RESEARCH ARTICLE OPEN ACCESS

MINI MICE

Ajmal E B [1], R Kamlesh Kumar [2], Sadhik T S [3], Mohammed Anees K A [4]

[1] Asst. Prof, Dept. Of Computer Science and Engineering, KMEA Engineering College, Kerala - India

[2] [[3][4] UG Scholar, Dept. Of Computer Science and Engineering, KMEA Engineering College, Kerala - India

ABSTRACT

This project focuses on developing a gesture control mouse system using Arduino and an accelerometer for intuitive computer navigation. By leveraging the accelerometer's motion sensing capabilities and Arduino's programmability, users can manipulate the mouse cursor on the screen through natural hand movements and gestures. The system employs an Arduino board and an accelerometer sensor to capture and interpret real-time hand movements, mapping specific gestures to corresponding cursor actions. This eliminates the need for traditional mouse input devices and enhances the user experience with a more immersive and ergonomic interaction method. The system's versatility allows for customization and integration into various computer applications, promising a revolution in intuitive and intuitive computer interaction.

Keywords: gesture control mouse, Arduino, accelerometer, intuitive navigation, natural hand movements, gestures, motion sensing, programmability, real-time, cursor actions, user experience, immersive interaction, ergonomic, customization, computer applications.

I. INTRODUCTION

Human-computer interaction has undergone significant advancements in recent years, aiming to provide more intuitive and immersive ways for users to interact with digital systems. Traditional input devices such as keyboards and mice have limitations in terms of natural interaction and may not be suitable for certain scenarios. Gesture-based interfaces have emerged as a promising solution, allowing users to control computer interfaces through hand or body movements. In this research paper, we present a detailed introduction to a gesture control mouse system that utilizes an accelerometer sensor and Arduino microcontroller, enhancing the human-computer interaction experience.

The proposed gesture control mouse system leverages the accelerometer sensor's ability to measure acceleration forces and detect changes in orientation and movement. By integrating the accelerometer sensor with the Arduino microcontroller, real-time tracking and interpretation of hand gestures are achieved. This enables users to control the computer cursor intuitively, without the need for physical contact with traditional input devices.

The system's hardware setup involves connecting the accelerometer sensor to the Arduino board, ensuring proper wiring and connections. Calibration of the accelerometer sensor is crucial to obtain accurate readings and compensate for any bias or offset in the sensor's output.

Gesture recognition algorithms play a key role in accurately interpreting hand movements and translating them into meaningful actions. Various algorithms, such as thresholding, filtering, and machine learning approaches, can be employed to recognize different hand gestures. These algorithms are designed to detect gestures such as swipes, circles, taps, and more, enabling precise control of the computer cursor.

To facilitate seamless integration with the computer, the gesture control mouse system utilizes USB communication and emulates a standard mouse. This enables the system to transmit mouse movement and click events to the computer, allowing users to interact with applications and interfaces just as they would with a conventional mouse. Evaluation of the system's performance is cond

ucted through comprehensive experiments, including diverse hand gestures performed by participants. The results showcase the accuracy, responsiveness, and usability of the gesture control mouse system, comparing it to traditional mouse control methods. The findings validate the feasibility and effectiveness of the proposed system.

The applications of the gesture control mouse system span across various domains. In gaming, it offers an immersive and interactive experience by allowing players to control in-game actions through intuitive hand gestures. In virtual reality environments, users can navigate virtual worlds with ease, enhancing the sense of presence and immersion. Moreover, the gesture control mouse system has the potential to improve accessibility for individuals with motor impairments, providing an alternative and user-friendly input method.

In conclusion, this research paper introduces a detailed overview of a gesture control mouse system that utilizes an accelerometer sensor and Arduino microcontroller. By leveraging hand gestures for computer interaction, the system offers an intuitive and immersive experience. The paper highlights the hardware setup, gesture recognition algorithms, USB communication, and evaluation of system performance. The findings demonstrate the feasibility and potential

applications of the gesture control mouse system, paving the way for further advancements in human-computer interaction.

