TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

KATHMANDU ENGINEERING COLLEGE

KALIMATI, KATHMANDU

MAJOR PROJECT PROPOSAL REPORT ON

H.A.N.D. : HAPTIC ANALYSIS FOR NATURAL DEXTERITY

BY

KRITIKA DAHAL - KAT078BEI007

MAHAN SIGDEL - KAT078BEI009

NISHANA THAPA - KAT078BEI011

SABINA GHIMIRE - KAT078BEI016

TO

DEPARTMENT OF ELECTRONICS, COMMUNICATION AND INFORMATION ENGINEERING

KATHMANDU, NEPAL

MAY , 2024

Acknowledgement

We would like to express our sincere gratitude to the Department of Electronics, Communication, and Information, Kathmandu Engineering College, for providing us with an opportunity to initiate our major project as a part of a syllabus. And a special thanks goes to our supervisor, **Er. Sagun Manandhar** and **Er. Anmol Bajracharya**, for assisting and aiding us in every possible way in this project. We are deeply thankful to the Project Coordinator, **Er. Sujan Shrestha**. We would also like to mention a special note of thanks to the esteemed Head of the Department of Electronics, Communication, and Information, **Er. Suramya Dahal**. We would also like to extend our gratitude to every teacher of the Department of Electronics, Communication and Information, for their guidance.

Table of Contents

List of Figures

List of Tables

List of Abbreviation

MPU: Motion Processing Unit

IMU: Inertial Measurement Unit

HCI: Human-Computer Interaction

DoF: Degrees of Freedom

VR: Virtual Reality

AR: Augmented Reality

PLA: Polylactic Acid

ESP: Espressif Systems

IoT: Internet of Things

VAR: Virtual Augmented Reality

MEMS: Micro-Electro-Mechanical Systems

DMP: Digital Motion Processor

TCP/IP: Transmission Control Protocol/Internet Protocol

CPU: Central Processing Unit

GPU: Graphics Processing Unit

Wi-Fi: Wireless Fidelity

RISC: Reduced Instruction Set Computer

IEEE: Institute of Electrical and Electronics Engineers

HTTP: Hypertext Transfer Protocol

IDE: Integrated Development Environment

API: Application Programming Interface

EEVEE: Enhanced Efficient Virtual Environment Engine

Chapter 1: Introduction

1.1 Background Theory

The rapid advancement of human-computer interaction (HCI) technologies has paved the way beyond traditional input devices towards more intuitive interfaces. Among these, Gesture-based systems have emerged as a powerful mode of interaction where users engage with digital systems through physical movements, particularly of the hands. Human hands are capable of complex movements and precise control, making them an ideal medium for gesture-based input. This evolution is particularly significant in applications such as gaming, virtual and augmented reality, assistive technology, and robotics.

Leveraging this, modern wearable systems incorporate a variety of sensors to translate hand and finger gestures into digital commands. The MPU6050, a 6-DoF inertial measurement unit (IMU) combining a 3-axis gyroscope and a 3-axis accelerometer, is widely used for real-time tracking of wrist orientation and movement, crucial for recognizing hand gestures [1], [2]. For detecting finger movements, flex sensors are employed, which vary resistance based on bending, providing analog signals proportional to flexion [3]. Additionally, mechanical switches, activated by levers attached to finger components, offer a digital input method with tactile feedback. Data from these sensors are processed by microcontrollers like the Arduino Mega, which provides ample I/O support. For seamless interaction with external applications, the ESP8266 Wi-Fi module facilitates wireless data transmission, ensuring low-latency communication between the wearable device and the host system , ensuring real-time interaction within the gaming environment [4].

