Haptic Glove to Control Pointer

Project Report
Electronics System Design

By

Parshwa Shah (1641068)
Maharsh Suryawala (1641029)
Yash H Patel (1641044)

Motivation:

Traditional methods of input like mouse have been used since a long time but they lack a sense of immersion and are also not really easy to use. With computers being so widely used now these traditional inputs need to be improved. Millions of old people in retail as well as other sectors find it really difficult to use mouse either due to old age or disabilities. New method of input will also help people with arthritis that cannot use the mouse effectively. With virtual reality (VR) and augmented reality (AR) growing rapidly in popularity, computers can finally escape the physical limitations of a flat screen and make the leap into the depth of a truer third dimension and hence a haptic mouse pointer can be a input device for controlling the futuristic computers.

Solution:

Because of limitations of current systems as mentioned above we are trying to make a wearable haptic glove which would provide easy access to PC and its surrounding environment and which would feel even more real. A normal computer has input devices such as keyboard and mouse and an output device i.e. The monitor. In futuristic computers, a monitor will most likely to be replaced by an AR headset and input devices such as keyboard will be replaced by microphone(using speech recognition) and the mouse with a haptic pointer device. A haptic pointer device would change user's interface with his computer, as he can control all pointing and scrolling based on the orientation of the finger and contact between fingers i.e. our finger can be used to control cursor of our laptop.

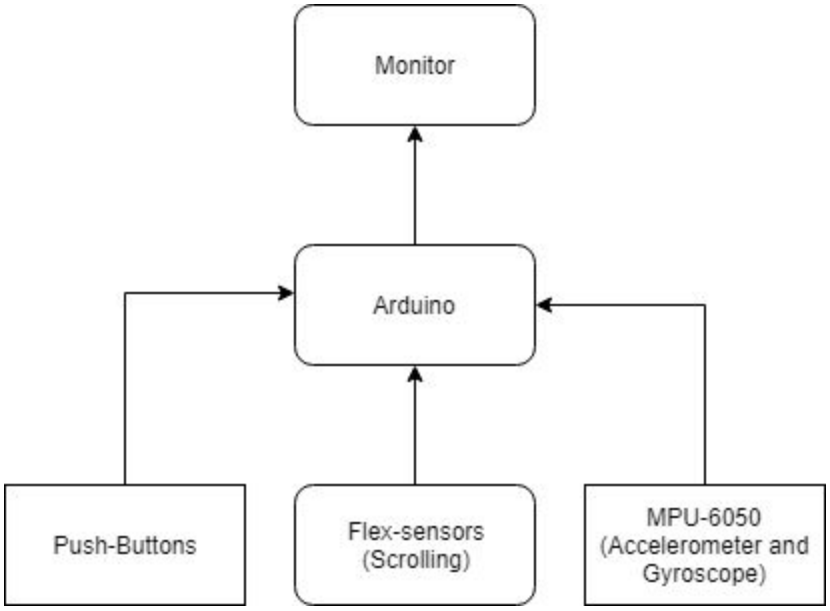
Description:

The circuit will comprise of the MPU-6050 sensor being powered by the arduino through the Vcc pin. Vcc will be connected to a 10k ohm resistor then that resistor will be connected to the push button. This completes the base circuit. This circuit will then be placed on a glove that will be used for input.
As the user moves his finger ie points on the screen at different positions the cursor(pointer) would move and the press button is used to give the effect of switches flex sensors would give the effect of scrolling.

Final Outcome:

The final project will comprise of a glove that can be used to control a computer's mouse functions without needing a surface or even any wires. You can control mouse movement by waving your hand around and control left click, right click , scroll up and down by flexing each of your fingers. This glove will work with any pc and even functions as bluetooth device for android phones and tablet.While we can connect it with AR applications to connect it with real world. This mouse can also be used for gaming purposes so that the user can get a better experience.

Block Diagram



Components Required:

1. 1 x Arduino Uno
2. 1 x MPU-6050
3. 1 x Glove
4. 1x Mini usb to usb cable
5. Jumper cables
6. 1 x 10K ohm resistor
7. 1 x BreadBoard
8. 1 x Push button
9. 1 x Flex sensor