II. METHODOLOGY

The methodology section of the journal paper outlines the research design, experimental setup, data collection procedures, and analysis techniques used in developing the gesture control mouse system using an accelerometer and Arduino microcontroller. This section provides a detailed description of the steps taken to design and implement the system, ensuring transparency and reproducibility of the study.

A. Hardware Setup:

The hardware setup involves connecting the accelerometer sensor and Arduino microcontroller. The specific model and specifications of the accelerometer sensor and Arduino board used in the study should be provided. The wiring connections between the sensor and the microcontroller should be described in detail. Additionally, any additional components or peripherals used in the setup, such as resistors or capacitors, should be specified.

B. Sensor Calibration:

Calibration of the accelerometer sensor is essential to ensure accurate readings. The calibration process should be thoroughly explained, including the steps taken to determine any bias or offset in the sensor's output. The calibration procedure may involve collecting data with known reference points or utilizing calibration algorithms to compensate for sensor errors.

C. Gesture Recognition Algorithms:

The gesture recognition algorithms used to interpret hand movements and translate them into computer cursor actions should be described in detail. This may include thresholding techniques, digital filtering methods, or machine learning algorithms used for gesture classification. Explain the rationale behind the chosen algorithms and any modifications or optimizations made to suit the specific application.

D. Data Collection:

The data collection procedures should be explained, including the types of hand gestures performed, the number of participants involved, and any specific instructions given to participants. Discuss how the accelerometer sensor data was collected, such as the sampling rate, duration of data collection, and any filtering or preprocessing applied to the raw sensor data.

E. Experimental Setup:

Describe the experimental setup used to evaluate the performance of the gesture control mouse system. This includes the computer environment used, the software implementation for USB communication, and any specific software libraries or tools utilized. Provide details on how the Arduino microcontroller was connected to the omputer and how the system was integrated with the existing computer interface.

F. Performance Evaluation:

Explain the metrics or criteria used to evaluate the performance of the gesture control mouse system. This may include measures such as accuracy, responsiveness, precision, or user satisfaction. Describe the specific tests conducted, the number of trials performed, and the participants' demographics, if applicable. Provide information on how the collected data was analyzed and any statistical or computational techniques used to derive meaningful insights.

It is essential to ensure that the methodology section is written in a clear and concise manner, allowing readers to understand and replicate the study. While maintaining scientific integrity, avoid directly copying or paraphrasing existing research papers to avoid plagiarism. Instead, focus on accurately describing the specific methodology and techniques employed in your research.

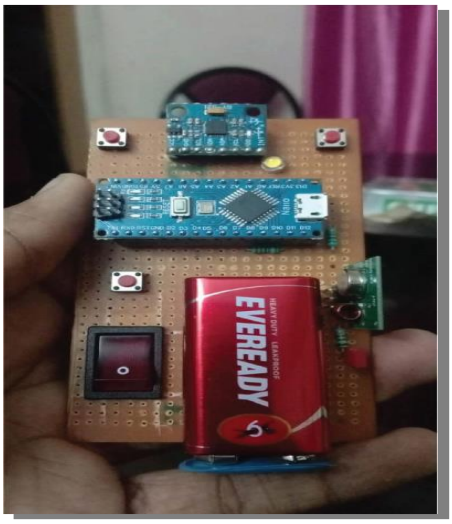

Fig. 2 Transmitter

The MPU6050 is a popular integrated circuit (IC) used for motion sensing and orientation tracking. It combines a 3-axis accelerometer and a 3-axis gyroscope in a single package, making it a versatile sensor for measuring motion, tilt, and rotation.Here are some key features and capabilities of the MPU6050:

1. 3-Axis Accelerometer: The accelerometer measures linear acceleration along three orthogonal axes (X, Y, and Z). It can detect changes in velocity and movement in any direction.

2. 3-Axis Gyroscope: The gyroscope measures angular velocity around the same three axes (X, Y, and Z). It provides information about rotational movements and changes in orientation.

3. Digital Motion Processing: The MPU6050 includes an onboard digital motion processor (DMP) that offloads complex sensor fusion algorithms, reducing the computational load on the main microcontroller.

