UMBRA

VIT®
AP

MEMBERS INVOLVED:

1. Chainathan SS [18BCD7130]
2. Saswata Haldar [18BCD7096]
3. Naga Jayanth Chennupati [18BCN7084]
4. Mohammad Sadaf [18BCD7062]
5. Dasari Mohan Amritesh [18BCN7038]
6. Viswanadha Vedvyas [18BCE7190]

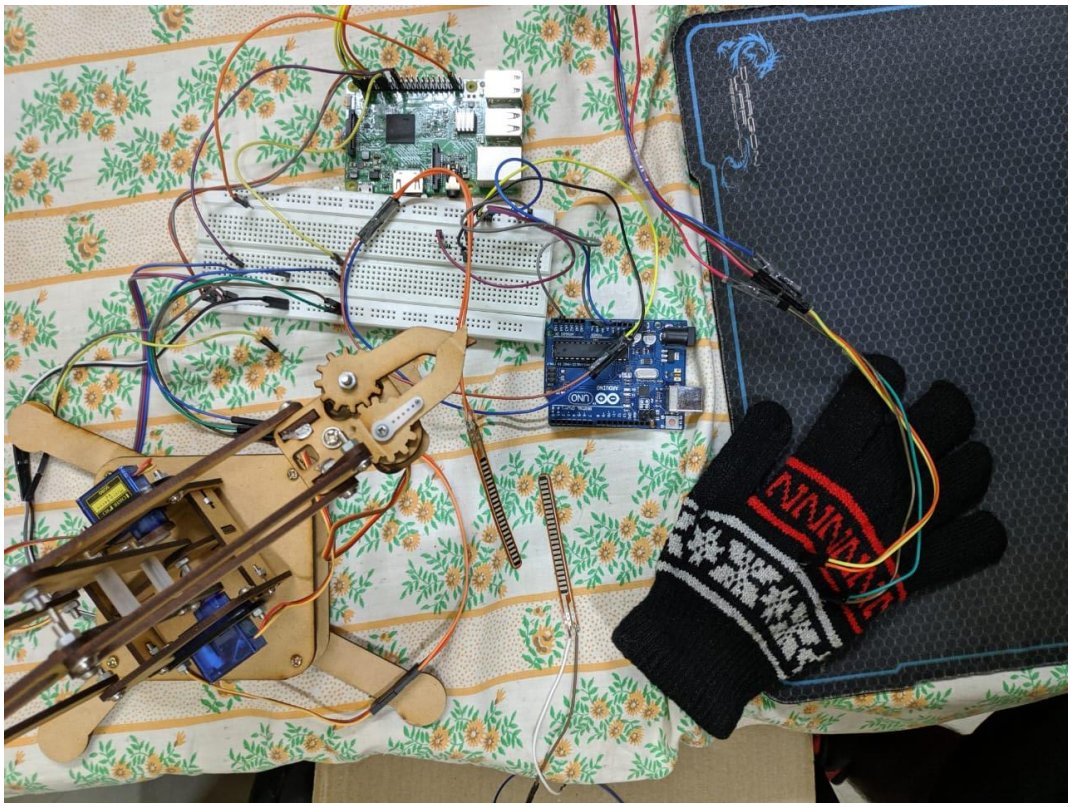
UNDER THE GUIDANCE OF:

Prof. Balu Laxman Parne

Department of Computer Science and Engineering.

ABSTRACT

- Robotics, a field as fascinating as it is revolutionary.
- One aspect of robotics that has demanded constant attention of enthusiasts is the mimicry of human gestures and maneuvers.
- Project UMBRA strives to achieve compete replication of the gestures and maneuvers of the human arm.
- Sensors read the gestures from a human hand and convey the same to a robotic arm assembled from hydraulics, motors and levers.
- The primary goal is to achieve real-time replication of gestures which may later be extended to storing one or more set of gestures to indefinitely replicate them.
- This increases the scope of application of the project by leaps and bounds.
- All in all the system will provide a great advantage over the existing emergency response models in a very consumer friendly and cost effective way.