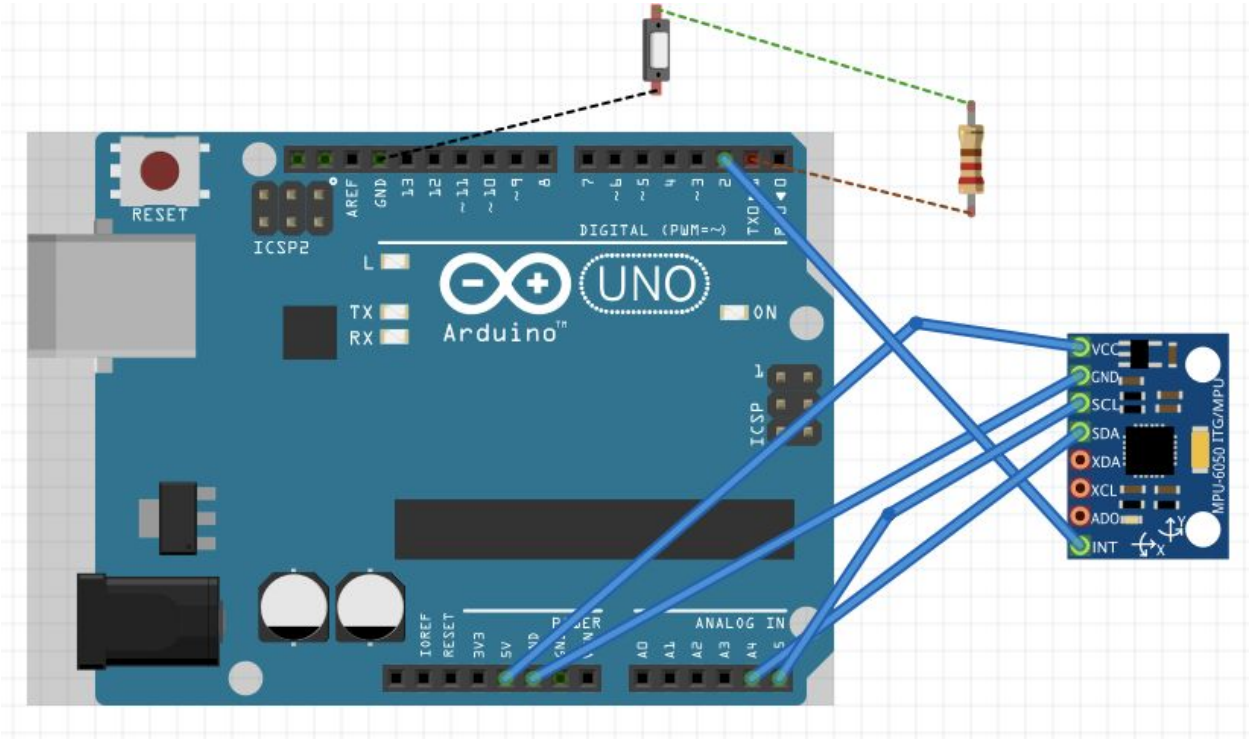
Selection Criteria of Components:

Why use MPU-6050 ?
MPU-6050 sensor contains a MEMS accelerometer and a MEMS gyro in a single chip. It is very accurate, as it contains 16-bits analog to digital conversion hardware for each channel. Therefor it captures the x, y, and z channel at the same time. The sensor uses the I2C-bus to interface with the Arduino.
The MPU-6050 is not expensive, especially given the fact that it combines both an accelerometer and a gyroscope which would fulfill all our requirements.

