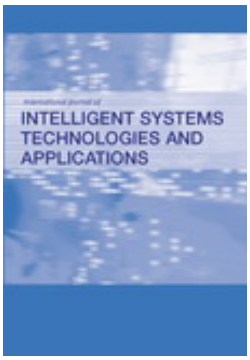
INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research



International Journal of Intelligent Systems Technologies and Applications

ISSN online: 1740-8873 - ISSN print: 1740-8865
https://www.inderscience.com/ijista

Gesture-based mouse control system based on MPU6050 and Kalman filter technique

Narayanan Prasanth, Kaustubh Shrivastava, Abhishek Sharma, Aritri Basu, Ritwik A. Sinha, S.P. Raja

DOI: 10.1504/IJISTA.2023.10055775

Article History:

Received:	10 June 2022
Last revised:	09 December 2022
Accepted:	03 January 2023
Published online:	27 April 2023

Gesture-based mouse control system based on MPU6050 and Kalman filter technique

Narayanan Prasanth*, Kaustubh Shrivastava, Abhishek Sharma, Aritri Basu, Ritwik A. Sinha and S.P. Raja

School of Computer Science and Engineering,
Vellore Institute of Technology,
Vellore-632 014, Tamil Nadu, India
Email: n.prasanth@vit.ac.in
Email: kaustubh.shrivastava2019@vitstudent.ac.in
Email: abhishek.sharma2019@vitstudent.ac.in
Email: aritri.basu2019@vitstudent.ac.in
Email: ritwik.sinha2019@vitstudent.ac.in
Email: avemariaraja@gmail.com
*Corresponding author

Abstract: The recognition of gestures with accuracy is still a widespread challenge which leads to prevention in the usage of wearable gesture recognition systems. In this paper, we proposed a gesture-based mouse, a smart glove based system for gesture recognition and remote control. To reduce the dependency of the gesture recognition system on external conditions like light and network connection, we eliminate the need of a camera and use a glove fitted with the MPU6050 sensor to detect motion accurately with minimal latency. The elimination of the camera component also reduced the overall component cost significantly. In this paper, we discuss the algorithm followed to map hand motion to cursor movement on the screen via the pseudocode and flowchart. Using this methodology, basic functionalities of the mouse and its operations were implemented with less expense and to more effectiveness.

Keywords: Kalman filter; gesture based mouse; NodeMCU; MPU6050; PyAutoGUI.

Reference to this paper should be made as follows: Prasanth, N., Shrivastava, K., Sharma, A., Basu, A., Sinha, R.A. and Raja, S.P. (2023) 'Gesture-based mouse control system based on MPU6050 and Kalman filter technique', *Int. J. Intelligent Systems Technologies and Applications*, Vol. 21, No. 1, pp.56–71.

Biographical notes: Narayanan Prasanth is an Associate Professor at the School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India. His research interest includes network switch scheduling, switching architecture, high performance computing and IoT. He has published more than 30 papers in various conferences and journals. He is a life member of ISTE and member of CSTE.

Kaustubh Shrivastava completed his BTech in Computer Science and Engineering from the Vellore Institute of Technology, Tamilnadu, India. His research interest includes data structures, computer architecture, theory of computation and compiler.

Abhishek Sharma completed his BTech in Computer Science and Engineering from Vellore Institute of Technology, Tamilnadu, India. His research interest includes cryptography, network security, cyber security and vulnerability assessment.

Aritri Basu completed her BTech in Computer Science and Engineering from the Vellore Institute of Technology, Tamilnadu, India. Her research interest includes data structures, computer architecture, theory of computation and compiler.

Ritwik A. Sinha completed his BTech in Computer Science and Engineering from the Vellore Institute of Technology, Tamilnadu, India. His research interest includes cryptography, network security, cyber security and vulnerability assessment.

S.P. Raja completed his BTech in Information Technology in 2007 from the Dr. Sivanthi Aditanar College of Engineering, Tiruchendur. He completed his ME in Computer Science and Engineering in 2010 from the Manonmaniam Sundaranar University, Tirunelveli. He completed his PhD in 2016 from the Manonmaniam Sundaranar University, Tirunelveli. His area of interest is image processing and cryptography. He is having more than 14 years of teaching experience. He has published 44 papers in international journals, 25 in international conferences and 12 in national conferences.

1 Introduction

A computer mouse is a pointing device that is used mainly in graphical user interface (GUI)-based computer systems to help the user navigate throughout the system screen. The operating systems which are GUI-based such as Apple's MacOS and Microsoft's Windows that command over 90% of the desktop OS market (2020), hence, the computer mouse plays an essential role in modern human-computer interactions, which cannot be underestimated. Though the mouse has become an important part of input systems, its pitfalls cannot be ignored. One of the severe but rare problems that may be faced by extensive mouse users is carpal tunnel syndrome (CTS). A literature survey was conducted (Thomsen et al., 2008) spanning 253 studies, which found at least nine studies that show the problems that are related to carpal tunnel pressure while using a finger or wrist. These observations lead to the conclusion that it is impossible for someone to say that using the conventional mouse (Keir et al., 1999) for a long period of time may not cause some of individuals to pathophysiological events that lead to CTS and other musculoskeletal disorders.

In this particular study (Fagarasanu and Kumar, 2003) the researchers found out after doing their experiments that 64% out of the total time that a person spends using a mouse, there is an ulnar deviation of more than 15 degrees. In some cases 30% of the total time of mouse use, the ulnar deviation exceeded 30 degrees. The factors such as ulnar deviation and total forearm pronation play a very significant role in CTS pathogenesis. Controlling the cursor is very difficult for elderly people (Smith et al., 1999). People with motor defects are one of the most common groups that face difficulties while working with a mouse (Trewin and Pain, 1999).