CONTENTS

LIST OF FIGURES

INTRODUCTION

- We live in a society where automation and robotics are the new oil.
- When it comes to dealing with personal or institutional work, there's no need for one to physically do much work.
- For a long time now, automation has impacted our life more than we realize.
- Through the years, many businesses and institutions all over the world now have turned to automate their work mostly using robots to.
- As a result, all these organisations make sure that they deploy the best available resources possible.

Some of the identified core elements of the robotics and automation field are:

- The robotics revolution is rapidly accelerating, as fast paced technological advances in automation, engineering, energy storage, artificial intelligence, and machine learning converge. The result has transformed the capabilities of robots and their ability to take over tasks once carried out by humans.
- The number of robots in use worldwide multiplied three-fold over the past two decades, to 2.25 million. Trends suggest the global stock of robots will multiply even faster in the next 20 years, reaching as many as 20 million by 2030, with 14 million in China alone. The implications are immense, and the emerging challenges for governments and policymakers are equally daunting in their scale.
- The rise of the robots will boost productivity and economic growth. It will lead, too, to the creation of new jobs in yet-to-exist industries.
- Our device is one such device that employs and makes use of the modern technology to ensure the same. It's kind of a stand-alone device that's powered by a Raspberry Pi, an Arduino board, servo motors and a few more other components that help in the smooth working of the system.

Though at present our project has integrated everything that can be done in a given short span of time, we believe it still has the potential to incorporate more details and facilities, which will take our project to the next level with several other add-ons making the procedure of replication of human gestures way faster and easier to implement.

BACKGROUND

- The reason behind the success of this project resides with the six-member team that toiled days behind this project, trying to resolve the unforeseen issues that aroused from the same. Even though the project might seem to be simple in concept, the implementation requires a lot of coding for each component to be integrated.
- When the time came for the team to choose a topic for the project, the team picked this one because after all we believed that lack of an effective and cheap implementation of a robotic arm that could help reduce human effort was non-existent. And if there's something that we could do help improve this situation, it would definitely help to improve life in one way or another.
- So, then the whole team put their hearts and will to it, started from scratch, improvising solutions to all the problems and finally integrated all the components so as to deliver our promise.
- We are also extremely grateful and indebted to our project guide and all other friends who gave us a hand during the difficult times.
- The detailed facts and technical details regarding the project will be discussed in the following pages of the report.

PROBLEM DEFINITION

In some circumstances, close emulation of the human hand is desired, as in robots designed to conduct bomb disarmament and disposal. Robotic Arms are supposed to be used in places where the human mind can err. It is a basic human trait to be nervous around high tension situations. The lives of many depend on one man's actions and what if this one man is nervous or not ready to deal with this situation. These Robotic Arms can be controlled from afar, hence removing the need for the presence of a human personnel at the situation. Robotic arms can also used in places where it is not safe for a human to work, for example inside a nuclear power plant. Robotic Arms can be used as a useful tool for a surgery, where precise hand movements are required. When the robotic arm is attached to an RC car, it can reach places where humans cannot. The cost of Robotic Arms is still very high when compared to their use cases. The only reason why this is not a wide-spread implementation is because of its very high price. Through this project, we are trying to make a fully functional Robotic Arm which can be controlled by our Hand Gestures while still trying to remain cost-effective.

OBJECTIVES

- To map hand gestures using data from Flex sensor and Accelerometer.
- Processing of data from Flex and transferring data to Raspberry pi using Arduino.
- Predefined instructions to Servo motors based on input using Raspberry pi.
- Integrating the code and the components effectively.
- Reducing time latency.
- Present the prototype.