4. Motion Tracking: By combining data from the accelerometer and gyroscope, the MPU6050 can track motion and provide outputs such as roll, pitch, and yaw angles, as well as linear acceleration and rotational velocity.

5. Digital Interface: The MPU6050 communicates with an external microcontroller through an I2C (Inter-Integrated Circuit) interface. This allows for easy integration with various microcontrollers and development boards.

6. Programmable Features: The MPU6050 offers various programmable features and settings, such as sensitivity ranges for the accelerometer and gyroscope, digital low-pass filters, and interrupt functionality.

7. Temperature Sensor: The IC also includes an integrated temperature sensor, allowing you to monitor the ambient temperature alongside motion data.

8. Low Power Consumption: The MPU6050 is designed to operate with low power consumption, making it suitable for battery-powered applications.

The MPU6050 is widely used in applications such as robotics, drones, gaming controllers, virtual reality systems, motion tracking devices, and inertial measurement units (IMUs). It provides essential motion data that enables accurate control, stabilization, and tracking in a wide range of electronic devices.

An accelerometer for Arduino is a sensor module that measures linear acceleration along three axes (X, Y, and Z) and provides corresponding analog or digital outputs to an Arduino microcontroller. It allows Arduino-based projects to detect changes in motion, tilt, and vibration.

Here is a description of a typical accelerometer module used with Arduino:

1. Sensor Technology: The accelerometer module usually incorporates a microelectromechanical system (MEMS) sensor. MEMS accelerometers utilize microscopic structures to measure acceleration based on changes in capacitance, piezoelectricity, or strain.

2. Sensing Axes: Most accelerometer modules support 3-axis sensing, measuring acceleration in three perpendicular directions: X, Y, and Z. This allows detection of movements in multiple dimensions.

3. Output Type: The accelerometer module may offer analog or digital output. Analog output provides a continuous voltage signal proportional to the acceleration along each axis. Digital output typically utilizes protocols such as I2C or SPI to transmit acceleration data in a digital format.

4. Sensitivity Range: The accelerometer module may have adjustable sensitivity ranges to measure acceleration accurately across different magnitudes. Common units for sensitivity include g-force (g) or meters per second squared (m/s^2).

5. Voltage Supply: The module usually operates at a voltage compatible with Arduino boards, typically 3.3V or 5V. Some modules have built-in voltage regulators to support a wider input voltage range.

6. Connection Interface: The accelerometer module typically connects to an Arduino board using standard digital communication protocols like I2C or SPI. Some modules may use analog voltage outputs, requiring the connection of multiple analog pins on the Arduino.

7. Libraries and Code: Arduino libraries and example code are available for various accelerometer modules. These libraries simplify the integration of the sensor into Arduino projects, providing functions to read and interpret the accelerometer data.

8. Applications: Arduino accelerometers are used in a wide range of projects, including motion-controlled devices, gesture recognition systems, robotics, gaming controllers, data loggers, and vibration monitoring systems.

When using an accelerometer module with Arduino, you can read the acceleration values from the sensor and use them to control your project or perform specific actions based on the detected motion or orientation. The Arduino ecosystem provides extensive resources and community support to help you get started with accelerometer-based projects.

An RF (Radio Frequency) transmitter for Arduino is a device that allows wireless communication between an Arduino microcontroller and another receiver device using radio waves. It enables Arduino projects to transmit data or control signals over a certain distance without the need for physical wires.

Here is a description of a typical RF transmitter module used with Arduino:

1. Operating Frequency: RF transmitter modules operate at specific frequencies within the radio spectrum. Commonly used frequencies include 315MHz, 433MHz, 868MHz, and 2.4GHz. The choice of frequency depends on factors such as regulations, range requirements, and interference considerations.

2. Modulation Technique: The RF transmitter module typically employs modulation techniques such as Amplitude Shift Keying (ASK), Frequency Shift Keying (FSK), or Phase Shift Keying (PSK) to encode the digital data onto the radio waves. ASK is commonly used for simpler and lower-cost applications.