A computer mouse is a handheld pointing device used to control the cursor in most graphical user-based computer systems. Due to disorders like carpal tunnel syndrome faced by extensive mouse users, there is a scope for the development of gesture based pointing devices. In the current market, only a few stand-alone devices are available for wearable gesture recognition, yet most of them are expensive and not appropriate for comfortable daily wear. Most of them do not have an affordable cost for large scale adoption. Modern gesture recognition algorithms are mostly designed for discrete gestures.

In this paper, we propose a gesture-based mouse control model that would lay the foundation for the development of gesture-based recognition in the field of input and output systems in laptops and PCs. To implement this we have used an inertial measurement unit (IMU) sensor – MPU6050 which is basically a device that senses movement. It combines a 3-axis accelerometer, a 3-axis gyroscope and a digital motion processor (DMP). This sensor uses the I2C (inter-integrated circuit) bus interface to communicate with microcontrollers. With the help of this sensor, we detect gestures and use it to control the mouse of the computer it is connected to and provide some basic functionality based on gestures. This proposed design allows for a more interactive mouse that users can interact with more seamlessly with less physical contact. Thus, this mouse can be used in various fields such as for personal use by professionals and people with disabilities, in education to control smart boards, in medicine for patients requiring assistive technology and so on. When combined with a Wi-Fi module, its use can be further extended as a long-range virtual mouse that can be useful in various domains. For example, it can be used to control slideshows with simple hand gestures and can be used for presentations at conferences.

The remaining section of the paper is organised as follows: Section 2 contains the related literature reviews. Section 3 contains the proposed work. Section 4 provides results and discussions. Section 5 concludes the paper followed by future work.

2 Related work

Image processing and computer vision-based techniques have been deployed for controlling a computer mouse via gestures. Two trackers are introduced (Argyros and Lourakis, 2006); one tracks only the 2D hand gestures, while the other also tracks 3D hand gestures. The 2D hand tracker uses a Bayesian classifier to track the hands of the user by detecting the blobs possessing the desired colour distribution. The 3D hand tracker employs a stereoscopic camera system that captures two synchronised video streams, both of which are then processed by a 2D hand tracker. A 3D reconstruction is then obtained from the processed video streams using the iterative closest point algorithm. After tracking the hands, a finger detection method is used to detect fingers on each hand in the viewpoint, this finger detection data is then used for detecting gestures.

The researchers in Ko and Yang (1997), presented a device that would be able to recognise hand gestures to move the cursor as well as perform various mouse functions. They came up with an algorithm that would be able to trace the path of the index finger as well as identify unistroke gestures. The gestures may copy, rotate, move, enlarge, reduce, etc. It incorporates the use of input patterns making it suitable to precisely identify finger movement. The algorithm applied for recognising gestures is trainable; this means that we can define new gestures in the system. The researchers (Tsai et al.,

2020) presented a cheap system with hand recognition. Many different types of vision techniques have been used. The region of interest from the regions is being captured using skin and motion detection. A connected component labelling algorithm is used to find the centroid of an object. A convex hull algorithm is used to find out the exact area of the hand gesture and remove the arm area. They experienced some interference in a simulated environment but the recognition rate was still high. The authors (Kim et al., 2010) proposed an optimised mouse to control the head of the mouse on a computer screen with head rotations and eye blinks so that severely disabled people can use a computer. The cursor position was calculated using the values of the angular velocity of head rotators that they got from the gyro sensors, whereas the click and double click events of the mouse were detected by the Optical Sensors which are triggered by the blinking of the user's eyes. The mouse has shown better performance in about only 21% of the experiments which included about 25% of Dasher, 20 times clicking and about 37% of the on-screen keyboard, respectively.

The researchers (Sziladi et al., 2017) have made software that with the help of hand and eye gestures controls the mouse cursor. The researchers here used an analysing software whose main purpose is to keep track of the eye and mouse and then store the data, known as open gaze and mouse analyser (OGAMA). A leap motion controller was used to detect the movement of the hand. The stored data is then used to analyse and process these movements. Another purpose of the software is to filter the data of hand and eye movement. It incorporates database pre-processing solely for this purpose. According to the researchers they have worked on two types of movement, i.e., fixation and saccade. So, fixation is basically that the person is maintaining his gaze at one place on the screen whereas saccade is a very quick movement of eyes from one point on the screen to the other point on the screen.

The researchers (Perng et al., 1999) demonstrated that accelerometers could be used to determine and convert hand gestures into machine-interpreted signals. The glove model they proposed consisted of six accelerometers and a wrist controller. They positioned five accelerometers on the fingertips and one on the dorsum of the hand. They were able to develop a pointing device with the idea of a gravity-induced accelerometer. The authors gave the liberty to choose the model of accelerometers, however, they used ADXL202 (accelerometer module) with +/- 2g of range. They also pointed out that the energy dissipation of the glove totalled about 45 mW–50 mW making it extremely resourceful in a constrained environment like the IOT network. The smart gloves introduced by Xia and Du (2019) are based on the acceleration based gesture recognition method. The recognition accuracy of the smart glove is improved by using the attitude data information. A threshold model is used to segment the collected data and the feature extraction is done using the ant colony algorithm. The identification of gestures is done using the hidden Markov model used for modelling and training the dataset. The collected data of the glove is stable and the signal processing is simple though the equipment is more complicated and expensive and the utility is relatively low. The authors (Mhetar et al., 2014) have proposed the addition of some functionality to a virtual marker. The algorithm used by this paper can be divided into five sections: object recognition, trace object, coordinate calculation, setting cursor position and event generation. The algorithm has been briefly described below. The IR camera helps in recognising a source emitting IR radiations. Thus, it will be able to detect the IR light from the mouse. The researchers suggested the use of Teensy or external crystal to

generate the frequency required for the setup (24 MHz). An IR camera traces both the coordinate axes (X and Y) with the help of the 24 MHz clock generated by teensy. This helped them obtain an accurate estimate of the position of the object emitting IR radiation. The camera subsequently directs this position data to the microcontroller. Once the data of the position has been received the coordinate conversion algorithm devised by the researchers is used to read the X and Y coordinates of the object. The mouse pointer on the screen is moved with the help of human interface devices (HID) function embedded in the microcontroller. The HID function comes as an already implemented method in teensy. The X and Y are given as input to this HID function which helps position the mouse pointer correctly. During the entire process of action, the camera traces the IR light emitted by the light emitting diode (LED) and consequently outputs the same on the screen.

