XEROBOT

PROJECT REPORT

Submitted by

MIHIR SHYAMSUNDAR DESHINGKAR	**111107017**
PANKAJ RAGHUNATH GABALE	**111107020**
RAHUL SURESH KHEDKAR	**111107032**

In partial fulfillment for the award of the degree of

B.Tech Electronics and Telecommunication Engineering

Under the guidance of

Prof. Mrs. Vidya N. More

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION

COLLEGE OF ENGINEERING, PUNE-411105

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION

COLLEGE OF ENGINEERING, PUNE

CERTIFICATE

Certified that the project, titled 'XEROBOT', has been successfully completed and executed by the following students

MIHIR SHYAMSUNDAR DESHINGKAR	**111107017**
PANKAJ RAGHUNATH GABALE	**111107020**
RAHUL SURESH KHEDKAR	**111107032**

And is approved for the partial fulfilment of the requirements for the degree of "B.Tech" in "Electronics and Telecommunication Engineering"

Prof. Mrs. Vidya N. More	**Dr. Ashok M. Sapkal**
Project guide	Head
Department of E & TC,	Department of E & TC,
College of Engineering, Pune	College of Engineering,Pune
Shivajinagar, Pune	Shivajinagar,Pune
411005.	411005.

This project report entitled

XEROBOT

By

MIHIR SHYAMSUNDAR DESHINGKAR	**111107017**
PANKAJ RAGHUNATH GABALE	**111107020**
RAHUL SURESH KHEDKAR	**111107032**

is approved for the degree of

Bachelor of Technology

Of

Department of Electronics and Telecommunication

College of Engineering Pune

Examiners	**Name**	**Signature**
1. External Examiner	________________________	
2. Internal Examiner	________________________	
3. Supervisor (s)	________________________	
	________________________	

Date:

Place:

ACKNOWLEDGEENT

It gives us immense pleasure to present our work on the topic of “XEROBOT.” This project has been carried out at the department of Electronics and telecommunication at College Of Engineering, Pune. We would like to thank our project guide **Prof.Mrs. Vidya N. More for** constant help, inspiration and encouragement. We would also like to thank our honorable head of department **Dr. A.M. Sapkal** for his valuable suggestions and support.

We would also like to convey our sincerest gratitude and indebtedness to all other faculty members and staff of the department , who bestowed their great effort and guidance at appropriate times without which it would have been very difficult on our project work .

Our thanks and appreciations also go to our friends in developing the project and people we are willingly helped us out with their abilities.

MIHIR SHYAMSUNDAR DESHINGKAR	**111107017**
PANKAJ RAGHUNATH GABALE	**111107020**
RAHUL SURESH KHEDKAR	**111107032**

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

(Name of the student)

(MIS No.)

Date:

Place:

Abstract

In today's growing industrial world the things are involving more newer and powerful technology equipment. In this world of growing technology, every aspect of human life is involving robotics and automation in it. New companies are also focusing on making the product which will help the human life easier and will involve the less human efforts. Robotics is entering into the household purposes in this new world. Robotics is right now an inseparable part of the industries and will get similar expansion with household use in coming years. The robotics available right is involving the preprogrammed aspects but now the evolving technologies are involving self-programmable or also called autonomous operation. However the autonomous operation doesn't involve the actual copying of the things. *XEROBOT* is the new concept which will change the human life to the next level. *XEROBOT* is a human action copying robot, it performs in the same way as taught by mentor or the operator. While *XEROBOT* can be a next step ahead in the world it's uses cannot be limited at certain extent. This report gives the glimpse of the prototype of the *XEROBOT* and its operation.

Table of Contents

- # Chapter 7

OBSERVATIONS AND RESULTS

Table of Figures

• Chapter 1

INTRODUCTION

1.1 Concept of Xerobot

The objective of this project is to make a task learning and self-programmable device, which will be helpful in industrial as well as military applications. For multitasking of the robot there are multiple ways to make them use for various activities either programming them or by attaching them with different hardware.

There are several categories of robots depending upon their usages like industrial robot, household robot, Military robots etc. These robots may need continuous monitoring and special programming depending upon their usage. While in industries, they invest lots of money on these robots to carry out different tasks. The cost for one robot will be in the range of $1000 to $100,000. [1]

Though spending this much amount of money, robots are bounded for one task only. Even in military purposes, the robots are attached for a particular task. Some approaches have been made in the direction of the multitasking robot, but that needs intensive task specific programming. Even if this task is achieved, these robots costs above $1mn, which is subsequently not serving the purposes of small scale business. [1]

So the current requirements in the field of robotics are:

1) The robot should be self-programmable
2) The robot should be cost efficient
3) The robot should have capability of multitasking

XEROBOT includes all these requirements, it is self-programmable, cheap in cost and multitasking. Though it may use many sensors, it will not cost more than $2000. It will be storing the activities done by the host or operator and can perform the same whenever asked to.

The main features of XEROBOT systems are:

1) Utilizes the minimal possible hardware
2) Human action tracking and decodes that
3) Higher processing power with ability to switch
4) Usable for small scale business or within range

1.2 Literature Survey

XEROBOT will have two modes:

1) Learning mode
2) Execution mode

Learning mode has following subparts:

1.2.1 Motion detection

The motion detection is the important feature of this robot. For this a 3D camera Microsoft Kinect is used. It is a camera with in-built projection of an IR frame. It detects the human skeleton and its joints including the depth of different levels. It uses the IR camera to analyze the depth of the body or the distance of the body from a point. It also have RGB camera to detect color and saturation. [2]

1.2.3 Decoding Software

The main objective of the prototype is decoding human actions from the data coming from the camera. Processing is the software tool used for this type of real time image calculations. It include image and video processing IDE which is generally used for any cameras. [3]

1.2.4 Wireless communication

The motions decoded by the decoding software will be transmitted to the robot through wireless communication. We are using the X-Bee Transceiver module for this. This module has the range between 5 meters of the spherical distance around it. [13]

1.2.5 Robotic Hardware

This bot will be a prototype bot for the XEROBOT. It includes the robotic arm with flexible motions on it. It will have 4 degrees of freedom. The robotic arm will be mounted on a car. It will operate with the commands provided from the controller through wireless communication. [8]

1.2.6 Auto-adjustment

As the bot will be autonomously working after the initial setting has been done. It needs to get the feedback about the actual movement of itself. It can be calibrated using the Gyroscope and Accelerometers. It is provided as a feedback control system for the bot to make it auto adjustable and for fine handling.

1.2.7 Execution mode

In the execution mode the robotic arm actually performs the task taught to the robotic hand. It also involves the actual change of position of the bot and taking it to the destination of the object given and will accordingly drive the motor.

• Chapter 2

Microcontroller Configuration

2.1 Controller overview

XEROBOT uses Arduino microcontroller as a main unit to monitor and analyze the data. Arduino is a single-board microcontroller, intended to make the application of interactive objects or environments more accessible. The hardware consists of an open-source hardware board designed around an 8-bit Atmel AVR microcontroller, or a 32-bit Atmel ARM. Current models features a USB interface, 6 analog input pins, as well as 14 digital I/O pins which allows the user to attach various extension boards.

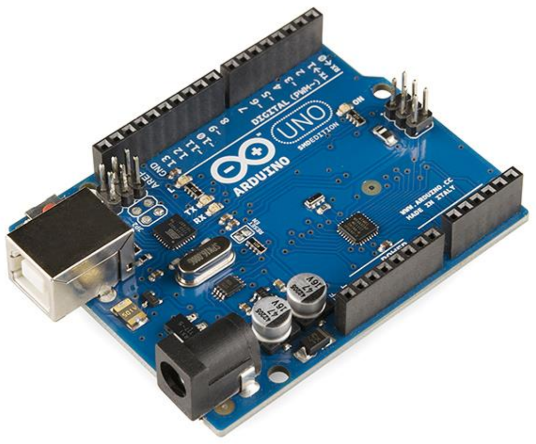


Figure 2.1 (Arduino UNO)

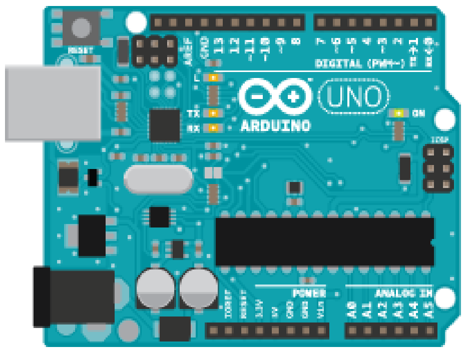


Figure 2.2 (UNO PCB layout)

2.2 Programming

Arduino's microcontroller is also pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory, compared with other devices that typically need an external programmer. This makes using an Arduino more straightforward by allowing the use of an ordinary computer as the programmer.

At a conceptual level, when using the Arduino software stack, all boards are programmed over an RS-232 serial connection, but the way this is implemented varies by hardware version. Serial Arduino boards contain a level shifter circuit to convert between RS-232-level and TTL-level signals.

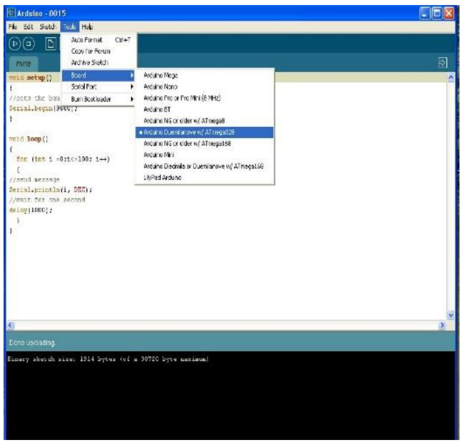
Current Arduino boards are programmed via USB, implemented using USB-to-serial adapter chips such as the FTDI FT232. Some variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, Bluetooth or other methods. (When used with traditional microcontroller tools instead of the Arduino IDE, standard AVR ISP programming is used.)

2.3 Software

The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and is derived from the IDE for the Processing programming language and the Wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a "sketch".

Arduino programs are written in C or C++. The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output operations much easier. Users only need define two functions to make a runnable cyclic executive program:

- setup(): a function run once at the start of a program that can initialize settings
- loop(): a function called repeatedly until the board powers off


Arduino - 0015
File Edit Sketch Tools Help
Auto Format Ctrl+T
Copy for Forum
Archive Sketch
Board
Serial Port
Burn Bootloader
mine
Arduino Mega
Arduino Nano
Arduino Pro or Pro Mini (8 MHz)
Arduino BT
Arduino NG or older w/ ATmega8
• Arduino Duemilanove w/ ATmega328
Arduino NG or older w/ ATmega168
Arduino Mini
Arduino Diecimila or Duemilanove w/ ATmega168
LilyPad Arduino

```
void setup()
{
//sets the bau
Serial.begin(9600);
}

void loop()
{
  for (int i =0;i<=100; i++)
  {
//send message
Serial.println(i, DEC);
//wait for one second
delay(1000);
  }
}
```

```
Done uploading.
Binary sketch size: 1914 bytes (of a 30720 byte maximum)
```

Figure 2.3 Arduino IDE

It is a feature of most Arduino boards that they have an LED and load resistor connected between pin 13 and ground; a convenient feature for many simple tests. The previous code would not be seen by a standard C++ compiler as a valid program, so when the user clicks the "Upload to I/O board" button in the IDE, a copy of the code is written to a temporary file with an extra include header at the top and a very simple main () function at the bottom, to make it a valid C++ program.

