Sudan University of science and Technology

College of Graduate Studies

Simulation of 3D Car Painting Robotic Arm

محاكاة ذراع روبوت ثلاثي الابعاد لتلوين السيارات

A Thesis Submitted to the College of Graduate Studies at Partial Fulfillment of the Requirements for the Degree of M.Sc. in Mechatronics Engineering

By:

Alaa Mohamed AbdAlmagiedSead Ahmed

Supervisor:

D.rFathElrahman Ismael Khalifa Ahmed

August 2015

الآية

بسم الله الرحمن الرحيم

قَالَ تَعَالَىٰ:

﴿ وَقُل رَّبِّ زِدۡنِي عِلۡمٗا ١١٤ ﴾

صدق الله العظيم

سورة طه

DEDICATION

I want to dedicate this work for my family and friends. It is through their support me that I have been able to focus on my work. My parents taught me that value of learning early on, and have always been supportive me of various decisions to continue my education.

ACKNOWLEDGEMENT

The first thanks and appreciation is for our creator Allah who choose me for this path, and never abandoned me from his mercy. Allah guided me and enlightened our souls and minds through the last semester getting me bounded together as one person and allowing me to know a new definition of the coming life to be which is based on mature thinking and organized group work.

Also each of gratitude and thanksgiving for Dr. Fath Elrahman Ismael Khalifa Ahmed as my supervisor, for his continuous guidance and suggestions throughout the preparation of this research.

Last but not least thanks are also for my family and friends.

ABSTRACT

Robotic is an electromechanical system, help improve production accuracy, and reduces time and increase production.

In this project a study of 3D robotic ARM design, simulation concepts and theories for the painting of cars that deals with the design, construction, operation, and application of robots is conducted. The simulation covers different parameters to evaluate the design. Moreover simulation environment help to test these parameters cost effectiveness of building robots and selecting optimal component, dimensions and Accuracy is used.

COMSOL application was selected as a simulation design and modeling program, that give the ability to fabricate 3D Objects and simulate the movement, the matlab program was used to control the object movement through commands stored in M File linked to the graphical user interface.

The 3D robotic arm was tested in MATLAB environment. The test has been applied for car painting in its three parts front, middle and end of the car.

المستخلص

الروبوت هو نظام الكتروميكانيكي يساعد على تطوير دقه الانتاج تزامنا مع زياده الانتاج وتقليل الزمن والفاقد.

في هذا المشروع سيتم دراسه تصميم ذراع روبوت ثلاثي الابعاد وعمل نمذجه للافكار والنظريات لتلوين السيارات والتعامل مع التصميم والتركيب والتشغيل وتطبيقات الروبوتات .

النمذجه تغطي عوامل متغيرات التجربه المختلفه لتقييم التصميم وايضا البيئه المحيطه للنمذجه تساعد على اختبار هذه العوامل من دون زياده التكلفه لبناء الروبوت واختيار المكونات مع مراعه الدقه والابعاد.

تم اختيار تطبيقCOMSOLلتنفيذ وعمل نمذجه البرنامج الذي يسمح بعمل او تصميم اي جسم في االنظام ثلاثي الابعاد وتحركاته حيث تم ربطه ببرنامج الماتلاب للتحكم في حركه هذا الجسم من خلال اوامر محفوظه في الماتلاب.

تم تحقيق الهدف المقصود من المشروع باختبار ذراع الروبوت ثلاثي الابعاد بواسطه برنامج الماتلاب للعمل على تلوين سياره وذلك بتقسيم السياره الى ثلاثه اجزاء امام ووسط والجزء الخلفي للسياره.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

L: Length

W: weight

M: torque

V: Velocity

R: radius

Y: rpm

N: momentum of inertia

A: angular acceleration

LIST OF ABBREVIATIONS

CFD: Computational Fluid Dynamics

CPU: Central Processing Unit

CNC: Computer Numerical Control

ERBS: Electrostatic Rotary Bell Sprayer

FCC: Fraunhofer-Chalmers Centre

GUI: graphical user interface

ICs: Integrated Computer solutions

ISO: International Organization for Standardization

RAM: Random-Access Memory

PCB: Printed Circuit Board

PWM: Pulse Width Modulation

CHAPTER ONE

INTRODUCTION

1.1 Preface

A robot is an electro-mechanical system which can perform any of the operation commanded by the user with the help of input devices like remote controller or any type of sensor, which are interfaced with some computer and electronic programming. Certain algorithms are the controlling medium created by the user so that it performs the specified functions. Majorly any robot can be of two types: semi-autonomous and autonomous. [1]

Robotics is the branch of technology that deals with the design, construction, operation, and application of robots, as well as computer systems for their control, sensory feedback, and information processing in this project High risky harmful duties in different fields such as industrials, radioactive lab and the high power tension generators, Cost effectiveness of building robots without using a simulation program to select the optimal materials, component and dimensions Accuracy on movement and Lake of accuracy while using manpower.[2]

1.2 Problem Statement

The cost effective design of 3D robotic arm to be used in industrial activities such as car painting require extensive simulation in order select optimal component and dimensions to achieve the required accuracy and time response.

1.3Proposed Solution

Design and simulate a robotic arm using 3D mechanical design simulation using COMSOL software with an application of 3D painting of cars in car industrial production lines.

1.4 Thesis Aim and Objectives

The General aim of this research is to design and simulate a 3D robotic arm using a simulation and design software COMSOL, in order to use it as an for painting cars on industrial production line.

Specific objectives

- To calculate loads that will be handled by the robotic arm.
- To choose motors types regarding the load and desired torque.
- To choose components used to construct robotic arm.
- To design and simulate the robotic arm based on the calculations into COMSOL simulation and design software.
- To evaluate the 3D arm robotic which used for car painting.

1.4 Methodology

In order to achieve a fixable smooth design of robotic arm a mathematical calculation and consideration isused along with applying them into the COMOL software to produce the simulation of the robotic arm. The COMSOL has been interfaced with MATLAB code saved in M file. After simulation is done a degree of motion will be set to each motor in the robot. The maximum and minimum values will be set, and then results will be obtained. The control of the robot is done through graphical user interface.

1.5 Thesis Outlines

The thesis contains five chapters, chapter one is an introduction that includes preface, problem statement of the project also the objectives of the project and the methodology. Chapter two explains a literature review of the robotics and background of the study. Chapter three discusses the system design of the project including the tools and requirements and chapter four explain the simulation and results. Finish of the thesis with solid conclusion and good recommendations for future work are.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

The earliest known industrial robot, conforming to the ISO definition was completed by "Bill" Griffith P. Taylor in 1937 and published in Meccano Magazine, March 1938.[3] The crane-like device was built almost entirely using Meccano parts, and powered by a single electric motor. Five axes of movement were possible, including grab and grab rotation. Automation was achieved using punched paper tape to energize solenoids, which would facilitate the movement of the crane's control levers. The robot could stack wooden blocks in pre-programmed patterns. The number of motor revolutions required for each desired movement was first plotted on graph paper. This information was then transferred to the paper tape, which was also driven by the robot's single motor. Chris Shute built a complete replica of the robot in 1997. [4]

George Devol applied for the first robotics patents in 1954 (granted in 1961). The first company to produce a robot was Unimation, founded by Devol and Joseph F. Engelberger in 1956, and was based on Devol's original patents. Unimation robots were also called programmable transfer machines since their main use at first was to transfer objects from one point to another, less than a dozen feet or so apart. They used hydraulic actuators and were programmed in joint coordinates. Unimation later licensed their technology to Kawasaki Heavy Industries and GKN, manufacturing Unimates in Japan and England respectively. For some time Unimation's only competitor was Cincinnati Milacron

Inc. of Ohio. This changed radically in the late 1970s when several big Japanese conglomerates began producing similar industrial robots.[5]

In 1969 Victor Scheinman at Stanford University invented the Stanford arm, an all-electric, 6-axis articulated robot designed to permit an arm solution. This allowed it accurately to follow arbitrary paths in space and widened the potential use of the robot to more sophisticated applications such as assembly and welding.

At the height of the robot boom in 1984, Unimation was acquired by Westinghouse Electric Corporation for 107 million U.S. dollars. Westinghouse sold Unimation to StäubliFaverges SCA of France in 1988, which is still making articulated robots for general industrial and clean room applications and even bought the robotic division of Bosch in late 2004.

2.2Background

Robotics is the branch of technology that deals with the design, construction, operation, and application of robots, as well as computer systems for their control, sensory feedback, and information processing.

These technologies deal with automated machines that can take the place of humans in dangerous environments or manufacturing processes, or resemble humans in appearance, behavior, and/or cognition.

A robot is a machine capable of physical motion for interacting with the environment. Physical interactions include manipulation, locomotion, and any other tasks changing the state of the environment or the state of the robot relative to the environment. A robot has some form of mechanisms for performing a class of tasks. A rich variety of robot mechanisms has been developed in the last few decades. In this chapter, we will give an overview on the various types of mechanisms used for generating robotic motion.[5]

2.2.1Types of robotic arm

Types of robotic arms where developed such as arms:

- Cartesian robot / Gantry robot: Used for pick and place work, application of sealant, assembly operations, handling machine tools and arc welding. This robot arm has three prismatic joints, whose axes are coincident with a Cartesian coordinator. Cartesian robot illustrated in figure (2.1)

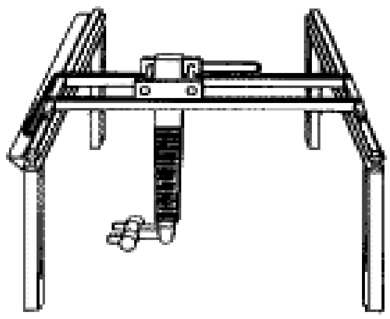


Figure2.1: Cartesian robot

Cylindrical robot: Used for assembly operations, handling at machine tools, spot welding, and handling of die casting machines. It's a robot whose axes form a cylindrical coordinate system .Cylindrical robot shown in figure (2.2).

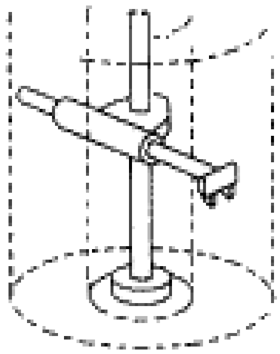


Figure2.2: Cylindrical robot

- Spherical robot / Polar robot (such as the Unimate): Used for handling machine tools, spot welding, die casting, fettling machines, gas welding and arc welding. It's a robot whose axes form a polar coordinate system. Spherical robot illustrated in figure (2.3).