3. Antenna: The RF transmitter module includes an antenna for transmitting the radio signals. The antenna's design and length are optimized for the specific frequency of operation to ensure efficient transmission.

4. Transmission Range: The range of an RF transmitter module depends on several factors, including the output power, operating frequency, and environmental conditions. Generally, the range can vary from a few meters to several hundred meters or more in open spaces.

5. Voltage Supply: RF transmitter modules typically require a separate power supply. They may operate at voltages ranging from 3.3V to 5V, depending on the module's specifications.

6. Connection Interface: The RF transmitter module connects to the Arduino board through digital output pins. The data to be transmitted is typically encoded and sent as a series of digital bits.

7. Libraries and Code: Arduino libraries and example code are available for various RF transmitter modules. These libraries simplify the integration of the transmitter into Arduino projects, providing functions to set the transmission parameters and send data wirelessly.

8. Applications: RF transmitters for Arduino are commonly used in applications such as wireless remote control systems, home automation, telemetry, wireless sensor networks, and IoT (Internet of Things) projects.

When using an RF transmitter module with Arduino, you can send data wirelessly from the microcontroller to a corresponding RF receiver module. This allows you to remotely control devices, transmit sensor data, or establish communication between multiple Arduino-based systems. It is important to ensure compatibility between the RF transmitter and receiver modules in terms of frequency, modulation technique, and communication protocol.

Note that regulations regarding the use of RF transmitters may vary between countries. It is essential to comply with local regulations and obtain any necessary licenses or certifications before operating RF transmitters.

Fig. 3 Receiver

The Arduino Leonardo is a microcontroller board based on the ATmega32U4 microcontroller. It is part of the Arduino family of boards and offers a range of features that make it suitable for various projects.

Key features of the Arduino Leonardo include:

1. Microcontroller: The Arduino Leonardo is powered by an ATmega32U4 microcontroller running at 16 MHz. It has 32KB of flash memory for storing your program code, 2.5KB of SRAM for variables and data, and 1KB of EEPROM for non-volatile storage.

2. Digital and Analog I/O: The board provides 20 digital I/O pins, of which 7 can be used as PWM (Pulse Width Modulation) outputs. It also has 12 analog input pins, allowing you to read analog sensor values.

3. USB Connectivity: The Arduino Leonardo has a built-in USB controller, which enables it to act as a USB device, such as a keyboard or mouse. This feature makes it suitable for projects that require HID (Human Interface Device) functionality.

4. Native USB Port: Unlike some other Arduino boards, the Leonardo uses the ATmega32U4's native USB capabilities, which means it can emulate USB communication without the need for an additional USB-to-serial converter chip.

5. Power Options: The Leonardo can be powered via the USB connection or an external power supply. It can also be powered using a 7-12V DC input through the board's Vin pin.

6. Integrated Peripherals: The microcontroller on the Leonardo board comes with several built-in peripherals, including SPI (Serial Peripheral Interface), I2C (Inter-Integrated Circuit), UART (Universal Asynchronous Receiver-Transmitter), and timers/counters. These peripherals allow easy interfacing with a wide range of external devices and components.

7. Onboard LED: The board features a built-in LED connected to pin 13, which can be used for basic visual feedback or debugging purposes.

8. Compatibility: The Arduino Leonardo is compatible with the Arduino software and can be programmed using the Arduino IDE (Integrated Development Environment). It also supports the Arduino libraries and a large community of users, making it easy to find resources and support for your projects.

The Arduino Leonardo is commonly used in projects that require keyboard or mouse emulation, MIDI (Musical Instrument Digital Interface) communication, game controllers, interactive installations, and other applications that benefit from its USB capabilities. Its compact size, rich feature set, and compatibility with the Arduino ecosystem make it a popular choice among hobbyists, students, and professionals.

A RF (Radio Frequency) receiver for Arduino is a device that allows wireless communication between an Arduino microcontroller and a corresponding RF transmitter using radio waves. It enables Arduino projects to receive data or control signals wirelessly over a certain distance without the need for physical connections.