Device housing commonly utilizes 3D-printed PLA filament, chosen for its lightweight properties, customizability, and suitability for ergonomic wearable enclosures that maintain sensor positioning and comfort during extended use [5]. In the software domain, gesture data is mapped to real-time responses within a digital environment. Game engines, such as Unreal Engine, support external hardware integration, enabling the visualization of hand movements from a first-person perspective [6]. This establishes a seamless connection between physical actions and virtual reactions, particularly effective in immersive puzzle-based or simulation games. Ultimately, gesture-based input systems enhance user experience by aligning digital control with natural human motion, proving ideal for applications ranging from gaming to assistive technology.

1.2 Problem Statement

Modern digital interaction interfaces, particularly in gaming, continue to struggle with creating truly immersive user experiences. Traditional input devices like keyboards, mouse, and gamepads impose artificial constraints on user interaction, fundamentally disconnecting players from the virtual environment. Although modern technologies have introduced motion controllers and VR peripherals, they are often expensive, bulky, or dependent on proprietary software and hardware ecosystems, making them inaccessible for general-purpose or educational use. Existing gesture-based systems typically employ either flex sensors or IMU sensors alone, leading to either limited resolution in motion tracking or high computational complexity. These systems often fail to provide intuitive, responsive control without noticeable latency or drift, especially in low-cost implementations, rendering them ineffective for seamless, real-time interactive experiences.

This project addresses these critical challenges by developing an innovative, low-cost, real-time gesture recognition system by integrating a hybrid sensor approach combining MPU6050, flex sensors, and tactile switches into a custom wearable interface. The system will serve as a natural, responsive hand-based input interface for controlling a first-person perspective simulation developed in Unreal Engine. The system aims to minimize computational latency, enhance gesture recognition accuracy, and provide tactile feedback while being affordable, modifiable, and scalable. This project also explores the potential of wearable gesture systems in creating immersive interactive experiences by translating physical hand movements into precise, real-time virtual actions.

1.3 Objectives

- To design and develop a wearable hardware system that captures real-time hand and finger movements using MPU6050, flex sensors, and mechanical switches.
- To integrate the hardware inputs into a simulation-based game environment developed in Unreal Engine, enabling real-time gesture-based control.
- To process and transmit gesture data from the wearable device to a host system using Arduino Mega and ESP8266.
- To evaluate the system's responsiveness and accuracy in mapping physical gestures to virtual hand movements for immersive interaction.

1.4 Scope

The project encompasses several key areas for development and future expansion:

- Integration with machine learning models to improve gesture recognition accuracy
- Addition of haptic feedback for more immersive interaction
- Expansion to full-body motion capture using additional wearable sensors
- Development of a mobile or desktop interface for visualizing and mapping gestures
- Incorporating voice + gesture multimodal control systems

1.5 Applications

The data glove system finds applications in various fields including:

- Virtual and Augmented Reality (VAR)
- Assistive Technology
- Gaming
- Robotics Control
- Smart Environments / IoT Applications
- Educational Tools

Chapter 2: Literature Review

2.1 Literature Review

Rautaray, S. S., & Agrawal, A. (2013) [7], focuses on developing a vision-based gesture recognition system that eliminates the need for physical input devices, allowing users to interact with digital objects using natural hand movements. The system employs computer vision algorithms for gesture detection, segmentation, tracking, and recognition, converting hand gestures into meaningful commands for real-time applications. The study highlights the limitations of traditional input methods, such as keyboards and mice, and emphasizes the advantages of markerless tracking systems in improving usability and immersion in dynamic environments. The proposed system contributes to the growing field of gesture-based interfaces.

Chowdhury and Haque (2013) [8], developed an animatronic hand controller that utilized flex sensors and servo motors, showcasing an innovative approach to robotic hand manipulation. The flex sensors played a crucial role in detecting finger bending motions, translating them into electrical signals that corresponded to hand gestures. These signals were then processed by an Arduino microcontroller, which controlled the servo motors responsible for mimicking natural finger movements. By integrating this sensor-motor system, the researchers enabled precise gesture-based control, allowing for applications in robotic prosthetics, remote-controlled manipulators, and assistive technology. Their work underscored the potential of Arduino-driven systems for creating cost-effective and accessible animatronic solutions, paving the way for further advancements in human-machine interaction and gesture-controlled robotics.