Two problems with the approach proposed (Argyros and Lourakis, 2006) were pointed out. Firstly, it requires a camera setup, especially for the 3D tracker and secondly, it can fail in bad lighting situations. Hence, we cannot use this technique as a day-to-day replacement for the computer mouse. The researchers (Dang et al., 2020) have done a comprehensive survey on human activity recognition (HAR) which is of two types, i.e., vision-based HAR and sensor-based HAR. The weaknesses, characteristics and strengths of machine learning and deep learning models were analysed. They found out that vision-based HAR is better than sensor-based HAR because the vision data is quite easily collected and is affordable. The convolutional neural network (CNN) model that they made had a recognition rate of 91.33%. There were certain drawbacks to the model proposed (Kim et al., 2010), the system required about 3 to 4 iterations to control the mouse position precisely because its accuracy was low. Although they had designed an optical circuit to reduce the effects of ambient light, the output of the optical sensor was affected by ambient light. This severely affected the mouse-clicking operations (double-clicking was the most cumbersome). Lastly, the researchers also pointed out that it was inconvenient to use the proposed since it requires assistance while wearing and taking off the mouse. The researchers (Sziladi et al., 2017) conducted various experiments and found that the results obtained from the test users were not satisfactory. The test subjects complained that they had to concentrate too much on the screen to move the mouse around the screen which tired their eyes. So, it was basically harder to use than the conventional mouse. They also analysed the data that they had gathered using OGAMA software and they found out that the hand and eye gesture-based mouse was much more ambiguous than the conventional mouse. One limitation of the model proposed (Mhetar et al., 2014) was the cumulative cost of the project. PixArt Wii controllers cost around 3,000–8,000 INR and a Teensy 4.0 Development board costs around 2,000 INR. Overall, the project cost comes to around 10,000 INR, which is too expensive for mass consumption.

Illnesses arising from the extensive use of computer mouse have been a problem worrying researchers across the globe for the past two decades. The problem is particularly interesting due to the fact that none of the previously implemented/mentioned models has been able to solve the issue completely. In addition, very few manufacturing companies actually pay heed to these problems and solely direct their efforts towards the performance of the mouse. Certain existing models have tried to address issues, particularly carpal tunnel pressure. Even though they were able to solve it to some extent, it was ultimately a compromise on the mouse's performance. We aim to propose a balanced mouse model keeping in account both the performance and the diseases

associated with its use (Kamel Benamara et al., 2021; Verma et al., 2020; Verma and Singh, 2021).

3 Proposed methodology

The approach taken in the paper to control the mouse combines the usage of Python automatic graphic user interface (PyAutoGUI) and accelerometer (MPU6050) whose data is filtered using Kalman filter (Alfian et al., 2021). This particular approach was chosen because though there were many attempts in the past to replace the conventional mouse, there were many problems with them such as a lack of efficiency in moving the mouse cursor. The main objective of the paper is to develop a wearable mouse pointer through which the user can control the cursor operations on any computer system. The shardware setup comprises the MPU6050 attached to the glove which in turn is connected to the ESP8266 via I2C communication (Pandey et al., 2018) lines. The ESP8266 is powered by connecting it to a USB port of the computer system. The Arduino script uploaded in the ESP8266 converts the static accelerometer data into Euler angles using the following equations (finding angle around x-axis and y-axis using the static accelerometer data)

$$\text{angleY} = \arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right), \text{angleX} = \arctan\left(\frac{a_y}{\sqrt{a_x^2 + a_z^2}}\right) \tag{1}$$

angleY is the angle around the y-axis and angleX is the angle around the x-axis. These are then combined with the data from the gyroscope using the Kalman filter to produce the final readings that will be used by the python script to control the cursor with the help of functions provided by pyautogui library.

3.1 Used components

- *PyAutoGUI* – we are going to move the cursor on the screen using a Python script which uses the python library called PyAutoGUI (2020), which is a cross-platform GUI automation Python module. The data that we are going to get from the ESP8266 will go through the USB port which uses the UART Protocol and then to the python script. It has functions such as click(), press(), move(), etc. for controlling the cursor on the computer system.
- *ESP8266* – the ESP8266 (2018) is a low-powered and low-cost microcontroller with the support of the TCP/IP protocol. Following are some of the important features of the chip:
  - a small form factor
  - b 128 kilobyte of RAM
  - c 4 megabit of in built flash memory
  - d built-in Wi-Fi and Bluetooth support.
  - e Harvard architecture based 32 bit processor that has 16 bit instruction set.
  - f 17 general purpose input and output pins.

ESP8266 was chosen for this IoT project mainly because its small form factor allows it to be easily attached to a glove without increasing the weight or reducing the functionality and has enough memory and RAM to efficiently execute the stored program.