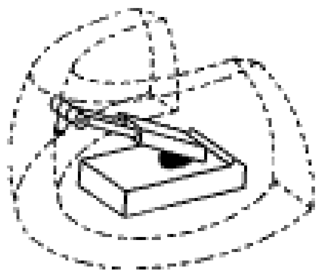


Figure2.3: Spherical robot

- SCARA robot: Used for pick and place work, application of sealant, assembly operations and handling machine tools. This robot has two parallel rotary joints to provide compliance in a plane .SCARA robot shown in figure (2.4).

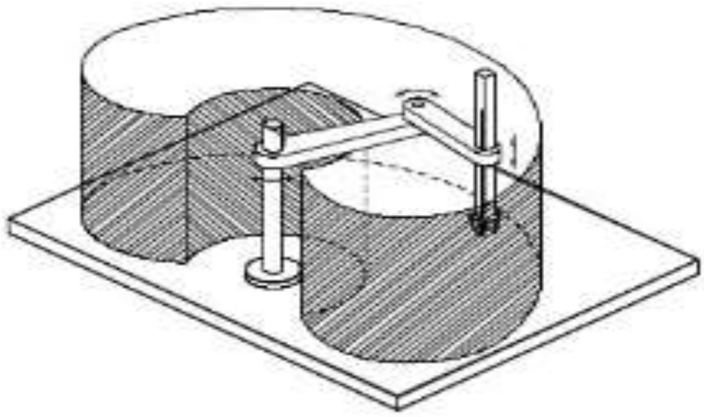


Figure2.4: SCARA robot

- Articulated robot: Used for assembly operations, die casting, fettling machines, gas welding, arc welding and spray painting. This type of robot arm has at least three rotary joints. Articulated robot illustrated in figure (2.5).

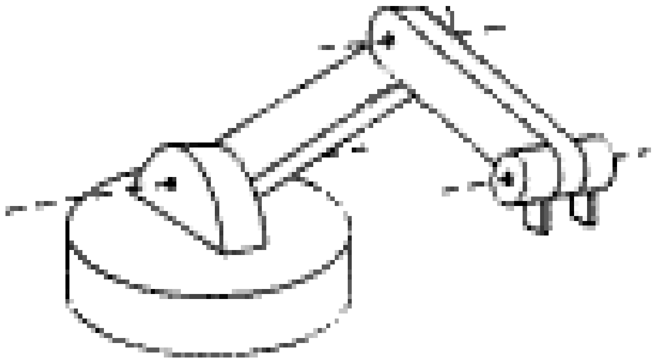


Figure2.5: Articulated robot

- Parallel robot: used in mobile platform for handling cockpit flight simulators. It's a robot whose arms have concurrent prismatic or rotary joints. Parallel robot illustrated in figure (2.6).

❖

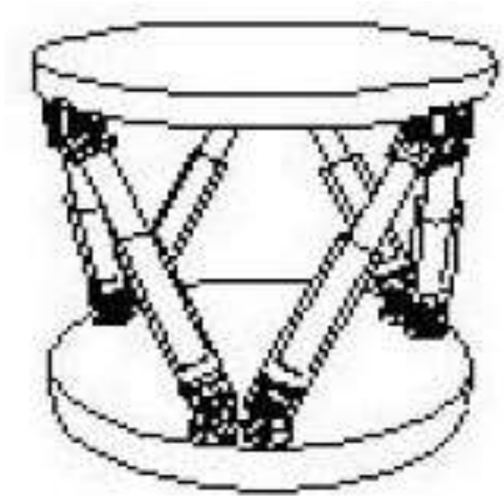


Figure2.6: Parallel robot

- Anthropomorphic robot: Similar to the robotic hand Luke Skywalker receives at the end of The Empire Strikes Back. It is shaped in a way that resembles a human hand, i.e. with independent fingers and thumbs .Anthropomorphic robot shown in figure (2.7).

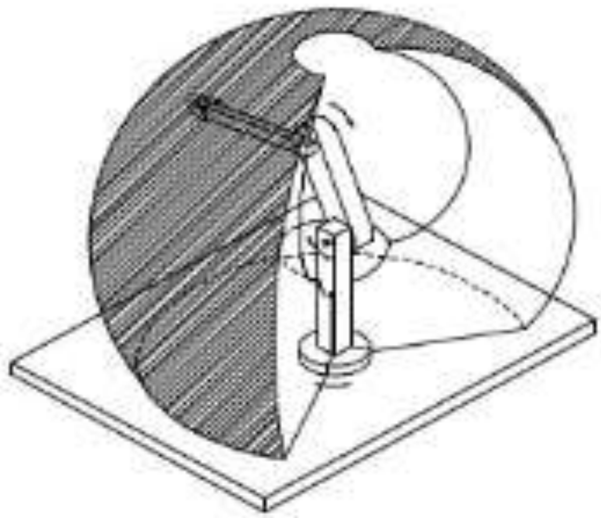


Figure2.7: Anthropomorphic robot

2.2.2Applications of robotic arm

❖ The robotic arm can be designed to perform any desired task such as welding, gripping, spinning etc., depending on the application. For example robot arms in automotive assembly line perform a variety of tasks such as wielding and parts rotation and placement during assembly.

- ❖ In space the space shuttle Remote Manipulator System have multi degree of freedom robotic arms that have been used to perform a variety of tasks such as inspections of the Space Shuttle using a specially deployed boom with cameras and sensors attached at the end effectors.
- ❖ The robot arms can be autonomous or controlled manually and can be used to perform a variety of tasks with great accuracy. The robotic arm can be fixed or mobile (i.e. wheeled) and can be designed for industrial or home applications. Robotic hands often have built-in pressure sensors that tell the computer how hard the robot is gripping a particular object. This keeps the robot from dropping or breaking whatever it's carrying.

2.3 Related Works

In [7], the author developed mechanism of a robotic arm that serves as a tool to lift an object from one place to another where it is widely used in the factory. The study of the material was analyzed using computer software that can calculate the finite element of linear stress analysis of each mechanical components of robotic arm. the Results of the analysis was used as a reference to select suitable material. The aluminum 6061 was used. In addition, the selection of electrical components used in the robotic arm is also taken into account by calculating the inverse kinematic and forward kinematic of this robotic arm movement. Besides that, the forces exerted on the robotic arm are also calculated to ensure that mechanical components of the robotic arm is not easily broken or damaged. Referring to the result obtained, a robotic arm resistance depends on the motor used. Therefore, the compatibility of motor torque with the robotic arm design is made is important because it affects the stability of the robotic arm.

The author in [7] deals with the aluminum as a hard element to be broken, and the best fabrication notes and consideration.

The main problem of the work in [7] is that it doesn’t deal with the joint and the calculations of the robot structure.

In [8] , designing a robot arm control using MATLAB is consedered. It is designed to be used in the movement either to the left and right and is also used to lift an object using MATLAB a robot arm is controlled. The project uses a PIC 16F877A microcontroller circuit as the basic circuit. 3 servo motors were used as an application extension to make movements and lifting an object. In[8], the PIC microcontroller to be programmed instructions to control the servo motor. This project uses MATLAB as a graphical user interface (GUI) for controlling the movement of this robot. The microcontroller software will be standardized to achieve the simulation is not always limited to the convergence between the tools used by the circuit. These types is also designed in such electronics industry and the manufacturing industry in order to achieve the objectives, there are several scope had been outlined. The scope of this project includes using PICBASIC PRO programming to program microcontroller PIC16F877A, build PIC microcontroller circuit and robotic arm for the system, and interface the microcontroller to computer by using RS232 serial port communication. Servo motors are also used for robotic arm and control by MATLAB. MATLAB is used to control PIC microcontroller which is connected to robotic arm. Robotic arm movement can be done by moving to the left or right to take and put things in different places.

The author in [8], covers the rs232 connection to the MATLAB and robotic arm, including the procedures and subroutines and the limitations.

In [9], Robotic arms are used in lifting heavy objects and carrying out tasks that require extreme concentration and expert accuracy. This study mainly focuses on the accuracy in control mechanism of the arm while gripping and placing of objects. A design has been proposed to replicate an industrial robot arm with a reach in a three dimensional space which could pick and place objects specified. The three dimensional space access mechanisms operate in cylindrical coordinate system. The operating domain is a cylindrical sector of a fixed radius and height and limited rotation. A four jaw angular gripper will be of use to grip the object firmly with a precise stress.

The gripping precision could be defined for objects within the specified dimension of the object. The stress on the object can be controlled. The system facilitates autonomous object detection within its limitations. A user interface is incorporated with the system for human input feed on the desired destination within the working frontiers.

The targeted destination is specified in terms of height, radius and angle. In addition the orientation of the object can be provisioned along with the destination.

In [9], the design of the circuit is in focus including the microcontroller and sensors with detectors, and the limitations of controlling robotics using microcontroller.

In [10], many robots do jobs that are hazardous to people such as defusing bombs, exploring shipwrecks, and mines. In this way electrical field has wide area where we can use the robot instead of our conventional methods. In power system generation boiler is one of the place where the robot technology is widely used for maintenance purpose because human maintenance is too difficult due to high temperature but robot can easily perform the maintenance in live condition also In a narrow space or a hazardous space human effort is not applicable properly in that case robot can do important task to perform proper maintenance. i.e. it can be observe the facts like Crack detection test in boiler inner wall, Any type of physical operation at nuclear reactor, Clearing any obstacle in a hollow channel, Commissioning any instrument in a narrow space and Spot welding etc. This type of operation by human is very troublesome, injurious and harmful.

Sometimes it may causes death. The implementation of robotics in a complicated, hazardous or narrow space is an efficient and economic approach.

In [10], The benefits of the robotic in industrial that protect the human from harmful and dangerous equipment. Also the author focused in the basic operation of robot in narrow and hazardous space is stated.

In [11] ,The Electrostatic Rotary Bell Sprayer (ERBS) technique for painting process where a high-speed rotating device with high electric potential atomizes the paint producing charged droplets and sprays them through the air onto a surface was developed . These devices commonly use a shaping-air system, which in conjunction with the electrostatic forces present, direct the particles to the target.

In work [11], IBOFlow, a Computational Fluid Dynamics (CFD) solver developed at the Fraunhofer-Chalmers Centre (FCC) is employed to perform simulations on an ERBS system. The results are then compared with empirical measures obtained at the Swerea IVF research laboratories looking for validation. Subsequently, the focus is set specifically on simulating the region near the applicator bell. In the present work, a rotary-bell spray painting system was successfully modeled and simulated. Simultaneously, a parametric study of an electrostatic rotary-bell spray painting system was carried out at the Swerea IVF laboratories using the laser light sheet visualization technique. From these methodical trials important features of the system were identified, and a better understanding of the paint spray structure and the transfer process was gained.