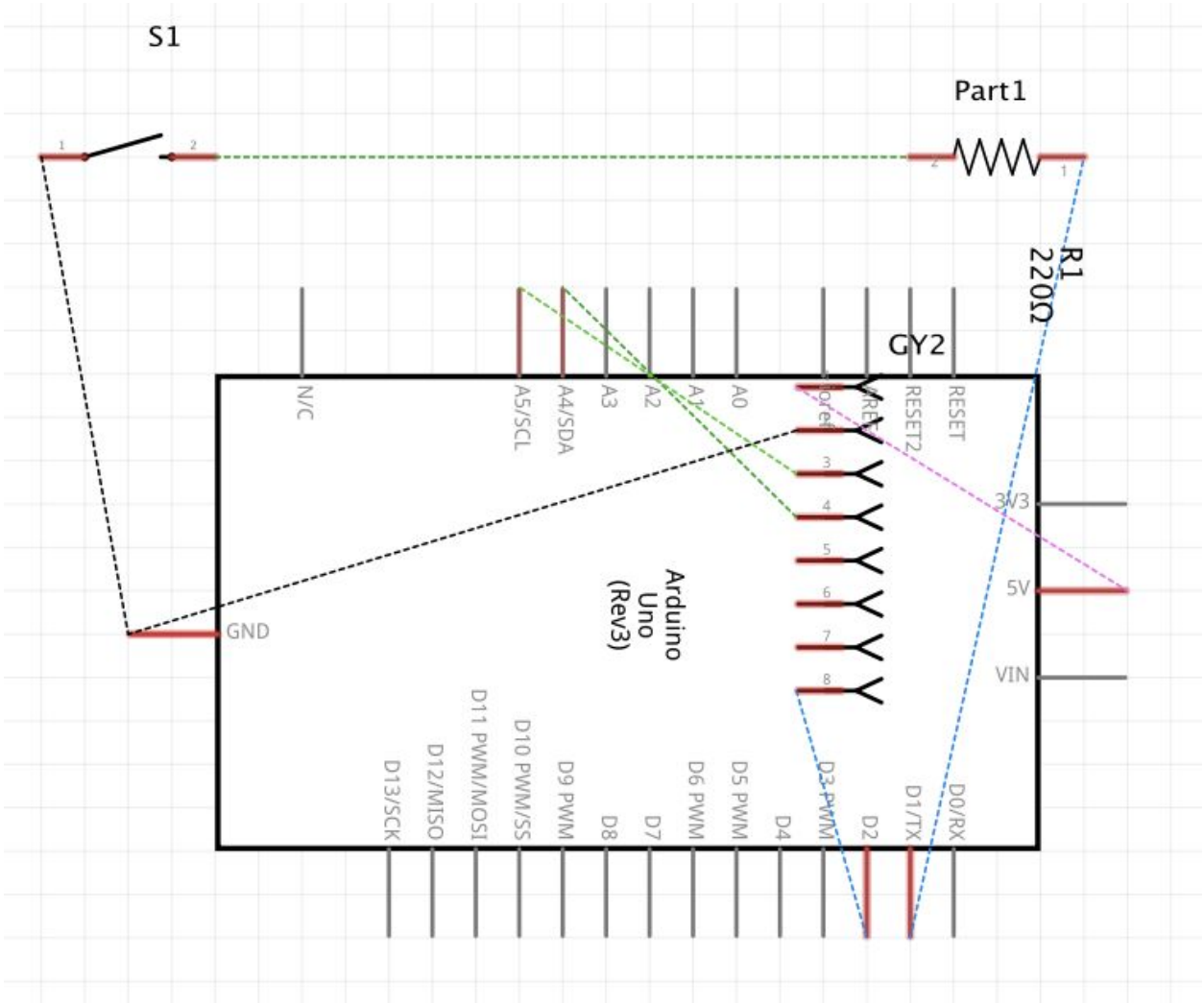
Why Flex Sensors?
They are use to get the moment of finger of the user so we can select about how much amount we can scroll the screen.

Circuit Diagram:

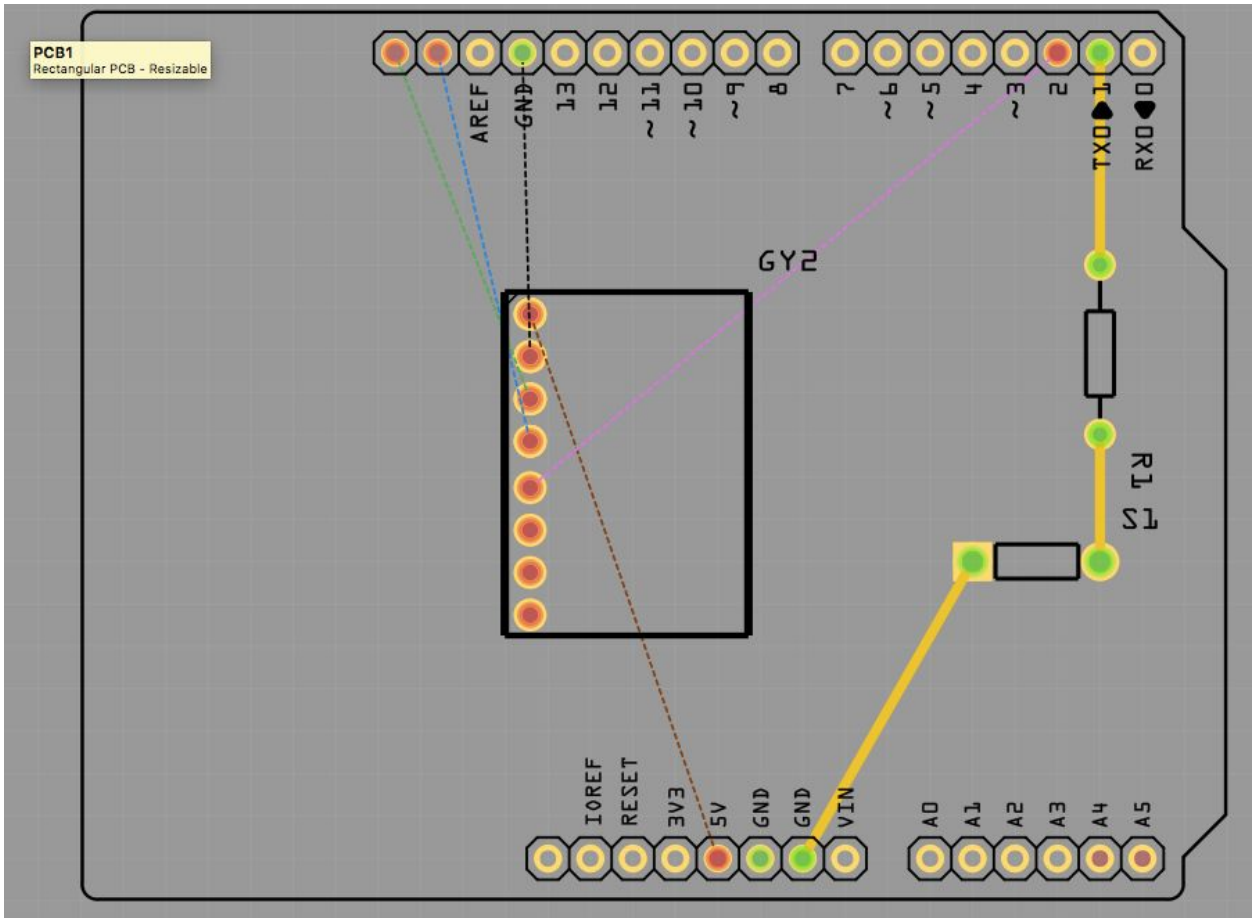
CIrcuit:



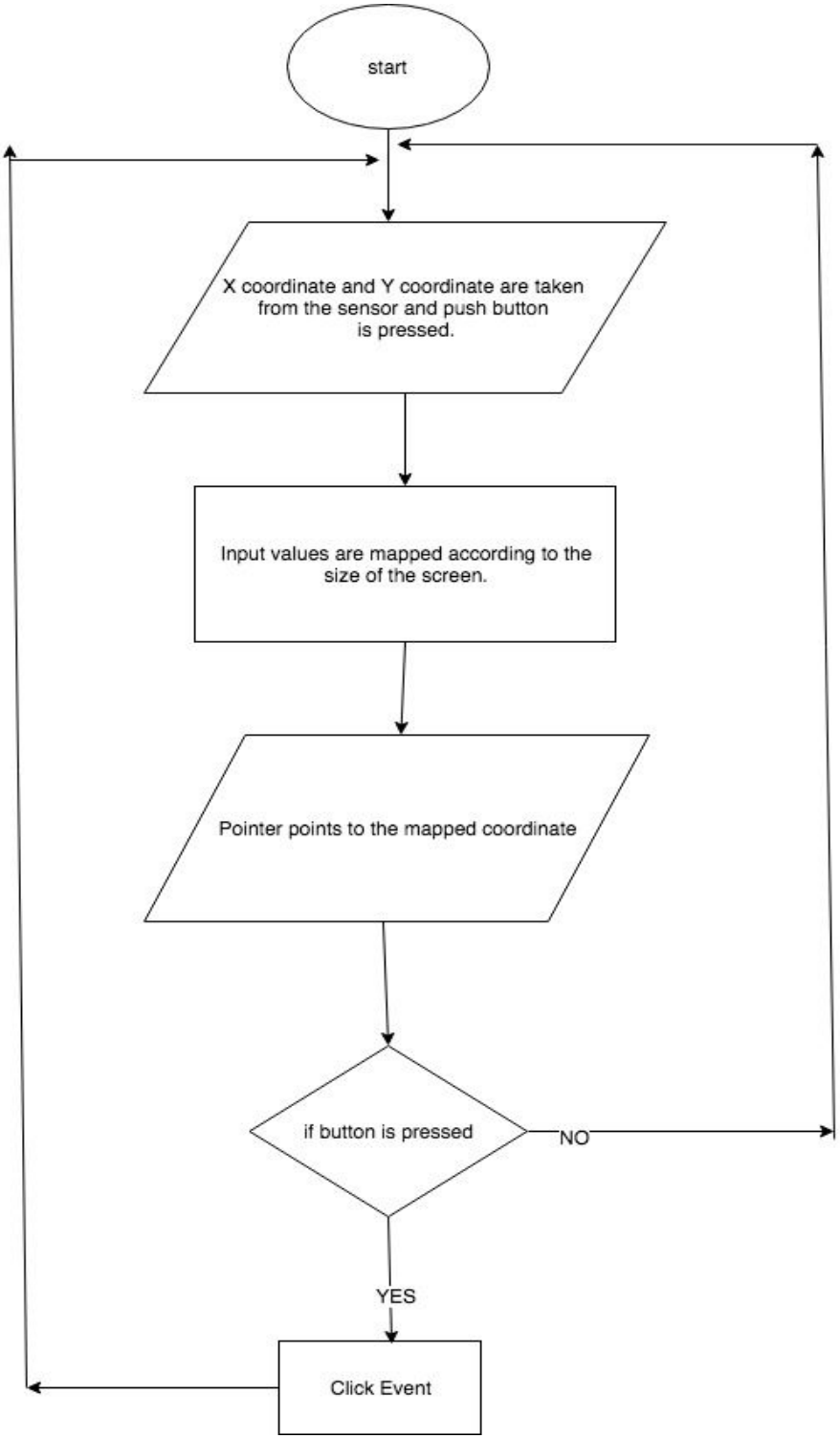
Schematic:


S1
Part1
R1
220Ω
GY2
N/C
A5/SCL
A4/SDA
A3
A2
A1
A0
AREF
RESET2
RESET
3V3
5V
VIN
GND
Arduino
Uno
(Rev3)
D13/SCK
D12/MISO
D11 PWM/MOSI
D10 PWM/SS
D9 PWM
D8
D7
D6 PWM
D5 PWM
D4
D3 PWM
D2
D1/TX
D0/RX

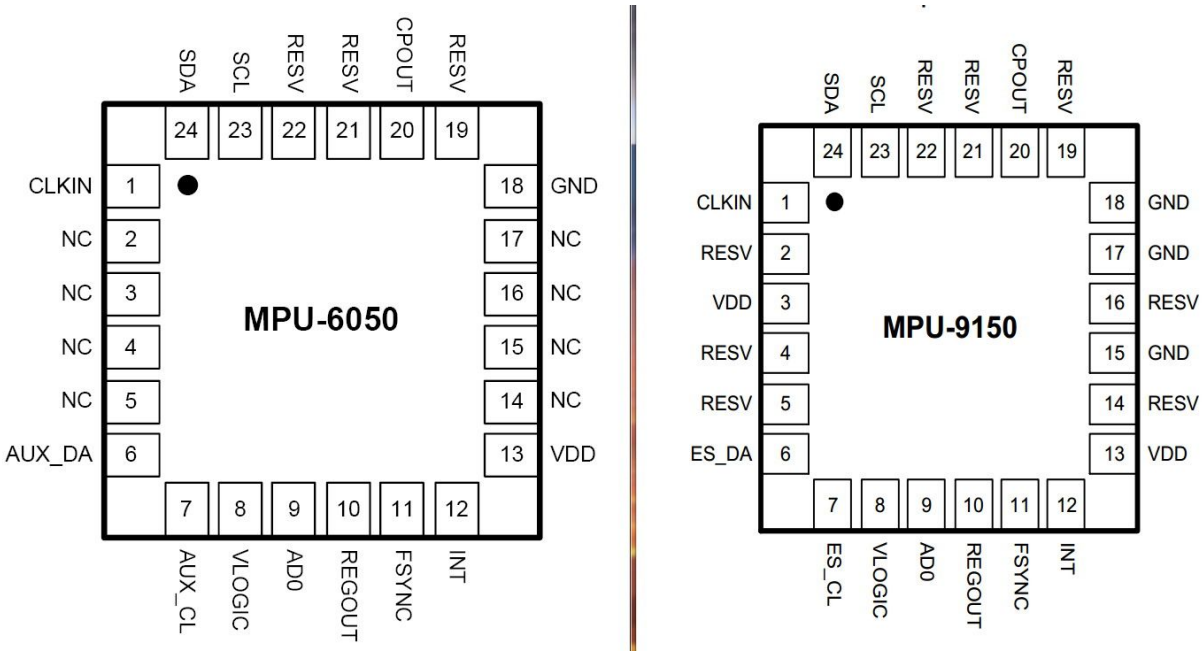
PCB:


PCB1
Rectangular PCB - Resizable
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
GY2
R1
1S
IOREF
RESET
3V3
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5