Kumar et al. (2012) [3], introduced the DG5 VHand 2.0, a wireless data glove designed to enhance gesture-based interactions in virtual and augmented reality. The glove featured flex sensors, which captured precise finger movements, and Bluetooth connectivity, ensuring seamless real-time communication with external devices. To process gesture inputs effectively, the researchers employed a K-Nearest Neighbors (K-NN) classifier, allowing their system to recognize complex hand motions with high accuracy. The glove was particularly effective for applications such as air-writing, where users could trace letters mid-air, and 3D sketching, enabling intuitive and immersive design processes within virtual environments. Their findings underscored the potential of wearable sensor technology in expanding the boundaries of human-computer interaction, demonstrating how real-time gesture recognition could improve accessibility, user experience, and creative workflows in emerging technologies.

Sturman and Zeltzer (1994) [9], conducted an extensive survey of early glove-based input devices, highlighting the technological innovations that laid the foundation for modern hand-tracking systems. Among these, the VPL DataGlove utilized optical fibers to measure finger

flexion, providing a level of precision that was groundbreaking at the time. Meanwhile, the Dexterous HandMaster (DHM) relied on Hall-effect sensors, which detected changes in magnetic fields to determine finger movements. These devices represented significant strides in human-computer interaction, offering high-accuracy motion tracking crucial for applications in virtual reality (VR) and robotic control.

Huang (2017) [10], explored the application of the MPU6050 sensor in flight control systems, utilizing its integrated accelerometers and gyroscopes to detect tilt angles with high accuracy. This sensor's ability to measure both linear acceleration and angular velocity made it ideal for stabilizing aircraft movements and ensuring precise orientation tracking.

Chapter 3: Related Theory

3.1 Hardware

MPU6050 Sensor: The MPU6050 is a widely used 6-axis MEMS-based Inertial Measurement Unit (IMU) that integrates a 3-axis accelerometer and a 3-axis gyroscope within a single chip.It is capable of detecting linear acceleration in the range of ±2g to ±16g and angular velocity from ±250°/s to ±2000°/s, making it highly suitable for motion tracking and gesture recognition applications. A notable feature of the MPU6050 is its onboard Digital Motion Processor (DMP), which performs real-time sensor fusion using algorithms such as Kalman or complementary filtering. This significantly reduces noise and drift in gyroscopic data, enabling stable orientation tracking through the calculation of quaternions or Euler angles (roll, pitch, and yaw). The sensor communicates with microcontrollers through the I²C interface, supporting clock speeds between 100 kHz and 400 kHz for efficient data exchange. Its compact design, reliability, and real-time capabilities make it ideal for wearable systems. It enables precise and responsive motion capture for interactive systems in gesture-based applications.

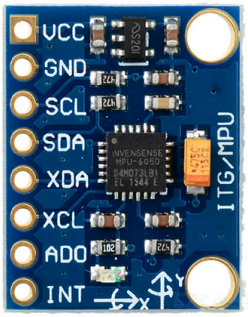


Figure 3.1: MPU6050 Sensor

Arduino Mega: The Arduino Mega is an open-source microcontroller board based on the ATmega2560, designed for projects requiring extensive input/output operations and greater memory capacity. It features 54 digital I/O pins, 16 analog inputs, and four UARTs for serial communication, making it suitable for complex hardware interfacing. With 256 KB of flash memory and a 16 MHz clock speed, it can handle multiple sensors and real-time data processing efficiently. In this project, the Arduino Mega serves as the central controller, managing input from multiple MPU6050 sensors and switches to ensure accurate and synchronized gesture-

based interactions within the game environment.