- *MPU6050* – MPU6050 (2019) is a motion sensor that consists of an accelerometer and a gyroscope. It is used to measure velocity, orientation, acceleration and displacement. An MPU6050 is designed for providing accurate motion detection in power, cost and space constrained environments. The internal calculations of this sensor are performed by its in-built DMP efficiently. MPU6050 consists of a 16-bit analog to digital converter (ADC) hardware to track three dimensional motions. The sensor communicates with ESP8266 via the I2C interface. MPU6050 is small sized and less expensive, making it the ideal sensor for wearable products like a virtual mouse.
- *Kalman filter* – the following section on Kalman filter is written in its entirety by referring to (2018) and the work proposed by Simon (2001). Kalman filter is an algorithm which is used to reduce noise and variance from a series of measurements over time. The Kalman filter uses all the past measurements and the current measurement to find the best estimate of the measured value at the present time. In our approach we are using an accelerometer and a gyroscope together to compute the direction of the tilt of the glove. Although either of the sensors is sufficient for computing the tilt of the sensor, the gyroscope suffers from an accumulation of the errors from integration approximation which causes it to drift and the data from the accelerometer is very sensitive to the changes in the tilt of the sensor. This means that we can only trust the data from the gyroscope for a short time and the data from the accelerometer for a longer time. The Kalman filter is used to get the best estimate of the tilt of the sensor from both the sensor's data; this solves the sensitivity and the drift problems of the accelerometer and gyroscope respectively.

3.2 Kalman filter operation

Prediction: we first estimate the current state of the system based on all the previous estimates.

Next we estimate the error covariance matrix based on the previous error covariance estimate.

Initially we have

$$P = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

This matrix signifies our trust in the current value of the system state estimate, the smaller the value of the elements of the matrix, higher the trust.

Updating: we compute a parameter called innovation.

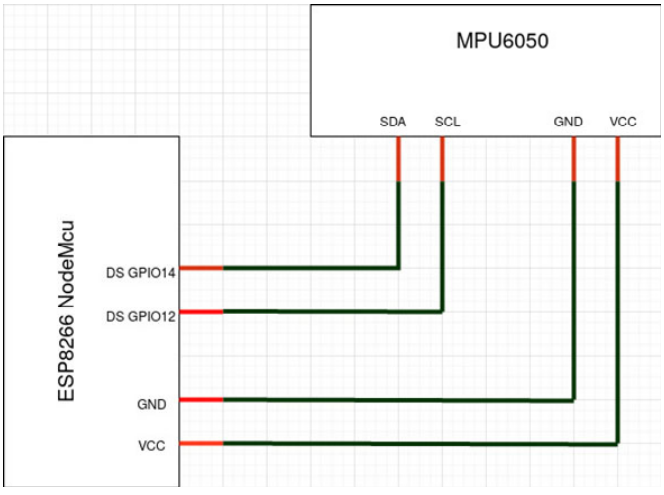
Next, we calculate the innovation covariance matrix and the Kalman gain.

The innovation variance predicts how much we should trust the measurement value from the sensor and the Kalman gain tells us how much we trust the innovation. If the value is large, it shows that we do not put much weight into the incoming sensor data.

The value for Kalman gain will be high if we trust the innovation but do not trust the estimation of the current state.

Finally, we calculate the current estimate of the state, and the error covariance matrix.

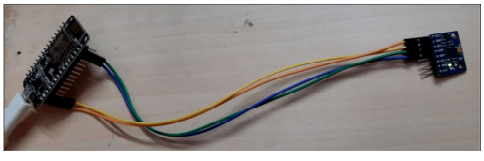
Figure 1 Proposed hardware design (see online version for colours)



The design for our project was fairly simple as shown in Figures 1–3. The connections between the various components constituting the system have been described below:

- The service capability layer (SCL) and secure device access (SDA), which facilitate the serial communication of data, pins of MPU6050 are connected with the D5 general purpose input/output (GPIO) and D6 GPIO interface of the ESP8266.
- The GND (ground) of MPU6050 is connected to the GND (ground) of ESP8266.
- The VCC (common collector voltage) of MPU6050 is connected to the 3.3 V pin of ESP8266.
- A USB type-B was used to power the ESP8266 kit and upload the Arduino code.

Figure 2 Hardware setup I (see online version for colours)



As shown in Figure 4, the ESP8266 uses the MPU6050 to get the accelerometer and the gyroscope data as it is programmed to get the data from the MPU6050 sensor and then filter the acceleration and the gyroscopic data so that the disturbance or the noise can be cancelled out. ESP8266 puts all the filtered data in the COM4 serial in the byte format. We get these values from the serial port using a python script. We are going to get the byte values from the serial port. We have used the python library PyAutoGUI and came

up with an algorithm which revolves around PyAutoGUI to move the mouse around the computer screen and click around according to the user's will. The flow chart of the algorithm is shown in Figure 5. One of the benefits of using our approach over an image processing based is that we do not need a camera setup or lighting setup to detect the hand movements and since we are directly collecting the movement data, the processing required is less than that for image processing techniques and image processing algorithms used to detect the gestures of the fingers can fail at times. Compared to approaches based on eye movement that can lead to problems to the eye as it requires more concentration for moving the mouse around while we are directly getting the data from the glove to move around the mouse. In our approach, the data is directly collected from the sensor, so no such problems are faced.

Figure 3 Hardware setup II (see online version for colours)

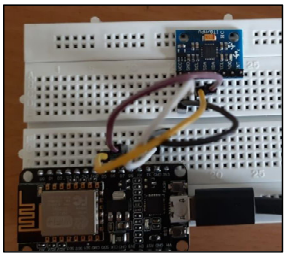


Figure 4 Data flow diagram (see online version for colours)

Additionally, we have refrained from using optical sensors in our implementation as it is greatly affected by the surrounding environment's light causing great deviations in their readings. This is clearly not a very suitable way of incorporating mouse-clicking operations in such projects. It was also pointed out that the mouse was not very comfortable to wear which itself, is a great hurdle. Keeping this in mind, we will just be using a glove to make it more comfortable to use. Keeping in mind the cost of this project, we have used less as well as cheaper components to build our system. The entire project cost a total of Rs.920 to build which is astonishingly less for such a gesture-based control system. Additionally, with fewer components, we can integrate this system onto a wearable component like a glove that can be used to navigate the mouse cursor effortlessly.