2.4 Alternatives evaluated

2.4.1 Raspberry Pi

The Raspberry Pi is on the opposite end – it features a fully-fledged operating system loaded on an SD card. It also has audio out, HDMI and RCA video output and an Ethernet port. This allows you to use your Raspberry Pi as a computer, complete with internet browsing, games and more. Plug in your keyboard, mouse and monitor, and you have an amazingly cheap computer.

The Raspberry Pi projects are more software based than hardware based. As it is simply a Linux computer, most projects are based around software hacks, media centers and graphics/sound and multimedia. It can however do some simple hardware control with the GPIO pins.

2.4.2 Beaglebone

The Beaglebone is the perhaps the least known of these platforms, but an incredibly capable board worthy of consideration for many projects. It is a powerful Linux computer that fits inside an Altoid’s mint container. However, the Beaglebone is similar of that kind of the raspberry pi but it is more expensive and not widely used. One has to do lots of presetting’s while using the Beaglebone.

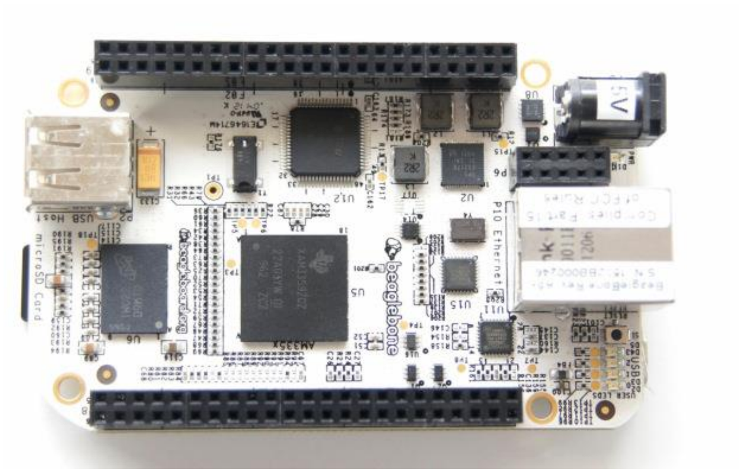


Fig. 2.4 Beaglebone Board

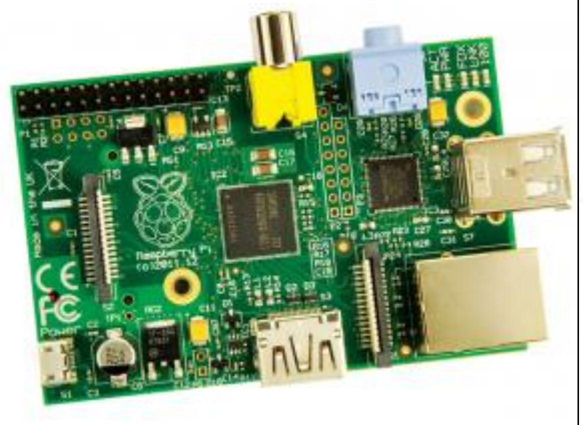
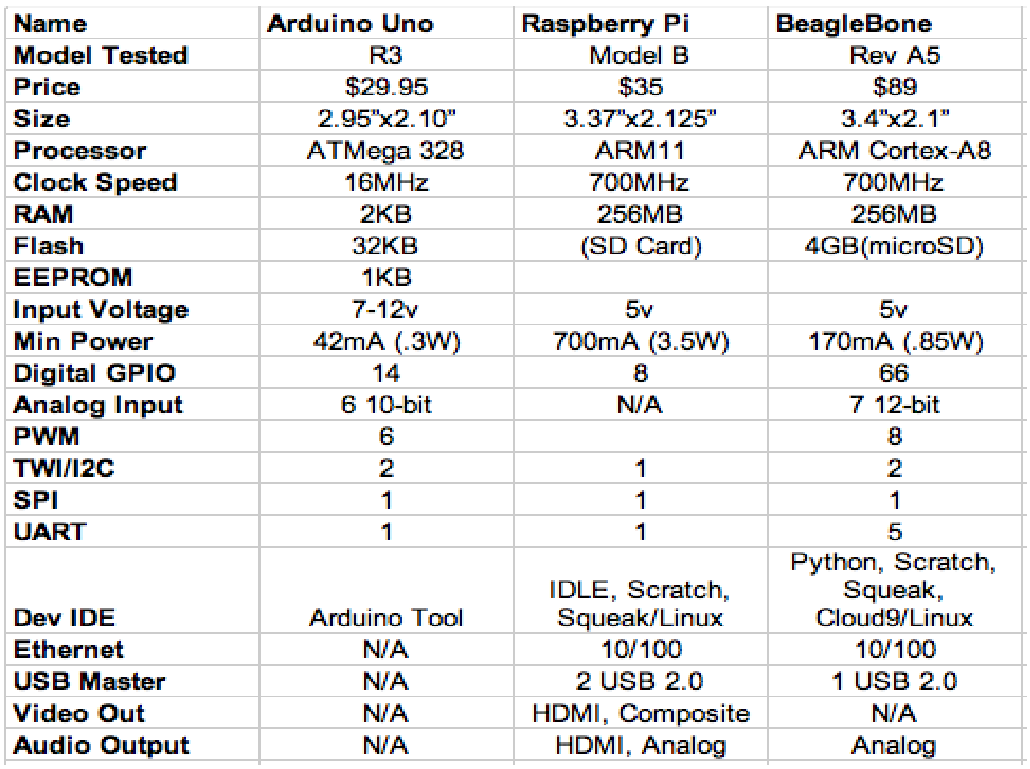


Fig.2.5 Raspberry pi board



Name	Arduino Uno	Raspberry Pi	BeagleBone
Model Tested	R3	Model B	Rev A5
Price	$29.95	$35	$89
Size	2.95"x2.10"	3.37"x2.125"	3.4"x2.1"
Processor	ATMega 328	ARM11	ARM Cortex-A8
Clock Speed	16MHz	700MHz	700MHz
RAM	2KB	256MB	256MB
Flash	32KB	(SD Card)	4GB(microSD)
EEPROM	1KB		
Input Voltage	7-12v	5v	5v
Min Power	42mA (.3W)	700mA (3.5W)	170mA (.85W)
Digital GPIO	14	8	66
Analog Input	6 10-bit	N/A	7 12-bit
PWM	6		8
TWI/I2C	2	1	2
SPI	1	1	1
UART	1	1	5
Dev IDE	Arduino Tool	IDLE, Scratch, Squeak/Linux	Python, Scratch, Squeak, Cloud9/Linux
Ethernet	N/A	10/100	10/100
USB Master	N/A	2 USB 2.0	1 USB 2.0
Video Out	N/A	HDMI, Composite	N/A
Audio Output	N/A	HDMI, Analog	Analog

Table 2.1 Comparison between UNO, R-pi and Beaglebone

2.5 Design criterion:

1. Easy handling
2. Large PWM availability
3. Portability
4. Low power source required
5. Easy coding or programming

2.6 Final Selection results

I. Arduino has inbuilt ADC which provides lots of flexibility while designing the circuit for the analog sensors.

II. It comes with 6 PWM's which gives easy hand over SPO2 sensor and ECG.

III. It is a controller not processor (unlike Beaglebone and raspberry pi) so it need not to be programed every time. When the code is burned in to the controller same code remains in that controller and continues to evaluate it after again power supply is provided

IV. It uses C and C++ language hence can be easily programmed and debugged.

V. It has many libraries already defined inbuilt so that it should provide flexibility to programmers while developing the system.

VI. It is low in cost so it also decreases the product cost and in the way it is also smaller in size than other controllers hence it also provides small size of the product

VII. Arduino has many shields already provided with it such as Ethernet shield and LCD display which can easily configured with Arduino.

VIII. It can be programmed with USB or TTL cables and do not require HDMI cable.

IX. Ethernet shield and other shield doesn’t require other power supply and have very little power requirement.

• Chapter 3

MICROSOFT KINECT

3.1. Introduction to Kinect

Recent advances in 3D depth cameras such as Microsoft Kinect sensors have created many opportunities for multimedia computing. Kinect was built to revolutionize the way people play games and how they experience entertainment. With Kinect, people are able to interact with the games with their body in a natural way. The key enabling technology is human body language understanding; the computer must first understand what a user is doing before it can respond. This has always been an active research field in computer vision, but it has proven formidably difficult with video cameras.

Kinect's impact has extended far beyond the gaming industry. With its wide availability and low cost, many researchers and practitioners in computer science, electronic engineering, and robotics are leveraging the sensing technology to develop creative new ways to interact with machines and to perform other tasks, from helping children with autism to assisting doctors in operating rooms. Microsoft released the Kinect Software Development Kit (SDK) for Windows which will undoubtedly amplify the Kinect Effect. The SDK will potentially transform human-computer interaction in multiple industries—education, healthcare, retail, transportation, and beyond. [2]

Figure 3.1 Kinect camera

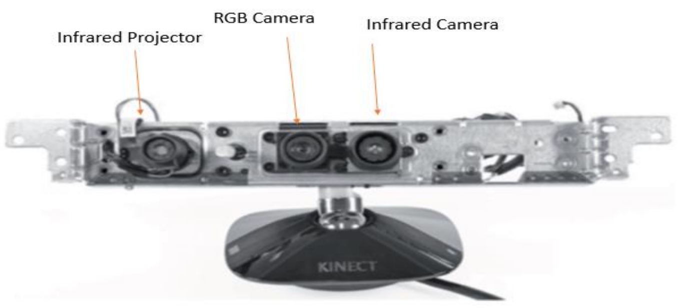


Figure 3.2 internal structure of Kinect

3.2 Kinect Sensor

The Kinect sensor incorporates several advanced sensing hardware. Most notably, it contains a depth sensor, a color camera, and a four-microphone array that provide full-body 3D motion capture, facial recognition, and voice recognition capabilities. Infrared (IR) projector, the color camera, and the IR camera. The depth sensor consists of the IR projector combined with the IR camera, which is a monochrome complementary metal oxide semiconductor (CMOS) sensor [3].The depth-sensing technology is licensed from the Israeli company Prime Sense although the exact technology is not disclosed, it is based on the structured light principle. The IR projector is an IR laser that passes through a diffraction grating and turns into a Set of IR dots. Figure below shows the IR dots seen by the IR camera.

The relative geometry between the IR projector and the IR camera as well as the projected IR dot pattern are known. If we can match a dot observed in an image with a dot in the projector pattern, we can reconstruct it in 3D using triangulation. Because the dot pattern is relatively random, the matching between the IR image and the projector pattern can be done in a straightforward way by comparing small neighborhoods using, for example, normalized cross correlation.