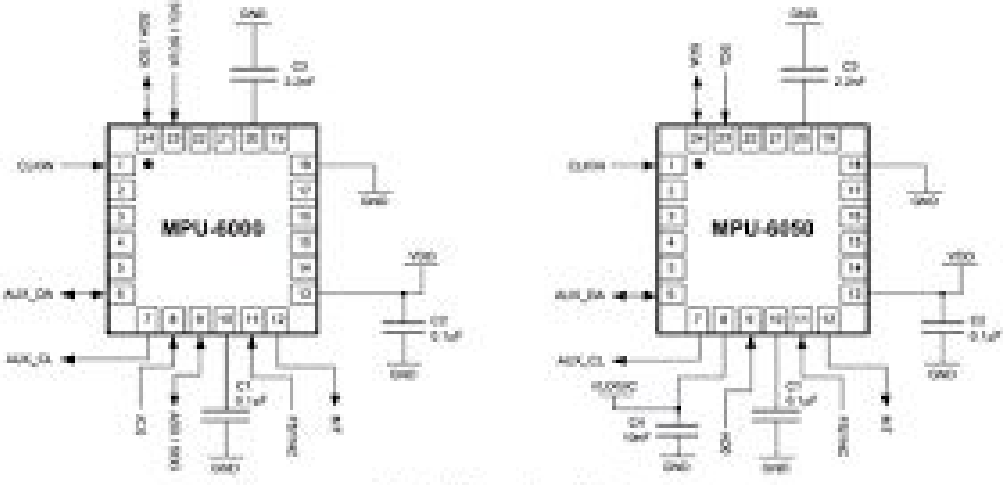
Flow Chart:


start
X coordinate and Y coordinate are taken from the sensor and push button is pressed.
Input values are mapped according to the size of the screen.
Pointer points to the mapped coordinate
if button is pressed
NO
YES
Click Event

Datasheet Of MPU-6050 Accelerometer and Gyroscope:



7.2 Typical Operating Circuit



Typical Operating Circuits

Timeline :

6/03/2018	Topic selection, selecting components, circuit diagram, flow chart.
20/03/2018	Understanding the working of components and working on arduino code.
30/03/2018	Debugging the code and perfecting the design.
1/04/2018 - 7/04/2018	Final presentation

Operating Principle:

The goal is to create mouse whose pointer has a velocity that is controlled via the tilt of the device. So, when the mouse is held flat in the air, the cursor should be stationary. If it is tilted to the left, the cursor should start to move the left of the screen proportional to the tilt.The MPU 6050 is a 6 DOF (degrees of freedom) or a six-axis IMU sensor, which means that it gives six values as output: three values from the accelerometer and three from the gyroscope. We use the accelerometer to fetch the x,y,z coordinates corresponding to the hand movement. An accelerometer works on the principle of the piezoelectric effect like a cuboidal box with a small ball inside it. The walls of this box are made with piezoelectric crystals. Whenever you tilt the box, the ball is forced to move in the direction of the inclination due to gravity. The wall that the ball collides with creates tiny piezoelectric currents. There are three pairs of opposite walls in a cuboid. Each pair corresponds to an axis in 3D space: X, Y, and Z axes. Depending on the current produced from the piezoelectric walls, we can determine the direction of inclination and its magnitude. A python script is then used to control the mouse pointer according to the coordinates obtained through MPU 6050.

Operating principle of push button when you push a button, press a micro switch or flip a toggle switch, two metal parts meetup. It may appear that the contact is made right away. That isn't exactly right. Inside the switch there are moving parts. When you push

the switch, it at first reaches the other metal part, but just in a brief split of a microsecond. At that point it reaches somewhat more, and after that again somewhat more. At last the switch is completely closed. The switch is bouncing between in-contact, and not in-contact. When the switch is closed, the two contacts actually separate and reconnect, typically 10 to 100 times over a period of about 1ms.Usually, the hardware works faster than the bouncing of the switch, which results in that the hardware thinks you are pressing the switch several times. While working with microcontrollers, we can deal with switch bounce by providing a delay of 50 to 100 microseconds using timer0 so that we don't have to care about switch debounce again and again.