In [11], the discussion of the spray automotive calibration is done through calculation of air flow and other parameters that affect the painting process.

CHAPTER THREE

SYSTEM DESIGN

3.1 Introduction

A robotic arm is a robotic manipulator, usually programmable, with similar functions to a human arm. This Robotic arm is programmable in nature and it can be manipulated. The robotic arm is also sometimes referred to as anthropomorphic as it is very similar to that of a human hand. Humans today do all the tasks involved in the manufacturing industry by them self's. However, a Robotic arm can be used for various tasks such as welding, drilling, sprayingand many more. A self sufficient robotic arm is fabricated by using components like microcontrollers and motors. This increases their speed of operation and reduces the complexity. It also brings about an increase in productivity which makes it easy to shift to hazardous materials. In the implementation process, the necessary components of structure ICs, blocks and power supply are all assembled on the PCB .The main part of the design is microcontroller which coordinates and controls the product's action. This specific micro controller is used in various types of embedded applications. Robotics involves elements of mechanical and electrical engineering, as well as control theory, computing and now artificial intelligence. According to the Robot Institute of America, "A robot is a reprogrammable, multifunctional manipulator designed to move materials, parts, tools or specialized devices through variable programmed motions for the performance of a variety of tasks. The robots interact with their environment, which is an important objective in the development of robots. This interaction is commonly established by means of some sort of arm and gripping device or end effectors. In the robotic arm, the arm has a few joints, similar to a human arm, in addition to shoulder, elbow, and wrist, coupled with the finger joints; there are many joints. [12]

The design process is clearly explained in the next section with detailed information regarding the components which are used, followed by the implementation leading to results and finally ends with conclusion.

The Robotic Arm is designed using the Microcontroller this process works on the principle of interfacing servos and potentiometers. The Potentiometers play an important role the remote is fitted with potentiometers and the servos are attached to the body of the robotic arm. The potentiometer converts the mechanical motion into electrical motion. Hence, on the motion of the remote the potentiometers produce the electrical pulses, which are en route for PCB board. The board then processes the signals received from the potentiometers and finally, converts them into requisite digital pulses that are then sent to the servomotors. This servo will respond with regards to the pulses which results in the moment of the arm.

3.2Control Board Block Diagram

The block diagram of the robotic arm consist of a microcontroller that used to control the movement of the motor regarding the input control signal, the microcontroller reads the axesand coordinates from dataset that includes a set of points that each part of the arm should move, and the dimension of the painted object, here the microcontroller convert the file into bunch of motor movement control pulses and through motor driver it moves the actuator (motors), the input switches represents and external interrupt to the process. The block diagram shown in figure(3.1).

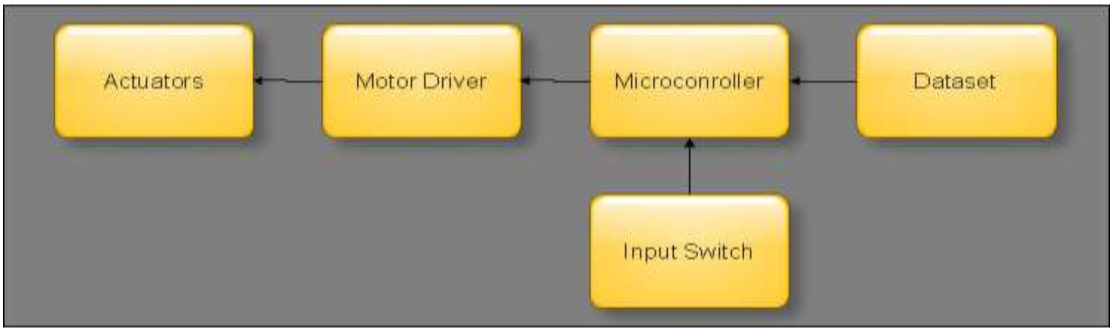


Figure 3.1: System Block Diagram.

3.3 Mathematical Model of Design

In order to design and fabricate the robotic arm some calculations should be done as the following:

3.3.1Force calculations of joints

The point of doing force calculations is for motor selection. Chosen motors must not only support the weight of the robot arm, but also what the robot arm will carry (the blue ball in the image below).

The first step is to label your FBD, with the robot arm stretched out to its maximum length. Figure 3.2 illiterates the weight and length of each part of the robot W1 represents the shoulder weight W2 is the weight on risk and gripper while W3 represent the weight on gripper and W4 represents the risk load weight. M1andM2 is the torque at jonts, and L1,L2 and L3 represents the length of the arm parts.

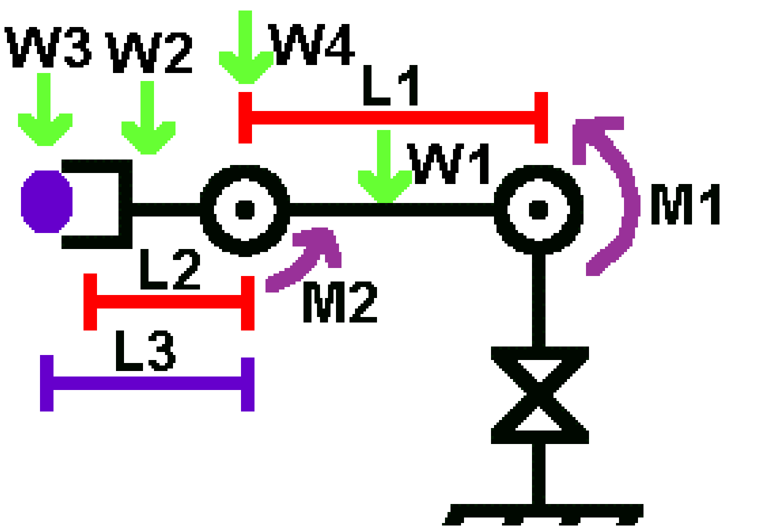


Figure 3.2: Robotic Arm Sketch

These parameters used in mathematical model to design the robotic arm along with the weight of each linkage, weight of each joint, weight of object to lift and length of each linkage**,** this calculation must be done for each lifting actuator.

Using the above parameters the following equation will be used to calculate and design the robotic arm.

- Motor torque joint (I)

Joint (I) represents the overall weight on the arm which be calculated depending on the motor torque, and it was used here to detect the motor specifications that needed to handle the load weight. Equation (3.1) illustrates motor1 torque calculation

$$M1 = {}^{L1}/_{2} * W1 + L1 * W4 + \left(L1 + {}^{L2}/_{2}\right) * W2 + (L1 + L3) * W3 \qquad (3.1)$$

Where:

L: Length

W: weight

M: torque

- Motor torque joint (II)

Joint (II) represents the overall weight on the arm which be calculated depending on the motor torque, and it was used here to detect the motor specifications that needed to handle the load weight. Equation (3.2) illustrates motor 2 torque calculations.

$$M2 = {}^{L2}/_{2} * W2 + L3 * W3 \quad (3.2)$$

Where:

L: Length

W: weight

M: torque

And it was found that shorter arm lengths allow for smaller torque requirements. Each motor after the main load motor M (I) reduces the equation to the front motor, because no load from behind except a potential load for the motor.[13]

3.3.2Velocity and motion planning

Calculating end effecter velocity is mathematically complex. The simplest way to do it is assuming that robot arm (held straight out) is a rotating wheel of L diameter. The joint rotates at Y rpm, so therefore the velocity is:

$$V = 2 * \pi * R * Y \quad (3.3)$$

Where:

V: Velocity of end effecter on straight arm

R: radius

Y: rpm

The end effecter does not just rotate about the base, but can go in many directions. The end effecter can follow a straight line, or curve, etc.

With robot arms, the quickest way between two points is often not a straight line. If two joints have two different motors, or carry different loads, then max velocity can vary between them. [13]

3.3.3 Momentum calculation

There are many deciding factors. Usually you want straight lines when the object the arm moves is really heavy, as it requires the momentum change for movement .momentum equation is represent in equation (3.4)

$$N = W * V \tag{3.4}$$

Where:

N: momentum

W: Weight

V: Velocity

But for maximum speed (perhaps the arm isn't carrying anything, or just light objects) you would want maximum joint speeds. [13]

If joint J0 to rotate 180 degrees in less than 2 seconds, what torque does the J0 motor need? Well, J0 is not affected by gravity, so all we need to consider is momentum and inertia. Putting this in equation form we get this:

$$M = N * A \tag{3.5}$$

Where:

M: Torque

N: momentum of inertia

A: angular acceleration

3.4Requirements and Components

The main require components is:

3.4.1Controller

Controller is the "brain" of the industrial robotic arm and allows the parts of the robot to operate together. It works as a computer and allows the robot to also be connected to other systems. The robotic arm controller runs a set of instructions written in code called a program. The program is inputted with a teach pendant. Many of today's industrial robot arms use an interface that resembles or is built on the Windows operating system. Moreover the microcontroller considered as one of the best controllers in arm robotic and thus it has been chosen in this design.

Microcontrollerit was used to control the circuit of the robot and it can be defined as MCU or µC is a functional computer system-on-a-chip. It contains a processor core, memory, and programmable input/output peripherals. Microcontrollers include an integrated CPU, memory (a small amount of RAM, program memory, or both) and peripherals capable of input and output. Microcontroller architecture illustrates in figure (3.3).