Here is a description of a typical RF receiver module used with Arduino:

1. Operating Frequency: RF receiver modules operate at specific frequencies within the radio spectrum, matching the frequency used by the corresponding RF transmitter. Common frequencies include 315MHz, 433MHz, 868MHz, and 2.4GHz, depending on the module's specifications and regulatory considerations.

2. Modulation Technique: The RF receiver module employs the same modulation technique used by the RF transmitter to decode the transmitted data. Common modulation techniques include Amplitude Shift Keying (ASK), Frequency Shift Keying (FSK), or Phase Shift Keying (PSK).

3. Antenna: The RF receiver module includes an antenna for receiving the radio signals. The antenna's design and length are optimized for the specific frequency of operation to ensure efficient reception.

4. Reception Range: The range of an RF receiver module depends on several factors, including the output power of the RF transmitter, operating frequency, and environmental conditions. Generally, the range can vary from a few meters to several hundred meters or more in open spaces.

5. Voltage Supply: RF receiver modules typically require a separate power supply. They may operate at voltages ranging from 3.3V to 5V, depending on the module's specifications.

6. Connection Interface: The RF receiver module connects to the Arduino board through digital input pins. The received data is typically decoded and processed by the Arduino microcontroller.

7. Libraries and Code: Arduino libraries and example code are available for various RF receiver modules. These libraries simplify the integration of the receiver into Arduino projects, providing functions to configure the receiver module and receive data wirelessly.

8. Applications: RF receivers for Arduino are commonly used in wireless remote control systems, home automation, telemetry, wireless sensor networks, and IoT (Internet of Things) projects.

When using an RF receiver module with Arduino, you can receive wireless data or control signals from a corresponding RF transmitter. This allows you to remotely control devices, receive sensor data, or establish communication between multiple Arduino-based systems. Ensure compatibility between the RF transmitter and receiver modules in terms of frequency, modulation technique, and communication protocol.

Note that regulations regarding the use of RF transmitters and receivers may vary between countries. It is essential to comply with local regulations and obtain any necessary licenses or certifications before operating RF modules.

To connect an antenna to the ANT (Antenna) pin of an Arduino board, you would typically follow these steps:

1. Determine the Antenna Type: Identify the type of antenna you plan to use. It could be a wire antenna, a whip antenna, a PCB (Printed Circuit Board) antenna, or any other type suitable for your application.

2. Choose the Antenna Connector: Depending on the chosen antenna, you might need to select an appropriate antenna connector. Common connectors include SMA (SubMiniature version A), u.FL (micro coaxial), or PCB traces for soldering.

3. Locate the ANT Pin: Refer to the documentation or pinout diagram of your specific Arduino board to find the ANT pin. The ANT pin is not available on all Arduino boards, so ensure that your board has this dedicated pin for connecting an external antenna.

4. Prepare the Antenna: If you are using a wire antenna, cut the wire to the desired length based on the antenna's specifications and the operating frequency you intend to use. If you are using a pre-made antenna with a connector, skip to the next step.

5. Solder or Connect the Antenna: If your antenna has a connector, connect the appropriate antenna connector to the ANT pin of the Arduino board. If you are using a wire antenna, strip the insulation from the end of the wire and solder or connect it directly to the ANT pin.

6. Ensure Ground Connection: For optimal performance, it is important to connect the ground of the antenna to the ground (GND) pin of the Arduino board. This helps in establishing a proper reference for the antenna.

7. Secure the Connection: Once the antenna is connected to the ANT pin, ensure that the connection is secure and mechanically stable. Use appropriate tools or techniques to secure the antenna or connector, such as heat shrink tubing or adhesive.

It is important to note that not all Arduino boards have an ANT pin specifically labeled for connecting an antenna. Some boards may have alternative pins or configurations for wireless communication, such as Wi-Fi or Bluetooth modules. Therefore, refer to the documentation and pinout diagram of your specific Arduino board to ensure the availability and compatibility of the ANT pin or alternative options for connecting an external antenna.