PROCEDURE/METHODOLOGY

1. CIRCUIT BUILDING

- In the beginning gather all the components i.e. Arduino Uno, Raspi, Servo motors, Flex sensors, Accelerometer, Breadboard or PCB, Connecting jumper wire, Robotic arm, Power supply.
- Connect the accelerometer's VCC, ground, SDA and SCL pins to raspberry pi's 5v, ground, SDA and SCL pins respectively.
- Next connect the servo motors' power, ground and input pins to raspberry pi. And then connect flex sensor's ground and output pin to Arduino's ground and analog pin with a resistor in between. Finally connect Arduino to raspberry pi through the usb cable and provide raspberry pi power supply.

2. CODING

- Open Raspi OS and write the code.
- Compile the code.
- Upload the code to the board by clicking on the upload button and connecting the board to one of the USB ports on the pc. Ensure that you have selected the correct port.
- Once the code is uploaded the prototype will be active.
- Allow the module to stabilize.
- The system should be running well and the robotic arm should be able to replicate the gestures.

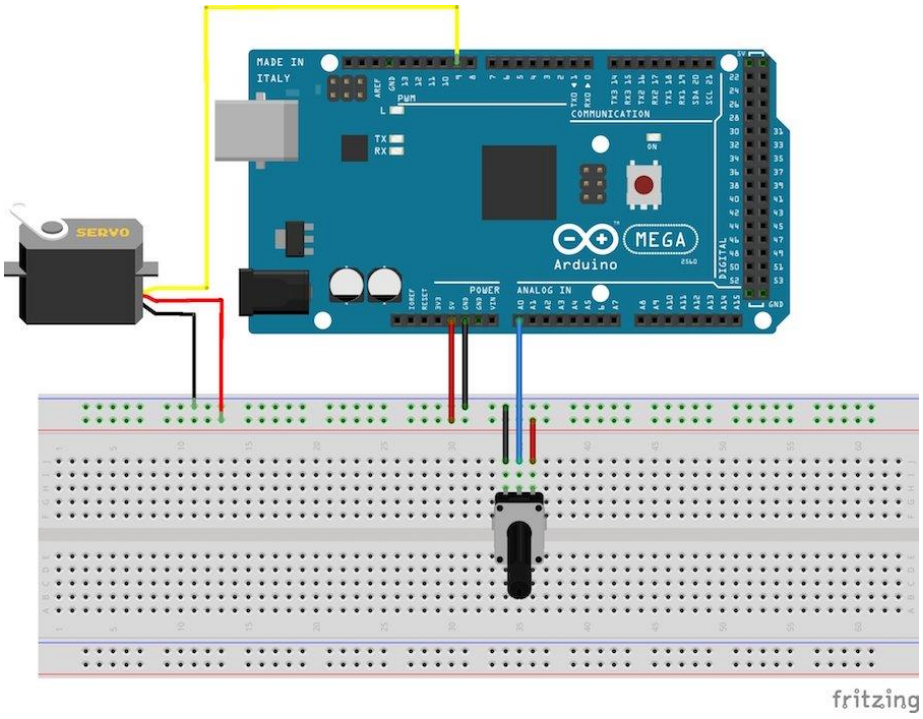
3. RUNNING THE SYSTEM

- The provided glove is the driver of the project.
- The user needs to be familiar with the pre-defined gesture to be able to interact with the prototype.
- Once the user is familiarized with the gestures, the robotic arm can be made to replicate the gestures.

4. COMPONENTS EXPLAINED

1. Servo Motor

- Servo systems include three primary components: a motor, a drive (also referred to as an amplifier), and a feedback mechanism. Also typically included are a power supply and a servo controller capable of controlling either a single axis or coordinating the motion of multiple axes.
- Servo motors can be either AC or DC types, with AC servos being more suitable for constant speed applications and DC servos for variable speed applications. DC servo motors can also be either brushed or brushless.
- A servo drive amplifies the signal from a master controller to provide sufficient current (power) to the motor to generate speed and produce torque. In a rotary motor, current is proportional to torque, so the servo drive directly controls the torque produced by the motor. Similarly, in a linear motor, current is proportional to force, so the drive controls the force produced by the motor.
- The work of the module in our project is to replicate and control the motion of the robotic arm.