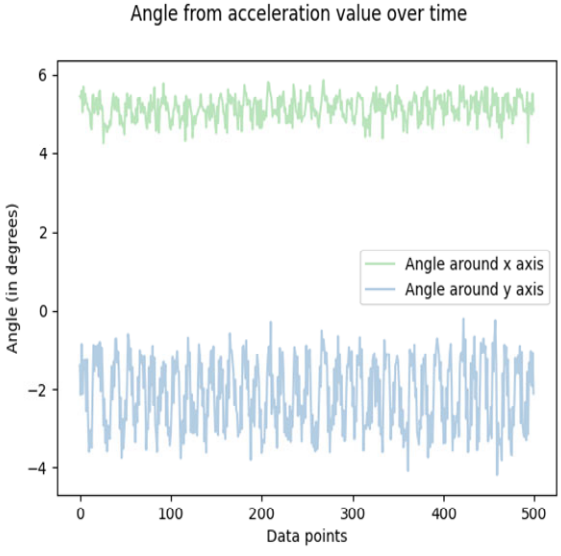
4 Results and discussion

4.1 Finding measurement noise variance

To find the variance of measurement noise of the accelerometer reading, we kept the sensor undisturbed and recorded around 500 readings as shown in Figure 6.

The value we obtained for a measurement noise variance for the angle around the x-axis is 0.094 and for the angle around the y-axis it is 0.788. We take the average of these two as the measurement noise variance value in our system, which is 0.441.

Figure 5 The angle measured from the accelerometer readings in degrees is plotted on the Y-axis and the number of data points is on the X-axis (see online version for colours)



4.2 Finding acceleration variance and variance of the gyroscope bias

To find the acceleration variance and variance of the gyroscope bias we tested different values for both and chose the one with the least mean squared error for the tilt reading from the sensor. We took the raw readings from the gyroscope and accelerometer around any one axis and then used them against the data from the Kalman filter to compute the MSE.

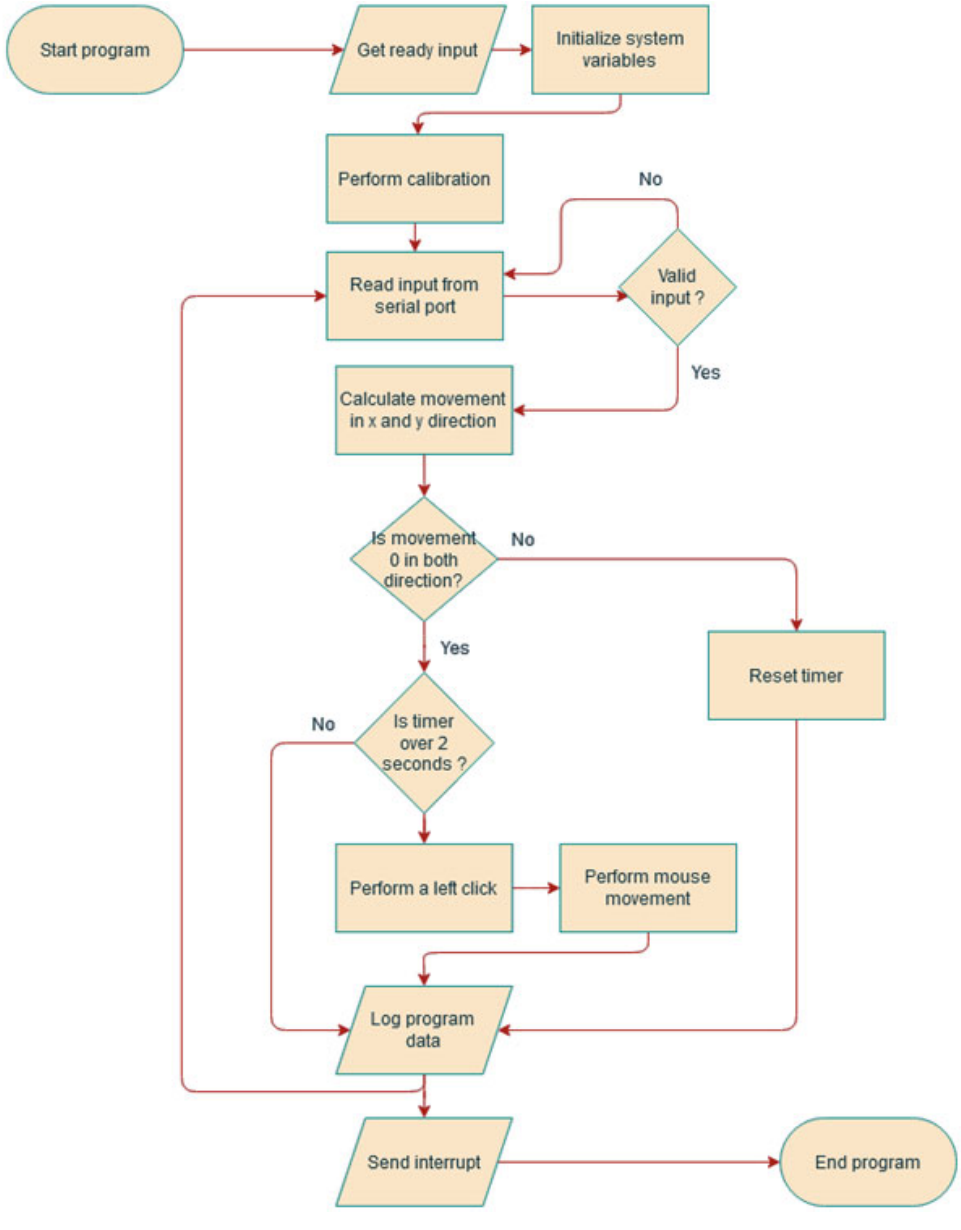
From Table 1, for each column (except the first one) the first value in the tuple is the value used for accelerometer variance and the second value in the tuple is the value used for the variance of the gyroscope bias. The values in each cell of Table 1 is the MSE obtained against a given accelerometer variance and gyroscope bias. The formula for MSE is particularly simple (formula for calculating MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{2}$$