Figure 3.3 shows the depth map produced by the Kinect sensor for the IR image in Figure 2. The depth value is encoded with gray values; the darker a pixel, the closer the point is to the camera in space. The black pixels indicate that no depth values are available for those pixels. This might happen if the points are too far (and the depth values cannot be computed accurately), are too close

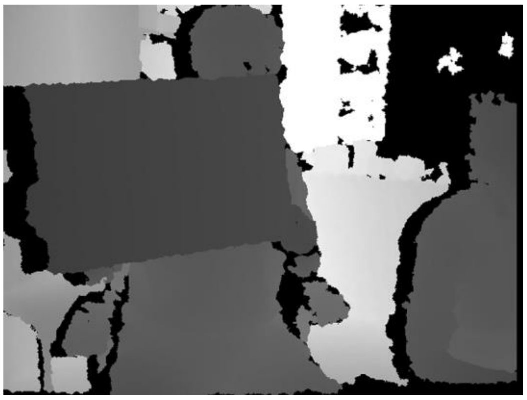


Figure 3.3 Depth image of Kinect sensor

(There is a blind region due to limited fields of view for the projector and the camera), are in the cast shadow of the projector (there are no IR dots), or reflect poor IR lights (such as hairs or specular surfaces)

The depth values produced by the Kinect sensor are sometimes inaccurate because the calibration between the IR projector and the IR camera becomes invalid. This could be caused by heat or vibration during transportation or a drift in the IR laser. To address this problem, together with the Kinect team, a recalibration technique is shipped with the Kinect sensor. If users find that the Kinect is not responding accurately to their actions, they can recalibrate the Kinect sensor by showing it the card. The depth value produced by the Kinect sensor is assumed to be an affine transformation of the true depth value—that Z measured Z = αZ+ β which we found to be a reasonably good model. The goal of recalibration is to determine α and β. (We could also use a more complex distortion model that applies the same technique.) Using the RGB camera, the recalibration technique determines the 3D coordinates of the feature points on the calibration card in the RGB camera's coordinate systems which are considered to be the true values. At the same time, the Kinect sensor also produces the measured 3D coordinates of those feature points in the IR camera's coordinate system. Minimizing the distances between the two point sets, the Kinect sensor can estimate the values of a and b and the rigid transformation between the RGB camera and the IR camera. [3]

3.3 Skeleton Tracking

Skeletal tracking must ideally work for every person on the planet, in every household, without any calibration. A dauntingly high number of dimensions describe this envelope, such as the distance from the Kinect sensor and the sensor tilt angle. Entire sets of dimensions are necessary to describe unique individuals, including size, shape, hair, clothing, motions, and poses. In skeletal tracking, a human body is represented by a number of joints representing body parts such as head, neck, shoulders, and arms each joint is represented by its 3D coordinates. The goal is to determine all the 3D parameters of these joints in real time to allow fluent interactivity. Rather than trying to determine directly the body pose in this high-dimensional space, through Kinect it is able to display the things per pixel. [4]

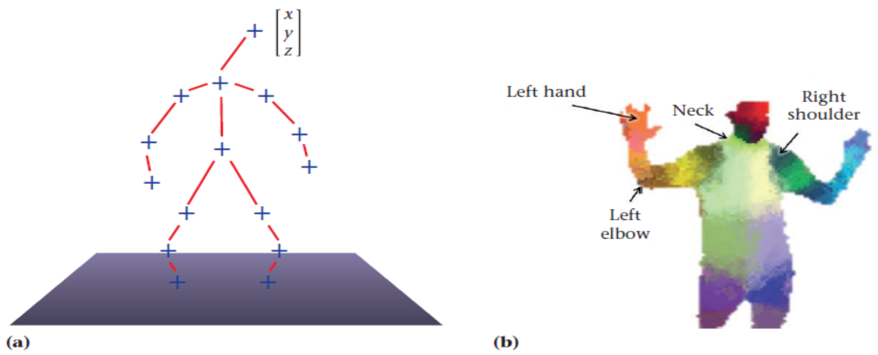


Figure 5. Skeletal tracking. (a) Using a skeletal representation of various body parts, (b) Kinect uses per-pixel, body-part recognition as an intermediate step to avoid a combinatorial search over the different body joints.

Figure 3.4 skeleton defining by Kinect sensor

The segmentation of a depth image as a per-pixel classification task (no pairwise terms or conditional random fields are necessary). Evaluating each pixel separately avoids a combinatorial search over the different body joints. For training data, we generate realistic synthetic depth images of humans of many shapes and sizes in highly varied poses sampled from a large motion-capture database. We train a deep randomized decision forest classifier, which avoids over fitting by using hundreds of thousands of training images. Simple, discriminative depth comparison image features yield 3D translation invariance while maintaining high computational efficiency. [4]

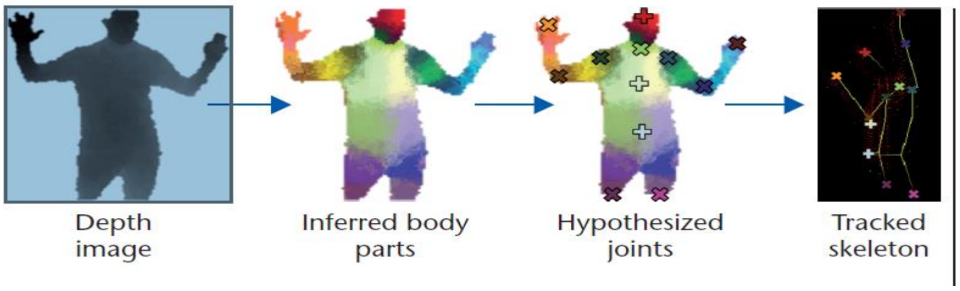


Figure 3.5 tracking the human skeleton

For further speedup, the classifier can be run in parallel on each pixel on a graphics processing unit (GPU). Finally, spatial modes of the inferred per-pixel distributions are computed using mean shift resulting in the 3D joint proposals. An optimized implementation of our algorithm runs in under 5 ms per frame. It works frame by frame across dramatically differing body shapes and sizes, and the learned discriminative approach naturally handles self-occlusions and poses cropped by the image frame. The first step is to perform per-pixel, body-part classification. The second step is to hypothesize the body joints by finding a global centroid of probability mass (local modes of density) through mean shift. The final stage is to map hypothesized joints to the skeletal joints and fit a skeleton by considering both temporal continuity and prior knowledge from skeletal train data.

3.4 Three main hardware parts of the Kinect

- **Color VGA video camera -** This video camera aids in facial recognition and other detection features by detecting three color components: red, green and blue. Microsoft calls this an "RGB camera" referring to the color components it detects.

- **Depth sensor -** An infrared projector and a monochrome CMOS (complimentary metal-oxide semiconductor) sensor work together to "see" the room in 3-D regardless of the lighting conditions.

• **Multi-array microphone** - This is an array of four microphones that can isolate the voices of the players from the noise in the room. This allows the player to be a few feet away from the microphone and still use voice controls.

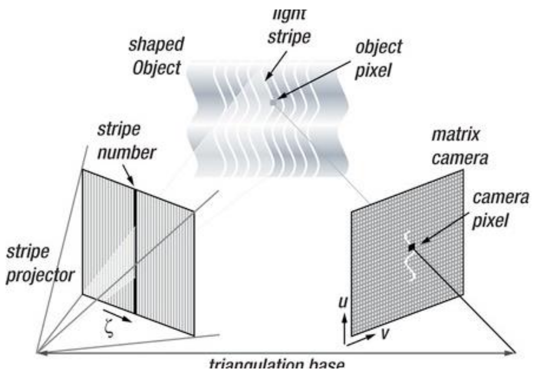


Figure 3.6 structured light scanning

3.5 Structured-Light 3D Scanning

Most structured-light scanners are based on the projection of a narrow strip of light onto a 3D object, using the deformation of the stripe when seen from a point of view different from the source to measure the distance from each point to the camera and thus reconstitute the 3D volume. This method can be extended to the projection of many stripes of light at the same time, which provides a high number of samples simultaneously.

3.6 Microsoft Kinect for Windows or Drivers

In order to access the Kinect data streams, it is necessary need to install the necessary drivers on your computer. Microsoft announced the release of the official Microsoft Kinect SDK for non-commercial use. This SDK offered the programmer access to all the Kinect sensor capabilities plus hand/skeleton tracking. At the time of writing, Kinect for Windows SDK includes the following:

•Drivers for using Kinect sensor devices on a computer running Windows 7 or Windows 8 developer preview (desktop apps only)

•APIs and device interfaces, along with technical documentation

3.7 OpenNI

OpenNI breaks the dependency between the sensor and the middleware, the API enables middleware developers to develop algorithms on top of raw data formats, independent of the sensor device that is producing the data. In the same way, sensor manufacturers can build sensors that will work with any OpenNI-compliant application.

• Chapter 4

PROCESSING

4.1 Introduction to processing

Processing IDE much resembles Arduino’s. In fact, the Arduino IDE was based on the Processing IDE Processing is an open source programming language and environment for people who want to create images, animations, and interactions. Initially developed to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context, Processing also has evolved into a tool for generating finished professional work.

Today, there are tens of thousands of students, artists, designers, researchers, and hobbyists who use Processing for learning, prototyping, and production. Processing is based in Java, one of the most widespread programming languages available today. Java is an object-oriented, multi-platform programming language. The code written in Java is compiled into something called byte code that is then executed by the Java Virtual Machine living in your computer. This allows the programmer to write software without having to worry about the operating system (OS) it will be run on. This is a huge advantage when one is trying to write software to be used in different machines. Processing programs are called sketches because Processing was first designed as a tool for designers to quickly code ideas that would later be developed in other programming languages. As the project developed, though, Processing expanded its capabilities, and it is now used for many complex projects as well as a sketching tool.

4.2 Processing IDE

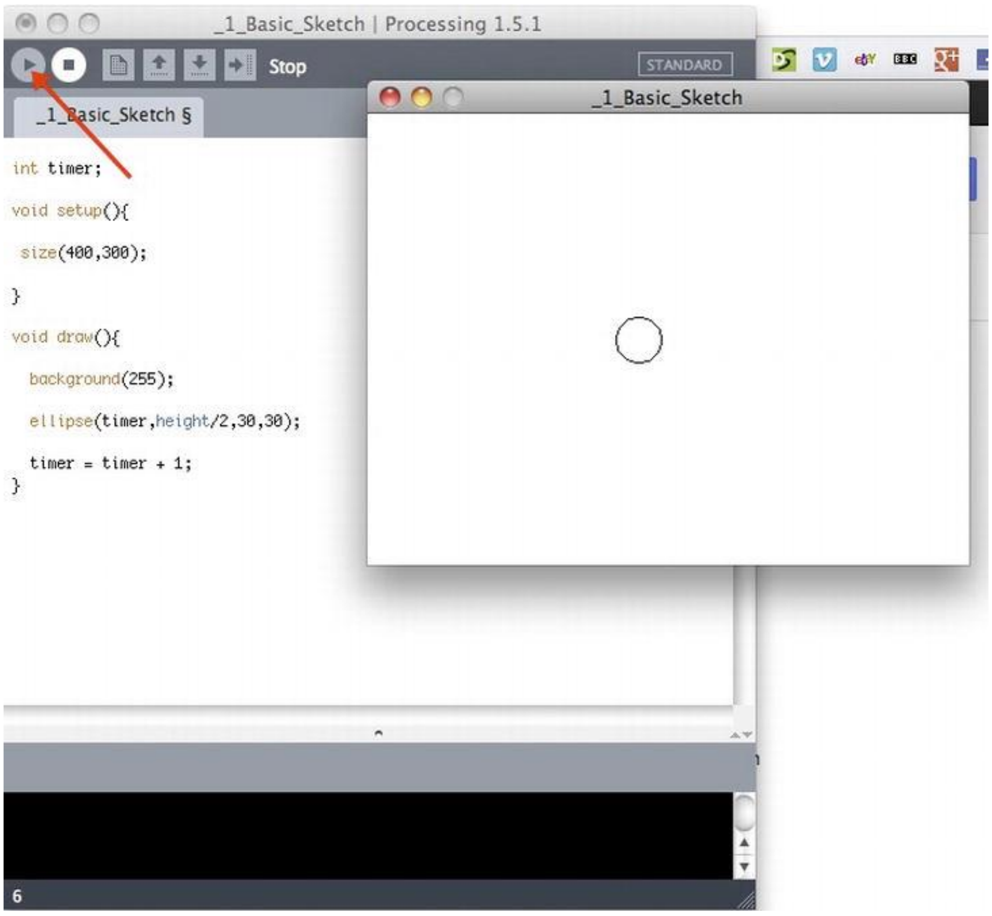
As mentioned, the Processing IDE is very similar to the Arduino IDE. There is a console at the bottom of the Processing IDE; a message area that will warn you about errors in the code and at runtime; a text editor where you will write your code; plus a tab manager and a toolbar at the top of the IDE. There is another button called Export PApplet; pressing it will convert the Processing code into Java code and then compile it as a Java Applet, which can be embedded in a web browser. This is the way you’ll find Processing Applets, or PApplets, displayed on the Internet. [5]