2. Arduino

- The Arduino uses ATmega328 microcontroller which requires a working voltage of 5V. But the input voltage can vary between 7V to 20V.There are 14 digital pins in Arduino and 6 Analog pins. There are two power ports, one of 5V other of 3.3V. It has 3 GND pins.
- The Arduino is capable of collecting data and do some calculations using that data and give output has per the result of the function and conditions written in the code. The programming for Arduino is done in Arduino IDE which is a free software.

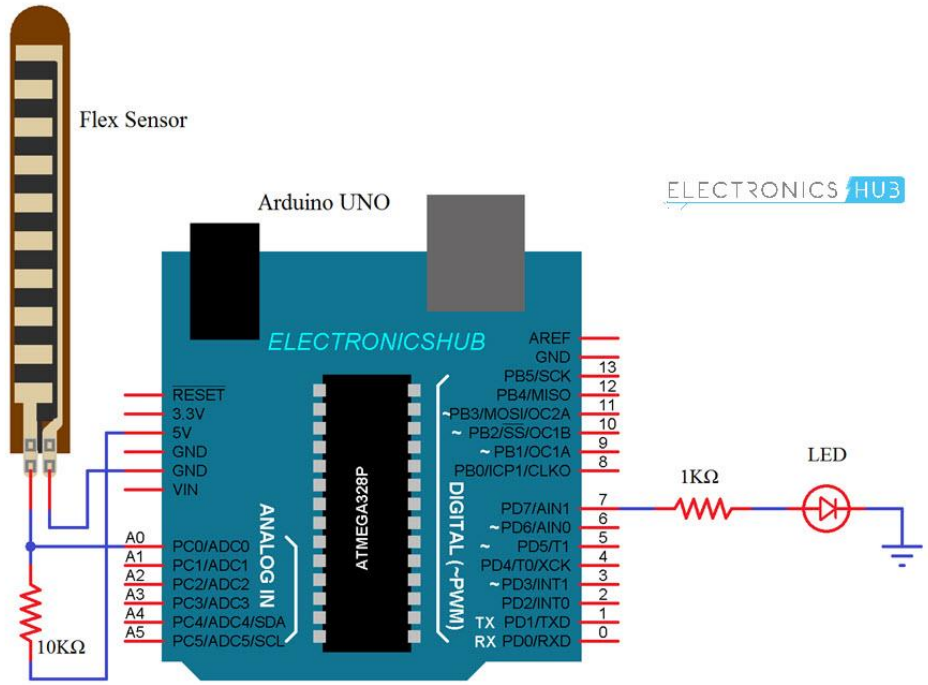
- In our project the Arduino is used to control the GSM module, the LCD display and the keypad and it helps in sending and receiving a message or calling.

3. Raspberry Pi

- The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse.
- It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python.
- It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