Here, Y_i is the value observed from the sensor and the $\hat{Y}_i$ is the predicted/filtered value from the Kalman filter. For instance the value in the first row and first column is the MSE between the gyroscope sensor value and the Kalman filter value around the x-axis. If we carefully observe the MSE between the Kalman filter value and the accelerometer sensor values for a given pair of (accelerometer variance, gyroscope bias), we see that there is not a lot of deviation between the values. We thus went for the MSE between the

gyroscope sensor value and Kalman filter value as there are deviations between these values. The reason we chose both accelerometer variance and gyro bias as 0.089 is because we got the least MSE (between Kalman filter and sensor value) for gyroscope values in this case.

Figure 6 Flow chart diagram of proposed system (see online version for colours)



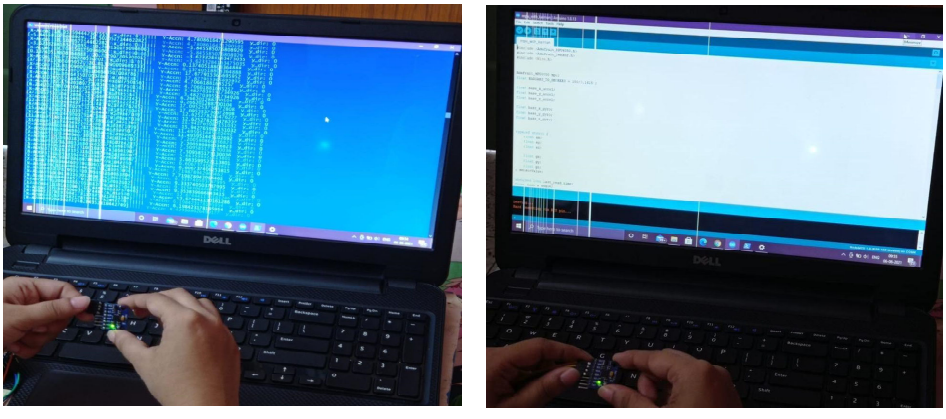
4.3 Qualitative results

The cursor on the screen can be moved to the desired direction by tilting the MPU6050 sensor in that direction as shown in Figures 7(a) and 7(b). There is very negligible lag between the motion of the sensor and the motion of the cursor. The cursor can be used to click at any desired position by keeping the sensor stationary for two seconds. There is very negligible lag between the motion of the sensor and the motion of the cursor. Compared to the normal touchpad or wired mouse, using the sensor gives us a more hands-free experience and feels comfortable.

Table 1 Comparing MSE between filtered values from Kalman filter and values from the accelerometer and gyroscope for different values of acceleration and variance of gyroscope bias

Parameter	*(0.096, 0.0)*	*(0.042, 0.063)*	*(0.056, 0.041)*	*(0.061, 0.066)*	*(0.054, 0.043)*	*(0.089, 0.089)*	*(0.00, 0.082)*	*(0.08, 0.026)*	*(0.08, 0.085)*
With gyroscope around X axis	3021.06	385.58	266.63	1373.12	105.99	174.05	348.36	267.30	390.77
With accelerometer around X axis	4.31	4.51	2.58	4.33	2.97	3.34	1.27	2.58	4.80
With gyroscope around Y axis	496.0	209.80	70.67	90.61	1087.02	129.03	144.28	70.67	40.29
With accelerometer around Y axis	1.70	1.97	1.31	1.22	1.91	1.71	1.56	1.31	3.38

Figure 7 Cursor movements using: (a) mouse I and (b) mouse II (see online version for colours)



(a) (b)

As we have found above the least MSE came out for 0.089 for both accelerometer variance and variance of gyroscope bias. We also tried the different values of accelerometer variance and variance of gyroscope bias as stated in Table 1. While testing for the different values, we also found out the same result, i.e., the mouse was working

more efficiently when the accelerometer variance and variance of gyroscope was 0.089. This further proves that the comparison of MSE between filtered values from Kalman filter and values from the accelerometer and gyroscope for different values of acceleration variance and variance of gyroscope bias is valid.

The mouse that we have made was tested using the Dasher program. The Dasher program is a software that is used to write sentences by just dragging the mouse cursor towards the letter that to write. When we used our proposed mouse to enter text using Dasher, the overall speed of our mouse was less than a conventional mouse. We tried to type the sentence 'The quick brown fox' with both our mouse and conventional mouse and found that it took 30 s for the former and 10s for the latter. Thus, we cannot claim that our proposed mouse is much better compared to the conventional mouse in terms of speed, but if we compare it to other gesture based mouse's then maybe we can say that our proposed mouse is better in terms of performance. We also tried our mouse on the on-screen keyboard but the problem with the trial on-screen keyboard is that our mouse takes some time to initiate the click so the time it takes against a conventional mouse is even more as compared to when the Dasher program was used. Though the speed of our proposed mouse may be slow, it is pretty accurate in doing the things it is supposed to do.