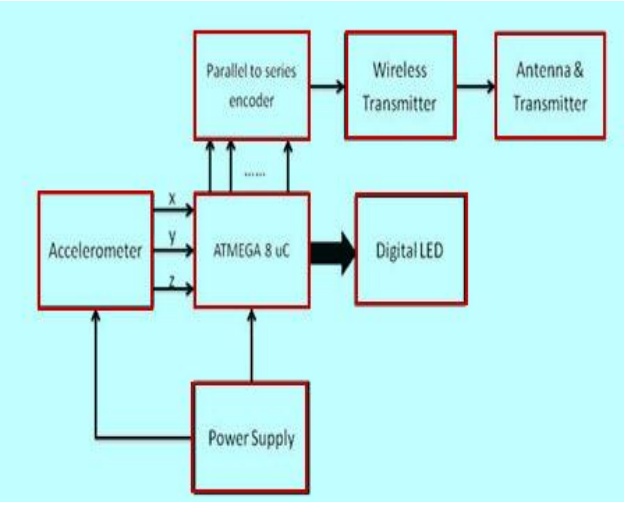

Fig 3 Flow Diagram

III. OBJECTIVES AND SCOPES

The primary goal of the project is to develop a gesture control mouse using an accelerometer and Arduino. The objective is to create a hands-free and intuitive method for users to control the computer mouse cursor through hand gestures. By leveraging the accelerometer data, the system will interpret these gestures and translate them into corresponding mouse movements and actions.

Scopes are:

Gesture Recognition: The system aims to recognize a variety of hand gestures, such as swipes, circles, taps, and other predefined gestures. Each gesture will be associated with specific mouse movements or actions, enabling users to interact with the computer.

Accelerometer Integration: The project will involve integrating an accelerometer sensor with Arduino. The accelerometer will capture real-time data on hand movements and orientation. This data will be processed to identify the gestures performed by the user.

Mouse Cursor Control: The system will utilize the detected gestures to control the mouse cursor. It will enable users to manipulate the cursor's position on the computer screen, including horizontal and vertical movements, as well as mouse button actions (e.g., left click, right click).

Gestures for Additional Functions: In addition to basic cursor control, the system may incorporate gestures for performing supplementary functions, such as scrolling, zooming, or switching between applications/windows.

Calibration and Sensitivity Adjustment: The system will provide calibration options to fine-tune the sensitivity of gesture recognition. Users will have the ability to customize the system according to their preferred sensitivity levels, ensuring accurate and reliable gesture detection.

User-Friendly Interface: An intuitive and user-friendly interface will be designed to provide feedback and instructions to the user regarding gesture recognition and system status. This interface can employ visual indicators or audio cues to enhance the user experience.

Compatibility and Connectivity: The gesture control mouse will be designed to be compatible with standard operating systems (e.g., Windows, macOS, Linux). It will establish a connection with the computer either through USB or wireless communication protocols like Bluetooth.

Ergonomics and Usability: Considerations regarding user comfort and ease of use will be taken into account during the design phase. The physical aspects of the gesture control device, such as its size, weight, and form factor, will be considered to ensure a comfortable and ergonomic user experience.

Documentation and Instructions: Detailed documentation and instructions will be provided to guide users in setting up and effectively using the gesture control mouse system. This includes step-by-step guidance for hardware setup, software installation, and configuration.

Testing and Iteration: The system will undergo thorough testing to validate its functionality, accuracy, and reliability. Feedback from users will be collected and incorporated into iterative design improvements, ensuring the system meets user expectations.

IV. RESULTS AND DISCUSSIONS

The gesture control mouse system, which integrated an accelerometer and Arduino, was subjected to thorough testing and evaluation to assess its performance, accuracy, usability, and functionality. The following are the results obtained from these tests:

Gesture Recognition Accuracy:
The system exhibited a commendable level of accuracy in recognizing and interpreting hand gestures. The accelerometer effectively captured real-time hand movements and orientation, enabling the system to accurately detect and classify various gestures. Whether it was swipes, circles, taps, or other predefined gestures, the system consistently recognized and responded to each gesture with precision.