4.3 Processing Variables

Integers will be used to store positive and negative whole numbers (this means numbers without a decimal point, like 42, 0, and -56. An integer variable can store values ranging from 2,147,483,648 to 2,147,483,647 (inclusive). If you need to store larger numbers, you need to define your variable as a long. Whenever we need more precision, we will be using floating-point numbers, or floats, which are numbers with a decimal place (23.12, 0.567, and 234.63). The data type double works the same way as float, but it is more precise. In general, Processing encourages the use of floats over doubles because of the savings in memory and computational time [5]

Some important functions:

- Setup () function: This function is called once in the lifetime of a program (unless you call it explicitly from another function)
- Draw () function: a draw () function that will run as a loop until the program is terminated by the user.



```
int timer;
void setup(){
 size(400,300);
}
void draw(){
  background(255);
  ellipse(timer,height/2,30,30);
  timer = timer + 1;
}
```

Figure 4.1 Processing IDE output printing an ellipse

4.4 Accessing the Depth Map and RGB Image

First, import the Simple-OpenNI library, and declare the variable Kinect that will contain the Simple-OpenNI object, like so:

```
Import SimpleOpenNI.*;

SimpleOpenNI Kinect;
```

Kinect has a standard RGB camera and an infrared camera on board, used in combination with an infrared projector to generate the depth image for 3D scanning. Enable the depth map and RGB images in your Kinect object and the mirroring capability that will mirror the Kinect data so it's easier for you to relate to your image on screen

4.5 Kinect Space

We have already accessed Kinect's depth map in the "Accessing the Depth Map and RGB Image" section and wondered if you can get that map translated back into the real-scale 3D coordinates of the objects. The following sketch does that for you. It starts pretty much like the depth map example but includes a function that shows the real coordinates of the points. First, import the appropriate libraries, declare Simple-OpenNI and KinectOrbit objects, and initialize them in the setup () function, enabling the depth map only. [7]

```
importprocessing.opengl.*;

import SimpleOpenNI.*;

importKinectOrbit.*;
```

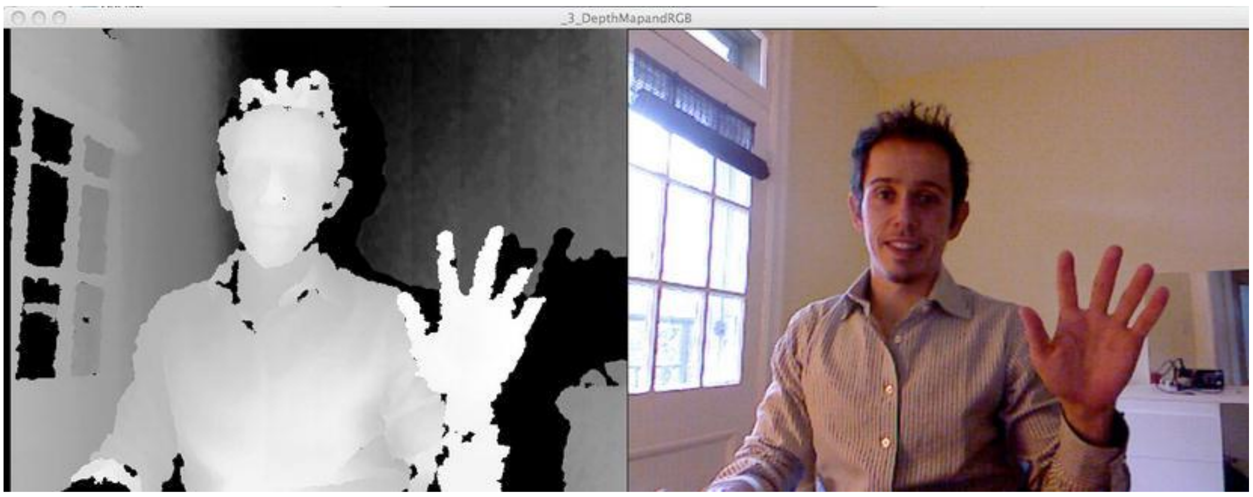


Figure 4.2 Depth map of RGB image in processing

4.6 Skeleton Tracking

Skeleton tracking is probably the most impressive of NITE’s, and therefore of Kinect’s, capabilities. This framework allows the computer to understand a person’s body position in 3D and to have a quite accurate idea of where the person’s joints stand in space at every point in time. This is quite an overwhelming feature if one thinks that not so long ago he needed to buy a daunting amount of material including several cameras, special suits, and so on in order to acquire the same data. The only downside of skeleton tracking (for certain projects) is the need for the user to stand in a certain “start pose” to allow the middleware to detect the user’s limbs and start the tracking. This inconvenience seems to have been fixed in the current version of the official Microsoft SDK and will likely disappear from NITE, but for the time being, you’ll have to make do with it.

• Chapter 5

ALGORITHMS

5.1 Overview

This chapter includes the actual operation of the Kinect and the processing software. The main objective of the project has been satisfied in here. In the working of the Kinect sensor the depth calculations have been given and their corresponding calibrations are also provided. The main task after detecting the skeleton will be the task of performing of the bot and the actions generated to it. Deciding the software was the main part in handling of the process of action detection as well as object detection.

Human skeleton can be detected using SimpleOpenNI library. This library enables to detect different joints of the human body. Pixel coordinates of required joints can be obtained using this library. Getting the different joints helps in actual observation of the movements of each part in terms of Angle and Distance. The observed distance will be used in actual movements of the motor. [8]

Different sections of software are -

Mode detection

Mode 0: Default or gesture detection mode

Mode 1: Task learning mode

Mode 2: Task execution mode

Before learning these modes some of the things which need to be known are as follows

5.2 Mode selection

5.2.1 Skeleton detection

Skeleton detection is required as we need to detect the left and hand joints and their positions, which are used in task learning. Using the coordinates of the detected joints, we also detected various gestures based on which the program decides the mode of operation.

5.2.2 Gesture detection

There is a unique gesture for each of the above mode. Initially, the program is in Mode 0. In this mode, gesture is continuously detected till a gesture for either Mode 1 or Mode 2 is detected. When a particular gesture is detected, it goes in the mode corresponding to that gesture. When the program is in Mode 1, it stays there till a stop gesture is performed, after which it goes to default mode, i.e. Mode 0. There is no gesture which will detect the end of the execution mode. The program stays in execution mode till it receives an acknowledgement from Arduino indicating that the task execution is complete. After that acknowledgement, the program goes back to Mode 0.

5.3 Object detection

Object detection is an important task for the whole process to be carried out. Object detection is done by the Kinect using both his RGB and IR camera. Object detection includes the calculation of the dimensions of the object including the co-ordinates of the object. Making these calculations will help in sorting the things like exact calibrations of the object. In this prototype feature the robotic arm is considered to copy the human actions copying. It will be good if the usage of both the things of the Kinect camera are used.

Object detection uses two methods depending upon the types:

1) **Purely color based**

This object detection method was not so reliable due to the limitations of resolution of Kinect. Also if the object is picked in different angle, the object will not be detected. Hence we discarded this algorithm. In color based object detection, the scanning area is same as explained in above method. We executed the codes for red, blue and green object detection. It was found that the hand was detected as green and hence we could not detect green objects. Both red and blue object detections gave good results. So we decided to use red object detection. The algorithm for color object detection is presented below.

2) **Dimension based**

In dimension and color based object detection, we mainly detected the object using its length, breadth and color of the object. In this we scanned a rectangle of 60*60 pixels with the right hand palm point at the center of the rectangle. A cluster of those points which were within the range of ±20cm was detected as the object and its maximum length and maximum depth was detected in real dimensions. As the dimensions are real dimensions and not pixel values, the user can stand at any distance from the Kinect. We performed object learning by storing down the dimensions and the maximum color present in the object. Parameters of different objects were stored in an array.

In object detection, the parameters of the detected object would be scanned from that array the corresponding object would be recognized. If the object was not found, a message would be displayed. The depths of the corresponding objects can be obtained by using the IR camera.

5.4 Object tracking algorithm

Now as the object is detected by the camera the main task is to keep track of that object and follow the same till the end of the process to occur and save it in its memory. This algorithm detects one of the blue, green or red objects according to the parameters set in the object detection program. Only the object which is in the vicinity of the right hand of the user is detected. In this experimentation the object is taken as blue because of the color properties of the Kinect. Kinect can distinguish blue color more specifically and distinctively.

5.4.1 Algorithm for object detection and tracking:

1. Track the palm of the right hand of the user
2. Scan a rectangular window of size 60*60 with the palm point at the center of the rectangle
3. If the depth of the pixel is within range of ±20cm of the palm depth, and if the pixel is blue, then Increment pixel count and add that pixel to object.
4. After the window is scanned completely, calculate the centroid of the object
5. Calculate the real world x, y and z (depth) parameters of the object in each frame and monitor them. Store the parameters when the object is first detected in their respective arrays, viz., xarr [], yarr [], zarr[]
6. If the change in any one parameter is greater than some threshold, then store that point, and modify the start point.
7. Continuously monitor the left hand gesture
8. Repeat steps 6 and 7 until the gesture for stopping the learning mode is detected
9. If gesture for stopping of learning mode is detected, go to mode 0 which continuously checks the gesture

5.4.2 Flow Diagram for the algorithm:

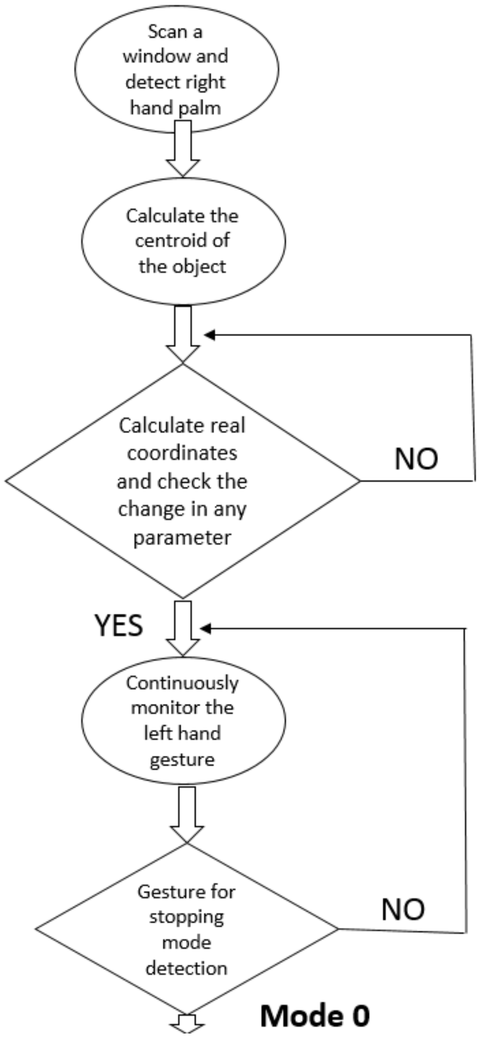

Scan a window and detect right hand palm
Calculate the centroid of the object
Calculate real coordinates and check the change in any parameter
NO
YES
Continuously monitor the left hand gesture
Gesture for stopping mode detection
NO
Mode 0

Figure 5.1 flow diagram object tracking

5.5 Execution mode

After determining the actual or real co-ordinates of the object the next job is to send that values to the hardware or bot. The wireless mechanism X-Bee is used for it. The main task is to send the values from one software processing to another Arduino IDE. This is major task to send information from one block to other block and to make it work on a hardware. The following algorithm gives the brief on how the data is sent from processing to Arduino IDE.

