HAND GESTURE BASED SMART WHEELCHAIR

A PROJECT REPORT

Submitted by

SRI RAMANA.S (2010505055)

KOUSHIK.B (2010505021)

VIGNESH.B (2010505063)

Under the guidance of

Dr.N.PAPPA

Sponsored under

RESEARCH SUPPORT SCHEME

of

CENTER FOR TECHNOLOGY DEVELOPMENT AND TRANSFER

ANNA UNIVERSITY

DEPARTMENT OF INSTRUMENTATION ENGINEERING

JANUARY 2014

CTDT PROJECT DETAILS

i) Students involved in the project

Name	*Department/ Centre*	*Roll Number*	*E-mail Id and Mobile number*
SRI RAMANA.S	EIE	2010505055	raminmyst@gmail.com +91-9952461750
KOUSHIK.B	EIE	2010505021	kshik.balas@gmail.com +91-8056325354
VIGNESH.B	EIE	2010505063	bvignesh2992@gmail.com +91-9629921090

ii) Faculty associated with the project

Name	*Department/Centre*	*Designation*	*Phone/Mobile Number & E-mail Id*
Dr.N. Pappa	EIE	Associate Professor	+919962560646 npappa@rediffmail.com

iii) Date of start of the project : 08.06.2013

iv) Duration of the project : 6 months

v) Total approved value : INR 25,000

vi) Amount spent : INR 17,944

1. ABSTRACT OF THE PROJECT

The main objective of this project is to construct a wheelchair fitted with a robotic arm and to control the same using hand gestures and also the robotic arm. This project uses a vision based scheme for gesture recognition. This project is aimed at those patients who are paralyzed at the legs and those who have weak reflexes. It enables them to move freely by navigating the wheelchair and using the same gesture pickup any object in the vicinity by controlling the robotic arm. Images taken by camera are processed in laptop and serial commands are given to the Arduino. An Arduino is used as the main processor to control both the navigation and the robotic arm. A relay circuit drives the wheelchair through signals from Arduino. The arm uses inverse kinematics to find its appropriate gripper position in 3-D co-ordinate space.

TOPIC **P.NO**

ABSTRACT **3**

LIST OF FIGURES **7**

1. LITERATURE SURVEY **8**

1.1 SMART WHEELCHAIR **8**

1.2 RECENT DEVELOPMENTS 8

2.WHERE OUR PROJECT STANDS: 10

2.1 DEMERITS OF USING NON-VISION TYPE/JOYSTICK BASED SMART WHEELCHAIR **11**

2.2 DEMERITS OF USING HEAD GESTURE BASED SMART WHEELCHAIR **11**

2.3 MERITS OF USING VISION TYPE BASED SMART WHEELCHAIR **11**

3. GENERAL INFORMATION:

3.1 ROBOTIC ARM: **11**

3.2 TYPES OF ROBOTIC ARMS **12**

3.3 IMAGE PROCESSING **13**

3.4 OPENFRAMEWORKS **14**

3.5 ARDUINO **15**

4. PROJECT WORK **16**

A) NAVIGATION MODE16

B) ROBOTIC ARM MODE16

4.1 MODE SELECTION 22

4.2 GESTURE RECOGNITION ALGORITHM 22

4.2.1 PREPROCESSING22

4.2.2 DETECTION OF GESTURE FROM THE MATRIX 22

4.2.3 DECIDING MODES 23

4.3 IMPLEMENTATION IN OFXOPENCV ADD-ON OF OPENFRAMEWORKS 23

4.4 SERIAL COMMUNICATION TO THE ARDUINO 24

4.5 INVERSE KINEMATICS DERIVATION 25

4.6 HARDWARE IMPLEMENTATION 28

5. CONCLUSION AND FUTURE SCOPE 33

APPENDICES

APPENDIX 1 34

APPENDIX 2 35

APPENDIX 3 43

APPENDIX 4 50

LIST OF FIGURES: PAGE NO:

1 ROUGH SKETCH OF WHEELCHAIR 17

2. OVERALL BLOCK DIAGRAM 18

3. GESTURES USED IN THIS PROJECT 19

4. ROBOTIC ARM FREE BODY DIAGRAM 25

5.1. OVERALL HARDWARE BLOCK DIAGRAM 28

5.2. RELAY CIRCUIT 29

6 . PICTURE OF ROBOTIC ARM 31

7. PICTURE OF WHOLE SETUP 32

APPENDIX-4

GESTURES IN BLACK AND WHITE IMAGES 50

1. LITERARTURE SURVEY

1.1 SMART WHEELCHAIR:

A smart wheelchair is any motorized platform with a chair designed to assist a user with a physical disability, where an artificial control system augments or replaces user control. Its purpose is to reduce or eliminate the user's task of driving a motorized wheelchair. Usually, a smart wheelchair is controlled by a computer, has a suite of sensors and applies techniques in mobile robotics, but this is not necessary. The interface may consist of a conventional wheelchair joystick, or it may be a "sip-and-puff" device or a touch-sensitive display connected to a computer. This is different from a conventional motorized or electric wheelchair, in which the user exerts manual control over motor speed and direction via a joystick or other switch- or potentiometer-based device, without intervention by the wheelchair's control system.

Smart wheelchairs are designed for a variety of user types. Some platforms are designed for users with cognitive impairments, such as dementia, where these typically apply collision-avoidance techniques to ensure that users do not accidentally select a drive command that results in a collision. Other platforms focus on users living with severe motor disabilities, such as cerebral palsy, or with quadriplegia, and the role of the smart wheelchair is to interpret small muscular activations as high-level commands and execute them. Such platforms typically employ techniques from artificial intelligence, such as path-planning.

1.2 RECENT DEVELOPMENTS:

Recent technological advances are slowly improving wheelchair and EPW technology. Some wheelchairs, such as the iBOT, incorporate gyroscopic technology and other advances, enabling the chair to balance and run on only two of its four wheels on some surfaces, thus raising the user to a height comparable to a standing person. They can also incorporate stair-climbing and four-wheel-drive feature motorized assists for hand-powered chairs are becoming more available and advanced. The popular Segway Personal Transporter is a mobility device that was a direct outgrowth of the development of the iBOT wheelchair. The Segway, which is basically an iBOT with two wheels removed, was developed explicitly to

increase the number of units produced and take advantage of the economies of scale to make the iBOT affordable to wheelchair users. The $25,000 iBot, which was developed as a joint venture between Johnson's Independence and Dean Kamen's DEKA Research, was discontinued in January 2009.

A variation on the hand-powered wheelchair is the Leveraged Freedom Chair (LFC), designed by the MIT Mobility Lab. This wheelchair is designed to be low-cost, constructed with local materials, for users in developing countries. Engineering modifications have added hand-controlled levers to the LFC, to enable users to move the chair over uneven ground and minor obstacles, such as bumpy dirt roads, that are common in developing countries. It is under development, and has been tested in Kenya and India so far.

The addition of geared, all-mechanical wheels for manual wheelchairs is a new development incorporating a hypocycloidal reduction gear into the wheel design. The 2-gear wheels can be added to a manual wheelchair. The geared wheels provide a user with additional assistance by providing leverage through gearing (like a bicycle, not a motor). The two-gear wheels offer two speed ratios- 1:1 (no help, no extra torque) and 2:1, providing 100% more hill climbing force. The low gear incorporates an automatic "hill hold" function which holds the wheelchair in place on a hill between pushes, but will allow the user to override the hill hold to roll the wheels backwards if needed. The low gear also provides downhill control when descending.

Others variants, that are being developed, are listed in the following section. A Standing wheelchair is one that supports the user in a nearly standing position. They can be used as both a wheelchair and a standing frame, allowing the user to sit or stand in the wheelchair as they wish. They often go from sitting to standing with a hydraulic pump or electric-powered assist. Some options are provided with a manual propel model and power stand, while others have full power, tilt, recline and variations of power stand functions available as a rehabilitative medical device. The benefits of such device includes, but is not limited to:

- Raises Independence
- Raises Self Esteem
- Heightens Social Status
- Allows For Easier Communication
- Extends Access Level

- Improved Quality of Life
- Increased Pressure Relief Improved functional reach to enable participation in ADL
- Improved Circulation, Enhance independence and productivity
- Improved Respiration, Maintain vital organ capacity, Reduce occurrence of UTI
- Improved Flexibility, Maintain bone mineral density, Improve passive range motion
- Improved Ease of Transfer, Reduce abnormal muscle tone and spasticity, pressure sores
- Reduce the pressure sores, skeletal deformities, and psychological well being[6]

` A bariatric wheelchair is one designed to support larger weights; most standard chairs are designed to support no more than 250 lbs. (113 kg) on average.

Paediatric wheelchairs are another available subset of wheelchairs. Hemi wheelchairs have lower seats which are designed for easy foot propulsion. The decreased seat height also allows them to be used by children and shorter individuals.

A power-assisted wheelchair is a recent development that uses the frame & seating of a typical manual chair while replacing the standard rear wheels with wheels that have small battery-powered motors in the hubs. A floating rim design senses the pressure applied by the users push & activates the motors proportionately. This result in the convenience, small size & light-weight of a manual chair while providing motorised assistance for rough/uneven terrain & steep slopes that would otherwise be difficult or impossible to navigate, especially by those with limited upper-body function.

2. WHERE OUR PROJECT STANDS:

Though there are many sophisticated devices they are not much accessible to the middle class people and most of them are still in the development stage or high end devices that are applicable only to aristocratic people. Our solution is rather a middle level cost solution that incorporates navigation and arm control using vision based gesture. Below we list some of the merits of our method in comparison with similar techniques that are implemented throughout the world.

2.1 DEMERITS OF USING NON-VISION TYPE/ JOYSTICK BASED SMART WHEELCHAIR:

- Needs complex wrist movement[2]
- Improper control may lead to accidents.[2]
- Difficult for elderly and certain disabled person because they lack the necessary motor skills, strength, or visual acuity.[2]
- User needs to wear gloves fitted with many flex sensors and wires and is very uncomfortable [2]
- Only limited no. of gestures possible.

2.2 DEMERITS OF USING HEAD GESTURE BASED SMART WHEELCHAIR:

- Head gesture-the response of the wheelchair is too slow due to the processing time, the occupant has to keep looking in that direction until the wheelchair reacts which might cause discomfort towards the user.[2]
- This method restricts the free head and gaze movement.