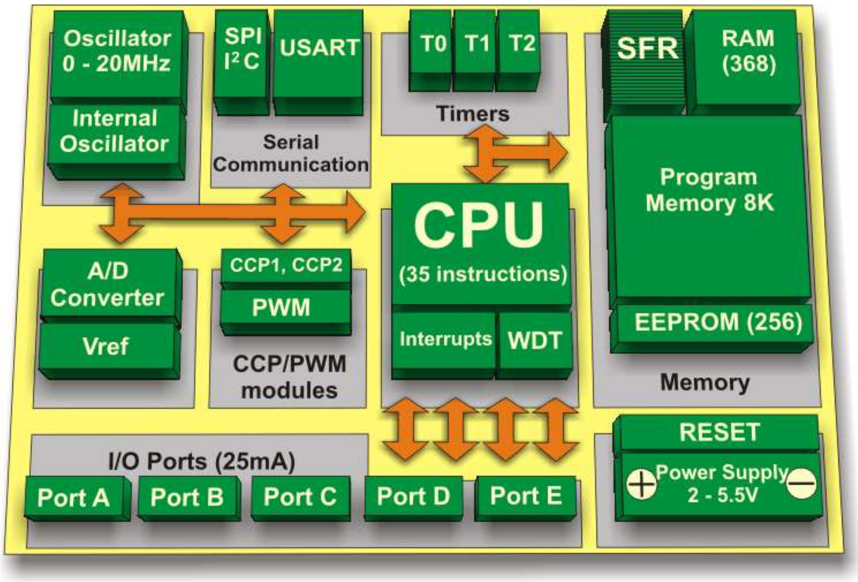


Figure 3.3: shows microcontroller architecture

Microcontroller programs must fit in the available on-chip program memory, since it would be costly to provide a system with external, expandable, memory. Compilers and assembly language are used to turn high-level language programs into a compact machine code for storage in the microcontroller's memory. Microcontrollers were originally programmed only in assembly language, but various high-level programming languages are now also in common use to target microcontrollers. These languages are either designed especially for the purpose, or versions of general purpose languages such as the C programming language. [14]

3.4.2Industrial robot arms

Industrial robot arms can vary in size and shape. The industrial robot arm is the part that positions the end effector. With the robot arm, the shoulder, elbow, and wrist move and twist to position the end effector in the exact right spot. Each of these joints gives the robot another degree of freedom. A simple robot with three degrees of freedom can move in three ways: up & down, left & right, and forward & backward. Many industrial robots in factories today are six axis robots

3.4.3The end effector

The **end effector** connects to the robot's arm and functions as a hand. This part comes in direct contact with the material the robot is manipulating. Some variations of an effector are a gripper, a vacuum pump, painting, magnets, and welding torches. Some robots are capable of changing end effectors and can be programmed for different sets of tasks.

3.4.4The drive

The **drive** is the engine or motor that moves the links into their designated positions. The links are the sections between the joints. Industrial robot arms generally use one of the following types of drives: hydraulic, electric, or pneumatic. Hydraulic drive systems give a robot great speed and strength. An electric system provides a robot with less speed and more strength. Pneumatic drive systems are used for smaller robots that have fewer axes of movement. Drives should be periodically inspected for wear and replaced if necessary. In this project electric drive system has been used to fulfill high accuracy in the application of car painting which needs less speed and more strength. A servo motor is the main part which operates the 3D arm robotic.

A servomotor is a rotary actuator that allows for precise control of angular position, velocity and acceleration.[1] It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.Servomotors are not a specific class of motor although the term servomotor is often used to refer to a motor suitable for use in a closed-loop control system.Servomotors are used in applications such as robotics, CNCmachinery or automated manufacturing.

To fully understand how the servo works, you need to take a look under the hood. Inside there is a pretty simple set-up: a small DC motor, potentiometer and a control circuit. The motor is attached by gears to the control wheel .As the motor rotates, the potentiometer's resistance changes, so the control circuit can precisely regulate how much movement there is and in which direction. Servo motor represent in figure (3.5).

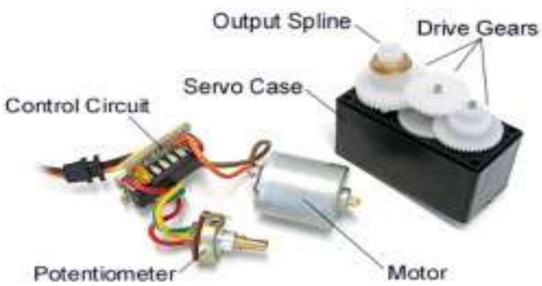


Figure3.5: servo motor

When the shaft of the motor is at the desired position, power supplied to the motor is stopped. If not, the motor is turned in the appropriate direction. The desired position is sent via electrical pulses through the signal wire. The motor's speed is proportional to the difference between its actual position and desired position. So if the motor is near the desired position, it will turn slowly, otherwise it will turn fast. This is called proportional control. This means the motor will only run as hard as necessary to accomplish the task at hand.

Servos are controlled by sending an electrical pulse of variable width, or pulse width modulation (PWM), through the control wire. There is a minimum pulse, a maximum pulse and a repetition rate. A servo motor can usually only turn 90° in either direction for a total of 180° movement. The motor's neutral position is defined as the position where the servo has the same amount of potential rotation in the both the clockwise or counter-clockwise direction. The PWM sent to the motor determines position of the shaft, and based on the duration of the pulse sent via the control wire the rotor will turn to the desired position. The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the 90° position. Shorter than 1.5ms moves it to 0° and any longer than 1.5ms will turn the servo to 180°, as diagramed below.[15]

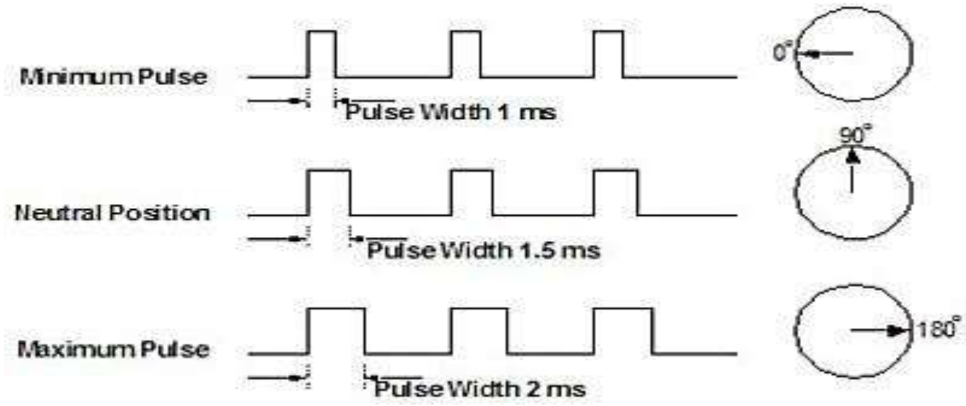


Figure3.6: Servo controller with (PWM)

When these servos are commanded to move, they will move to the position and hold that position. If an external force pushes against the servo while the servo is holding a position, the servo will resist from moving out of that position. The maximum amount of force the servo can exert is called the torque rating of the servo. Servos will not hold their position forever though; the position pulse must be repeated to instruct the servo to stay in position.

Motor driver integrated circuit (L293). The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

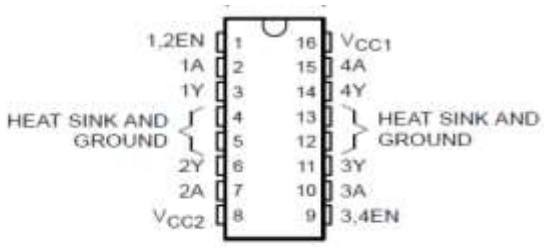


Figure3.7: DIP L293

All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN.When enable input is high, the associated drivers are enabled and their outputs are active and in phase with their inputs. When enable input is low, those drivers are disabled and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications. On the L293, external high-speed output clamp diodes should be used for inductive transient suppression. A VCC1 terminal, separate from VCC2, is provided for the logic inputs to minimize device power dissipation.

3.4.5 Sensors

Sensors allow the industrial robotic arm to receive feedback about its environment. They can give the robot a limited sense of sight and sound. The sensor collects information and sends it electronically to the robot controlled. One use of these sensors is to keep two robots that work closely together from bumping into each other. Sensors can also assist end effectors by adjusting for part variances. Vision sensors allow a pick and place robot to differentiate between items to choose and items to ignore.

PIR sensors allow you to sense motion, almost always used to detect whether a human has moved in or out of the sensors range. They are small, inexpensive, low-power, easy to use and don't wear out. For that reason they are commonly found in appliances and gadgets used in homes or businesses. They are often referred to as PIR, "Passive Infrared", "Pyroelectric", or "IRmotion" sensors. For many basic projects or products that need to detect when a person has left or entered thearea, or has approached, PIR sensors are great. They are low power and low cost, pretty rugged, have a wide lens range, and are easy to interface with. Note that PIRs won't tell you howmany people are around or how close they are to the sensor, the lens is often fixed to a certain sweep and distance (although it can be hacked somewhere) and they are also sometimes set off by housepets.PIR sensor illustrates in figure (3.7).

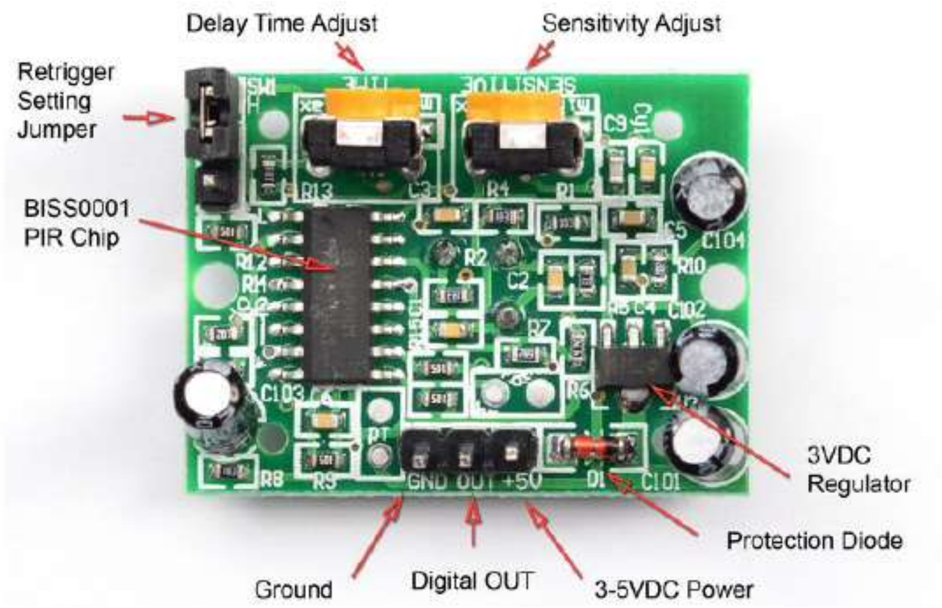


Figure3.7: PIR sensor

CHAPTER FOUR

DESIGN AND SIMULATION

4.1 Introduction

In this chapter the design and simulation of robotic arm included, and also including an introduction to simulation software and simulation steps along with the graphical user interface (GUI), that used to move each part of robotic arm separately.

Moreover a flowchart was made to illustrate the program flow and describe each process and subroutine used in the code.

4.2 Robotic Simulator Application (COMSOL)

A COMSOL robotics simulator is used to create embedded applications for a robot without depending physically on the actual machine, thus saving cost and time.