5.5.1 Algorithm in processing IDE

1. Calculate the difference between x, y and z parameters of current and next position of the object from xarr[], yarr[], zarr[]

2. Send the difference to Arduino

3. Repeat steps 1 and 2 until the three arrays are completely traversed

4. Go to mode 0

5.5.2 Flow diagram for processing algorithm

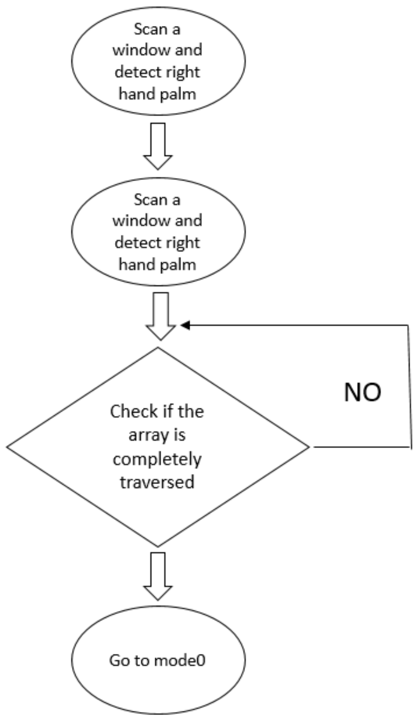

Scan a window and detect right hand palm
Scan a window and detect right hand palm
Check if the array is completely traversed
NO
Go to mode0

Fig. 14 Flow diagram for processing to IDE

The second task after sending the values to the Arduino will be, giving that values to the hardware. The Arduino software from the Arduino IDE will send according to the motor pins defined. As I2C expander circuit is used PC 8574, it gives the values to the I2C port expander.

5.5.3 Algorithm in Arduino IDE

1. Pick up the object

2. Read the values sent by processing

3. Calibrate the robot motors as per the received values and move the object

4. Repeat steps 2 and 3 until all values are received from processing

5.5.4 Flow diagram for Arduino algorithm

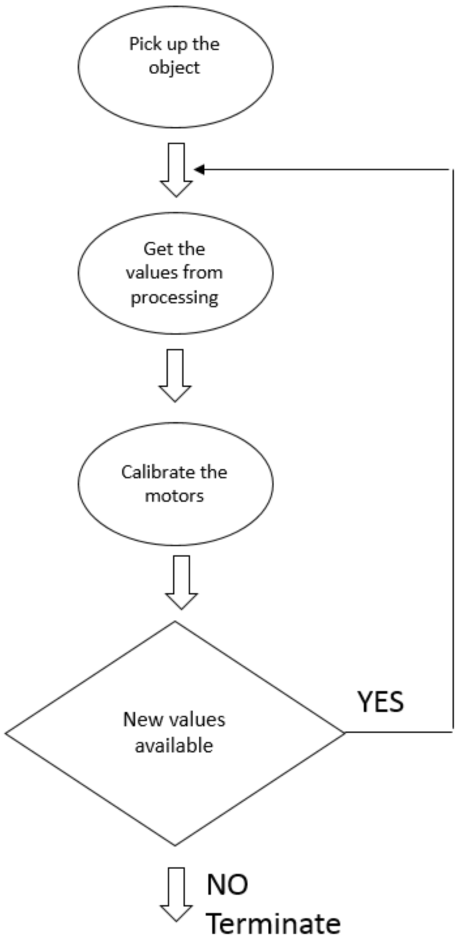


Figure 5.3 Flow chart for Arduino algorithm

The whole procedure is executed in the defined manner. No any step can be executed in between of the others. The whole System is executing the flow in which the procedure is carried out.

• Chapter 6

HARDWARE IMPLEMENTATION

After having the look at all the software elements of the XEROBOT, the actual execution depends on the Hardware involved with it. It involves the 4 main parts in the hardware including the electronic circuitry. The following are the parts included in the hardware description of the bot.

1) Robotic Arm
2) Circuits and IC's
3) Wireless Module X-bee
4) Mechanical Hardware

6.1 Robotic Arm

Robotic Arm is the main executor of the whole project. It has 5 DC motors inbuilt within it. It involves more of the mechanical work to make that out hence it is preferred to have the whole robotic assemble purchased. The following image is of the robotic arm used in this project. The 5 motors on it are defined like a parts of our hand to which it will copy. Gripper is also provided with it to have picking up of the objects. These motors are operated on the Motor driver IC L293d. [9]

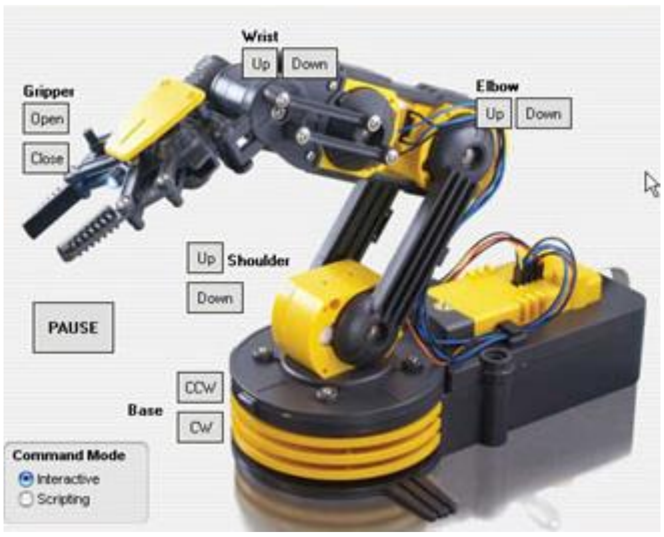


Figure 6.1 Robotic arm

6.1.1 These are the specifications of the robotic arm:

- Weight: 658 gms
- Lifting Capacity: 100 g

- 9.0" L x 6.3" W x 15.0" H
- Power Source: 4 D Cell Batteries (not included)
- Maximum Vertical Reach: 15"
- Maximum Horizontal Reach: 12.6"
- Wrist Motion Range: 120°
- Elbow Motion Range: 300°
- Base Motion (shoulder) Range: 180°
- Base Rotation Range: 270°

Depending upon the task allotted to each motor the rotations of the motors are made. These are the applications of the each motor part and its positions. [9]

6.1.2 Arm elements and their applications

1) **Gripper**: Gripper is provided to the robotic arm for its picking up of the object and it can pick up the objects with the length up to 10cm of the range. And can lift up to the weight of 100g as specified earlier.

2**) Base motor**: It is used for the round rotation of the robotic arm. It have the turning of 270°. The base motor is of same torque provided here but for good performances it should have high torque capability

3) **Elbow motor**: Elbow motor is similar hand elbow for the robotic arm and it acts as stretcher for the elbow. It comes to action while folding and unfolding of the arm stretch.

4) **Shoulder motor**: Shoulder motor acts like as and vertical movement for the robotic arm. It stretches the shoulder in order to move it up and down.

5) **Wrist motor**: Wrist motor is a handler for the gripper. It is the last motion required for the arm. When the object is need to be picked up the wrist motor is used to turn gripper.

The sophisticated and easy working of the robotic arm makes it compatible to work on the platforms like different controllers. However for the motion is need to be controlled over here and can only be controlled by using motor drivers and potentiometer is needed for the computation of actual movement of the robotic arm and its resetting. Potentiometers are attached with motors and feedback again to the Arduino to reset the motor position. The 10k potentiometers are used for the 4 motors viz. Base, elbow, wrist and base motor.

6.2 Wireless X-bee module

6.2.1 Overview of X-bee technology

In the prototype built the communication between main controller and the hardware needs to be done. As the things are quite large in terms of distance one needs to think upon the parameter that will affect the hardware's mobility. So for this prototype there is need of the wireless communication between the two sections of the prototype. X-bee is the new wireless technology which is efficient as well as economical for the controller. Its higher data rate capability gives it wide range of applications. It operates at 2.4GHz and have wide area coverage.

These modules use the IEEE 802.15.4 networking protocol for fast point-to-multipoint or peer-to-peer networking. They are designed for high-throughput applications requiring low latency and predictable communication timing." So basically X-bee is Digi's own Zig bee based protocol. The X-bee radios can all be used with minimum number of connections – power (3.3V), data in and data out (UART), Reset and Sleep [10]. Devices that have a UART interface can connect directly to the pins of the RF module as shown in the figure below.

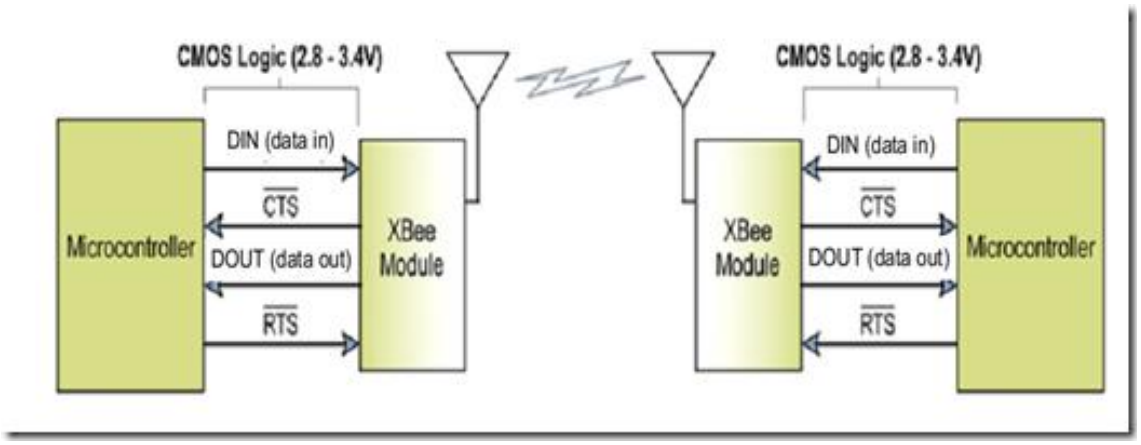


Figure 6.2 X-bee transmission and reception flow

API (Application Programming Interface) Operation is an alternative to the default Transparent Operation. The frame-based API extends the level to which a host application can interact with the networking capabilities of the module. When in API mode, all data entering and leaving the module is contained in frames that define operations or events within the module [11].