2.3 MERITS OF USING VISION TYPE BASED SMART WHEELCHAIR:

- Free bare hand movement no gloves or sensors needed.
- Simple design with a single camera.
- Allows large no. of gestures that can be used for various controlling purposes.
- Only method that can be used for sign language communication.

3. GENERAL INFORMATION:

3.1 ROBOTIC ARM:

A robotic arm is a type of mechanical arm, usually programmable, with similar functions to a human arm; the arm may be the sum total of the mechanism or may be part of a more complex robot. The links of such a manipulator are

connected by joints allowing either rotational motion (such as in an articulated robot) or translational (linear) displacement. The links of the manipulator can be considered to form a kinematic chain. The terminus of the kinematic chain of the manipulator is called the end effector and it is analogous to the human hand.

The end effector, or robotic hand, 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 lines perform a variety of tasks such as welding and parts rotation and placement during assembly. In some circumstances, close emulation of the human hand is desired, as in robots designed to conduct bomb disarmament and disposal.

3.2 TYPES OF ROBOTIC ARMS:

- CARTESIAN ROBOT / GANTRY ROBOT: Used for pick and place work, application of sealant, assembly operations, handling machine tools and arc welding. It's a robot whose arm has three prismatic joints, whose axes are coincident with a Cartesian coordinator.
- CYLINDRICAL ROBOT: Used for assembly operations, handling at machine tools, spot welding, and handling at die casting machines. It's a robot whose axes form a cylindrical coordinate system.
- SPHERICAL ROBOT / POLAR ROBOT (SUCH AS THE UNIMATE): Used for handling at machine tools, spot welding, die casting, fettling machines, gas welding and arc welding. It's a robot whose axes form a polar coordinate system.
- SCARA ROBOT: Used for pick and place work, application of sealant, assembly operations and handling machine tools. This robot features two parallel rotary joints to provide compliance in a plane.
- ARTICULATED ROBOT: Used for assembly operations, die casting, fettling machines, gas welding, and arc welding and spray painting. It's a robot whose arm has at least three rotary joints.
- PARALLEL ROBOT: One use is a mobile platform handling cockpit flight simulators. It's a robot whose arms have concurrent prismatic or rotary joints.

- 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.

3.3 IMAGE PROCESSING:

Image processing refers to processing of a 2D picture by a computer.

An image defined in the "real world" is considered to be a function of two real variables, for example, a(x,y) with a as the amplitude (e.g. brightness) of the image at the real coordinate position (x,y).

Modern digital technology has made it possible to manipulate multi-dimensional signals with systems that range from simple digital circuits to advanced parallel computers. The goal of this manipulation can be divided into three categories:

- Image Processing (image in -> image out)
- Image Analysis (image in -> measurements out)
- Image Understanding (image in -> high-level description out)

An image may be considered to contain sub-images sometimes referred to as regions-of-interest, ROIs, or simply regions. This concept reflects the fact that images frequently contain collections of objects each of which can be the basis for a region. In a sophisticated image processing system it should be possible to apply specific image processing operations to selected regions. Thus one part of an image (region) might be processed to suppress motion blur while another part might be processed to improve colour rendition. Sequence of image processing:

Most usually, image processing systems require that the images be available in digitized form, that is, arrays of finite length binary words. For digitization, the given Image is sampled on a discrete grid and each sample or pixel is quantized using a finite number of bits. The digitized image is processed by a computer. To display a digital image, it is first converted into analog signal, which is scanned onto a display.

Closely related to image processing are computer graphics and computer vision. In computer graphics, images are manually made from physical models of objects, environments, and lighting, instead of being acquired (via imaging devices such as cameras) from natural scenes, as in most animated movies. Computer vision, on the other hand, is often considered high-level image processing out of which a machine/computer/software intends to decipher the physical contents of an

image or a sequence of images (e.g., videos or 3D full-body magnetic resonance scans).

In modern sciences and technologies, images also gain much broader scopes due to the ever growing importance of scientific visualization (of often large-scale complex scientific/experimental data). Examples include microarray data in genetic research, or real-time multi-asset portfolio trading in finance.

Before going to processing an image, it is converted into a digital form. Digitization includes sampling of image and quantization of sampled values. After converting the image into bit information, processing is performed. This processing technique may be Image enhancement, Image restoration, and Image compression.

3.4 OPENFRAMEWORKS:

Open Frameworks is an open source C++ toolkit designed to assist the creative process by providing a simple and intuitive framework for experimentation. The toolkit is designed to work as general purpose glue, and wraps together several commonly used libraries, including:

- OpenGL, GLEW, GLUT, libtess2 and cairo for graphics
- rtAudio, PortAudio, OpenAL and Kiss,FFT or FMOD for audio input, output and analysis
- FreeType for fonts
- FreeImage for image saving and loading
- Quicktime, GStreamer and videoInput for video playback and grabbing
- Poco for a variety of utilities
- OpenCV for computer vision
- Assimp for 3D model loading

The code is written to be massively cross-compatible. Right now it supports five operating systems (Windows, OSX, Linux, iOS, Android) and four IDEs (XCode, Code::Blocks, and Visual Studio and Eclipse). The API is designed to be minimal and easy to grasp.

Open Frameworks is distributed under the MIT License. This gives everyone the freedoms to use open Frameworks in any context: commercial or non-

commercial, public or private, open or closed source. While many openFrameworks users give their work back to the community in a similarly free way , there is no obligation to contribute.

Simply put, openFrameworks is a tool that makes it much easier to make things with code. It is super useful for image processing techniques.

3.5 ARDUINO:

Arduino is an open source platform for development of applications. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software running on a computer (e.g. Flash, Processing, and MaxMSP).

The boards can be built by hand or purchased preassembled; the software can be downloaded for free. The hardware reference designs (CAD files) are available under an open-source license, we are free to adapt them to your needs.

4. PROJECT WORK:

A semi-autonomous hand gesture controlled wheelchair that works in two modes.(Fig 1)

- Navigation Mode
- Robotic Arm Mode.

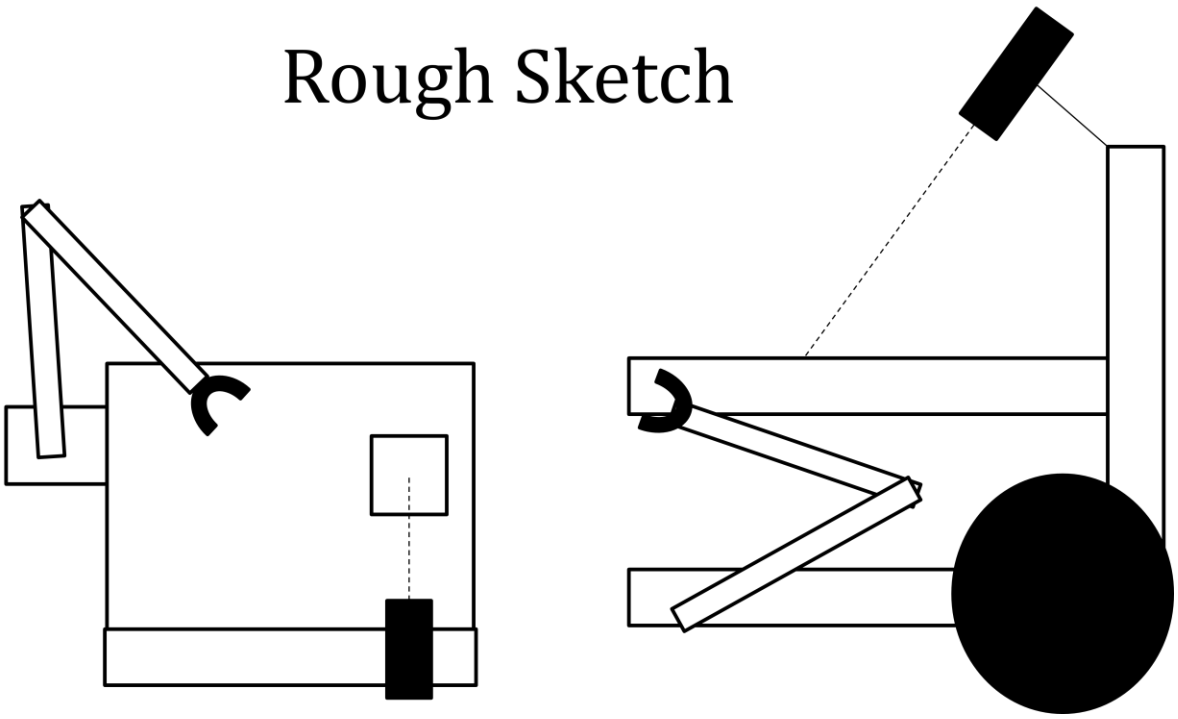
A) NAVIGATION MODE:

- The Control of the wheelchair motion is done by simply using easy to remember finger gestures.
- If index finger points forward, the wheelchair moves forward. Else it turns.
- At any point in time, a particular gesture causes the wheelchair to stop.

B) ROBOTIC ARM MODE:

- The arm, that we have constructed, is a 3DOF arm
- The wheelchair has 2DOF. The arm has totally 5 DOF since it will be remain attached to the wheelchair.
- The arm is also controlled by hand gestures to pick any object in the vicinity.

Imagine a user who cannot walk sitting in the wheelchair, and wants to grab a mug of coffee from the table at the far end of the room. He brings the wheelchair to the navigation mode by using the stop gesture followed by the pointer finger that stands for mode-1.Now he points his finger in the direction of the mug, the wheelchair moves smoothly towards the mug, once it is near the mug he makes the stop gesture. The wheelchair stops immediately. Now he is near the mug, but he cannot reach out and grab the mug. He chooses the mode selection or stop gesture to choose the robotic arm mode. Then he uses his fingers to navigate the gripper near the coffee mug and uses the ‘grab’ gesture to grab the mug. Then uses his finger to bring it near him and uses the ungrab gesture to place the mug on his other hand.