These applications can be transferred on the real robot (or rebuilt) without modifications. The term robotics simulator can refer to several different robotics simulation applications. For example, in mobile robotics applications, behavior-based robotics simulators allow users to create simple worlds of rigid objects and light sources and to program robots to interact with these worlds. Behavior-based simulation allows for actions that are more biological in nature when compared to simulators that are more binary, or computational. In addition, behavior-based simulators may "learn" from mistakes and are capable of demonstrating the anthropomorphic quality of tenacity. Figure (4.1) show Robologix robotics simulator. [16]

Figure4.1: Robologix robotics simulator.

One of the most popular applications for robotics simulators is for 3D modeling and rendering of a robot and its environment. This type of robotics software has a simulator that is a virtual robot, which is capable of emulating the motion of an actual robot in a real work envelope. Some robotics simulators, such as Robologix use a physics engine for more realistic motion generation of the robot. The use of a robotics simulator for development of a robotics control program is highly recommended regardless of whether an actual robot is available or not. The simulator allows for robotics programs to be conveniently written and debugged off-line with the final version of the program tested on an actual robot. Of course, this primarily holds for industrial robotic applications only, since the success of off-line programming depends on how similar the real environment of the robot is to the simulated environment. Sensor-based robot actions are much more difficult to simulate and/or to program off-line, since the robot motion depends on the instantaneous sensor readings in the real world. [16]

COMSOL Metaphysics is a finite element analysis, solver and Simulation software / FEA Software package for various physics and engineering applications, especially coupled phenomena, or metaphysics. The packages are cross-platform (Windows, Mac, Linux). In addition to conventional physics-based user interfaces, COMSOL Metaphysics also allows for entering coupled systems of partial differential equations (PDEs). The PDEs can be entered directly or using the so-called weak form an early version (before 2005) of COMSOL Metaphysics was called FEMLAB. [16]

The main product is COMSOL Desktop which is an integrated user interface environment designed for cross-disciplinary product development with a unified workflow for electrical, mechanical, fluid, and chemical applications. The add-on modules blend into COMSOL Desktop, and the way of operation of the software remains the same no matter which add-on products are engaged. [17]

COMSOL Metaphysics also provides application programming interfaces (APIs). The COMSOL API for use with Java comes included with COMSOL Metaphysics, and provides a programmatic way of driving the software through compiled object oriented code. Live Link for MATLAB allows working with COMSOL Metaphysics in combination with the MATLAB.

The Physics Builder, which is included in COMSOL Desktop, allows creating custom made physics interfaces accessible from the COMSOL Desktop with the same look-and-feel as the built-in physics interfaces. In the case of the Physics Builder, no programming is needed as it works in the COMSOL Desktop from the Physics Builder Tree, defining new user interface components.[16][17]

The Applications Builder is also available with COMSOL Desktop and allows saving models as specialized applications for use without going into the details of the simulations model. Two editors are available for designing applications; using drag-and-drop tools, in the Form Editor, or by programming using the Method Editor. There is scope to include specific features from the model or introduce new ones through programming using the Method Editor. [18]

4.3Flowchart

The following flowchart in illustrate the software flow inside the programmable logic controller as illustrated on figure (4.2).The system flow chart helps developers in writing their programs and simplifies the concept of the software to provide a full understanding of program sequence flow.

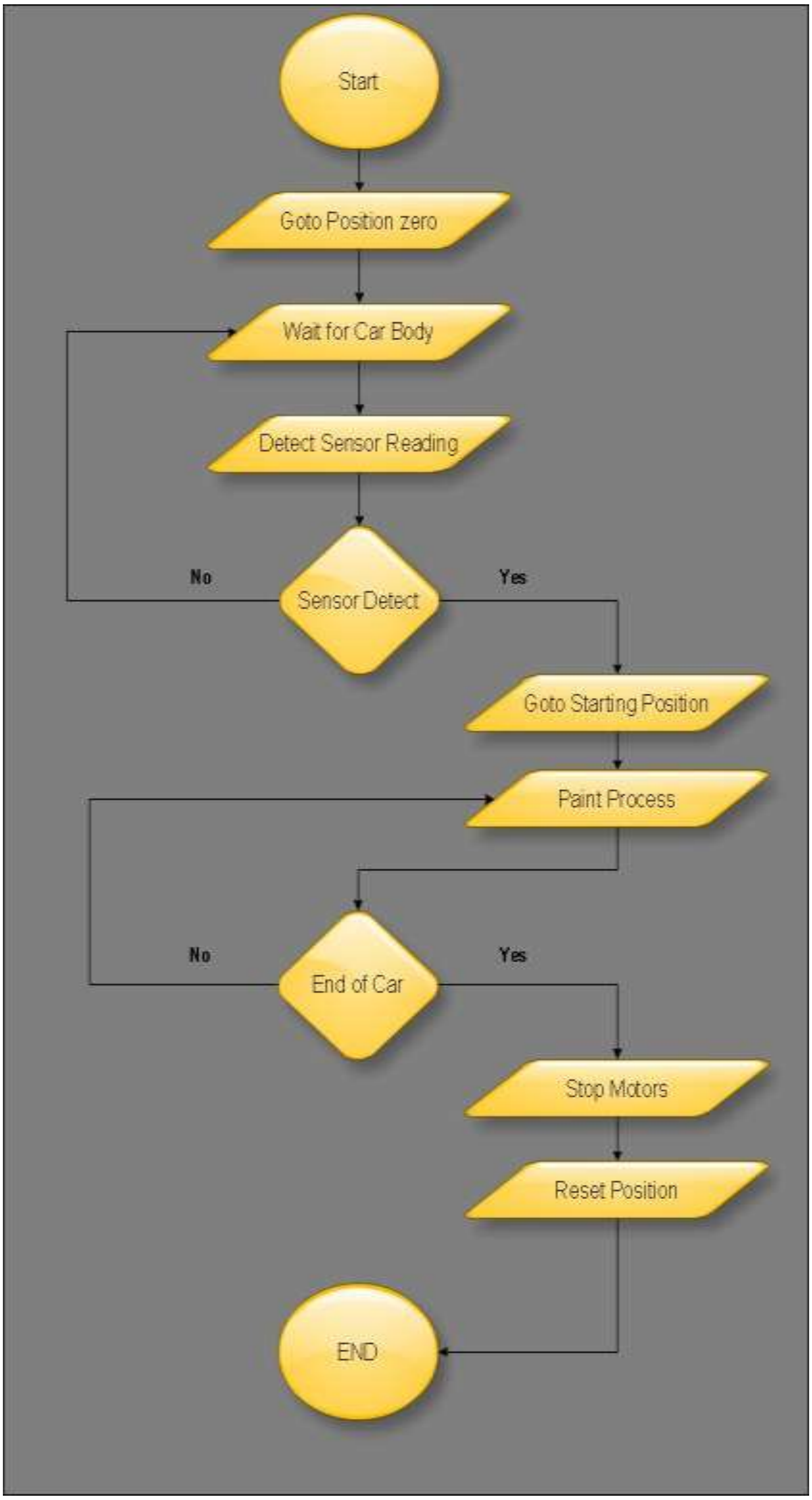


Figure4.2: Flowchart

4.3 3D Robotic Arm Parts and Movement

In this section the motors and robotic arm parts are represented in the following tables.

In table 4.1 six motors with main job are presented which represents the main construction of robotic arm movement, which has six major movement including Shoulder ,Elbow, Pitch, Roll, Yaw and Drill these parts give the robot the ability to paint in 3D or 2D.

Moreover the adjustment of movement degree was selected according to the movement of robot on the production line, some of arm motors must not pass the 45 degree and it is manually adjusted to fit the application of robotic arm As shown in table 4.1and through Matlab commands that stored into m-file and linked to graphical user interface (GUI) the motors will be controlled.

Table4.1: Motors and Arm Parts

No.	Motors	Arm Parts	Degree
1	Motor 1	Shoulder	0-45
2	Motor 2	Elbow	0-180
3	Motor 3	Pitch	0-45
4	Motor 4	Roll	0-360
5	Motor 5	Yaw	0-45
6	Motor 6	Drill	0-45

- Motor 1 (shoulder)

This motor handle the total weight of the robot including motor 1 itself to motor 6 and the degree of motion was set to 0-45 degree and through direct coupling it was joint to motor 2.shoulder motor illustrates in figure(4.3).

Figure4.3:Part of Shoulder (motor1)

- Motor 2 (Elbow)

This motor handle the total weight of the robot including motor 2 to motor 6 and the degree of motion was set to 0-180 in 3D x, y and z degree and through direct coupling it was joint to motor 3. Elbow motor represents in figure (4.4).

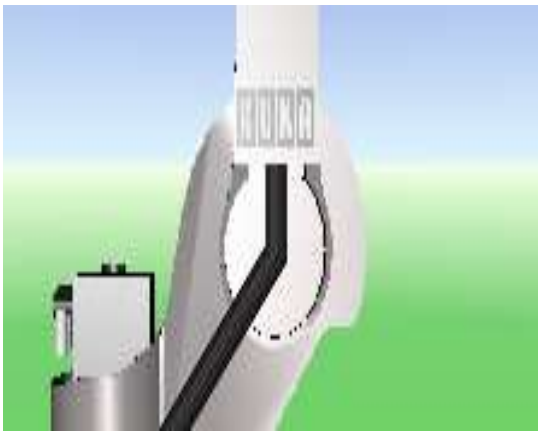


Figure 4.4:Part of Elbow(motor 2)

- Motor 3 (Pitch)

This motor handle the total weight of the robot including motor 3 to motor 6 and the degree of motion was set to 0-45 degree .Motor 3 through direct coupling it was joint to motor 4.Pitch motor shown in figure (4.5).

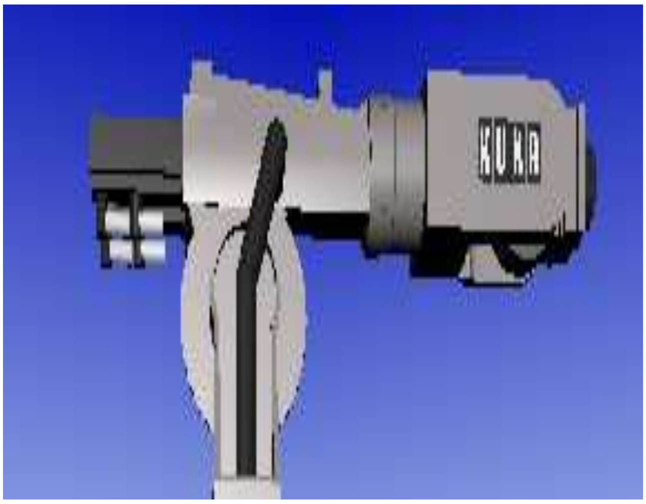


Figure4.5: Part of Pitch (motor3)

- Motor 4 (ROLL)

This motor handle the total weight of the robot including motor 4 to motor 6 and the degree of motion was set to 0-360 degree .Motor 4 through direct coupling it was joint to motor 5.Roll motor illustrates in figure (4.6).