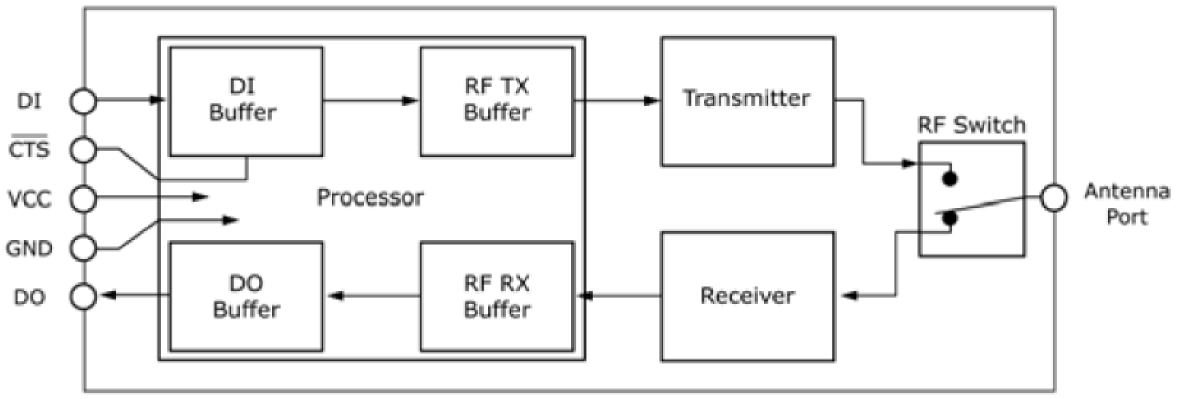


Figure 6.3 X-bee internal flow diagram

X-bee is attached with the controller similarly in both the transmitter and receiver sections. Both the control actions and signals are provided with the controller inputs and acknowledgements. The interfacing of the X-bee module Gen2 is given below

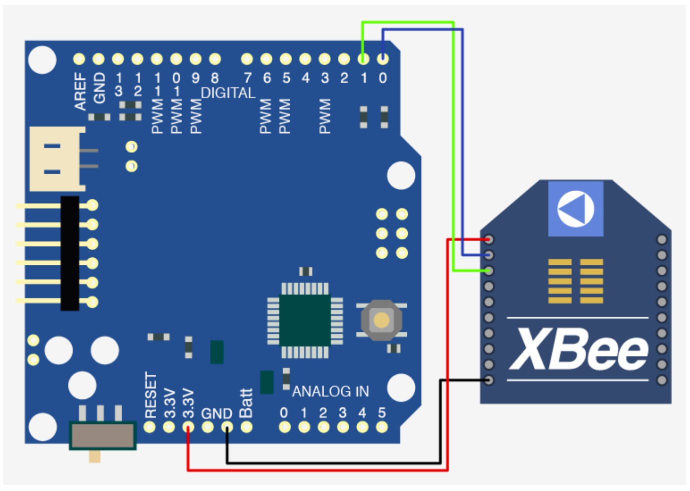
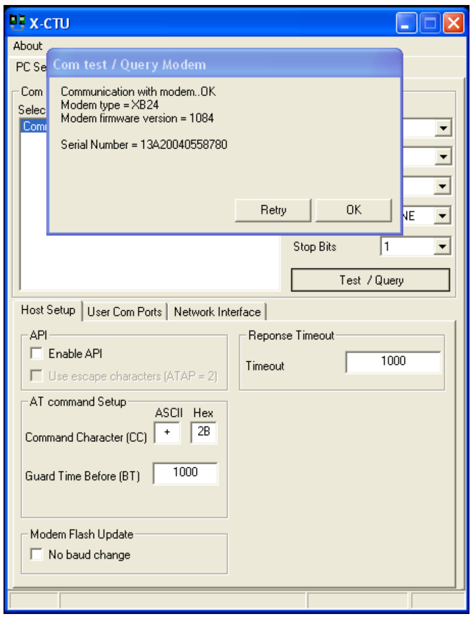


Figure 6.4 X-bee interfacing

6.2.2 X-CTU software

The X-CTU software is free to download and provides a simple interface to configure and update your X-bee transceivers. With this software firmware updates are a breeze and configuration is simple. It allows us to transmit data from the keyboard. It also has a feature of sending packet of data at a time instead of single characters. X-CTU is a terminal network for the operation of the X-bee module [12].



```
Communication with modem..OK
Modem type = XB24
Modem firmware version = 1084

Serial Number = 13A20040558780
```

Stop Bits: 1

Timeout: 1000

Command Character (CC): + 2B

Guard Time Before (BT): 1000

Figure 6.5 X-CTU terminals

After Opening the X-CTU one has to configure the X-bee address and its operation mode. In this prototype the device is used in its API mode. So by selecting the X-bee address and configuring the device the results will be obtained. The Results include the check bits and the Data transmitted by TX. Or Data to be sent.

6.3 Mechanical Hardware

6.3.1 Chassis Material

As the main base of the robot will be carrying the weight of around 1.5Kg, it is needed to have the strong chassis with flat surface and easy to drill and other applications with strong bond for the adhesives. The material is semi-fiber plastic also known as Acrylic and of the dimensions defined in the image shown below. The chassis structure is made in such a way that the arm should have no limitation with the movement of the shoulder and elbow.

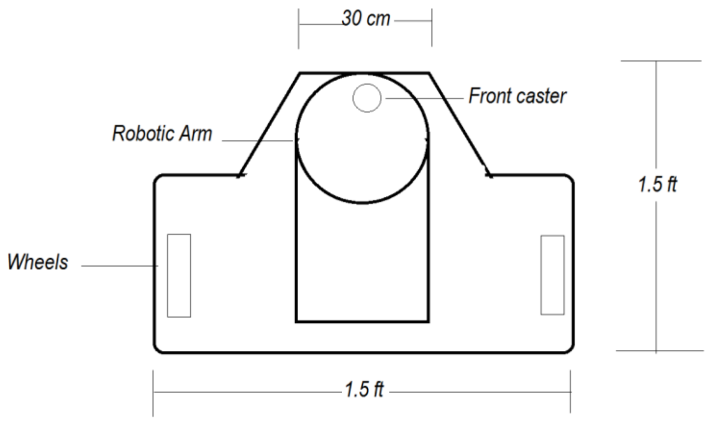


Figure 6.6 Structure of the base chassis

6.3.2 Wheels and Caster

Wheels and caster are provided for the movement of the chassis from one position to another. Wheels are driven by the DC motor with torque of around 1Kg. Two side motors are used. There is caster provided in front to save the space and to avoid other motors which are unnecessary. The caster provides the free movement of the front part allowing the omnidirectional movement of the bot.

Figure 6.7 Wheel *figure 6.8 Caster*

6.3.3 Motors

DC motors are used for driving the wheels. It involves the Electromagnetic coils to rotate the shaft. The motors used here are of the simple DC motors, they are installed with wheels as well as with robotic arm elements. The toy motors are used in the arm which requires the power supply of range 9-12 V and current up to 0.5A. The Wheel motor requires the same voltage levels but current rating is higher as they drive high torque.

Figure 6.9 Toy motor used Arm

6.4 Electronic Design

The electronic design includes the circuitry of the prototype project and have the schematic design of all the elements like controller, power supply and other IC's like drivers and port expanders.

The whole circuitry is divided into 2 different parts:

1) Motor divers circuits
2) Wireless module and controller section with power supply

6.4.1 Schematic Diagrams

The complete schematic of the project is as shown in the figure below

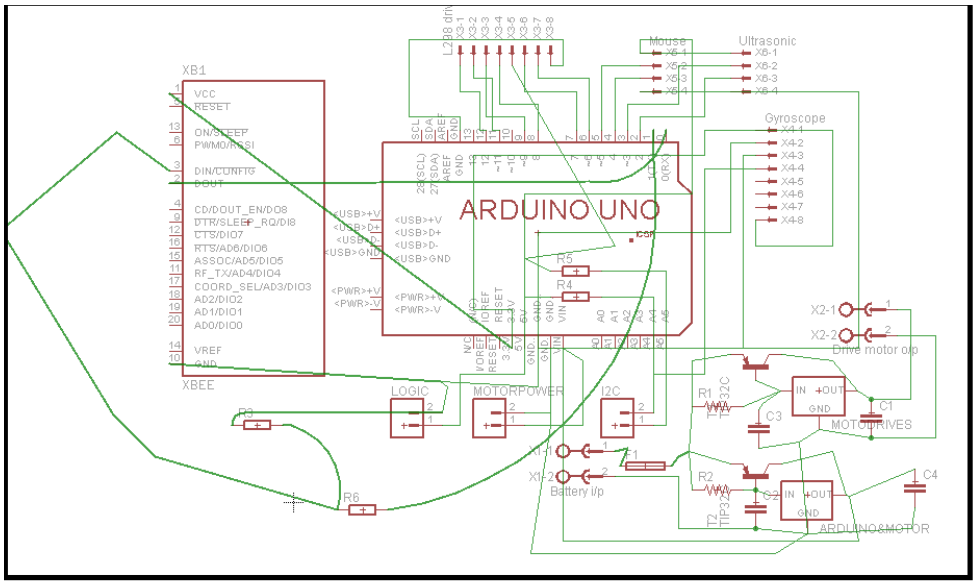


Figure 6.10 Schematic of controller, X-bee and power supply

This Schematic includes the Arduino and its interfacing with motor networks. It includes the reception from X-bee module and sends the data to the motor drivers. It also includes the readings from the potentiometer and again resends the information back to motors to have the motors at their reset position. The motor driving circuit and potentiometer circuit is give below.

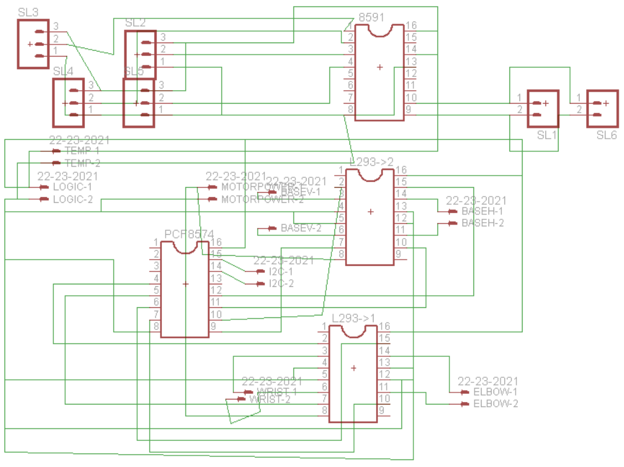


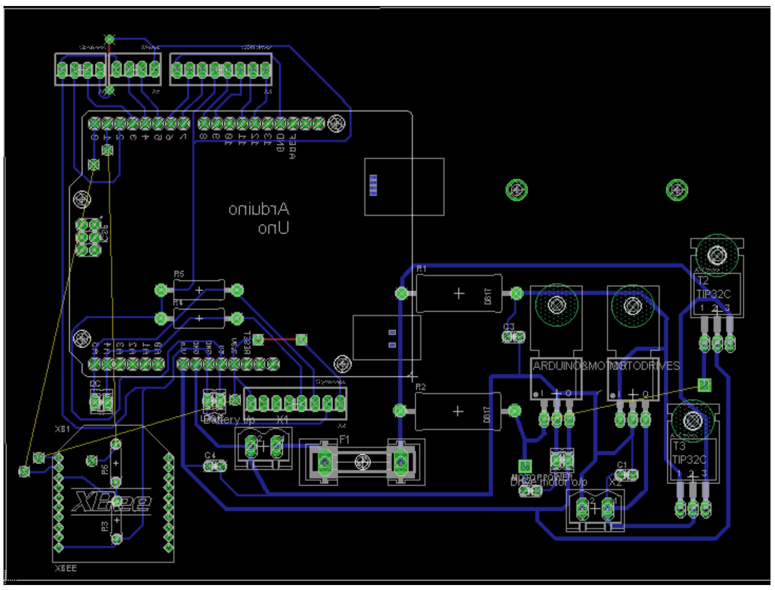
Figure 6.11 Schematic of motor driver circuit

In the above circuit IC PC8574 is used as I2C port expander and is addresed by the controller. It will be declared by Arduiuno by an address and it will also declare the I/O's of the devices connected to it. I2C allows configuring 2 master and 2 slave devices.