TOP VIEW SIDE VIEW

Fig 1

BLOCK DIAGRAM:

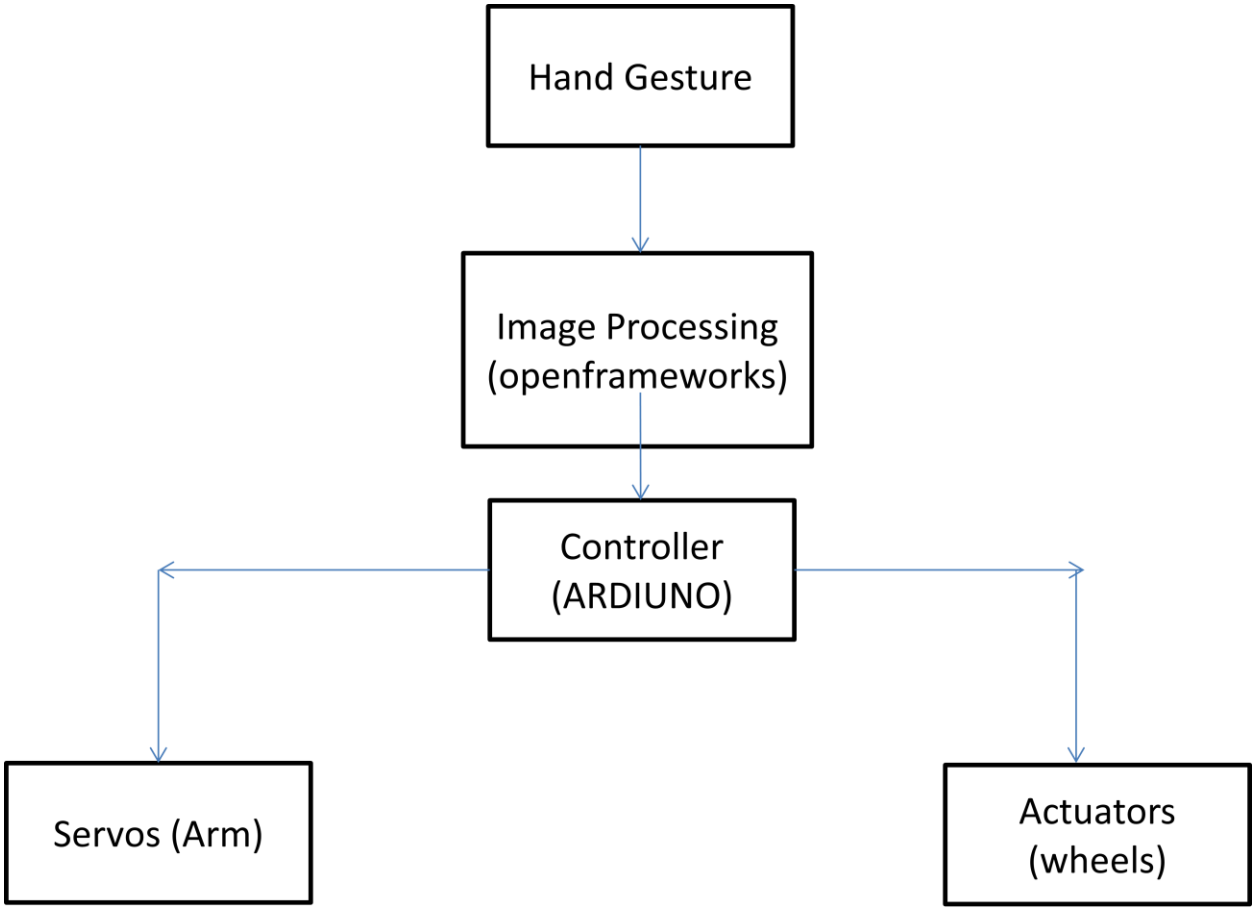


Fig 2

The overall block diagram of the setup is given in the Fig 2.The hand gestures are read as images through the camera attached to the wheelchair. Then, they are processed by the algorithm and then a control character is sent to Arduino by serial communication. The Arduino controls both the arm and the wheelchair based on its reception of the control character.

GESTURES USED IN THE PROJECT:(See appendix for more details 7.4)

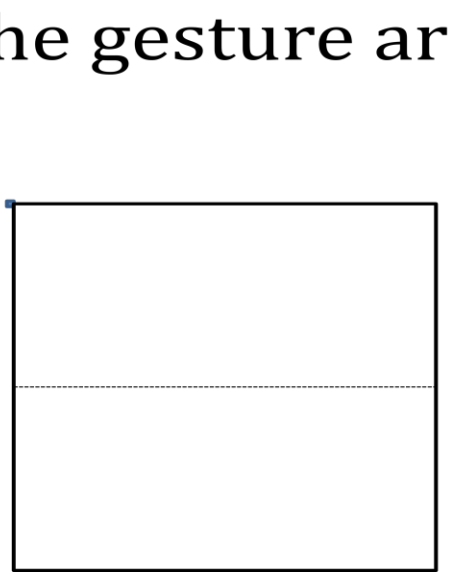


Fig 3.1

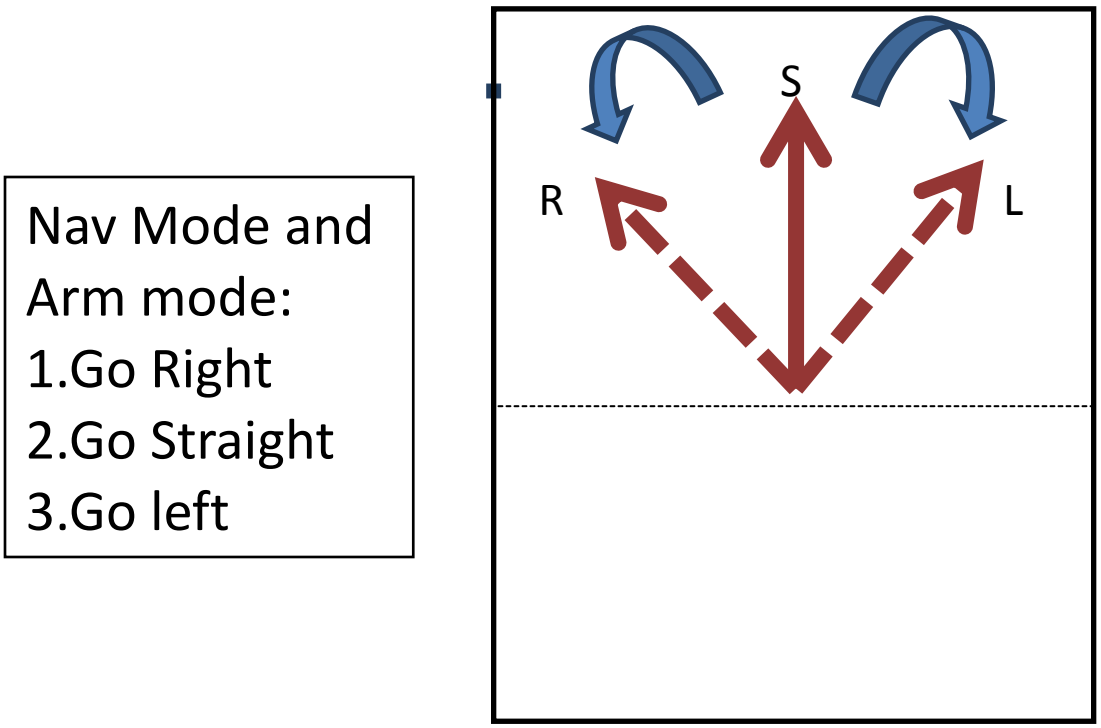


Fig 3.2

Both modes:
Reverse

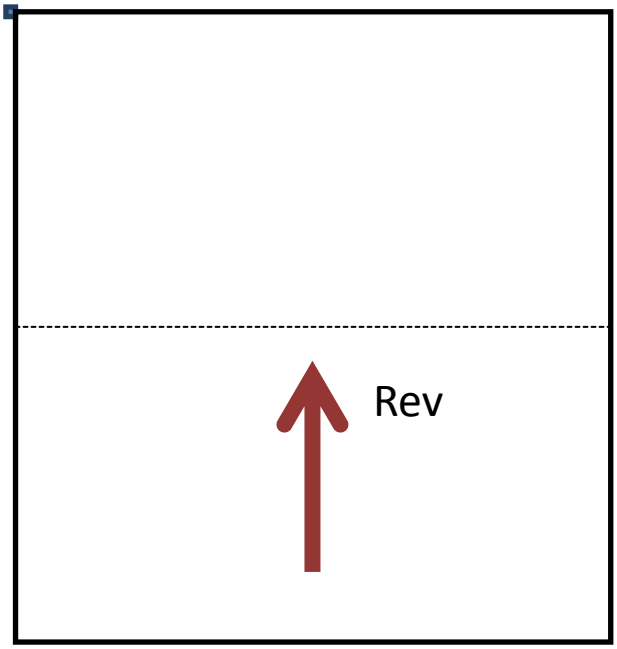


Fig 3.3

GRIPPER GRAB
and UNGRAB

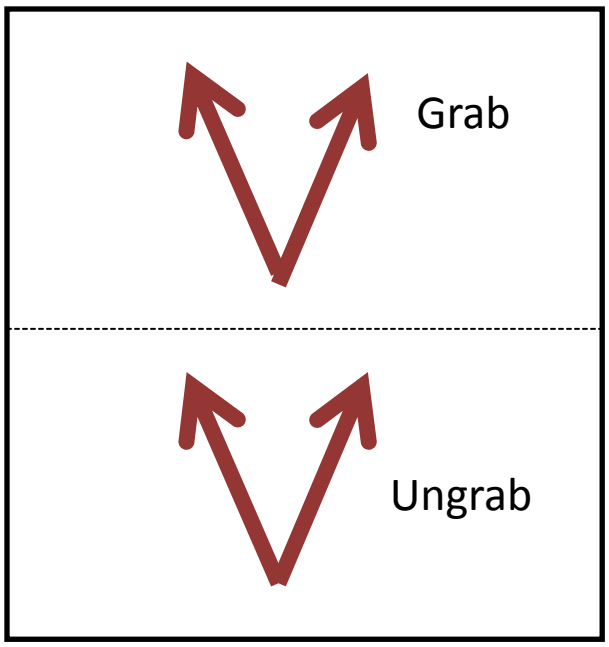


Fig 3.4

Gripper UP , DOWN and Reverse

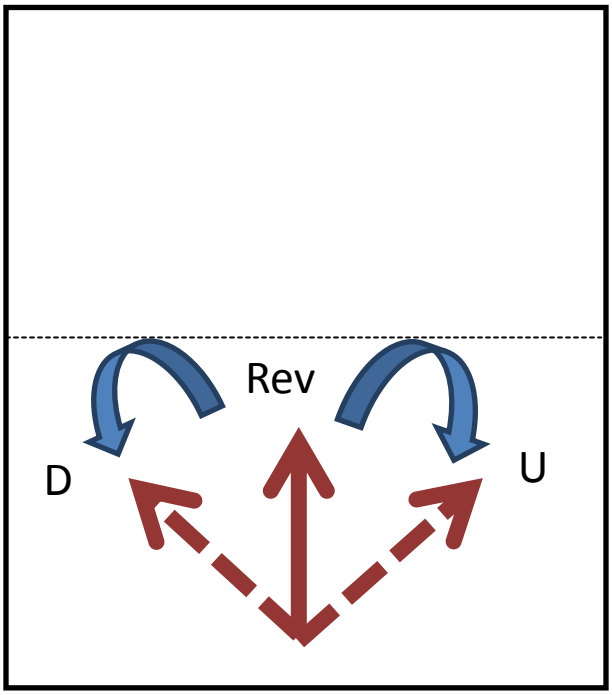


Fig 3.5

STOP:

A FIST IS MADE TO STOP IMMEDIATELY AND TO SELECT MODES

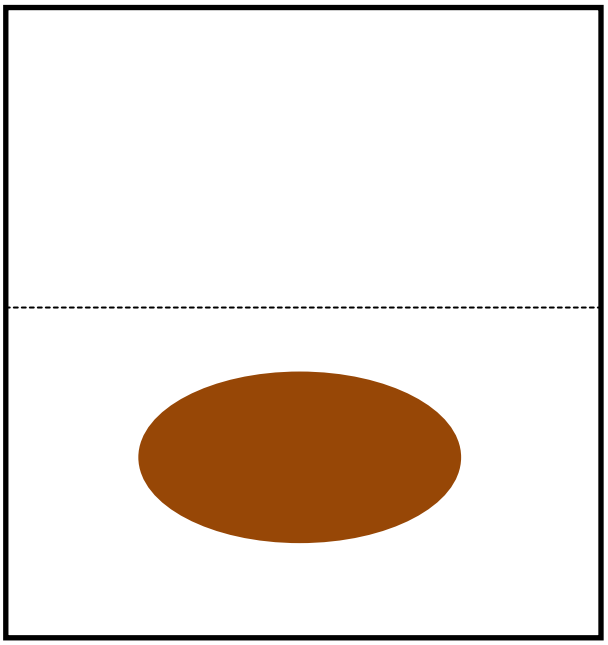


Fig 3.6

4.1 MODE SELECTION:

Whatever be the robots current position a stop gesture brings everything to a stop. Then a ‘pointer finger gesture or Go straight’ switches the wheelchair to the navigation mode. Instead a ‘Grab gesture’ switches the wheelchair to robotic arm control mode.

4.2 GESTURE RECOGNITION ALGORITHM:

4.2.1 PREPROCESSING:

1. It converts the colour image into a black and white image.
2. Define a matrix of size number of rows x 3.
3. The first column represents the number of columns to be taken for calculation.
4. The algorithm scans all rows and for every white to black and black to white transition, it records the centre pixel.
5. Then, it stores the number of times this transition occurs in the first column and the subsequent centre pixel as per the number of column indicated by the first column entry.

4.2.2 DETECTION OF GESTURE FROM THE MATRIX:

1. It is based on priority.
2. For the rows containing one pixel, the slope of line formed is taken as the angle of the finger.
3. If the column contains more than two pixels per row, then it is taken as gesture 2.
4. If first half of screen is empty, then the second half of the screen is examined for either the stop or up/down/reverse gesture.
5.

4.2.3 DECIDING MODES:

1. At any point in time, stop gesture brings back the user to mode selection.
2. The program waits for any one gesture.
3. The ‘go straight’ gesture indicates navigation mode.
4. The ‘Grab’ gesture indicates robotic arm mode.

4.3 IMPLEMENTATION IN OFXOPENCV ADD-ON OF OPENFRAMEWORKS:

The openframe works is a software framework that supports computer vision using ofxopencv addon that contains opencv libraries for vision application.The structure of opencv is that there are three important source files

1.main.cpp

2.testapp.cpp

3.testapp.h

The testapp.h and testapp.cpp is where we implement our program.

The main() is used to call the testapp.c as the procedure given below.

The testapp.h() is used to declare all the variables to be used by the testapp.c and also derive objects of predefined classes such as ofVideograbber , ofxcvcolorimage,..etc.

The testapp.c contains lots of predefined function for interfacing with PC such as functions that are executed during mouseclick, keypress etc. It has three most important functions

1.void setup()

2.void update()

3.void draw()

The setup() is executed only once at the starting of the program this is where we initialise all variables.

The update() is executed for every cycle continuously . It carries out all the non-graphics relate processing functions this is where we get the video grabber to get the incoming image and perform gesture recognition using the gesreg() function .We also perform the serial communication to the Arduino using ofserial class of openframeworks, based on the character that is returned from the gesreg() .

The draw() function is also executed on every iteration.It is where all the graphics related processing is done. In our project there is not much graphics required other that to simply display the image on the laptop screen.

(See appendix for program).

4.4 SERIAL COMMUNICATION TO THE ARDUINO:

- The gestures are communicated serially to arduino.
- The arduino takes decision based on the serially communicated value.
- It controls the wheelchair movement.
- It calculates inverse kinematics for the arm and drives the servo.

(See appendix for program).

4.5 INVERSE KINEMATICS DERIVATION:

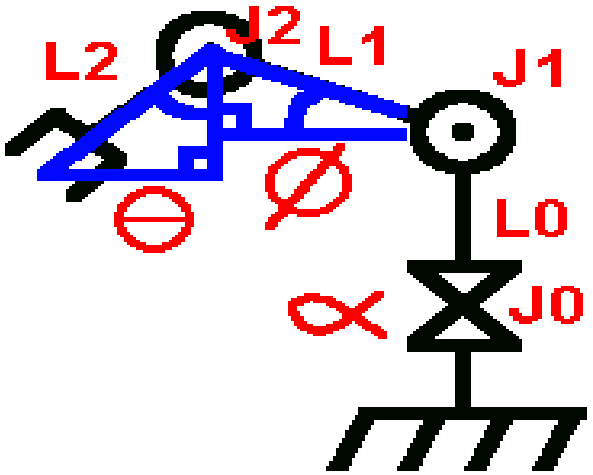


Fig 4

The mathematics of calculating the servo angles given the set of (x,y,z) co-ordinates is called inverse kinematics.

Let us shift the down by

$y = y - l_0$

First, we have to calculate the angle α of the base servo motor using the formula

$\alpha = -\tan^{-1}(x/z)$ or $(\pi - \tan^{-1}(x/z))$;

Now we calculate the x-co-ordinate when we rotate the arm back to the x-y plane.

$x = x/(\sin\alpha)$;

Since this is a 3-DOF arm

$L1\cos\phi+L2\sin\theta=x$ ----------(1)

$L1\sin\phi-L2\cos\theta=y$ ----------(2)

Multiply (1) by cosθ and (2) by sinθ

$x\cos\theta+y\sin\theta=L1\cos\phi\cos\theta +L1\sin\phi\sin\theta$

$x\cos\theta+y\sin\theta=L1\cos(\phi-\theta)$ ----------(3)

Multiply (1) by sinθ and (2) by cosθ

$x\sin\theta+y\cos\theta=L1\cos\phi\sin\theta-L1\sin\phi\cos\theta+L2(\cos^2\theta+\sin^2\theta)$

=L1cosϕinθ-L1sinϕcosθ+L2

xsinθ+ycosθ=L2-L1sin(ϕ-θ) ----------(4)

squaring (3) and (4) on both sides

$x^2(\cos^2\theta+\sin^2\theta)+y^2(\sin^2\theta+\cos^2\theta) = L1^2\cos^2(\phi-\theta) + L2^2+L1^2\sin^2(\phi-\theta)-2L1L2\sin(\phi-\theta)$

$x^2+y^2=L1^2+L2^2-2L1L2\sin(\phi-\theta)$

$2L1L2\sin(\phi-\theta)=L1^2+L2^2-x^2-y^2$

$\sin(\phi-\theta)=(L1^2+L2^2-x^2-y^2)/2L1L2$

$\phi-\theta=\sin^{-1}((L1^2+L2^2-x^2-y^2)/2L1L2)$

assume $f=(L1^2+L2^2-x^2-y^2)/2L1L2$

$\phi-\theta=\sin^{-1}f$

$\underline{\theta= \phi- \sin^{-1}f}$

sub θ in (1)

$L1\cos\phi+L2\sin(\phi- \sin^{-1}f)=x$

$(L1-L2f)\cos\phi+L2\cos(\sin^{-1}f)\sin\phi=x$ ----------(5)

It is of the form

$A\cos\phi+B\sin\phi=C$

Then

$R* \sin(\phi+\alpha)=C$; where $R= \text{sqrt}(A^2 + B^2)$ and $\alpha=\tan^{-1}(A/B)$

$\phi= \sin^{-1}(C/\text{sqrt}(A^2 + B^2)) – \tan^{-1}(A/B)$;

$\underline{\phi=\sin^{-1}(x/\text{sqrt}((L1-L2f)^2+L2^2\cos^2(\sin^{-1}f)))-\tan^{-1}((L1-L2f)/L2\cos(\sin^{-1}f)))}$

If y is +ve then we use the formula for (5) as

$A\cos\phi+B\sin\phi=C$

Then

$R* \cos(\phi-\alpha)=C$; where $R= \text{sqrt}(A^2 + B^2)$ and $\alpha=\tan^{-1}(B/A)$

$\phi= \cos^{-1}(C/\text{sqrt}(A^2 + B^2)) – \tan^{-1}(B/A)$;

$\underline{\phi=\cos^{-1}(x/\text{sqrt}((L1-L2f)^2+L2^2\cos^2(\sin^{-1}f)))+\tan^{-1}(L2\cos(\sin^{-1}f)/(L1-L2f)))}$

to uncover different solution from the solution space.