Figure4.6:Part of Roll(motor4)

- Motor 5 (YAW)

This motor handle the total weight of the robot including motor 5 to motor 6 and the degree of motion was set to 0-45 degree. Motor 5 through direct coupling it was joint to motor 6. YAW motor represents in figure (4.7).

Figure4.7: Part of YAW (motor5)

- Motor 6 (Drill)

This motor handle the total weight of motor 6 itself and the degree of motion was set to 0-45 degree and this degree was selected to cover painting process through spray nozzle that have also a spraining degree. Drill motor shown in figure (4.8).

Figure4.8: Part of Drill (motor 6)

In table 4.2 each motor has an initial position in degrees, the initial position, can be defined as a stage before the sensor triggers the robotic arm. Also motor initial position illustrates in figure (4.9).

Table4.2: Motors initial position

No.	Motor1	Motor2	Motor3	Motor4	Motor5	Motor6
Degree	3.4	2.3501	2.36	3.8	2.4	7

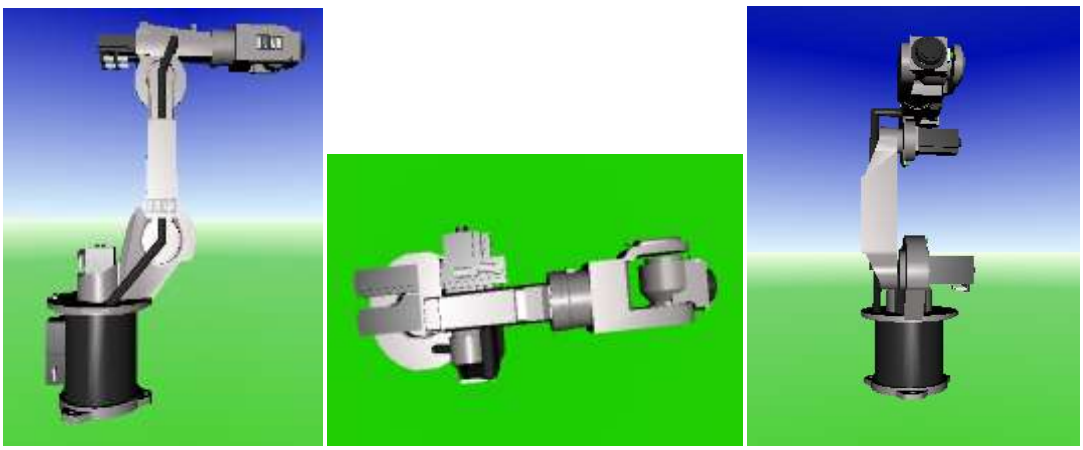


Figure 4.9 Motors initial position

In table 4.3 each motor has a minimum step depending on the motor size and its specifications so each motor has a minimum number of degree movements

Table 4.3 Increasing steps to each motor

No.	Motor1	Motor2	Motor3	Motor4	Motor5	Motor6
Forward	3.4	2.3501	2.36	3.8	2.4	7
Backward	-3.4	- 2.3501	-2.36	-3.8	-2.4	-7

3.4Graphical User Interface

The graphical user interface is a graphical representation to a command line code used to simplify the steps of movement to the end user. GUI with 3D robotic arm illustrates in figure (4.10).

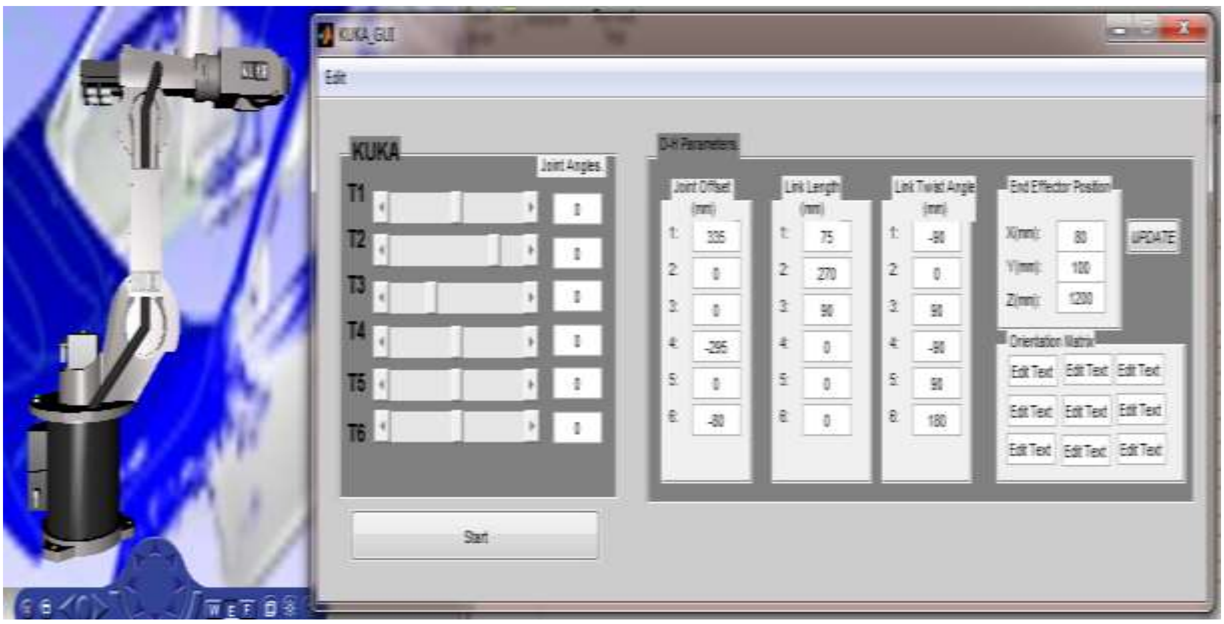


Figure 4.10: GUI with 3D robotic arm

The T1 to T6 represent the control of the motors which manually adjusted, joint offset to adjust the offset value to each motor, link length to adjust the length of the robotic arm for each part, twisting angle to set the angles of the rotation. Moreover the end effectors position is the setting and initializing of the robot position on display to for the movement.

A start bottom executes the subroutine program that loads a bunch of procedures into the program automatically without the usage of manual adjustment and it acts like an initiating button to start painting a car.

4.4 Results and Discussion

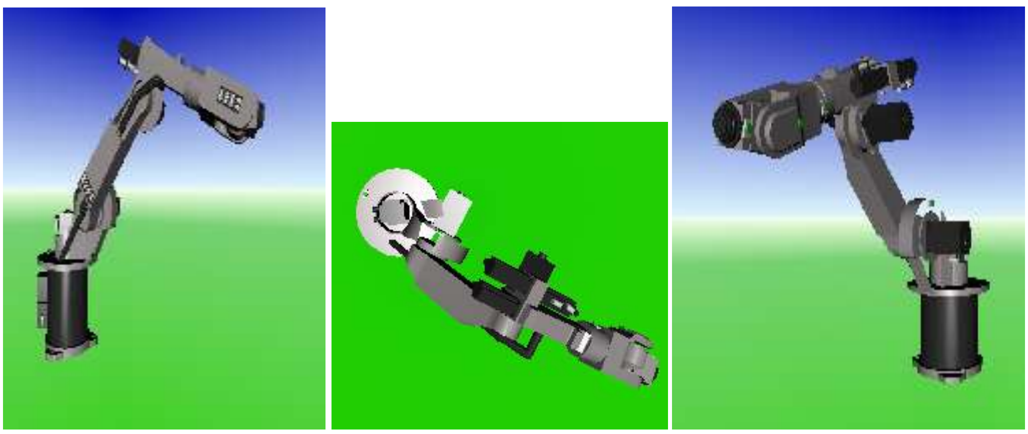
In this section three main parts of the car replaces the movement of the robot, front, middle and the end of car the robot was calibrated to move through steps starts arm initial position till the end of movement.

4.4.1Painting front of the car

Front of the car painting positions illustrated in table (4.4) and figure (4.11) through the three parts (A: front view, B: top view and C: right view). The painting of front car part appeared in figure (4.12).

Table 4.4: Painting car front

No.	Motor1	Motor2	Motor3	Motor4	Motor5	Motor6
Degree	-34	-46.9999	23.6	3.8	24/-24	7



A B C

Figure 4.11: 3D arm robotic positions in front of car painting

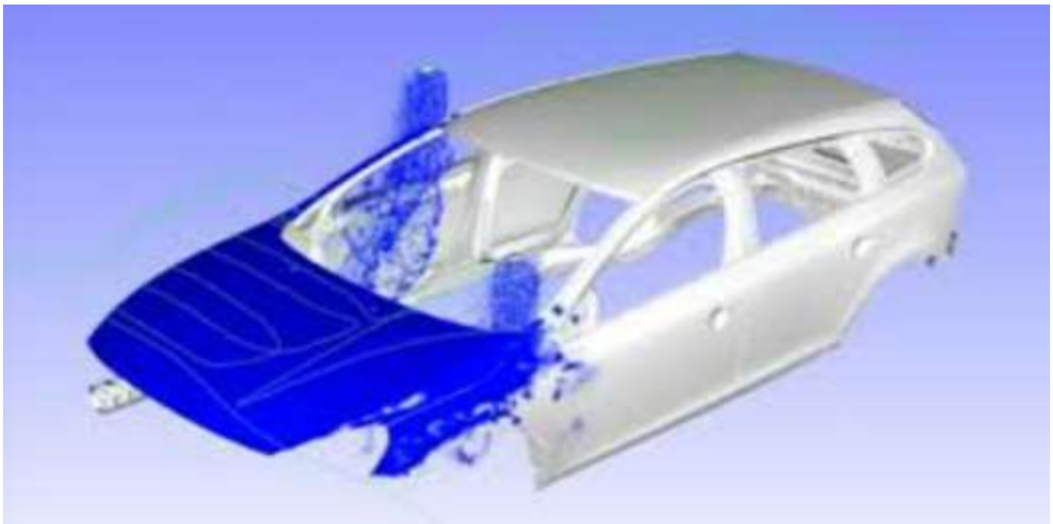


Figure4.12: painting Front of the car

4.4.2 Painting middle of the car

Middle of the car painting positions clarified in table (4.5) and figure (4.13) through the three parts (A: front view, B: top view and C: right view). Painting middle of the car part shown in figure (4.14).