6.4.2 PCB layout

There are 2 PCB's installed in the prototype

1) Arduino and Wireless controller PCB (*figure 6.12*)



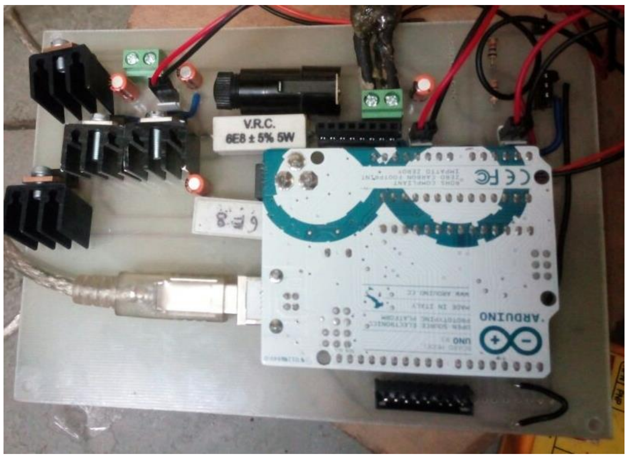


Figure 6.13 Actual assembly of the PCB

2) Motor Driver’s PCB

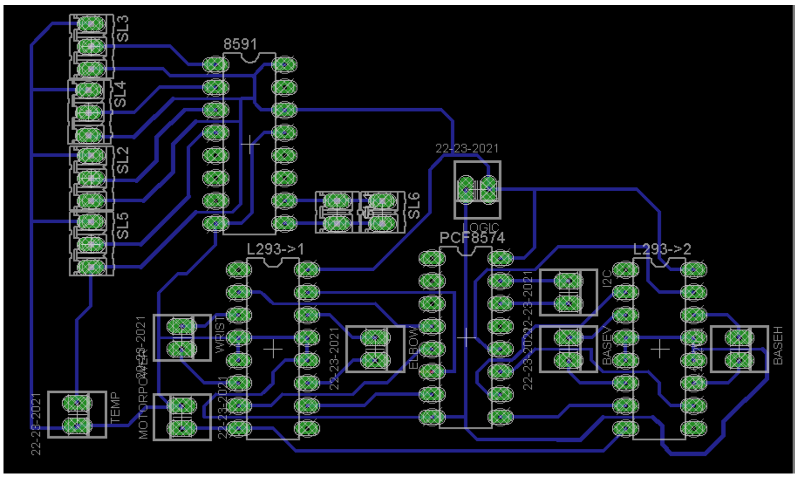


Figure 6.13 PCB layout for motor drivers

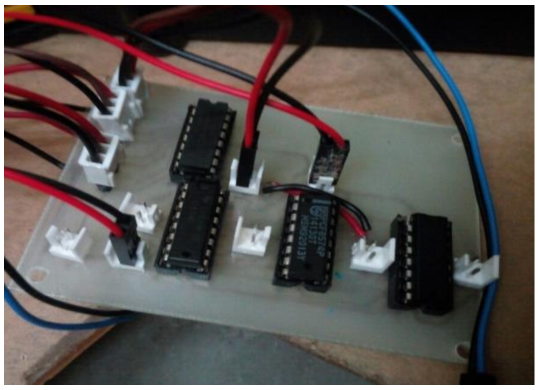


Figure 6.14 Actual assembly of the motor driver circuitry

• Chapter 7

OBSERVATIONS AND RESULTS

7.1 Flow of operation

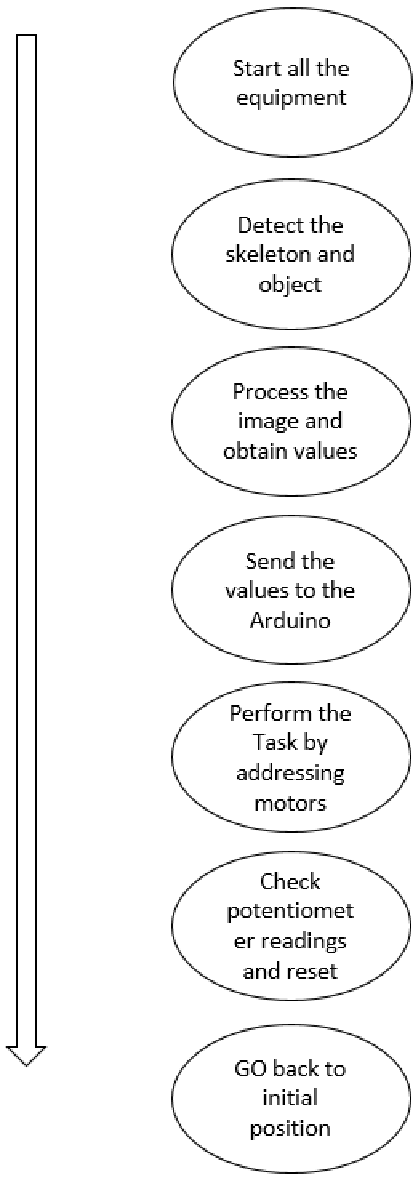

Start all the equipment
Detect the skeleton and object
Process the image and obtain values
Send the values to the Arduino
Perform the Task by addressing motors
Check potentiometer readings and reset
GO back to initial position

Figure 8.1 flow chart of the system

7.2 Kinect recordings

7.2.1 Skeleton detection

While taking the recordings from the Kinect a confidence variable is checked, if it gives the value 1, then that particular defined position is detected. The grayscale image is obtained through IR camera and the position of the **Elbow, Palm and shoulder is detected.** As the robotic arm is moving in the same direction like body arm, its edges should be properly visible in order to calculate the exact distances between the joints.

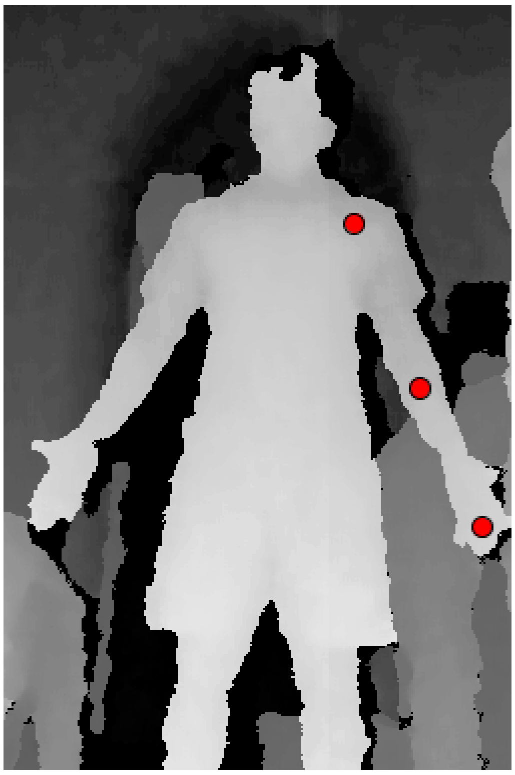


Figure 7.2 Palm, elbow and shoulder detection by Kinect

Following are the observations through the readings:

1) The distance between Kinect and the object or body should not be more than 20ft. (25ft is specified by the manufacturer).
2) While detecting palm and the elbow the confidence value should be checked and then data values have to taken
3) The object size should not be more than 10x 10cm as the palm will not be able to pick that object and will not be fit in the frame defined while detecting the object.

7.2.2 Object detection

The object detection is done in the fame of the palm area only, it will save the time to search the object in the whole frame. Searching the object in the complete frame will increase the complexity as well as the time delay between consecutive frames. As shown in the figure the gray box in the 2nd picture shows the new defined frame for object detection. 1st image is showing the actual RGB image of the person and object in his hand.

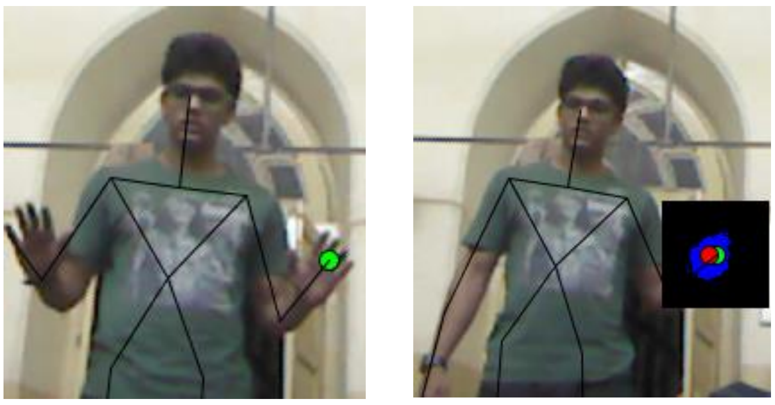


Figure 7.3 object tracking

The object is detected by the object detection algorithm specified in chapter 5. The object is detected and its real time co-ordinates are available at the serial port of the processing. These values of the

7.3 Potentiometer Calibrations

When motor rotates and starts to perform action it is necessary to remember the initial position of the motor. Even if the position of the particular part is known to the controller it should also remember the paths followed elbow and base motor mounted on arm. To keep track of the motor potentiometers are mounted on with the arm. The elastic windings are provided with it so that it will measure the rotated position of the Kinect. Accelerometer or Gyro meters can also be used but they need their different processing. The mounting of the potentiometer is as shown below

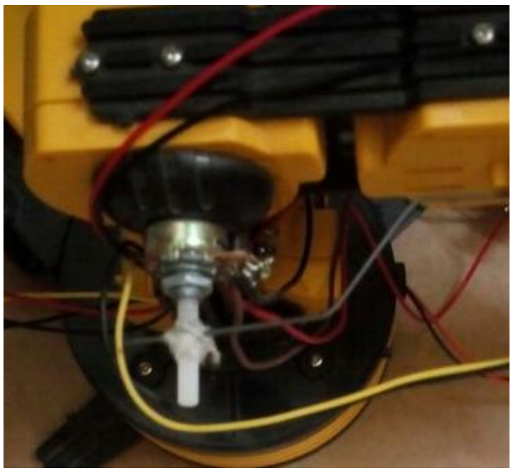


Figure 7.4 potentiometer

The observed readings of the arm with their minimum and maximum positions:

Name of Motor	Angle range(Deg.)	Value range(Analog values) Min-Max angle
Base motor	0-45	556-750
Elbow motor	0-90	900-750
Wrist motor	0-90	582-230

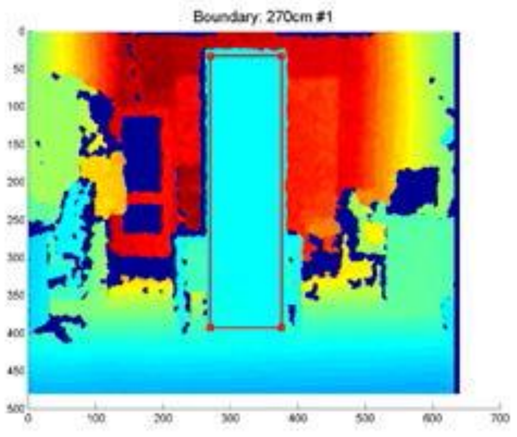
Table 2 Potentiometer readings

7.4 Test conditions

7.4.1 Repeatability test for Kinect

The repeatability of the range measurement is an essential aspect of using depth imaging sensors, as it provides the assessment of the ranging precision in short term. To determine the sensor repeatability performance, a planar target was imaged from a distance ranging from 0.5 m to 5 m in 0.1 m steps. Figure shows 3D (depth) images of the target with and without the environment. The measurement was repeated six times, so a total of 46 x 6 images were acquired and processed. The planar target has a size of 180 cm x 60 cm, so it's FOV in the image changes a lot. Consequently, the number of points obtained by the 3D sensor from the reference planar target varies over a large range, from 200K down to 10K [11].

However the range defined by the manufacturer is around 25ft, the Kinect camera start giving distortion after 17ft. The graph shows the results of repeatability test for Kinect.