Using ϕ, θ can be calculated as

$\underline{\theta= \phi- \sin^{-1}f;}$

4.6 HARDWARE IMPLEMENTATION:

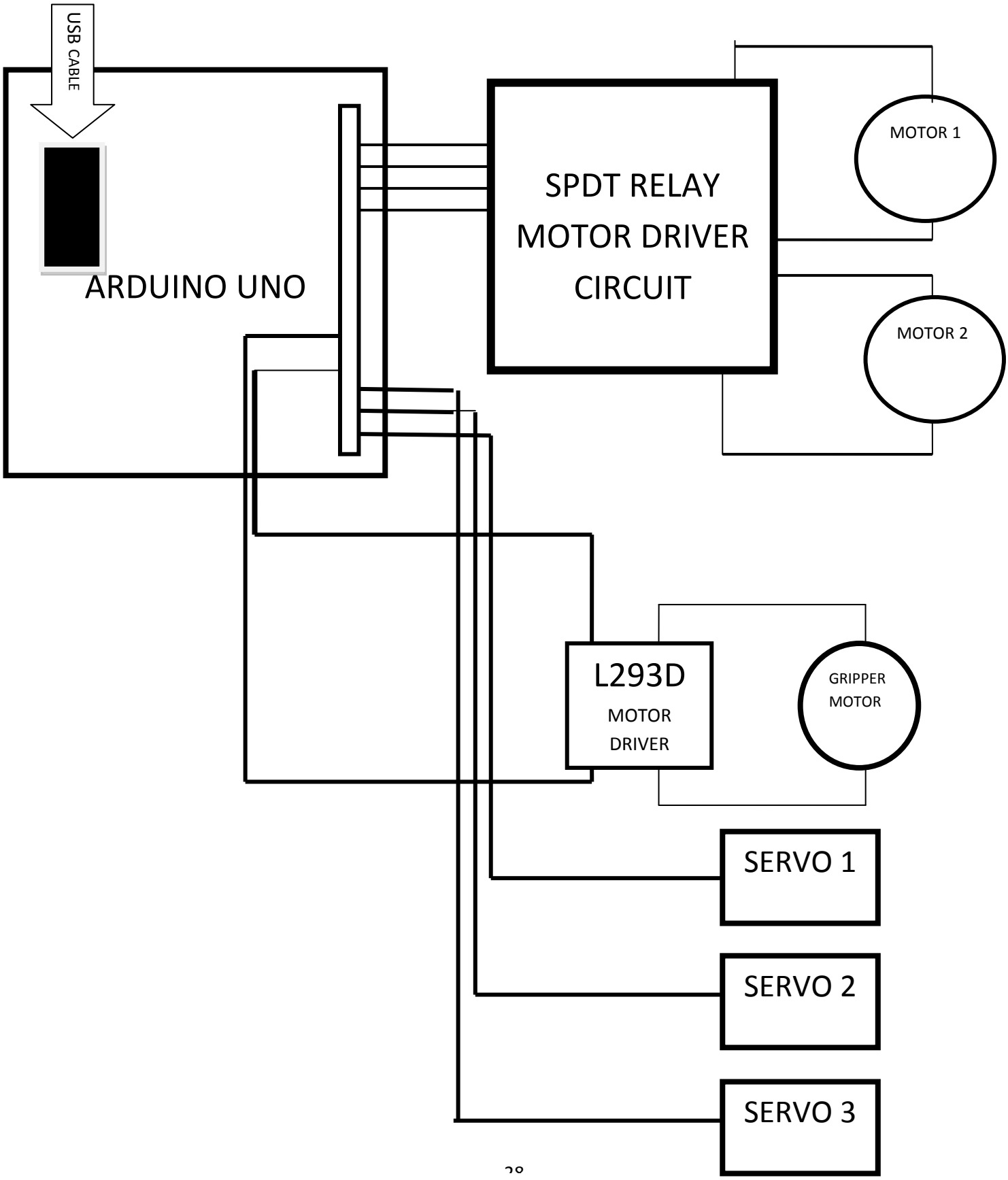


Fig 5.1

RELAY CIRCUIT DIAGRAM:

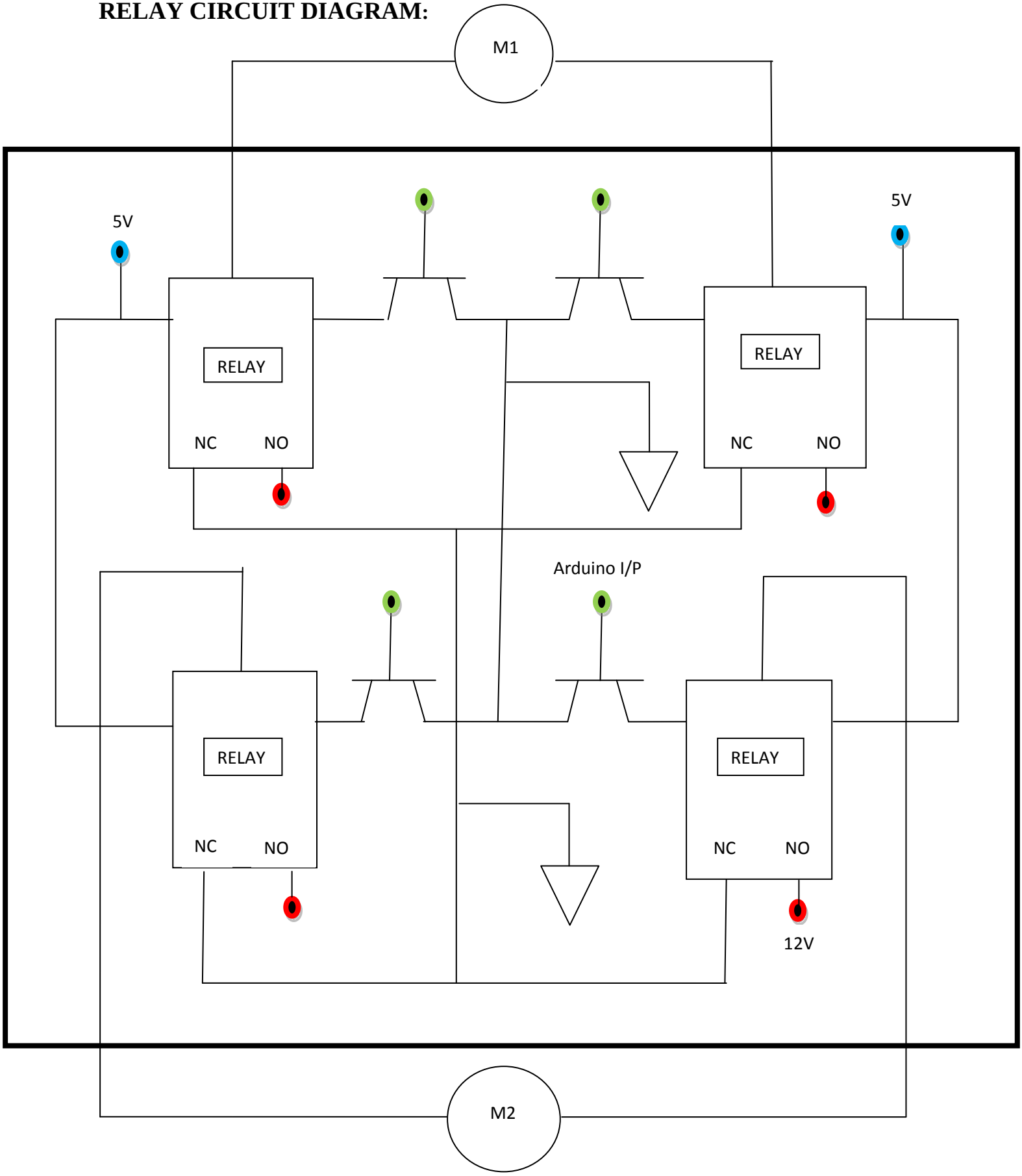


Fig 5.2

The overall circuit diagram of the project can be seen in Fig 5.1 . The project is implemented by using a webcam attached to a laptop that contains the gesture recognition program in open frameworks . The gesture area shown in the diagram is the area of focus for the camera. This is the region within which the user has to realize the finger gesture .So the camera continuously captures frames and feeds it to the laptop that does the image processing.

The laptop is serially connected to a **ARDUINO UNO BOARD** that acts as the main processor. It is **powered from the USB .**There are several pins in the board that perform both I/O operation and Pulse Width Modulation (PWM) required for servo controls. The Arduino pins 3,6,7 functions as both I/O and PWM but we use it as the latter to connect to the servo motor signal pin for the robotic arm.

The wheelchair is modified and fitted with a wiper motor of sufficient torque to drive the person-loaded wheelchair. The power supply for the wheelchair is from a high current 12V battery. The I/O pins of the Arduino 4, 8, 9, 10 are used to control the wheelchair motor **through the relay circuit** (Fig 5.2).

A relay circuit consisting of four SPST relays connected and controlled in such a way that the power supply is made to flow so as to change the direction of the motor. Since the wheelchair uses a differential drive technique for navigation, four Arduino signals are sufficient for two wheels.

Fig 6

The robotic arm is a 3-DOF arm with a base servo motor rotating on z-axis and shoulder and a elbow servo for planar motion. The base servo changes the plane of motion and hence a range of 3-D space can be traversed using this setup and depending on the length of each arm. It is found that a servo motor of **torque 15kgcm** is sufficient for all the joints. An end effector also called the gripper is

fixed at the end of the robotic arm and it is driven by a bomotor which is an ordinary D.C motor of low rpm.

The power supply for the robotic arm is from an ordinary lead acid 12V battery. The servo-motor operates at a voltage of **4.8-6V** hence a 7806 IC is used to bring it down to 6-V and then is given to the supply pin of the servo-motor. There is also a L293D for driving the gripper bomotor under 12V. This is also controlled by the 5, 12 pins of the arduino.