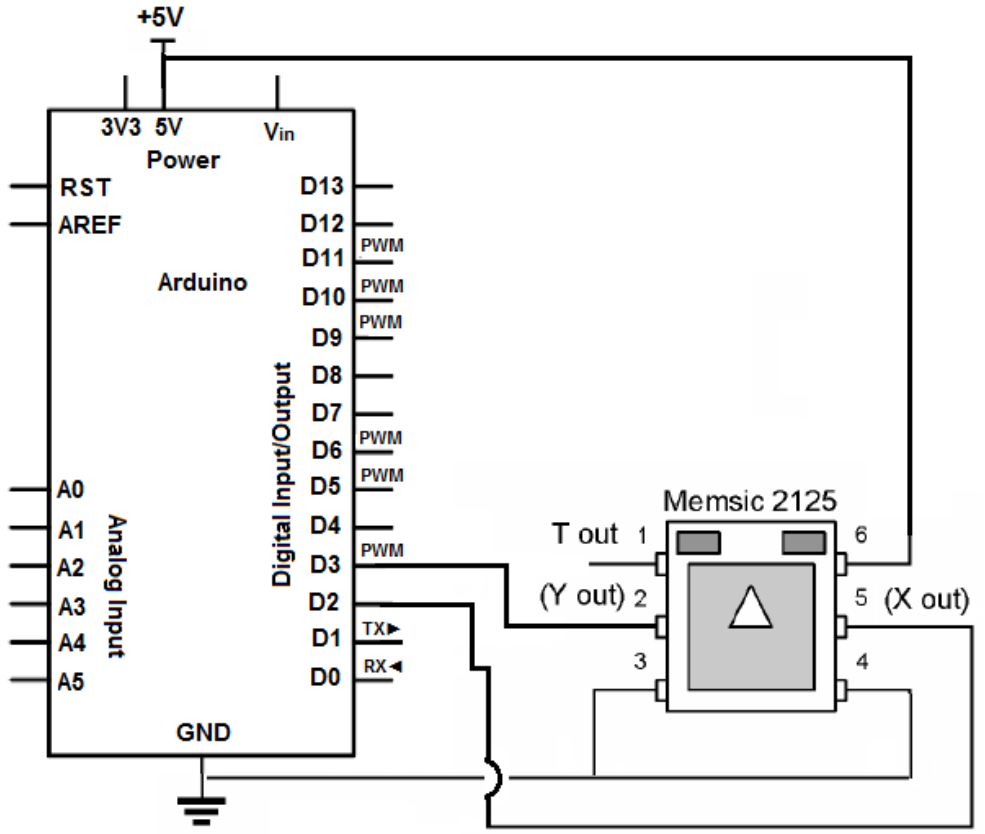
4. Flex Sensor

- A flex sensor or bend sensor is a sensor that measures the amount of deflection or bending.
- Usually, the sensor is stuck to the surface, and resistance of sensor element is varied by bending the surface.
- Since the resistance is directly proportional to the amount of bend it used as goniometer, and often called flexible potentiometer.



5. Accelerometer

- An accelerometer is an electromechanical device used to measure acceleration forces. Such forces may be static, like the continuous force of gravity or, as is the case with many mobile devices, dynamic to sense movement or vibrations.
- The accelerometer consists of many different parts and works in many ways, two of which are the piezoelectric effect and the capacitance sensor.
- The piezoelectric effect is the most common form of accelerometer and uses microscopic crystal structures that become stressed due to accelerative forces.
- These crystals create a voltage from the stress, and the accelerometer interprets the voltage to determine velocity and orientation.



6. Libraries

5.1 Software serial library

- The Arduino hardware has built-in support for serial communication on pins 0 and 1. But two pins are not enough when we use many modules at once. To overcome this problem the SoftwareSerial library has been developed to allow serial communication on other digital pins ofthe Arduino, using software to replicate the functionality (hence the name "SoftwareSerial").
- It is possible to have multiple software serial ports with speeds up to 115200 bps.

In our project we use a speed of 9600 bps for the include flex sensor module.