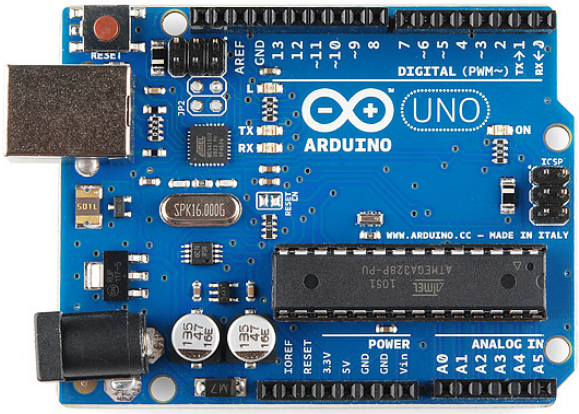


Figure 3.2: Arduino Uno Board

ESP8266: The ESP8266 is a low-cost, high-performance Wi-Fi microcontroller based on a 32-bit RISC CPU core (Tensilica L106), operating at 80–160 MHz. It integrates TCP/IP protocol stack and supports IEEE 802.11 b/g/n standards, enabling wireless connectivity with low power consumption (80 mA active mode). It enables devices to connect to wireless networks and communicate over the internet or within local networks. The ESP8266 supports multiple modes such as station, access point, and both simultaneously, making it highly versatile for wireless communication. It can be programmed using the Arduino IDE and is capable of handling HTTP requests, data transfer, and remote control functionalities. The ESP8266 is used to explore wireless communication possibilities between the hardware controller and the game system, potentially allowing untethered interaction.

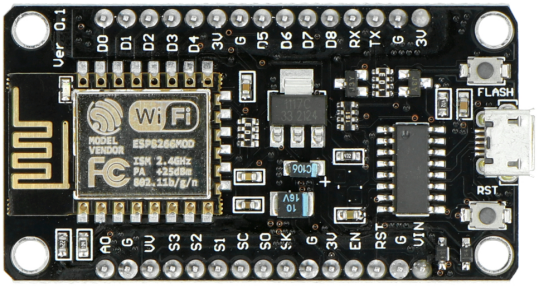


Figure 3.3: ESP8266 Wi-Fi Module

Flex Sensors: Flex sensors are passive resistive devices that change their resistance based on the amount of bend applied to them. Typically constructed using a flexible substrate coated

with conductive ink, their resistance increases as the sensor is bent. This property allows them to detect the degree of bending or curvature, making them suitable for applications involving motion capture, wearable electronics, and gesture recognition. When integrated with microcontrollers, the analog resistance change can be converted into meaningful input data. In this project, flex sensors are considered for detecting finger movements by measuring the degree of bend in each finger.

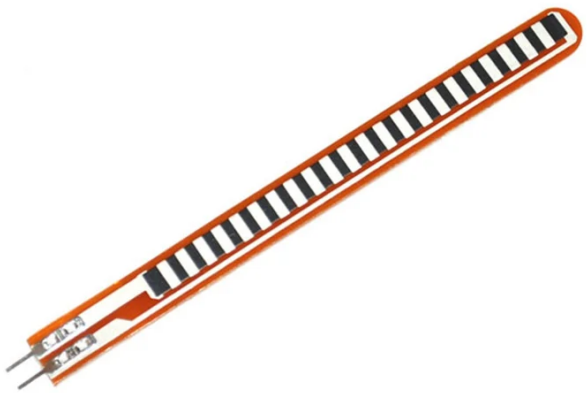


Figure 3.4: Flex Sensor

3.2 Software

Arduino IDE: The Arduino Integrated Development Environment (IDE) is the core programming tool for the Arduino Uno microcontroller in this project. Using C/C++, it configures the microcontroller to process digital signals from switches and analog data from MPU6050 sensors. Essential libraries like Wire.h and MPU650.h manage I²C communication, while serial protocols handle data transmission to the rendering engine. The IDE's debugging tools, including serial monitors, are crucial for verifying signal integrity and latency. Firmware algorithms integrate sensor fusion and debouncing logic for accurate gesture detection. This open-source platform significantly aids rapid prototyping and hardware-software integration for real-time interactive systems.