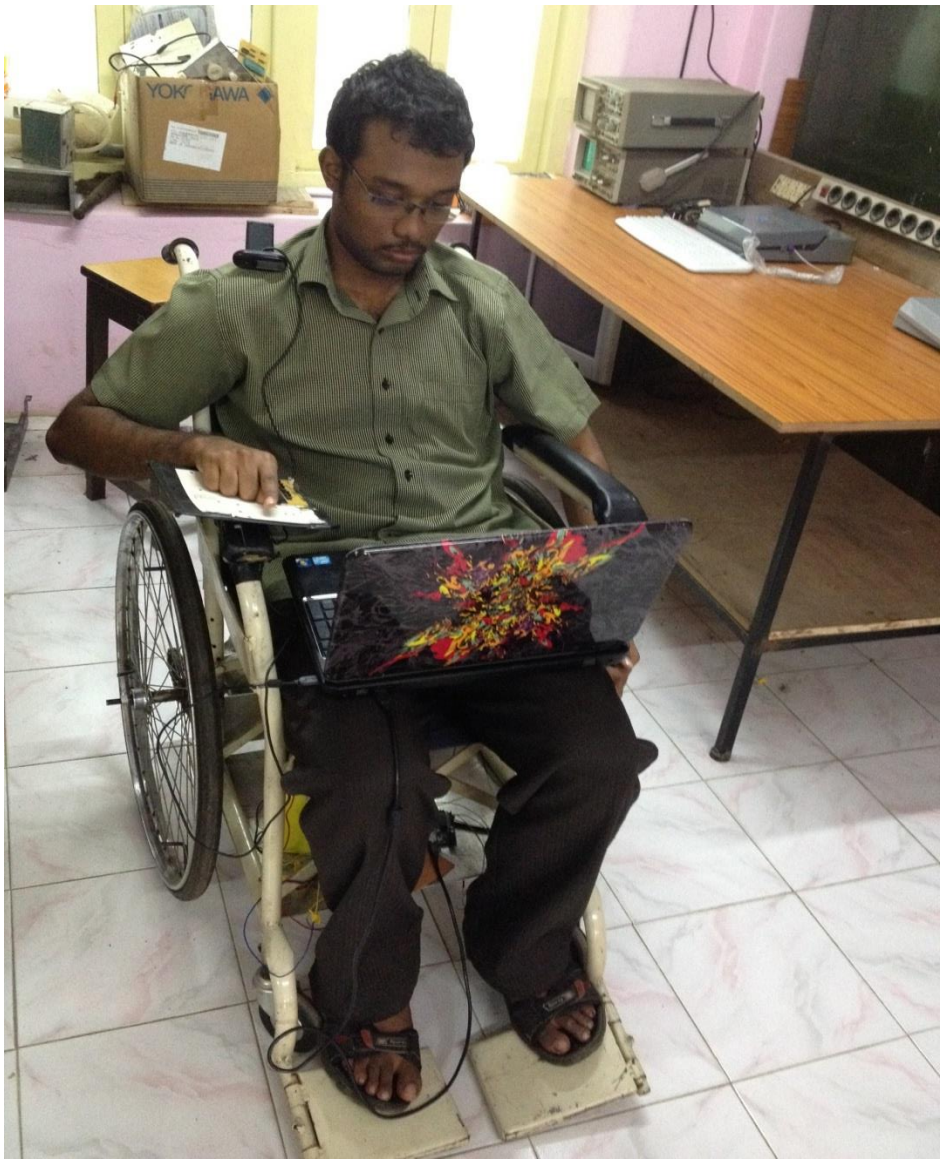


Fig 7

5.1 CONCLUSION

The above method employed was able to control both the arm and the wheelchair. In spite of certain absurdities in the logic, we were able to correct them and were able to perform the navigation and gripper movement using the arm successfully. The Arduino controller worked accordingly to give the right control signals and the actuators worked as expected.

5.2 FUTURE SCOPE THE PROJECT

Despite our success in the proposed work, there is still a lot of scope for improvement. The arm can be designed such that it not only picks and places objects for the patient but also feeds them. The wheelchair can be connected to a GPS system so that the patient's kith and kin know the location of the patient. Another gesture can be added which send an insecurity sign to the patient's immediate relatives. This may help them to prevent any untoward incident.

APPENDICES:

APPENDIX 1

HEADER PROGRAM:

```
#pragma once
#include "ofMain.h"
#include "ofxOpenCv.h"
class testApp : public ofBaseApp{
 public:
 void setup();
 void update();
void draw();
ofVideoGrabber vid;
ofxCvColorImage color;
ofxCvGrayscaleImage gr;
int h,w,times,timesb,tim;
double ges;
bool mode,stop;
double the;
unsigned char * pix;
ofSerial serial;
};
```

APPENDIX 2

PROGRAM:

```
#include "testApp.h"
//The gesreg function performs the recognition task and returns a double integer
double gesreg(unsigned char *pix,int w,int h)
{
 bool tog=0;
 long int sum=0,i=0,j=0;
  int avg=0,mat[240][3],m=0;
 double the=0;
 for(i=0;i<w*h;i++)
 sum+=pix[i];
 avg=sum/(i);
//Make it a black and white image
  for(i=0;i<w*h;i++)
 {
        if(pix[i]>(avg/1.5))
        pix[i]=255;
        else
        pix[i]=0;
  }
//Main algorithm starts here
for(i=0;i<h;i++)
{
        avg=0;sum=0;tog=0;m=1;mat[i][0]=0;
        for(j=0;j<w;j++)
        {
```

```
        if(pix[i*w+j]==tog*255 && ((j-sum)>5 || sum==0 ) && m<3 )
        {
        if(tog==0)
        {
        mat[i][0]++;
        }
        else
        {
        mat[i][m++]=(sum+j)/2;
        if((j-sum)>150)
        {
        mat[i][0]=10;
        }
        }
        sum=j;
        tog=!tog;
        }
        }
}
//Deriving the gesture from the matrix
sum=0;avg=0;
int stop=0,updwn=0,zero=0,real=0;
for(i=0;i<h;i++)
{
        if(mat[i][0]==1 && i<h/2)
        sum++;
       if(mat[i][0]==2 && i<h/2)
```

```
        avg++;
        if(i>h/2)
       {
       if(mat[i][0]==1)
       updwn++;
      else if(mat[i][0]==10)
      stop++;
     else if(mat[i][0]==2)
      real++;
     else
     zero++;
      }
 }
 if((h/2-sum) < 80)
{
    int temp=0;
    for(j=1;j<h/2 && mat[j][0]==1;j++)
{
        the+=atan2((float)(mat[j][1]-temp),1);
        temp=mat[j][1];
}
        the=the/sum;
        return the;
}
else if((h/2-avg) < 50)
        {
        return 4.0000;
```

```
        }
else if((real>updwn) && (real>stop))
        {   return 16.00000;
        }
else if(updwn>stop)
        {
          int temp=0,ff=1;
          the=0;
        for(j=h/2;j<h && mat[j][0]==1;j++)
        {      if(!ff)
        the+=atan2((float)(mat[j][1]-temp),1);
        temp=mat[j][1];
        ff=0;
        }
        the=the/updwn;
        return (the+12);
        }
        else if(updwn<stop)
        {
        return 5;
        }
        else
        {
        return 10;
        }
   }
```

```
//--------------------------------------------------------------
void testApp::setup()
{
	w=320;h=240;mode=0;ges=0,stop=1;times=7;timesb=0;tim=10;
	int baud = 9600;
	serial.setup("COM6", baud); //intialise serial communication
	vid.setVerbose(true);
	vid.listDevices();
	vid.setDesiredFrameRate(1);
	vid.initGrabber(w,h);
	color.allocate(w,h);
	gr.allocate(w,h);
}
//--------------------------------------------------------------
void testApp::update(){
	unsigned char getserial[3];
	memset(getserial,0,3);
	vid.grabFrame();
	if(vid.isFrameNew() && !(tim--))
	{
	tim=20;
	color.setFromPixels(vid.getPixels(), w,h);
	gr = color;
	pix=gr.getPixels();
	ges=gesreg(pix,w,h);
```

```
if(ges==5)
{
serial.writeByte('n');
stop=0;
}
if(!stop && !(times--))
{
times=7;
if(ges>-0.2 && ges <.2)
{
stop=1;
mode=0;
serial.writeByte(mode+48);
}
else if(ges==4)
{
mode=1;timesb=0;stop=1;
serial.writeByte(mode+48);
}
}
if(stop)
{
int alt=0;
 if(ges>11)
  {
  ges=ges-12;
  alt=1;
```

```
   }
   if(ges==4 && !(timesb--))
   {
   timesb=0;
   if(!alt)
 serial.writeByte('G');
   else
serial.writeByte('C');
   }
else if(ges>-0.2 && ges <.2)
{
timesb=0;
if(!alt)
serial.writeByte('S');
else
serial.writeByte('B');
}
else if(ges>-.45 && ges<-.2)
{
timesb=0;
if(!alt)
serial.writeByte('L');
else
serial.writeByte('U');
}
else if(ges>.2 && ges<.45)
{
```

```
        timesb=0;
        if(!alt)
        {
        serial.writeByte('R');
        }
        else
        {
        serial.writeByte('D');
        }
        }
        else if(ges==4)
        serial.writeByte('C');
        else if(ges==5)
        serial.writeByte('n');
        else
        serial.writeByte('o');
        }
        if(serial.available())
        {
        serial.readBytes(getserial,3);
        cout<<getserial;
        }
        gr.setFromPixels(pix,w,h);
        }
}
//--------------------------------------------------------------
void testApp::draw(){
```

```
        {
        gr.draw(w,h);
         ofFill();
        }
}
```

APPENDIX 3

7.3 ARDUINO PROGRAM :

```
#include <Servo.h>
#define lb .30
#define le .19
#define lw .28
#define gmp 12
#define gmn 5
#define mlp 4
#define mln 8
#define mrp 9
#define mrn 10
#define led 13
Servo sbas,sfst,ssec;
char ch='\0';
float c[3]={0};
int t=0,mode=0,stp=0,in=0,gbt=0;
float x=0,y=0,z=0;
void gost()
{
```

```
digitalWrite(mlp,HIGH);digitalWrite(mrp,HIGH);digitalWrite(mln,LOW);digital
Write(mrn,LOW);digitalWrite(led,HIGH);
}
void golt()
{
digitalWrite(mlp,LOW);digitalWrite(mrp,HIGH);digitalWrite(mln,HIGH);digital
Write(mrn,LOW);digitalWrite(led,LOW);
}
void gort()
{
digitalWrite(mlp,HIGH);digitalWrite(mrp,LOW);digitalWrite(mrn,HIGH);digital
Write(mln,LOW);digitalWrite(led,LOW);
}
void halt()
{
digitalWrite(mrp,LOW);digitalWrite(mlp,LOW);digitalWrite(mrn,LOW);digitalW
rite(mln,LOW);
}
void rev()
{
digitalWrite(mrn,HIGH);digitalWrite(mln,HIGH);digitalWrite(mrp,LOW);digital
Write(mlp,LOW);
}
void setup()
{
  Serial.begin(9600);
  pinMode(gmp,OUTPUT);
```

```
  pinMode(gmn,OUTPUT);
   pinMode(mlp,OUTPUT);
  pinMode(mln,OUTPUT);
  pinMode(mrp,OUTPUT);
  pinMode(mrn,OUTPUT);
  sbas.attach(3,544,2500);
  sfst.attach(6,544,2500);
  ssec.attach(7,544,2500);
  sbas.write(90);
  sfst.write(180);
  ssec.write(180);

}
void loop()
{
 if(Serial.available()>0)
  {
    ch=Serial.read();
Serial.write(ch);
 while(Serial.read()!=-1);
if(ch=='1' || ch=='0' || ch=='S' || ch=='R' || ch=='L' || ch=='U' || ch=='D' || ch=='B' ||
ch=='r' || ch=='n' || ch=='G' || ch=='C')
  {
  float st=.005;
  if(ch=='n')
  {
    stp=1;
```

```
  halt();
 }
 if(ch=='1' || ch=='0')
 {
  mode=ch-48;
  stp=0;
 }

 if(mode==0)
{
  switch(ch)
 {
  case 'S':gost();break;
  case 'R':gort();break;
  case 'L':golt();break;
  case 'B':rev();break;
 }
}

else if(mode==1)
 {
  if(in==0)
  {
  got(.04,.2,0);
  in=1;
  }
  getpos(&x,&y,&z);
```

```
switch(ch)
{
  case 'S':
  if(x<.30)
  got(x+st,y+st/3,z);break;
  case 'B':
  if(x>.04)
  got(x-st,y+st/3,z+st/4);break;
  case 'L':
  if(z>-.1)
  got(x,y+st/4,z-st);break;
  case 'R':
  if(z<.1)
  got(x,y+st/4,z+st);break;
  case 'U':
  got(x+st/4,y+st,z+st/6);break;
  case 'D':
 if(y>.01)
  got(x+st/4 ,y-st,z+st/6);break;
  case 'r':sbas.write(90);sfst.write(180);ssec.write(180);break;
  case 'G':
   {
          digitalWrite(gmp,HIGH);
            digitalWrite(gmn,LOW);
            delay(40);
            digitalWrite(gmp,LOW);digitalWrite(gmn,LOW);
    }break;
```