Arduino Code:

```
#include <Wire.h>
#include <I2Cdev.h>
#include <MPU6050.h>

MPU6050 mpu;
// 16 bits variables to get values of accelerometer and gyroscope in X, Y and Z axis.
int16_t ax, ay, az, gx, gy, gz;

// Variables in X direction and Y direction.
int vx, vy;

// Buttons for left and right click.
int button1 = 6;
int button2 = 7;
int buttonState1 = 0;
int buttonState2 = 0;
int leftButton;

void setup() {

    // To Initialize the Serial Bus for printing data.
    Serial.begin(9600);

    /*
    * To initiate the Wire library and join the I2C bus as a master or slave.
    * This function should normally be called only once.
```

```
    *
    * Parameters :- address: the 7-bit slave address (optional); if not specified, join the
bus as a master.
    *
    * Returns :- None
    */
    Wire.begin();

    //Buttons used for clicking
    pinMode(button1, INPUT);
    pinMode(button2, INPUT);
    pinMode(7, INPUT);

    // To initialize MPU 6050 with default values
    mpu.initialize();

    // Wait for MPU6050 to respond test connection function tries to recieve demo data
to check is MPU6050 connected perfectly
    if (!mpu.testConnection()) { while (1); }
}

void loop() {
    // Function used to get values of accelerometer and gyroscope in three directions.
    mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

    // Calculate velocities in X and Y directions
    vx = (gx+15)/150;
    vy = -(gz-100)/150;

    // Read data from pin7 to check has user clicked
    leftButton = digitalRead(7);
    delay(100);                 // Delay to stop switch Debouncing.

    // Print data serially along X axis
    Serial.print("X");
    Serial.println(vx+4);               // added values to calibrate sensors value
    delay(100);                 //Delay kept for instructions not to be skipped

    Serial.print("Y");
```

```
    Serial.print(vy-1);                 // added values to calibrate the sensors value
    delay(100);                 //Delay kept for instructions not to be skipped

    // When Switch is pressed low voltage (low logic level) is passed

    //To check and print on serial monitor that is PIN high or low i.e. is switched pressed
or not
    if(leftButton == HIGH){
        Serial.println(" 0");
        delay(100);
    }
    if(leftButton == LOW){
        Serial.println(" 1");
        delay(100);
    }


}
```

Python Script:

```
# Library to work with serial data
import serial

# Library for String manipulation
import re

# Library for system function usage
import sys

# Library move mouse pointer in python
import pyautogui

# Execption handling
try:
        # To select Serial port
        arduino = serial.Serial("/dev/ttyACM0",timeout=1)

        # Forever while loop
        while True:

         # Initialize string values to be read
        str1 = [0,0]
        str2 = [0,0]

   # Read data serially from given port and manipulate string data
        a1 = str(arduino.readline())
        a2 = str(arduino.readline())

        # To manipulate String and trim off other data
        if(a1[2] == 'X'):
        str1 = [int(s) for s in re.findall(r'-?\d+\.?\d*', a1)]
   # Else skip line
        else:
        arduino.readline()
```

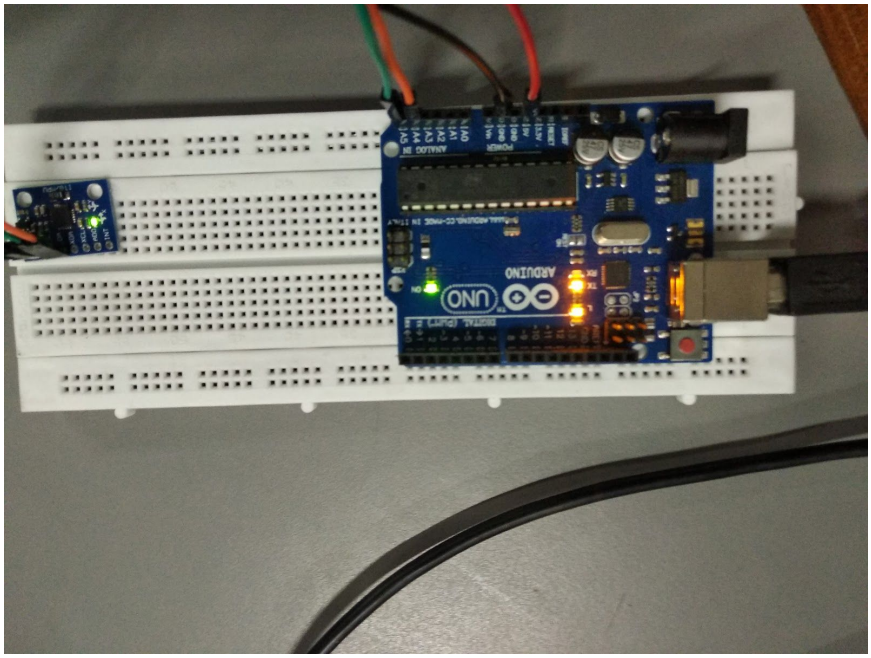
```
        # To manipulate String and trim off other data
        if(a2[2] == 'Y'):
        str2 = [int(s) for s in re.findall(r'-?\d+\.?\d*', a2)]


    # Move mouse Relatively by mapping the values given by MPU6050 and calibrate it
        pyautogui.moveRel((str2[0] )*2, (-2*(str1[0] )), duration = 0.01)

        #print all values of sensors
        print(str2[0] )
        print(-1*(str1[0] ))
        print(str2[0])

        # Check is pointer clicked
        if(str2[1] == 1):
        pyautogui.click(pyautogui.position())

        # Trying to catch an exception
except:
        print('check')
```