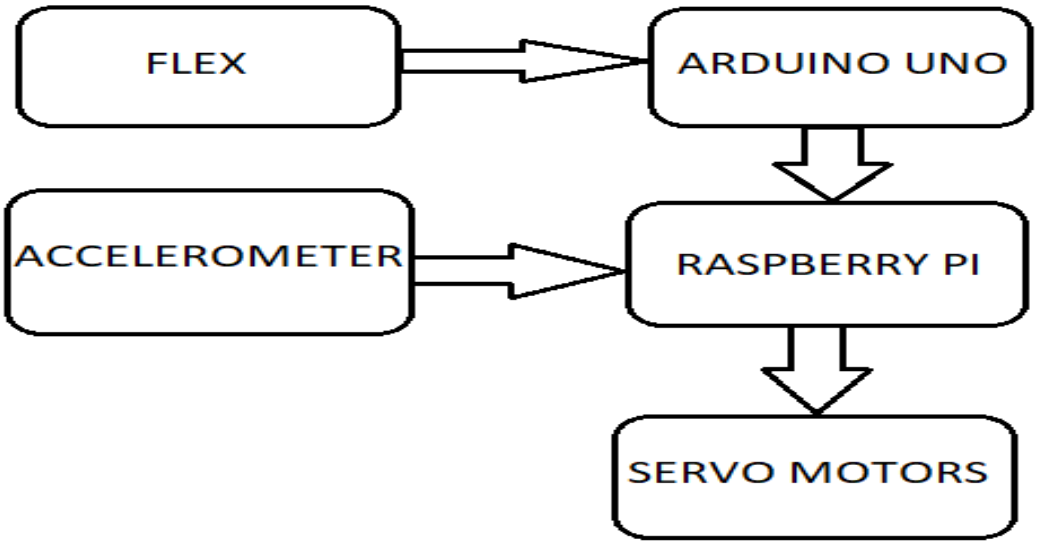
5.2 adafruit_adxl34x

- This library is a driver for the ADXL34x 3-axis accelerometer family. It has been developed by Bryan Siepert based on drivers by K. Townsend and Tony DiCola
- This library allows the Raspberry Pi to take input from the 3-axis accelerometer.
- This library when used with proper coding experience will return x, y and z axis accelerometer readings.

5.3 pigpio Library

- pigpio is a library for the Raspberry which allows control of the General Purpose Input Outputs (GPIO).
- pigpio works on all versions of the Pi.
- The Stepper Motors that we were using for our project worked on clock pulses, so the movements were very jittery.
- This pigpio on the other hand fixed that problem for us.
- This is the main library we used to configure all the GPIO ports.

Architectural Diagram:



RESULT AND DISCUSSIONS:

- The results of this project include the working of the servo motors after successfully fetching the data from the flex and accelerometer and feeding them to the motors.
- Even though we faced a bit of difficulty at first, the flex sensor, the accelerometer, the servo motors, run in unison with our robotic arm. We were able to achieve the gesture capturing and replicating functionality in our project.

Objectives Met:

1. Interlinking of the Arduino, Raspi and the Robotic Arm.
2. The passing of analog and digital sensor information from Arduino and Raspi to the main arm.
3. Reducing the latency time.
4. Setting multiple gestures for multiple actions.
5. Combining the circuit and code to work together in sync with each other.

Objectives Not-Met:

1. The robotic consists only of a pincher instead of a whole complete hand.

We had to re-configure the connections of the Arduino and Raspberry pi to so as to get a module working unison with the arm.

CONCLUSION AND FUTURE SCOPE

- The present study has been made to suggest and develop some tools which will eventually be useful to individuals, institutes, homes and any other space with human habitation to provide an alternate to human labor system at a reasonable cost and of a specified quality.
- This project is targeted towards financing, designing, implementing and operating robotic arm facilities and services that are traditionally expensive and complicated.
- As this project is simple to operate anyone with some basic knowledge of computers can operate this.
- Also because the project is not dependent on the internet the system will be always active independent of the internet and the fact that the system can run on any energy source (requires very low energy) largely reduces maintenance.
- As raspberry pi boards and components are the main building blocks of this system, this product is highly cost-effective and easy to mass produce.

That being said the current prototype can be highly improved in the coming years with more R&D.

- Better servo motors which do not operate on pulses if used will reduce the unwanted vibrations.
- Enable user to customize the functioning of the product as per requirement.
- A full hand can be developed in place of just a pincher.
- The model can be modified to work wirelessly.
- The model can be mounted upon a small RC vehicle which will allow it be portable and used over a larger area of operation.