Rendering Engine: The Rendering Engine is a fundamental software component that generates real-time visual output based on physical hand gestures captured by the hardware. Initially considering Unreal Engine, this prototype utilizes Blender's integrated EEVEE and Cycles engines for rendering and simulation. The engine displays a first-person perspective with visible virtual hands, mirroring gestures like finger bends and wrist rotations instantly. This real-time visual feedback is vital for player immersion and interaction. The engine processes data from the microcontroller, supplied through middleware, to dynamically adjust hand poses and object interactions. Its capability to reflect hardware input with minimal latency ensures intuitive and

responsive gameplay, serving as the core of the gesture-controlled gaming experience. Optimized for low latency with GPU acceleration, it manages environmental elements and physics simulations via Blender's Python API.

Blender: Blender is an open-source 3D creation suite which is utilized for designing and developing game assets. This includes 3D modeling, UV mapping, texturing, and rigging, crucial for creating realistic hand models and interactive objects. Blender's EEVEE and Cycles engines provide real-time previews and high-fidelity final renders, respectively. Its Python scripting capabilities facilitate customization and workflow automation. Blender functions as both a design tool and a visual integration layer, ensuring in-game visuals accurately reflect physical gestures captured by the hardware with minimal latency through automated workflows linking it to the Arduino's output.

Python: Python serves as the middleware layer, connecting hardware data with the rendering engine. Custom scripts, utilizing libraries like PySerial, parse serial data from the Arduino, converting raw sensor values into actionable game inputs. Python's integration with Blender via the bpy module maps gestures to in-game animations and automates tasks. It can also be used externally to interpret sensor data and relay it to the rendering engine via custom protocols. Python's versatility and extensive library support are crucial for efficient data translation and seamless interaction between the wearable hardware and the virtual environment, enabling real-time mapping of physical movements to game commands.

Chapter 4: Feasibility Study

4.1 Technical Feasibility

The project demonstrates strong technical feasibility by combining electronics, communica tion, and information processing in a practical and achievable manner. The Arduino Mega microcontroller acts as the brain of the system, providing sufficient processing power and I/O pins to accommodate multiple sensor inputs, including MPU6050 sensors, tactile push buttons, and optional flex sensors. These electronic components assist in accurately detecting hand gestures and finger movements. For communication, the ESP8266 module is considered for wireless data transfer between the hardware system and the game engine, facilitating real-time interaction with low delay. Serial communication protocols (either wired or wireless) will be utilized to ensure smooth and continuous data exchange from the Arduino to the computer running the game.

The information processing aspect involves translating raw sensor inputs into meaningful com mands for the game. This includes gesture detection using both switch activations and motion readings from the MPU. All of this will be managed using standard Arduino programming, with open-source libraries that expedite development. The modular hardware setup allows for f lexible testing, updates, and part replacement if necessary. Our design supports iterative devel opment, meaning we can test and improve as we build. Key strengths of the project include its adaptability, cost-effective components, and future scalability, making it a reliable and feasible system to implement with the tools and skills available to our team.

4.2 Economic Feasibility

The proposed system demonstrates strong economic feasibility with a cost-effective approach to hardware development and implementation. Most of the required electronic components and sensors are readily available in the local market and within our college resources. The primary components, like Arduino Mega, ESP8266, MPU sensors, and connecting wires, are economically accessible, with relatively low procurement costs compared to specialized gam ing interface systems. The use of PLA filament provides a budget-friendly prototyping solu tion, allowing multiple design iterations without significant financial investment. Open-source software platforms like Arduino IDE and Unreal rendering Engine further reduce development expenses by eliminating expensive proprietary software licensing costs.