Table 4.5: Painting car middle

No.	Motor1	Motor2	Motor3	Motor4	Motor5	Motor6
Degree	3.4	-21.1499	2.36	3.8	-26.4/+26.4	7

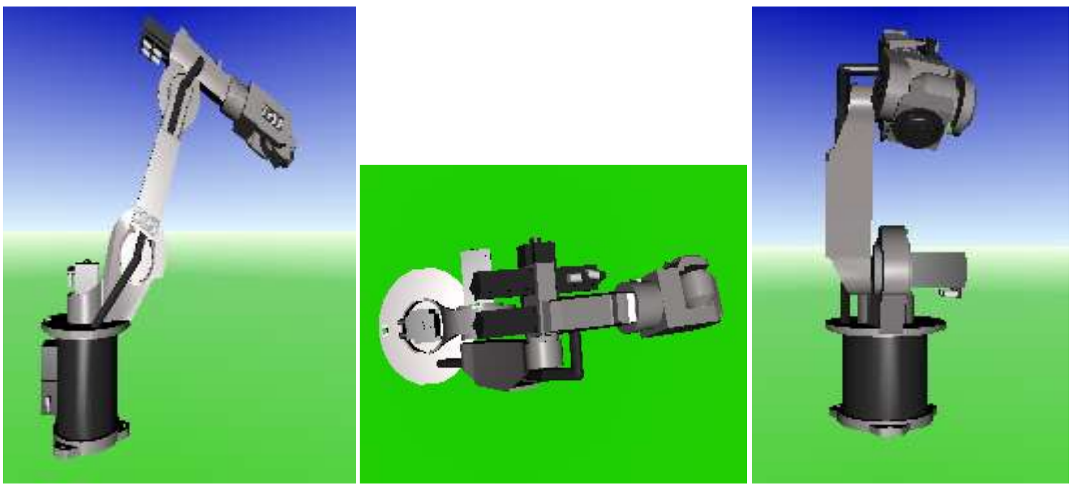


Figure 4.13:3D arm robotic positions in middle of car painting

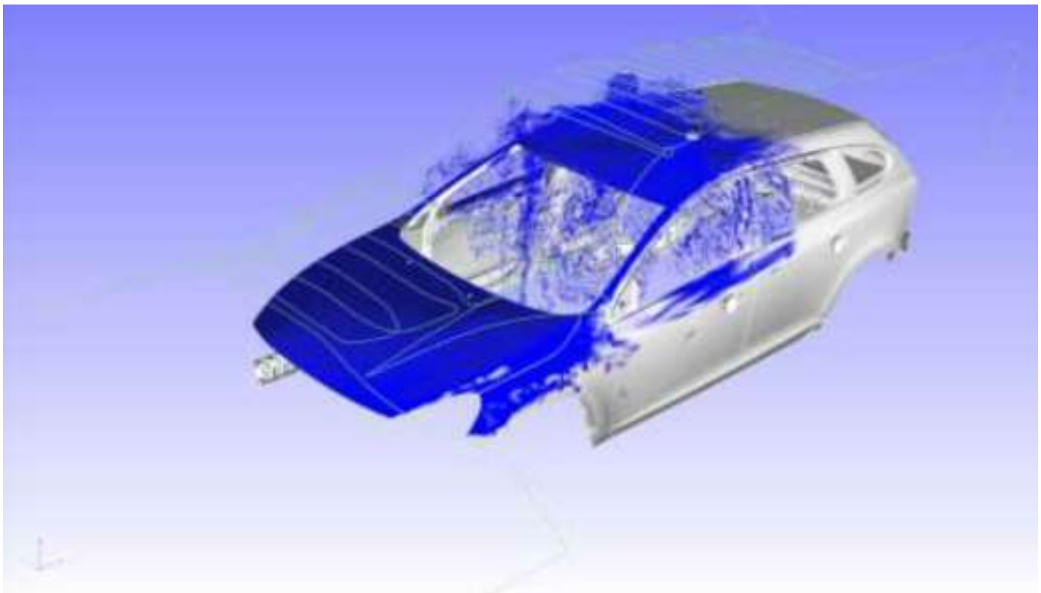


Figure4.14: painting middle of the car

4.4.3 Painting end of the car

End of the car painting positions illuminated in table (4.6) and figure (4.15) through the three parts (A: front view, B: top view and C: right view). Painting end the car part manifested in figure (4.16).

Table 4.6: Painting car end

No.	Motor1	Motor2	Motor3	Motor4	Motor5	Motor6
Degree	34	-46.9999	23.6	3.8	-36/+36	7

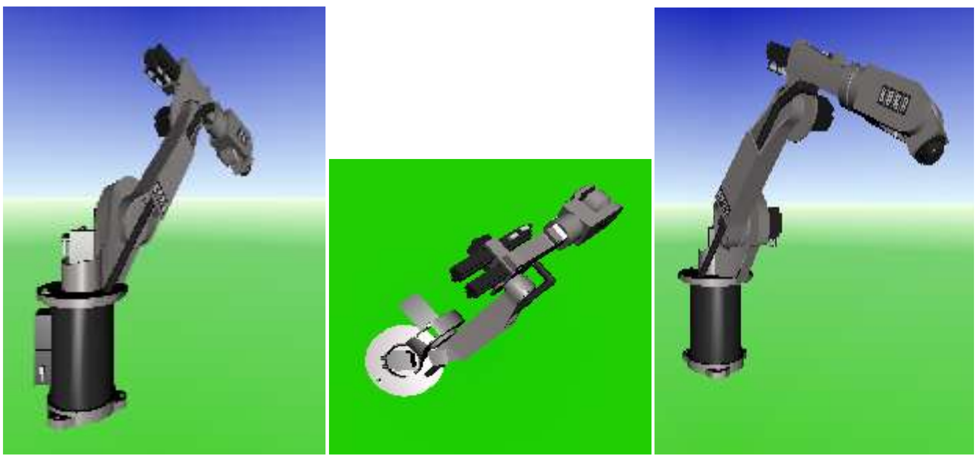


Figure 4.15: 3D arm robotic positions in end of car painting

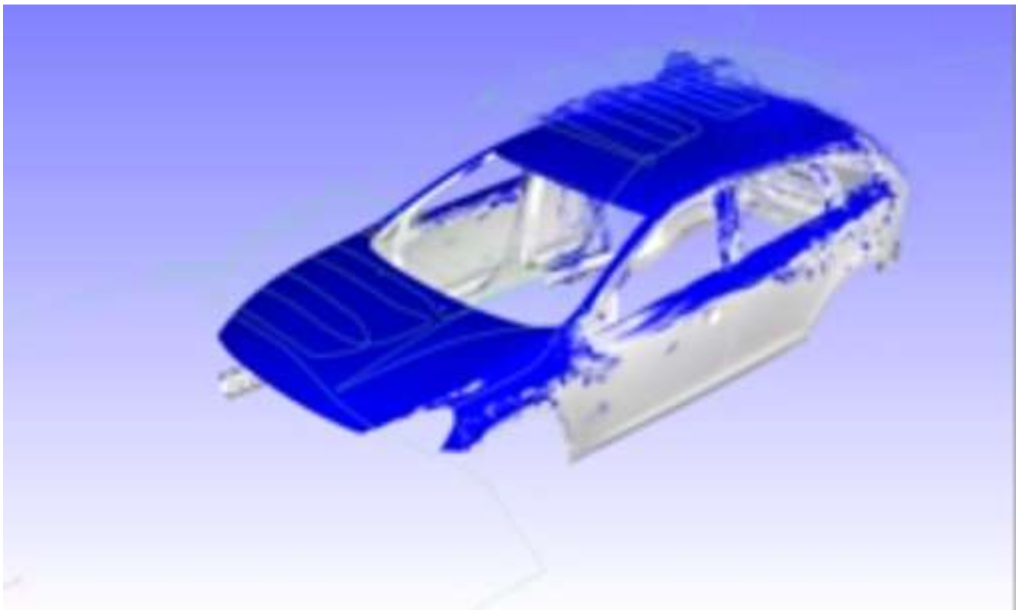


Figure4.16: painting end of the car

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

In this project the design and simulation of robotic arm was done successfully after deep investigation and analysis on 3D robotic ARM design and simulation concepts and theories for the painting of cars, COMSOL was chosen as a graphical 3D model creator that deals with the design, construction, operation, and application of robots in 3D presentation. The simulation covers different parameters to evaluate the design. Moreover simulation environment help to test these parameters without the cost effectiveness of building robots and selecting optimal materials, component, dimensions and Accuracy. Then MATLAB program used to control the object movement through MATLAB commands stored in M File linked to the graphical user interface.

The 3D robotic arm tested in MATLAB environment and it achieves project goals. The car object divided into three parts front, middle and end of the car, the robotic arm was first adjusted manual to reach the car body and the readings stored into a matrix and programmed to move the robot automatically, moreover a sensors used to detect the position of object while the robot is painting to stop the front subroutine program and execute the middle to reach the end.

5.2 Recommendations

In the end of this research, some points could be taken as a suggested future works:

- Simulate the control circuit into circuit simulator program.
- Design and implement a circuit board to control the robotic arm.
- Interface circuit to Matlab to move the robot depending on orders comes from microcontroller.
- Select materials to fabricate the robotic arm.
- Connect motor to design and interface circuit.

REFERENCES

1- Needham, Joseph . Science and Civilisation in China: Volume 2, History of Scientific Thought. Cambridge University Press. 1991 ISBN 0-521-05800-7.

2- Rosheim, Mark E. Robot Evolution: The Development of Anthrobotics. (1994). Wiley-IEEE. pp. 9–10. ISBN 0-471-02622-0.

3- Asimov, Isaac. "The Robot Chronicles". Gold. London: Voyager. (1996) [1995] pp. 224–225. ISBN 0-00-648202-3.

4- Crane, Carl D.; Joseph Duffy .Kinematic Analysis of Robot Manipulators. Cambridge University Press. ISBN 0-521-57063-8. Retrieved 2007-10-16.

5- Sandra Pauletto, Tristan Bowles, Designing the emotional content of a robotic speech signal. In: Proceedings of the 5th Audio Mostly Conference: A Conference on Interaction with Sound, New York, (2010). ISBN 978-1-4503-0046-9

6- R. Andrew Russell. Robot Tactile Sensing. New York: Prentice Hall.1990, ISBN 0-13-781592-1.