REFERENCES

- https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/
- https://www.researchgate.net/publication/320424417_Gesture_Control_Robotic_Arm_Using_Flex_Sensor
- https://www.ijcaonline.org/archives/volume173/number4/28323-2017915285
- https://www.raspberrypi.org/
- https://github.com/

APPENDIX

CODE:

```
import time
import Adafruit_ADXL345
import RPi.GPIO as GPIO
import serial as S1
import math
import pigpio

acceServoy=26
acceServox=17
acceServoz=23

boo=0


ser = S1.Serial('/dev/ttyACM0', 9600)


def choose():
    if(fin_bent()==1):
        SetAngle2(y1*100,acceServoz,piz)
    else:
        SetAngle3(y1*100,acceServoy,piy)
        SetAngle2(x1*100,acceServox,pix)



def fin_bent():
    line = ser.readline()
    for s in line.split():
        #if s.isdigit():
            f=int(s)
            if(f>(-600)):
                return 1
            else:
                return 0



def SetAngle2(angle,servoPin,pii):
    boo=0
    if(angle>157.0) :
       boo=150
    elif(boo<(-157.0)):
        boo=-150
    else:
        boo=translate(angle,-157,157,500,2500)

    pii.get_mode(servoPin)
    pii.set_servo_pulsewidth(servoPin, boo)
    pii.get_servo_pulsewidth(servoPin)
    return
```

```
def SetAngle3(angle,servoPin,pii):
    boo=0
    if(angle>157.0) :
       boo=150
    elif(boo<(-157.0)):
        boo=-150
    else:
        boo=translate(angle,-180,180,1500,500)

    pii.get_mode(servoPin)
    pii.set_servo_pulsewidth(servoPin, boo)
    pii.get_servo_pulsewidth(servoPin)
    return
def calc_xy(x,y,z):
    x2=x*x
    y2=y*y
    z2=z*z

    #res3 = math.sqrt(x2+y2)
    #res3=res3/z
    #accel_z=math.atan(res3)

    try:
        res=math.sqrt(y2+z2)
        res=x/res
        accel_x=math.atan(res)

        res1=math.sqrt(x2+z2)
        res1=y/res1
        accel_y=math.atan(res1)
    except:
        print('lol')
    finally:
        return accel_x,accel_y#,accel_z

def translate(value, leftMin, leftMax, rightMin, rightMax):
    # Figure out how 'wide' each range is
    leftSpan = leftMax - leftMin
    rightSpan = rightMax - rightMin

    # Convert the left range into a 0-1 range (float)
    valueScaled = float(value - leftMin) / float(leftSpan)

    # Convert the 0-1 range into a value in the right range.
    return rightMin + (valueScaled * rightSpan)

piy = pigpio.pi()
piy.set_mode(acceServoy, pigpio.OUTPUT)

pix = pigpio.pi()
pix.set_mode(acceServox, pigpio.OUTPUT)

piz = pigpio.pi()
piz.set_mode(acceServoz, pigpio.OUTPUT)

accel = Adafruit_ADXL345.ADXL345()

while True:
```

```
    x, y, z = accel.read()
    print('X={0}, Y={1}, Z={2}, '.format(x, y, z))
    x1,y1=calc_xy(x,y,z)
    print(x1*100)
    #print(y1*100)
    #print(z1*100)
    #xx=translate(x,-220,190,0,180)
    #zz=translate(y,-35,0,0,180)
    #yy=translate(y,-240,230,0,180)
    #print(yy)
    #zz=translate(z,-40,45,0,180)
    #print('Y={0}'.format(yy))
    #print('X={0}'.format(xx))
    #print('Z={0}'.format(zz))
    #time.sleep(0.2)
    choose()
    #SetAngle3(x1*100,acceServox,pix)
    choose()
    #SetAngle2(y1*100,acceServoz,piz)
    #SetAngle2(x1*100,acceServox,pix)
    #time.sleep(0.06)
    #translate(z,-20,45,0,180)
    #piy.set_servo_pulsewidth(acceServoy,1000)
    #pix.set_servo_pulsewidth(acceServox,500)
```