The project's modular design enables incremental development, meaning team members can progressively invest in components as needed, spreading out potential expenses. Potential cost

savings are achieved through utilizing existing college laboratory equipment and leveraging team members' existing technical skills, which minimizes additional training or external con sultation expenses. The overall economic viability is enhanced by the project's scalable na ture, potential for future refinement, and the use of widely available, low-cost technological components. The minimal financial requirements make this project an economically attractive research and development initiative within the current institutional infrastructure.

Chapter 5: Methodology

5.1 System Block Diagram

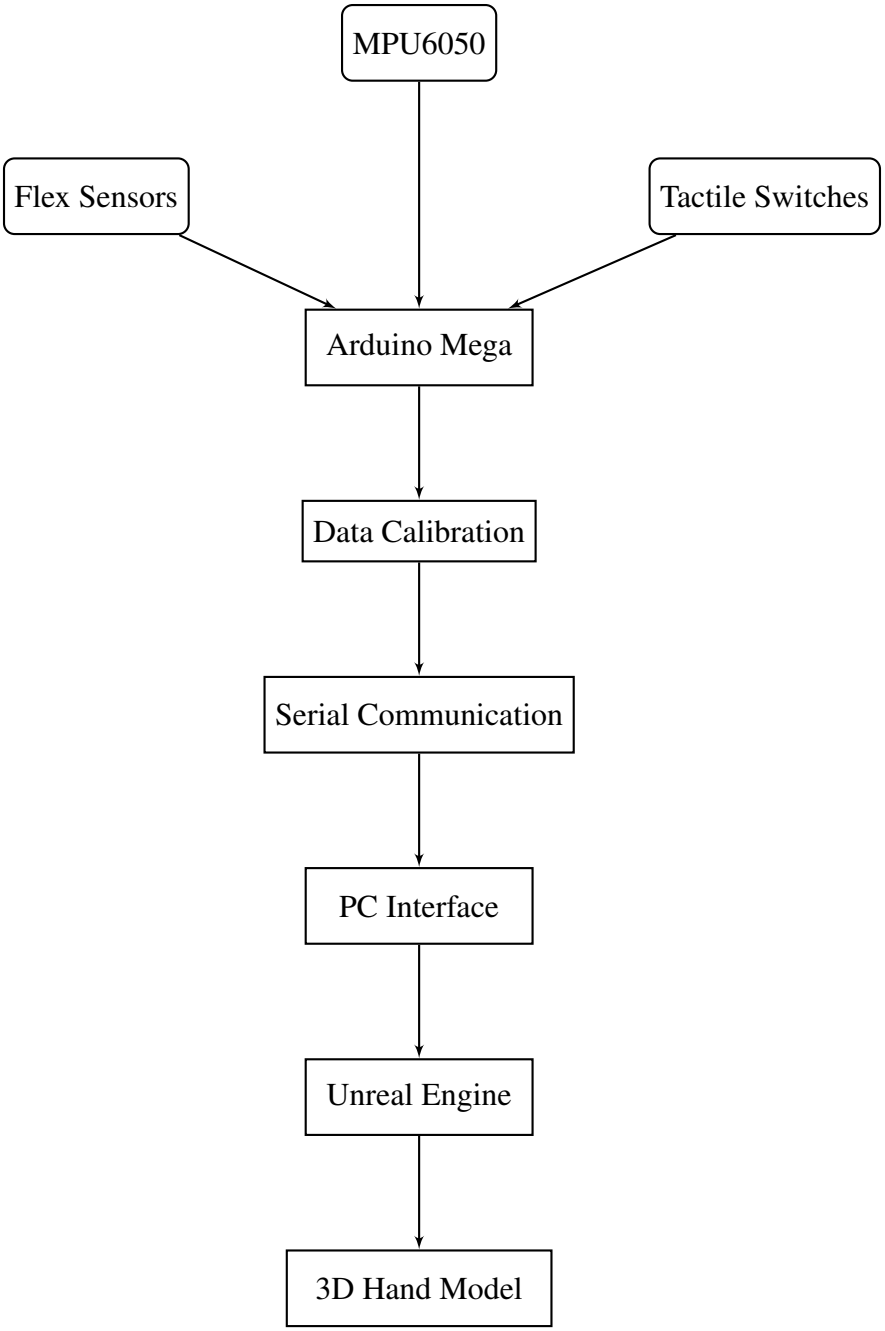


Figure 5.1: Detailed Block Diagram of H.A.N.D

5.2 Algorithm