5 Conclusions and future scope

In this work, we have implemented a gesture-based mouse that would be very useful for the elderly as well as for people who have motor diseases or others to reduce the strain caused by using a conventional mouse. One of the benefits of using our approach over an image processing based is that we do not need a camera setup or lighting setup to detect the hand movements and since we are directly collecting the movement data, the processing required is less than that for image processing techniques and image processing algorithms used to detect the gestures of the fingers can fail at times. Compared to approaches based on eye movement that can lead to problems for the eye as it requires more concentration for moving the mouse around while we are directly getting the data from the glove to move around the mouse. In our approach, the data is directly collected from the sensor, so no such problems are faced.

Additionally, we have refrained from using optical sensors in our implementation as it is greatly affected by the surrounding environment's light causing great deviations in their readings. This is clearly not a very suitable way of incorporating mouse-clicking operations in such projects. It was also pointed out that the mouse was not very comfortable to wear which itself, is a great hurdle. Keeping this in mind, we will just be using a glove to make it more comfortable to use. Keeping in mind the cost of this project, we have used less as well as cheaper components to build our system. The entire project cost a total of Rs.920 to build which is astonishingly less for such a gesture-based control system. Considering the fact that the ESP8266 module used comes with an inbuilt Wi-Fi module, this design can be easily extended to communicate wireless with a computer system with computer system while incurring additional cost. Furthermore, in the future, we can directly make device drivers control the mouse and make embedded hardware dedicated to doing any particular work so that the mouse implementation will work much faster. We can also enhance the system by using more sensors that will provide data according to the magnitude of the accelerometer and then the mouse can be

moved accordingly. Thus, with fewer components, we can successfully integrate this system onto a glove that the user can wear and navigate the mouse pointer effortlessly.

References

Alfian, R.I., Ma'arif, A. and Sunardi, S. (2021) 'Noise reduction in the accelerometer and gyroscope sensor with the Kalman filter algorithm', *Journal of Robotics and Control (JRC)*, Vol. 2, No. 3, pp.180–189.

Argyros, A. and Lourakis, M. (2006) *Vision-Based Interpretation of Hand Gestures for Remote Control of a Computer Mouse*, Vol. 3979, pp.40–51, DOI: 10.1007/11754336_5.

Dang, L.M. et al. (2020) 'Sensor-based and vision-based human activity recognition: a comprehensive survey', *Pattern Recognition*, Vol. 108, No. 1, p.107561.

Fagarasanu, M. and Kumar, S. (2003) 'Carpal tunnel syndrome due to keyboarding and mouse tasks: a review', *International Journal of Industrial Ergonomics*, Vol. 31, pp.119–136, DOI: 10.1016/S0169-8141(02)00180-4.

Kamel Benamara, N., Zigh, E., Boudghene Stambouli, T. and Keche, M. (2021) 'Towards a robust thermal-visible heterogeneous face recognition approach based on a cycle generative adversarial network', *International Journal of Interactive Multimedia and Artificial Intelligence*, In Press (In Press), pp.1–14, http://doi.org/10.9781/ijimai.2021.12.003.

Keir, P.J., Bach, J.M. and Rempel, D. (1999) 'Effects of computer mouse design and task on carpal tunnel pressure', *Ergonomics*, Vol. 42, No. 10, pp.1350–1360, DOI: 10.1080/001401399184992.

Kim, S., Park, M., Anumas, S. and Yoo, J. (2010) 'Head mouse system based on gyro-and opto-sensors', in *2010 3rd International Conference on Biomedical Engineering and Informatics*, IEEE, October, Vol. 4, pp.1503–1506.

Ko, B.K. and Yang, H.S. (1997) 'Finger mouse and gesture recognition system as a new human computer interface', *Computers & Graphics*, Vol. 21, No. 5, pp.555–561.

Mhetar, A., Sriroop, B.K., Kavya, A.G.S., Nayak, R., Javali, R. and Suma, K.V. (2014) 'Virtual mouse', *International Conference on Circuits, Communication, Control and Computing*, pp.69–72, DOI: 10.1109/CIMCA.2014.7057759.

Pandey, V.K., Kumar, S., Kumar, V. and Goel, P. (2018) 'A review paper on I2C communication protocol', *Int. J. Adv. Res. Ideas Innov. Technol*., Vol. 4, No. 2, pp.340–343.

Perng, J.K., Fisher, B., Hollar, S. and Pister, K.S. (1999) 'Acceleration sensing glove (ASG)', in *Digest of Papers. Third International Symposium on Wearable Computers*, IEEE, October, pp.178–180.

Simon, D. (2001) 'Kalman filtering', *Embedded Systems Programmin*g, Vol. 14, No. 6, pp.72–79.

Smith, M.W., Sharit, J. and Czaja, S.J. (1999) 'Aging, motor control, and the performance of computer mouse tasks', *Human Factors*, Vol. 41, No. 3, pp.389–396, DOI: 10.1518/001872099779611102.

Sziladi, G., Ujbanyi, T., Katona, J. and Kovari, A. (2017) 'The analysis of hand gesture based cursor position control during solve an IT related task', in *2017 8th IEEE International Conference on Cognitive Info communications (CogInfoCom)*, IEEE, September, pp.000413–000418.

Thomsen, J.F., Gerr, F. and Atroshi, I. (2008) 'Carpal tunnel syndrome and the use of computer mouse and keyboard: a systematic review', *BMC Musculoskelet Disord*, Vol. 9, No. 1, p.134.

Trewin, S. and Pain, H. (1999) 'Keyboard and mouse errors due to motor disabilities', *International Journal of Human Computer Studies*, Vol. 50, No. 1, pp.109–144.

