1+\theta_2) = \pm 4^\circ$ degrees

Average Tolerance level for $(\theta_1+\theta_2) = \pm 2.3^\circ$ degrees

Maximum Tolerance level for $(d_3) = \pm 3$ millimeters

Average Tolerance level for $(d_3) = \pm 1.4$ millimeters

4.2.1 Dimensions and work-space of Scara-manipulator

	Link1	Link2	Link3
Length (millimeters)	165mm	210mm	50mm

The Scara-manipulator has the following workspace:

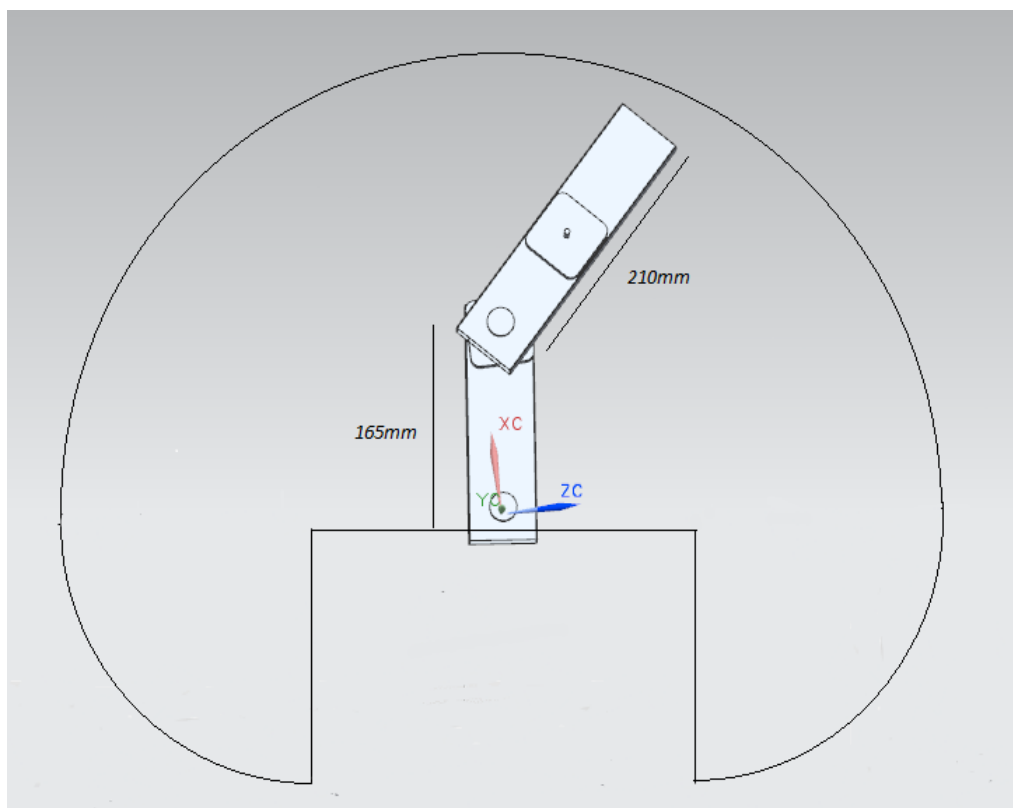


Figure 4.2.1 Robot Work space

Chapter 5

Conclusion and recommendations

5.1 Conclusion

This project started out with the objective of developing a 3 degree of freedom Scara-manipulator which could be used for a pick and place application. Initially the rough design of the robot was sketched and then detailed designs were drawn up with the use of CAD software packages. Once the dimensions and fixtures were finalized mechanical construction commenced. Equipment ranging from lathe machines to drills were utilized for the construction of the robot while the design was successfully completed as per the CAD designs.

The electronics part of the project was carried out after the mechanical construction was complete. This involved connecting the inputs and outputs from the arduino micro-controller to the motors and sensors etc...It was during the Programming phase of the project where I faced a few problems which involved the PWM signals not being transferred to the motor-driver board due to minor errors in the arduino code. These problems were however overcome and PID was successfully implemented on the Scara-manipulator. The final phase involved developing a graphical user interface in Visual Basic which allowed the user to define a point to which the robot-arm could move to.

As future improvements to the current system it could be suggested to use optical encoders as the sensory feedback which could provide more accurate and reliable feedback. Also when designing the mechanical structure CNC machines could be recommended which would result in a more mechanically sound structure.