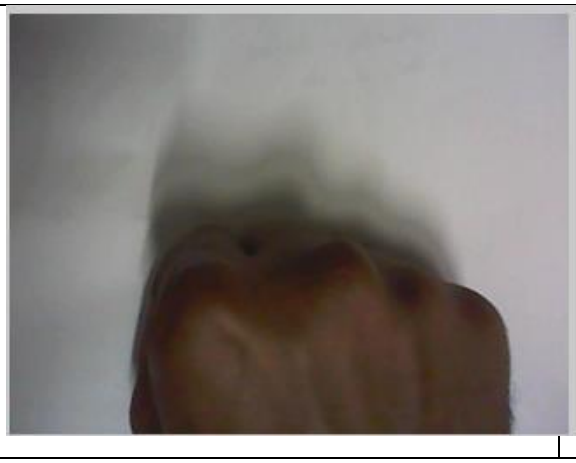
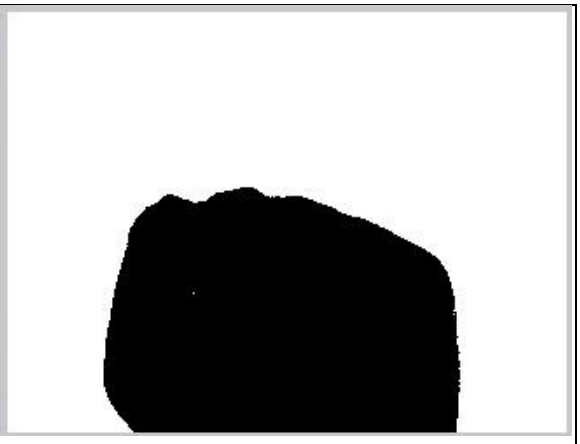
```
    case 'C':
            {
              digitalWrite(gmp,LOW);
              digitalWrite(gmn,HIGH);
              delay(40);
              digitalWrite(gmp,LOW);digitalWrite(gmn,LOW);
            }
  }
}
}
}
}
//function that makes gripper to move to a particular location
void  got(float x,float y,float z)
{
  float p=0,t=0,a=0;
 y=lb-y;
 if(z>=0)
 {
   a=(3.14-atan(x/z));
   x=x/sin(3.14-a);
 }
else
{
    a=-atan(x/z);
    x=x/sin(a);
}
```

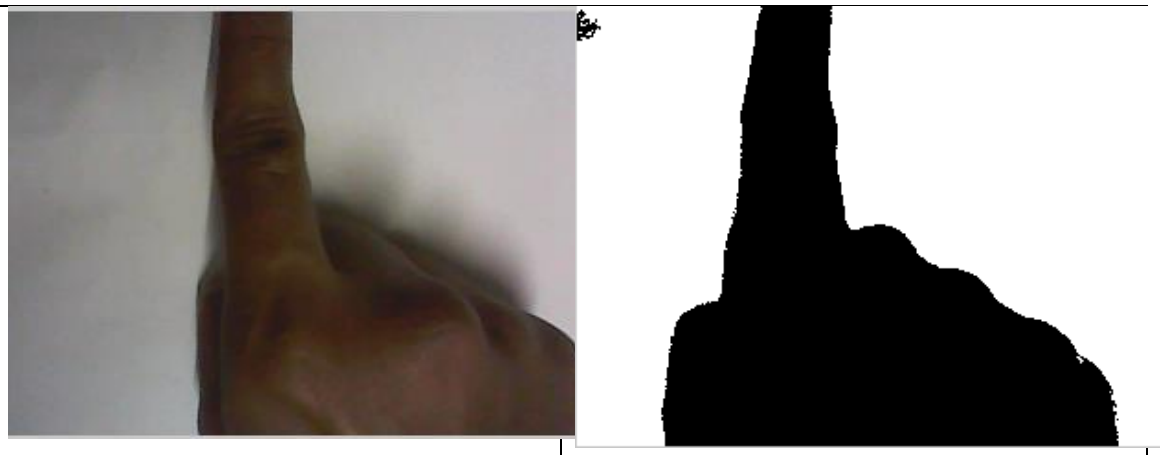
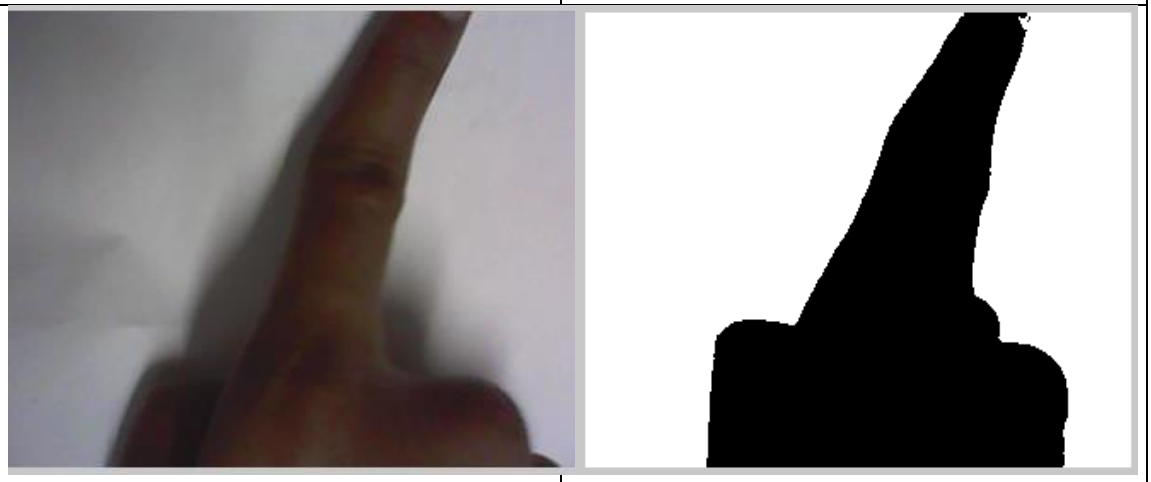
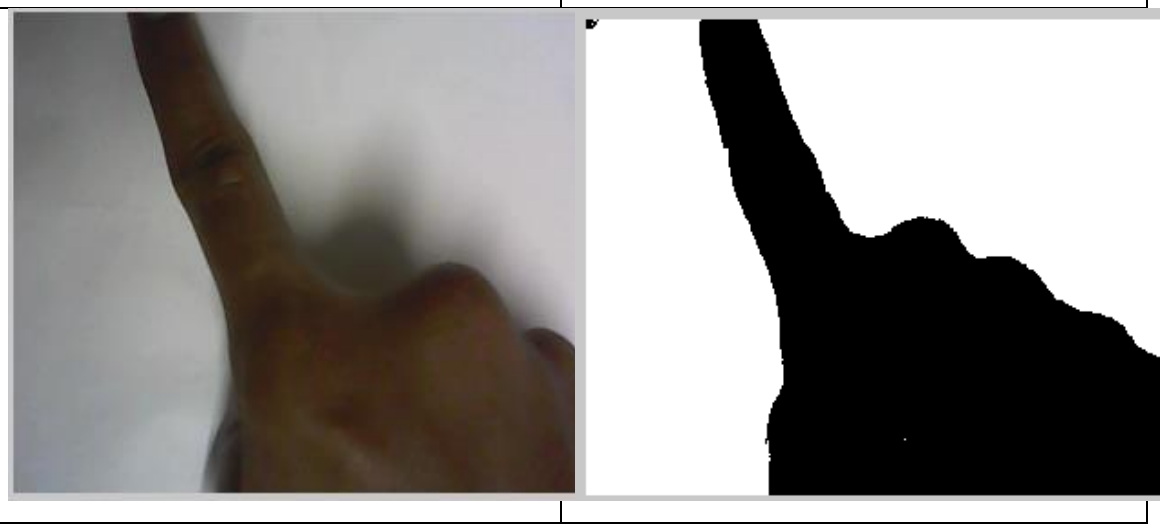
```
  a=(int)(a*180/3.14);
  float f=(le*le+lw*lw-x*x-y*y)/(2*le*lw);
  if(y>0)
  p=asin(x/(sqrt(pow((le-lw*f),2)+pow(lw*(cos(asin(f))),2))))-atan((le-
lw*f)/(lw*cos(asin(f))));
 else
{
p=acos(x/(sqrt(pow((lelw*f),2)+pow(lw*(cos(asin(f))),2))))+atan((lw*cos(asin(f))/
(le-lw*f)));
}
 t=(p-asin(f))*180/3.14;
 p=(int)(90+p*180/3.14);
 t=(int)(t-p+180)%360;
if(abs(a-sbas.read())>35 || abs((180-p)-sfst.read())>35 || abs(((180-t)-
sec.read()))>35)
  {
    int cnt=40;
while(((abs(a-sbas.read())>10 || abs((180-p)-sfst.read())>10 || abs(((180-t)-
ssec.read()))>10)) && cnt)
    {
    sbas.write(sbas.read()+(a-sbas.read())/30);
    sfst.write(sfst.read()+((180-p)-sfst.read())/30);
    ssec.write(ssec.read()+((180-t)-ssec.read())/30);
    cnt--;
    delay(200);
   }
   }
```