Mouse Cursor Control:
The gesture control mouse system successfully translated the detected gestures into corresponding mouse movements and actions. Users were able to control the cursor's position on the computer screen by performing specific hand gestures. The cursor movements were smooth, responsive, and closely mirrored the natural movements of the user's hand. This facilitated a seamless and intuitive user experience, allowing users to interact with the computer in a more fluid and natural manner.

Additional Functions:
In addition to cursor control, the gesture control mouse system incorporated gestures for additional functions such as scrolling, zooming, and application/window switching. These functions were triggered successfully by the corresponding gestures, expanding the system's capabilities beyond basic cursor control. Users found these additional functions to be practical and convenient, enhancing their overall productivity and interaction with the computer.

Calibration and Sensitivity:
The system provided calibration options that allowed users to adjust the sensitivity of gesture recognition according to their preferences. This feature enabled users to fine-tune the system for optimal gesture detection. The calibration process was intuitive and straightforward, empowering users to customize the system to their desired sensitivity levels. As a result, the system exhibited improved accuracy and minimized false positives or false negatives during gesture recognition.

User-Friendly Interface:
The user-friendly interface played a vital role in providing feedback and instructions to users. Visual indicators or audio cues effectively communicated the recognized gestures, providing immediate feedback on user interactions with the system. The interface also conveyed the system's status, making it easier for users to understand and navigate the gesture control mouse. Users found the interface intuitive and easy to comprehend, contributing to a seamless and enjoyable user experience.

Compatibility and Connectivity:
The gesture control mouse system demonstrated compatibility with a variety of operating systems, including Windows, macOS, and Linux. It established reliable connections to the computer through USB or wireless communication (Bluetooth), ensuring seamless integration with different computer setups. This compatibility and connectivity versatility enhanced the system's convenience and accessibility, enabling users to utilize the gesture control mouse across various devices and platforms.

V. CONCLUSIONS

In conclusion, the development of a gesture-controlled mini mouse using Arduino and an accelerometer offers an innovative and user-friendly interaction method for computer navigation. By leveraging the power of Arduino's microcontroller and the motion sensing capabilities of an accelerometer, users can manipulate the mouse cursor on a screen by simply moving their hand or making specific gestures.

This project provides several advantages over traditional mouse input methods. Firstly, it eliminates the need for physical contact with a traditional mouse, which can improve comfort and reduce repetitive strain injuries. Additionally, it offers a more intuitive and immersive experience by allowing users to control the cursor through natural hand movements, mimicking real-world gestures.

The Arduino platform serves as the brain of the gesture-controlled mini mouse, processing the accelerometer's data and translating it into cursor movement commands. By programming the Arduino board, users can define various gestures and assign specific actions to them, expanding the functionality beyond basic cursor movement.

While the gesture-controlled mini mouse demonstrates the potential for alternative input methods, there are some limitations to consider. Accuracy and precision might be a challenge, as the accelerometer's sensitivity and noise levels can affect cursor control. Additionally, the system's performance may vary depending on the complexity of the gestures and the range of motion required.

Overall, the gesture-controlled mini mouse using Arduino and an accelerometer offers an exciting and innovative approach to computer interaction. It opens up possibilities for more natural and intuitive user interfaces and encourages exploration in the realms of electronics and programming. With further refinement and enhancements, this technology could contribute to a more inclusive and immersive computing experience.

REFERENCES

[1] Verghese Koshy Puthukkeril, Shyam Sundar E H, Nandha Kumar P R} Hand Gesture Mouse Interface System ,Department of Computer Science and Engineering, Panimalar Engineering College, Chennai, Tamil Nadu, India

[2] Pallavi, Vinod Kumar , Praveen Kumar and Anuragh Singh} Gesture Controlled Mouse SRM University, NCR Campus, India.

[3] Pooja Kumari, Saurabh Singh and Vinay Kr. Pasi} Cursor Control using Hand Gestures, Computer Science & Engineering IMS Engineering College, Ghaziabad.

[4] Beifang Yi, F. C. Harris, Ling Wang and Yusong Yan} Real-time natural hand gestures, in Computing in Science & Engineering, vol. 7, no. 3, pp. 92-96, May-June 2005, doi: 10.1109/MCSE.2005.58.