On a final note it is noteworthy to mention that the objectives mentioned in the proposal for this project were achieved with the successful implementation of the Scara-manipulator.

References

- [1] Kinematic Modeling and Simulation of a SCARA Robot by Using Solid Dynamics and Verification by MATLAB/Simulink, Mahdi Salman Alshamasin, Department of Mechatronics Faculty of Engineering Technology, Albalqa' Applied University, Jordan, Florin Ionescu Mechatronics Institute Hochschule –Konstanz,Htwg, Germany,Riad Taha Al-Kasasbeh Department of Power Engineering Faculty of Engineering Technology, Albalqa' Applied University, Jordan, 2009

- [2] SCARA Robot: Modeled, Simulated, and Virtual-Reality Verified, Yousif I. Al Mashhadany (MIEEE, MIIE), Electrical Engineering Department, College of Engineering, University of Anbar, Baghdad, Iraq,2012

- [3] Trajectory Planning Design Equations and Control of a 4 - axes Stationary Robotic Arm
T.C. Manjunath, Student Member IEEE, SPIE, IOP, Life Member ISSS, ISOI, SSI, ISTE, 2007

- [4] Mathematical modelling, simulation and experimental verification of a Scara robot
M. Taylan Das, L. Canan Du' lger, Department of Mechanical Engineering, University of Gaziantep, 27310 Gaziantep, Turkey,2004

Appendix

GUI Code

```
Imports System.IO
Imports System.IO.Ports
Imports System.Threading

Public Class Form1
    Shared _continue As Boolean
    Shared _serialPort As SerialPort

    Private Sub Form1_Load (ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        SerialPort1.Close ()
        SerialPort1.PortName = "com3"
        SerialPort1.BaudRate = 9600
        SerialPort1.DataBits = 8
        SerialPort1.Parity = Parity.None
        SerialPort1.StopBits = StopBits.One
        SerialPort1.Handshake = Handshake.None
        SerialPort1.Encoding = System.Text.Encoding.Default
    End Sub

    Private Sub btnconnect_Click (ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnconnect.Click
        Try

            SerialPort1.Open()
            If SerialPort1.IsOpen Then
                btnconnect.Visible = False
                btndisconnect.Visible = True
            End If
        Catch
            SerialPort1.Close()
        End Try
    End Sub
```

```

Private Sub btndisconnect_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btndisconnect.Click
    Try
        SerialPort1.Close()
        btnconnect.Visible = True
        btndisconnect.Visible = False
        Exit Sub
    Catch
        MessageBox.Show("Error.")
    End Try
End Sub

```

```

Private Sub Form1_FormClosing(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosingEventArgs) Handles MyBase.FormClosing
    If SerialPort1.IsOpen() Then
        MessageBox.Show ("Disconnect before closing")
        e.Cancel = True
    End If
End Sub

```

```

Private Sub btnAngle1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAngle1.Click

```

```

    If SerialPort1.IsOpen() Then

        SerialPort1.WriteLine((txtAngle1.Text + "C") + (txtAngle2.Text + "R") +
(txtAngle3.Text + "L"))
        txtAngle1.Text = ""
        txtAngle2.Text = ""
        txtAngle3.Text = ""
    End If
End Sub

```

```

Private Sub Inv_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Inv.CheckedChanged

```

```

    txtAngle1.Visible = False
    txtAngle2.Visible = False

```

```
txtAngle1.Visible = False
txtAngle2.Visible = False
txtAngle3.Visible = False
btnAngle1.Visible = False
txtPoint1.Visible = True
txtPoint2.Visible = True
txtPoint3.Visible = True
btnInv.Visible = True
```

End Sub

```
Private Sub btnInv_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnInv.Click
```

```
    Dim inverse As String
    If Not SerialPort1.IsOpen() Then
        SerialPort1.Open()
    End If
    inverse = "inverse" + txtPoint1.Text + txtPoint2.Text + txtPoint3.Text
    SerialPort1.Write(inverse)
    txtPoint1.Text = ""
    txtPoint2.Text = ""
    txtPoint3.Text = ""
```

End Sub

```
Private Sub Fwf_CheckedChanged (ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Fwf.CheckedChanged
```

```
    txtPoint1.Visible = False
    txtPoint2.Visible = False
    txtPoint3.Visible = False
    btnInv.Visible = False
    txtAngle1.Visible = True
    txtAngle2.Visible = True
    txtAngle3.Visible = True
    btnAngle1.Visible = True
```

End Sub

End Class