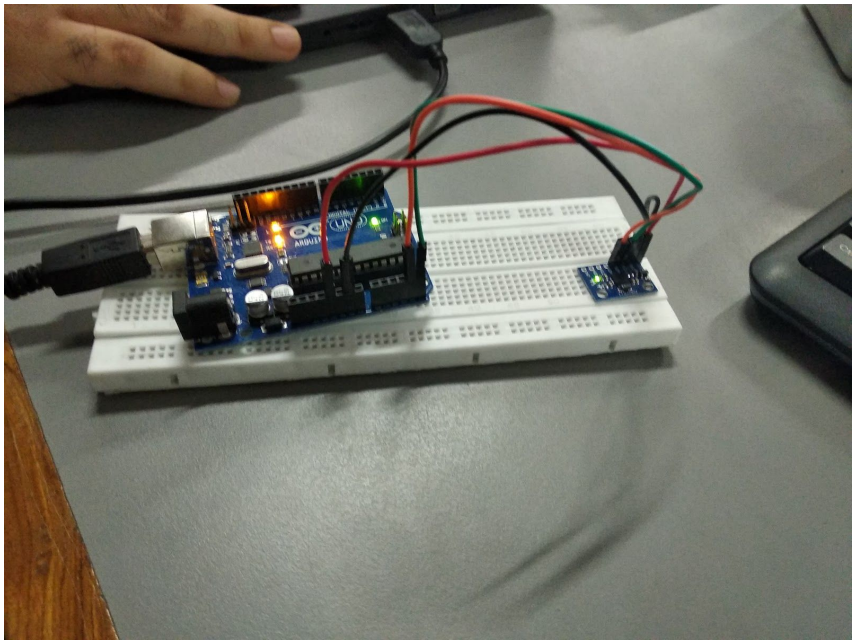
WebVr JavaScript Code:-

```
<script src="https://aframe.io/releases/0.5.0/aframe.min.js"></script>
<script
src="https://unpkg.com/aframe-teleport-controls@0.2.x/dist/aframe-teleport-controls.min
.js"></script>
<script
src="https://unpkg.com/aframe-controller-cursor-component@0.2.x/dist/aframe-controlle
r-cursor-component.min.js"></script>
<script
src="https://rawgit.com/ngokevin/kframe/csstricks/scenes/aincraft/components/random-
color.js"></script>
<script
src="https://rawgit.com/ngokevin/kframe/csstricks/scenes/aincraft/components/snap.js">
</script>
<script
src="https://rawgit.com/ngokevin/kframe/csstricks/scenes/aincraft/components/intersecti
on-spawn.js"></script>

<body>
  <a-scene>
        <a-assets>
        <img id="groundTexture" src="https://cdn.aframe.io/a-painter/images/floor.jpg">
        <img id="skyTexture" src="https://cdn.aframe.io/a-painter/images/sky.jpg">
        <a-mixin id="voxel"
        geometry="primitive: box; height: 0.5; width: 0.5; depth: 0.5"
        material="shader: standard"
        random-color
        snap="offset: 0.25 0.25 0.25; snap: 0.5 0.5 0.5"
        ></a-mixin>
        </a-assets>

        <a-cylinder id="ground" src="#groundTexture" radius="30"
height="0.1"></a-cylinder>

        <a-sky id="background" src="#skyTexture" theta-length="90"
radius="30"></a-sky>
```

```
        <!-- Hands. -->
        <a-entity id="teleHand" hand-controls="left" teleport-controls="type: parabolic;
collisionEntities: [mixin='voxel'], #ground"></a-entity>
        <a-entity id="blockHand" hand-controls="right" controller-cursor
intersection-spawn="event: click; mixin: voxel"></a-entity>

        <!-- Camera. -->
        <a-camera>
        <a-cursor intersection-spawn="event: click; mixin: voxel"></a-cursor>
        </a-camera>
  </a-scene>
</body>
```

Working conditions:





WebVr used for Demonstration