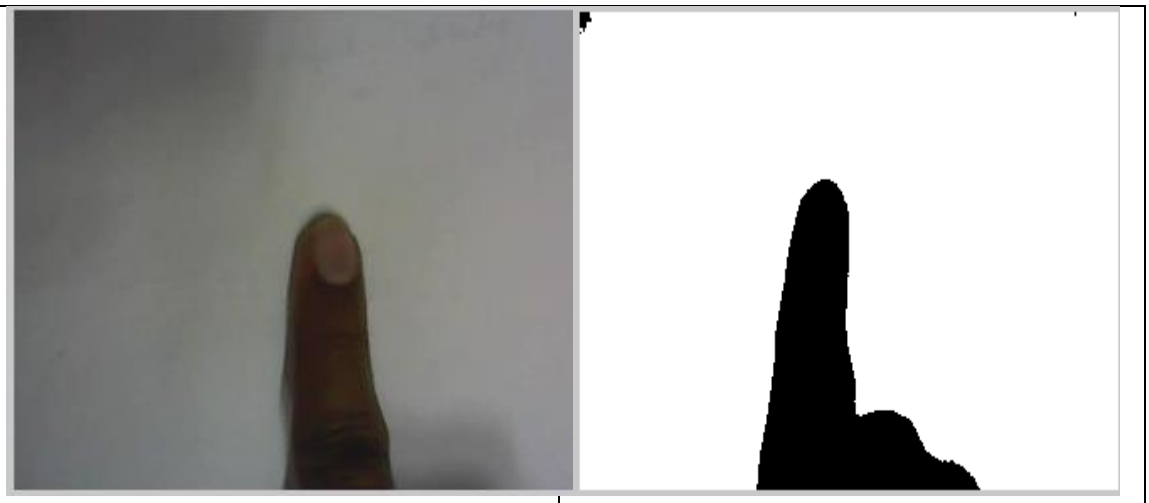
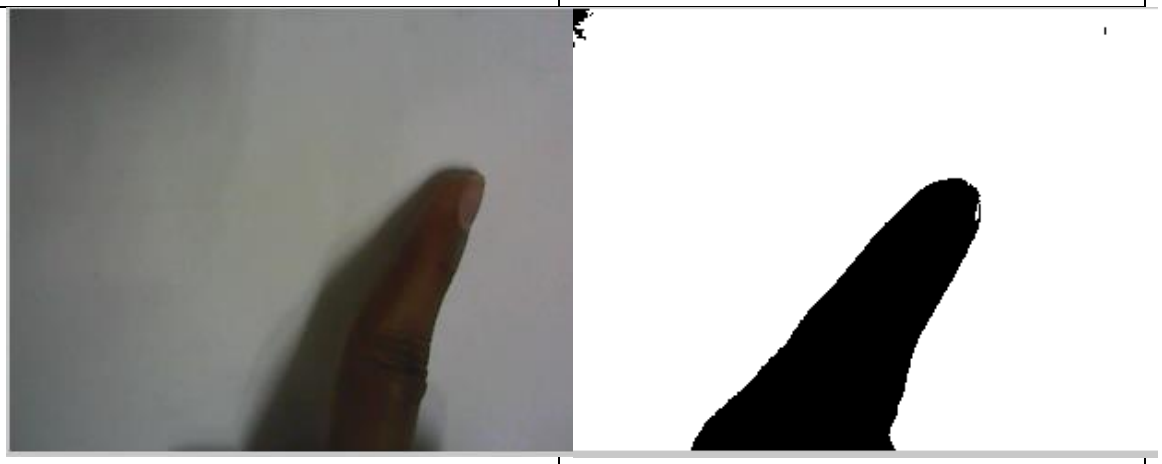
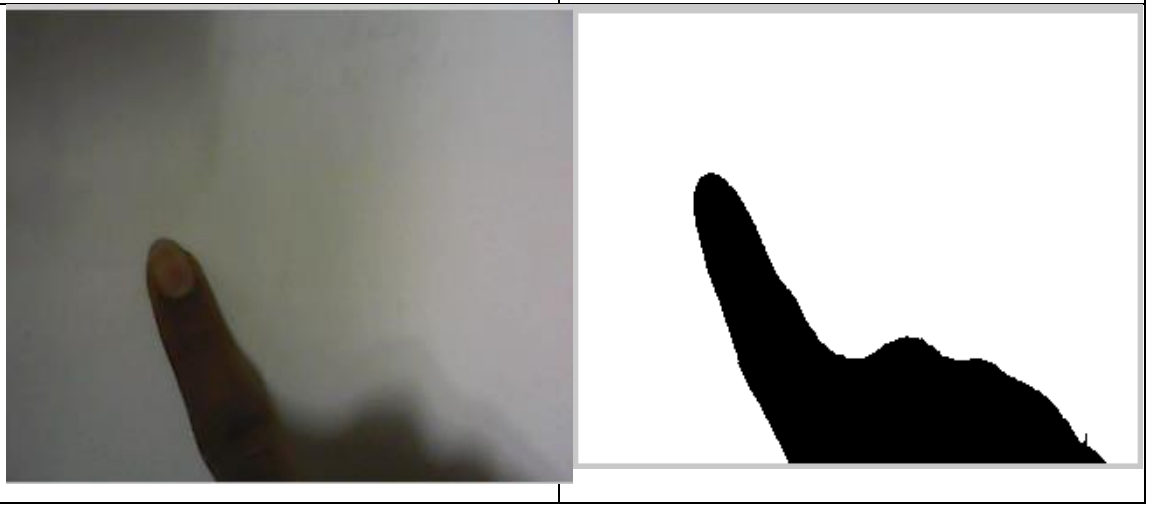
```
  sbas.write(a);
  sfst.write(180-p);
  ssec.write(180-t);
  }
//function to get current position of gripper
void getpos(float  *x,float *y, float  *z)
{
  int a=sbas.read(),p=180-sfst.read(),t=180-ssec.read();
*x=le*cos((float)(p-90)*3.14/180)+lw*sin((float)(t-90+(p-90))*3.14/180);
*y=lb+le*sin((float)(p-90)*3.14/180)-lw*cos((float)(t-90+(p-90))*3.14/180);
*z=cos((float)(180-a)*3.14/180)*(*x);
*x=sin((float)((180-a))*3.14/180)*(*x);
}
```

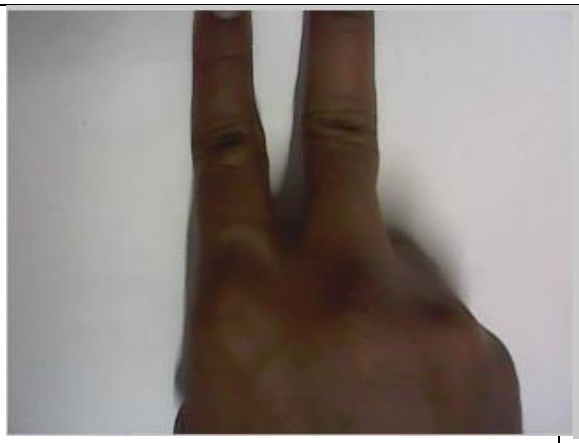
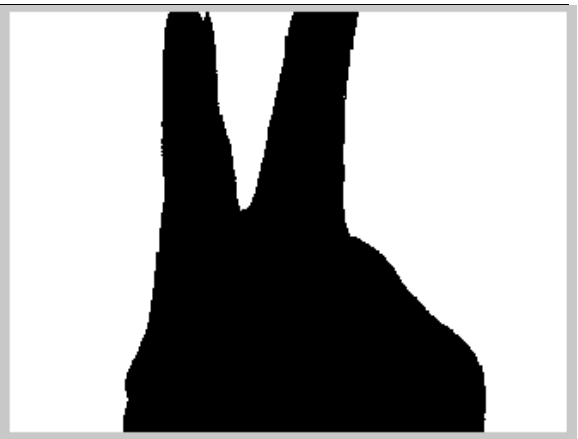
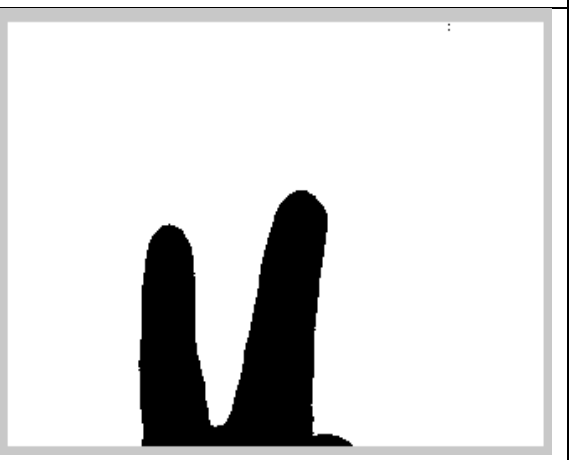
APPENDIX 4

GESTURES IN BLACK AND WHITE IMAGES:

STOP GESTURE		

GO STRAIGHT	
GO RIGHT	
GO LEFT	

REVERSE	
GRIPPER UP	
GRIPPER DOWN	

GRAB		
UNGRAB		

REFERENCES

1. P. Schrock, F. Farelo, R. Alqasemi, and R. Dubey (2009) "Design, Simulation and Testing of a New Modular Wheelchair mounted Robotic Arm to Perform Activities of Daily Living" -ICORR
2. Seong Pal Kang, "Virtual Human-Machine Interfaces and Intelligent Navigation of Wheelchairs" Thesis University of New South Wales. School of Mechanical and Manufacturing Engineering (2006).

3. Seong Pal Kang, Guy Rodnay, Michal Tordon,Jayantha Katupitiya (2003) ,“A Hand Gesture Based Virtual Interface for Wheelchair Control” IEEE conference
4. Yi Zhang, Jiao Zhang (2011) “A Novel Intelligent Wheelchair Control System Based On Hand Gesture Recognition” IEEE/ICME conference