Tsai, T-H., Huang, C-C. and Zhang, K-L. (2020) 'Design of hand gesture recognition system for human-computer interaction', *Multimedia Tools and Applications*, Vol. 79, No. 9, pp.5989–6007.

Verma, K.K. and Singh, B.M. (2021) 'Deep multi-model fusion for human activity recognition using evolutionary algorithms', *International Journal of Interactive Multimedia and Artificial Intelligence*, Vol. 7, pp.44–58, http://doi.org/10.9781/ijimai.2021.08.008.

Verma, K.K., Singh, B.M., Mandoria, H.L. and Chauhan, P. (2020) 'Two-stage human activity recognition using 2D-ConvNet', *International Journal of Interactive Multimedia and Artificial Intelligence*, Vol. 6, p.11, http://doi.org/10.9781/ijimai.2020.04.002.

Xia, P. and Du, Z. (2019) 'Research on intelligent recognition of intelligent gloves based on acceleration sensor', *3rd International Conference on Mechatronics Engineering and Information Technology (ICMEIT 2019)*, Atlantis Press.

Appendix

Kalman filter operation details

Introduction

Kalman filter is an algorithm which is used to reduce noise and variance from a series of measurements over time. The Kalman filter uses all the past measurements and the current measurement to find the best estimate of the measured value at the present time. In our approach we are using an accelerometer and a gyroscope together to compute the direction of the tilt of the glove. Although either of the sensors is sufficient for computing the angle of the tilt of the sensor, the gyroscope suffers from accumulation of the errors from integration approximation which causes it to drift and the data from the accelerometer is very sensitive to the changes in the tilt of the sensor. This means that we can only trust the data from the gyroscope in the short term and the accelerometer in the long term. The Kalman filter is used to get the best estimate of the tilt of the sensor from both the sensor's data; this solves the sensitivity and the drift problems of the accelerometer and gyroscope respectively.

As can be inferred from the above graph, the data from the gyroscope starts to drift over time even though the sensor is not disturbed. For the remainder of this section on Kalman filter implementation, we will Δt to denote the time elapsed between two readings.

Prediction: we first estimate the current state of the system based on all the previous estimates by

$$x_{k|k-1} = F.x_{k-1|k-1} + B.\theta_k$$

Next we estimate $P_{k|k-1}$, the error covariance matrix based on the previous error covariance estimate $P_{k-1|k-1}$ using

$$P_{k|k-1} = F.P_{k-1|k-1}.F^T + Q_k$$

Initially we have,

$$P = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

This matrix signifies our trust in the current value of the system state estimate, the smaller the value of the elements of the matrix, higher the trust.

Updating: we compute a parameter called innovation using the following equation:

$$\overline{y}_k = z_k - H.x_{k|k-1}$$

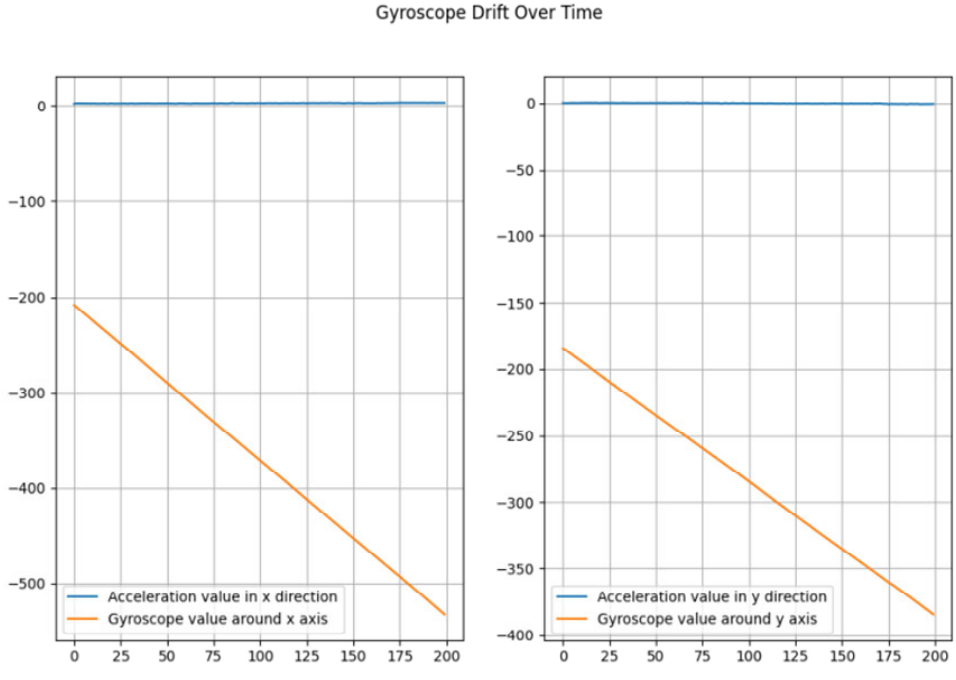
Next, we calculate the innovation covariance matrix and the Kalman gain.

$$S_k = H.P_{k|k-1}.H^T + R$$

$$K_k = H.P_{k-1|k}.H^T + S_k^{-1}$$

The innovation variance predicts how much we should trust the measurement value from the sensor and the Kalman gain tells us how much we trust the innovation. If the value S_k is large, it shows that we do not put much weight into the incoming sensor data. The value for Kalman gain will be high if we trust the innovation but do not trust the estimation of the current state.

Figure A1 Gyroscope drift over time (see online version for colours)



Finally, we calculate the current estimate of the state, and the error covariance matrix using the following formula:

$$x_{k|k} = x_{k|k-1} + K_k.\overline{y}_k$$

$$P_{k|k} = (I - K_k.H).P_{k|k-1}$$

This value $x_{k|k}$ data is further used for controlling the mouse cursor.