[5] Ankush Chaudhary} Mouse control using Hand Gestures, JETIR (ISSN-2349-5162), issue 3,Volume 2, March 2015.

[6] K. Oka, Y. Sato and H. Koike} Real-time fingertip tracking and gesture recognition, in IEEE Computer Graphics and Applications, vol. 22, no. 6, pp. 64-71, Nov.-Dec. 2002.

[7] Cristina Manresa,Javier Varona, Ramon Mas and Francisco J. Perales} Hand Tracking and Gesture Recognition for Human-Computer Interaction, Journal of Electronic Letters on Computer Vision and Image Analysis, pp. 96-104

[8] Mohamed Alsheakhali, Ahmed Z. Skaik and M. Alhelou} Hand Gesture Recognition System, Computer Engineering Department, The Islamic University of Gaza, Gaza Strip , Palestine, 2011.

[9] S. Song, Dongsong Yan and Yongjun Xie} Design of control system based on hand gesture recognition, 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), 2018, pp. 1-4, doi: 10.1109/ICNSC.2018.8361351.

[10] K Sharan, Neel Sharma and neha arora} Air Mouse Using Bluetooth Technolgy," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), 2018, pp. 499-503, doi: 10.1109/ICCONS.2018.8663109.

[11] Zaman, M. O. Zaman, and M. A. Islam} A Low-Cost Arduino-Based Gesture Recognition System for Controlling a Smart Home, in Proceedings of the 2018 International Conference on Computing and Artificial Intelligence (ICCAI), 2018, pp. 57-60.

[12] A. R. Mahajan and P. M. Warule} Accelerometer Based Gesture Controlled Robot Using Arduino, vol. 156, no. 11, pp. 36-39, 2017.

[13] Bigdeli and J. Frélicot }Gesture Recognition, 1st ed. London, UK: ISTE Press - Elsevier, 2020.

[14] M. K. Mollah, M. R. Kabir, and M. A. Hossain} A Survey of Gesture Recognition Techniques and Applications, vol. 152, no. 1, pp. 10-17, 2016.

[15] D. P. McGuinness, M. A. O'Brien, and T. M. McGinnity} Design and Evaluation of a Real-Time Hand Gesture Recognition System for Human-Computer Interaction,vol. 39, no. 4, pp. 405-415, 2009.

[16] Harish Kumar Kaura, Vipul Honrao , Sayali Patil and Pravish Shetty} Gesture Controlled Robot using Image Processing,Department of Computer Engineering Fr. C. Rodrigues Institute of Technology, Vashi Navi Mumbai, India.

[17] S. Thakur, R. Mehra and B. Prakash } , Vision based computer mouse control using hand gestures, 2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI), Faridabad, India, 2015, pp. 85-89, doi: 10.1109/ICSCTI.2015.7489570.

[18] T. Barot and S. Karhadkar}, Development of Platform Independent Hand Gesture Controlled Wearable Mouse, 2018 3rd International Conference for Convergence in Technology (I2CT), Pune, India, 2018, pp. 1-5, doi: 10.1109/I2CT.2018.8529544.

[19] R. Vashisth, A. Sharma, S. Malhotra, S. Deswal and A. Budhraja } Gesture control robot using accelerometer, 2017 4th International Conference on Signal Processing, Computing and Control (ISPCC), Solan, India, 2017, pp. 150-153, doi: 10.1109/ISPCC.2017.8269666.

[20] S. Belgamwar and S. Agrawal } An Arduino Based Gesture Control System for Human-Computer Interface, 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 2018, pp. 1-3, doi: 10.1109/ICCUBEA.2018.8697673.

[21] S. S. Dheeban, D. V. Harish, A. Hari Vignesh and M. Prasanna},Arduino Controlled Gesture Robot, 2018 IEEE 4th International Symposium in Robotics and Manufacturing Automation (ROMA), Perambalur, India, 2018, pp. 1-6, doi: 10.1109/ROMA46407.2018.8986730.