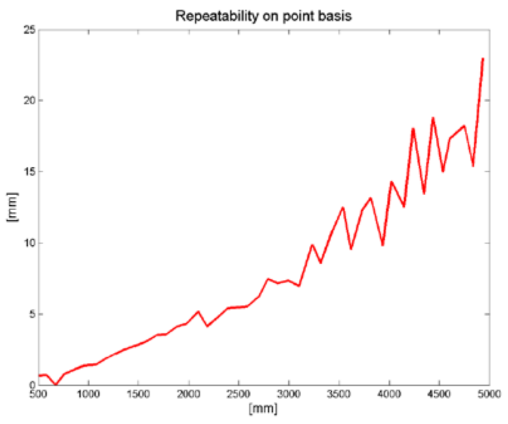


Figure 7.5 Testing object and repeatability test result

7.4.2 Distance test for Kinect

The Kinect is not a high resolution camera, it has a resolution of 640*480 pixels per frame. This causes Kinect to distort its image as the distance increases and the shadows are generated. Shadows causes the edges disturbances which makes user hard to determine the dimensions of the object. Depending upon the distance test the finalized distance is of 10ft. If the object is too close to the camera then Kinect will not detect the precise color depth, hence 10ft is an appropriate distance to perform the task. The graph showing the value of the Kinect at various depths [12].

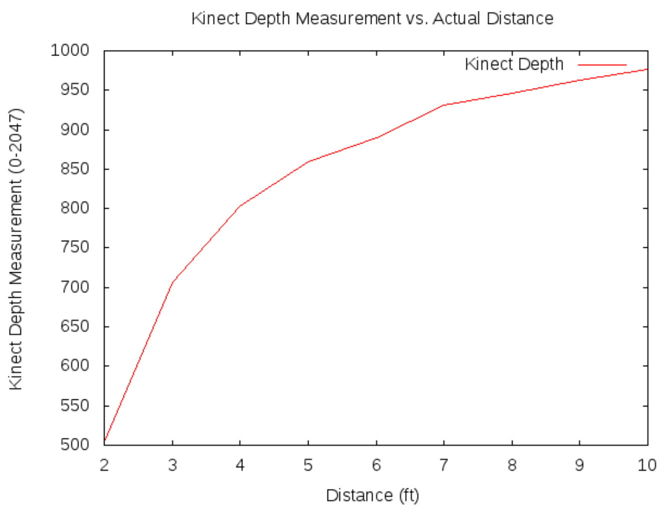


Figure 7.6 Distance vs. Depth value of Kinect

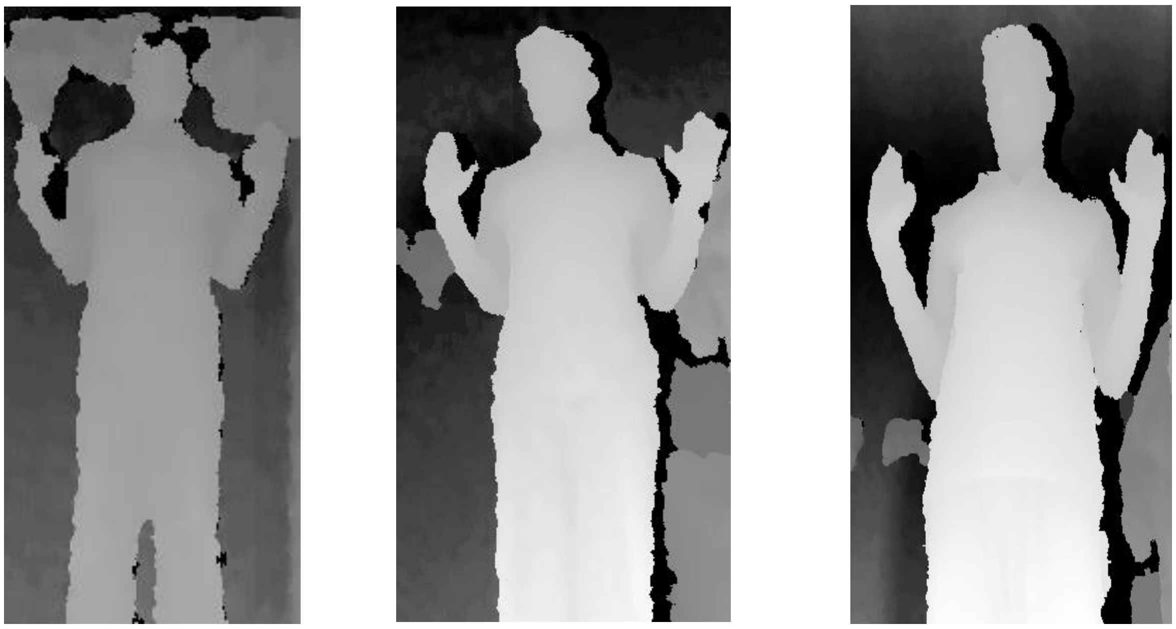


Figure 7.7 Depth images at distances of 15ft, 10ft & 5ft

7.5 Conclusions and Summary

7.5.1 Conclusions

1) The Xerobot works as multitasking robot and have ability to execute many jobs with high level of precision.
2) The image processing involved within the Processing software gives the dimensions of the object with negligible deviations
3) The arm motors requires some delay for the movement, hence the motions ae performed with the speed considering these delays
4) Kinect does not give the high resolution frames to process which causes some distortions.
5) The tests done on Kinect gives the variations in its reading depending upon the distance, rate of readings and have no effect of illumination in depth image for normal operations.

7.5.2 Limitations

1) The movements of the Xerobot are restricted because the height of the robotic arm as well as the gripper movement. It also involves too much calibrations in adjusting the position. Gripper cannot pick the objects more than 10x10cm.
2) Kinect camera is another limitation for the Xerobot as the frame resolution is lower the image quality decreases causing the inefficient processing of the image

3) Arduino is the controller limits the usage of Xerobot as it does not have the image processing capabilities as well as it cannot store high amount data into ROM. It requires PC to be mounted on the bot which makes it bulky.
4) Time delay between the processing of the image causes the delay between the readings

7.5.3 Remedies

1) Robotic arm should be put to the height of the body to avoid extra calibrations
2) The camera with higher resolutions will lessen the efforts of filtering of the image for various parameters.
3) Instead of Arduino if the processor is used which will eliminate the usage of PC attached with it. External memory can be provided with processor to avoid the problem of storage
4) High precise camera and its calibration will correspondingly decrease the time delay.

7.5.4 Future scope

Xerobot have wide application in many fields like Industrial, Household and military purpose. The applications of the Xerobot are not limited to any particular area as it has the capability of self-learning. If the further modification are done in calibrating and hardware arrangements the Xerobot can extend its application for the fields like Aeronautics, agriculture & research work. Xerobot is the future of the robotic available today.

Appendix I

THE LIST OF COMPONENETS

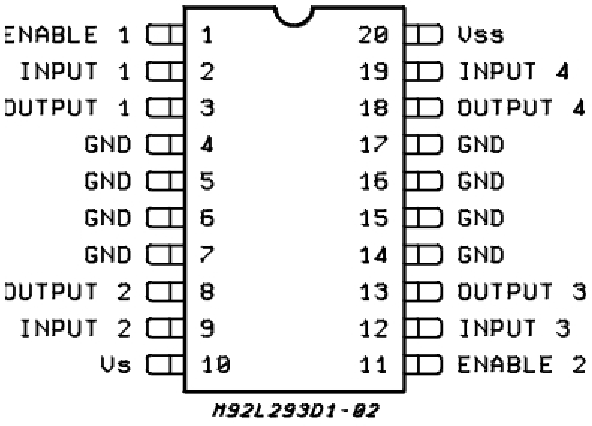
Sr. No.	Product Description	Quantity	Cost(Rs.)
1.	PCF 8574	1	40
2.	CAPACITORS 1uF	4	14
3.	RESISTORS 6.8k	2	8
4.	HEAT SINK	4	60
5.	BATTERY LiPo	1	2000
6.	ARDUINO UNO	1	1550
7.	RELIMATE COMNNECTORS	14	102
8.	PHEONIX CONNECTOR	2	30
9.	ROBOTIC ARM	1	3000
10.	POTENTIOMETER 10k	4	60
11.	POWER TRANSISTOR	2	40
12.	POWER RESISTOR	2	40
13.	PCB's	2	600
14.	L293d	4	40
15.	FUSE	1	100
16.	MICROSOFT KINECT	1	11000
17.	JUMPERS	20	20
18.	MALE CONNECTORS	5	15
19.	FEMALE CONNECTORS	3	15
20.	ACRYLIC MATERIAL	1	350
21.	CASTER	1	50
22.	BATTERIES 5V NiCad	2	20

Appendix II

IC SPECIFICATIONS

1) L293d

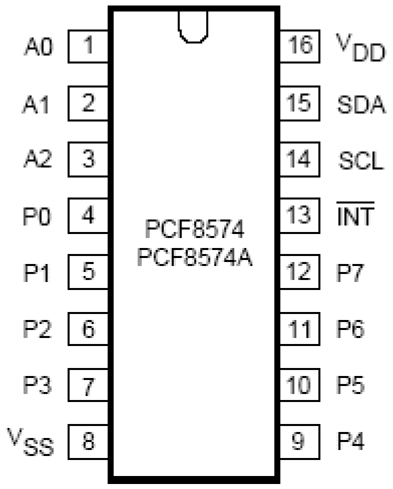
- Wide Supply-Voltage Range: 4.5 V to 36 V
- Separate Input-Logic Supply
- Internal ESD Protection
- Thermal Shutdown
- High-Noise-Immunity Inputs
- Functionally Similar to SGS L293 and
- SGS L293D
- Output Current 1 A per Channel
- (600 mA for L293D)
- Peak Output Current 2 A Per Channel
- (1.2 A for L293D)



2) PCF 8574

Operating supply voltage 2.5 to 6 V

- Low standby current consumption of 10 mA maximum
- I2C-bus to parallel port expander
- Open-drain interrupt output
- 8-bit remote I/O port for the I2C-bus
- Compatible with most microcontrollers
- Latched outputs with high current drive capability for
- directly driving LEDs
- Address by 3 hardware address pins for use of up to
- 8 devices (up to 16 with PCF8574A)
- DIP16, or space-saving SO16 or SSOP20 packages



REFRENCES

[1] http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5959090&tag=1 Multitasking robot

[2] http://eng.au.dk/fileadmin/DJF/ENG/PDF Kinect Specification

[3] http://www.arduino.cc/en/Main/ArduinoBoardUno Arduino UNO board

[4] http://www.arduino.cc/en/Main/Software Arduino IDE

[5] https://www.processing.org/ Processing basics

[6]http://www.instructables.com/id/Locket-of-Love/step2/OpenNI-Processing-Library/ OpenNI

[7]http://urbanhonking.com/ideasfordozens/2011/02/16/skeleton-tracking-with-kinect-and-processing/ skeleton tracking with processing

[8] http://in.mathworks.com/discovery/object-detection.html Real time object detection

[9] http://www.elenco.com/admin_data/pdffiles/OWI535USB.pdf Robotic arm

[10]http://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XXXIX-B3/543/2012/isprsarchives-XXXIX-B3-543-2012.pdf Repeatability test for Kinect

[11] http://mathnathan.com/2011/02/depthvsdistance/ Distance test for Kinect

[12] https://www.sparkfun.com/pages/X-bee_guide X-bee guide

[13]https://eewiki.net/display/Wireless/X-bee+Wireless+Communication+Setup#X-beeWirelessCommunicationSetup-Step2:PuttogetheryourX-beebreakoutboard X-bee XCTU