7- FRANCIS GIANG, ANAK JAPAR, Design andDevelopers Robotic Arm for Automatic Guided Conveyor, BACHELOR OF MECHANICAL ENGINEERING UNIVERSITI MALAYSIA PAHANG 2010

8- SITI HAJJAR BINTI ISHAK, Fakulti Kejuruteraan, Elektronik Dan KejuruteraanKomputer, DESIGN OF ROBOTIC ARM

CONTROLLER USING MATLAB, University Teknikal Malaysia Melaka APRIL 2011.

9- DIMENSIONAL REACH R.A.D.M.P.Ranwaka T. J. D. R. Perera, J. Adhuran, C. U. Samarakoon, R.M.T.P. Rajakaruna,MICROCONTROLLER BASED ROBOT ARM WITHTHREE, Department of Mechatronics Engineering, Faculty of Engineering, South Asian Institute of Technology and Medicine (SAITM), Sri Lanka.Email: piyumalranawaka@gmail.com

10- Antonio Bicchi and Giovanni Tonietti, Design, Realization and Control of Soft Robot Armsfor Intrinsically Safe Interaction with Humans,Centro Interdipartimentale di Ricerca "E. Piaggio" Universit`a di Pisa – Italia.

11- Lucas Ariel, Automotive Rotary-Bell Spray Painting Modeling and Simulation,Martinez, 11/6/2012.

12- Chee Kai Chua; Kah Fai Leong; Chu Sing Lim . Rapid Prototyping. 2003,World Scientific. p. 124. ISBN 978-981-238-117-0.

13- Felix Bopp. Future Business Models by Additive Manufacturing. Verlag. 4 July 2014.ISBN 3836685086. Retrieved.

14- John Iovine, PIC Microcontroller Project Book, New York Chicago San Francisco Lisbon London Madrid,2004, ISBN 0-07-143704-5

15- Wohlers Report, State of the Industry Annual Worldwide Progress Report on Additive Manufacturing, Wohlers Associates, 2009, ISBN 978-0-9754429-5-1

16- Parker, Dana T. Building Victory: Aircraft Manufacturing in the Los Angeles Area in World War II. Cypress,(2013) CA. ISBN 978-0-9897906-0-4.

17- Roe, Joseph Wickham . English and American Tool Builders. New Haven, CT: Yale University Press.(1916) LCCN 16011753.

18- Wells, David A. Recent Economic Changes and Their Effect on Production and Distribution of Wealth and Well-Being of Society. New York.(1890). D. Appleton and Co. ISBN 0-543-72474-3.

APPENDIX

MATLAB Code:

```
function varargout = KUKA_GUI(varargin)
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
               'gui_Singleton',  gui_Singleton, ...
               'gui_OpeningFcn', @KUKA_GUI_OpeningFcn, ...
               'gui_OutputFcn',  @KUKA_GUI_OutputFcn, ...
               'gui_LayoutFcn',  [] , ...
               'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
%%%      To operate this program:                              %%
%%%          1)     Make sure that KUKA.WRL, KUKA_GUI.fig,
KUKA_GUI.m     %%
%%%                 are in the appropriate path.               %%
%%%          2)     Execute KUKA_GUI.m                              %%


function KUKA_GUI_OpeningFcn(hObject, eventdata, handles,
varargin)
handles.output = hObject;
guidata(hObject, handles);
KUKA = vrworld('KUKA.wrl');
open(KUKA)
view(KUKA)

   radian=0
   KUKA.shoulder.rotation = [0, 1, 0, radian];

   KUKA.elbow.rotation = [0, 0, 1, radian];
```

```
   KUKA.pitch.rotation = [0, 0, 1, radian];

   KUKA.roll.rotation = [1, 0, 0, radian];

   KUKA.yaw.rotation = [0, 0, 1, radian];

   KUKA.drill.rotation = [1, 0, 0, radian];


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
KUKA = vrworld('KUKA.wrl');
open(KUKA)
T1=get(handles.slider1,'value');
set(handles.edit57,'String',num2str(T1));
guidata(hObject, handles);
   radian=T1*pi/180;
   q1=radian;
   KUKA.shoulder.rotation = [0, 1, 0, radian];

% --- Executes on slider movement.
function slider2_Callback(hObject, eventdata, handles)
KUKA = vrworld('KUKA.wrl');
open(KUKA)
T2=get(handles.slider2,'value');
set(handles.edit58,'String',num2str(T2));
guidata(hObject, handles);
   radian=T2*pi/180;
   q2=radian;
   KUKA.elbow.rotation = [0, 0, 1, radian];


% --- Executes on slider movement.
function slider3_Callback(hObject, eventdata, handles)
KUKA = vrworld('KUKA.wrl');
open(KUKA)
T3=get(handles.slider3,'value');
set(handles.edit59,'String',num2str(T3));
guidata(hObject, handles);
   radian=T3*pi/180;
   q3=radian;
```

```
   KUKA.pitch.rotation = [0, 0, 1, radian];



% --- Executes on slider movement.
function slider4_Callback(hObject, eventdata, handles)
KUKA = vrworld('KUKA.wrl');
open(KUKA)
T4=-get(handles.slider4,'value');
set(handles.edit60,'String',num2str(T4));
guidata(hObject, handles);
   radian=T4*pi/180;
   q4=radian;
   KUKA.roll.rotation = [1, 0, 0, radian];

% --- Executes on slider movement.
function slider5_Callback(hObject, eventdata, handles)
KUKA = vrworld('KUKA.wrl');
open(KUKA)
T5=-get(handles.slider5,'value');
   set(handles.edit61,'String',num2str(T5));
   guidata(hObject, handles);
   radian=T5*pi/180;
   q5=radian;
   KUKA.yaw.rotation = [0, 0, 1, radian];


   % --- Executes on slider movement.
function slider6_Callback(hObject, eventdata, handles)
KUKA = vrworld('KUKA.wrl');
open(KUKA)
T6=-get(handles.slider6,'value');
set(handles.edit62,'String',num2str(T6));
guidata(hObject, handles);
   radian=T6*pi/180;
   q6=radian;
   KUKA.drill.rotation = [1, 0, 0, radian];


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
```

```
% % % GIVEN JOINT VARIABLES

d=[335   0   0   -295    0   -80];
a=[75    270 90  0   0   0];

% % TRANSFORMATION MATRIX 1.
q1=get(handles.slider1,'Value');
T1=[    cos(q1)     0       -sin(q1) a(1)*cos(q1)
        sin(q1)     0       -cos(q1) a(1)*sin(q1)
        0           -1          0       d(1)
        0           0           0        1      ];

% % TRANSFORMATION MATRIX 2.
q2=get(handles.slider2,'Value');
T2=[    cos(q2)     -sin(q2)    0   a(2)*cos(q2)
        sin(q2)     cos(q2)     0   a(2)*sin(q2)
        0               0       1       0
        0               0       0       1   ];


% % TRANSFORMATION MATRIX 3.
q3=get(handles.slider3,'Value');
T3=[    cos(q3)         0    sin(q3)   a(3)*cos(q3)
        sin(q3)         0    -cos(q3)   a(3)*sin(q3)
        0               0       0       0
        0               0       0       1   ];
% % TRANSFORMATION MATRIX 4.
q4=get(handles.slider4,'Value');
T4=[    cos(q4)         0   -sin(q4)    0
        sin(q4)         0    cos(q4)    0
        0              -1       0      d(4)
        0               0       0       1   ];


% % TRANSFORMATION MATRIX 5.
q5=get(handles.slider5,'Value');
T5=[    cos(q5)     0       sin(q5)         0
        sin(q5)     0       -cos(q5)         0
        0           1          0             0
        0           0          0             1       ];
```

```
% % TRANSFORMATION MATRIX 6.
q6=get(handles.slider6,'Value');
T6=[    cos(q6)     -sin(q6)        0           0
        sin(q6)     cos(q6)     0               0
        0           0           1           0
        0           0           0               1   ];



A=T1*T2*T3*T4*T5*T6;

set(handles.edit42,'String',num2str(A(13)));
set(handles.edit43,'String',num2str(A(14)));
set(handles.edit44,'String',num2str(A(15)));
set(handles.edit48,'String',num2str(A(1)));
set(handles.edit49,'String',num2str(A(5)));
set(handles.edit50,'String',num2str(A(9)));
set(handles.edit51,'String',num2str(A(2)));
set(handles.edit52,'String',num2str(A(6)));
set(handles.edit53,'String',num2str(A(10)));
set(handles.edit54,'String',num2str(A(3)));
set(handles.edit55,'String',num2str(A(7)));
set(handles.edit56,'String',num2str(A(11)));
guidata(hObject, handles);


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

KUKA = vrworld('KUKA.wrl');
open(KUKA)
T1=-34;
set(handles.edit57,'String',num2str(T1));
guidata(hObject, handles);
   radian=T1*pi/180;
   q1=radian;
   KUKA.shoulder.rotation = [0, 1, 0, radian];
```

```
   KUKA = vrworld('KUKA.wrl');
open(KUKA)
T2=-46.999;
set(handles.edit58,'String',num2str(T2));
guidata(hObject, handles);
   radian=T2*pi/180;
   q1=radian;
   KUKA.elbow.rotation = [0, 1, 0, radian];


   KUKA = vrworld('KUKA.wrl');
open(KUKA)
T3=23.6;
set(handles.edit59,'String',num2str(T3));
guidata(hObject, handles);
   radian=T3*pi/180;
   q1=radian;
   KUKA.pitch.rotation = [0, 1, 0, radian];


   KUKA = vrworld('KUKA.wrl');
open(KUKA)
T4=3.8;
set(handles.edit60,'String',num2str(T4));
guidata(hObject, handles);
   radian=T4*pi/180;
   q1=radian;
   KUKA.roll.rotation = [0, 1, 0, radian];



      KUKA = vrworld('KUKA.wrl');
open(KUKA)
T5=2.4;
set(handles.edit61,'String',num2str(T5));
guidata(hObject, handles);
   radian=T5*pi/180;
   q1=radian;
   KUKA.yaw.rotation = [0, 1, 0, radian];
```

```
   KUKA = vrworld('KUKA.wrl');
open(KUKA)
T6=7;
set(handles.edit62,'String',num2str(T6));
guidata(hObject, handles);
   radian=T6*pi/180;
   q1=radian;
   KUKA.drill.rotation = [0, 1, 0, radian];
```