- Configure MPU6050 for motion tracking
- Set up analog inputs for flex sensors
- Configure digital pins for tactile switches
- Initialize serial communication port
- Read acceleration and rotation data from MPU6050
- Sample voltage values from flex sensors
- Check state of tactile switches
- Apply noise filtering to sensor readings
- Implement calibration offsets to raw data
- Calculate hand orientation from accelerometer/gyroscope data
- Convert flex sensor readings to finger joint angles
- Prepare data packet with structured format
- Calculate and append checksum for error detection
- Transmit formatted data through serial interface
- Receive data packets on PC side
- Validate incoming data integrity
- Update virtual hand skeleton parameters
- Apply calculated movements to 3D hand model

5.3 Flowchart

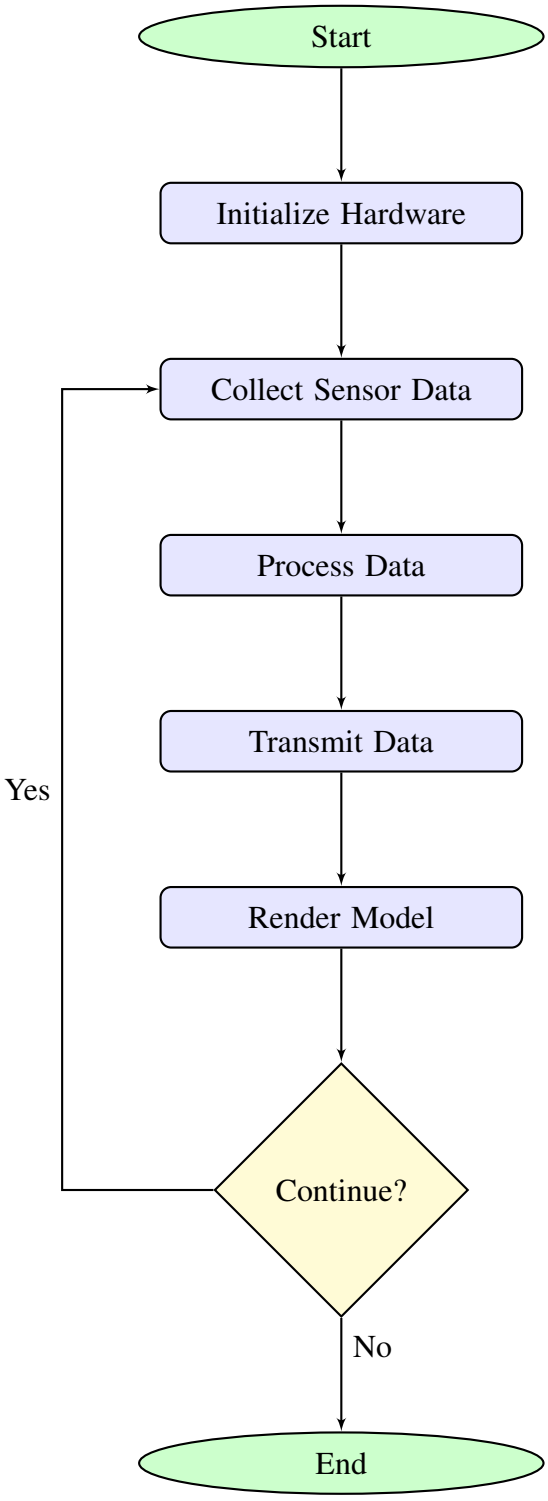


Figure 5.2: Algorithm Flowchart for H.A.N.D

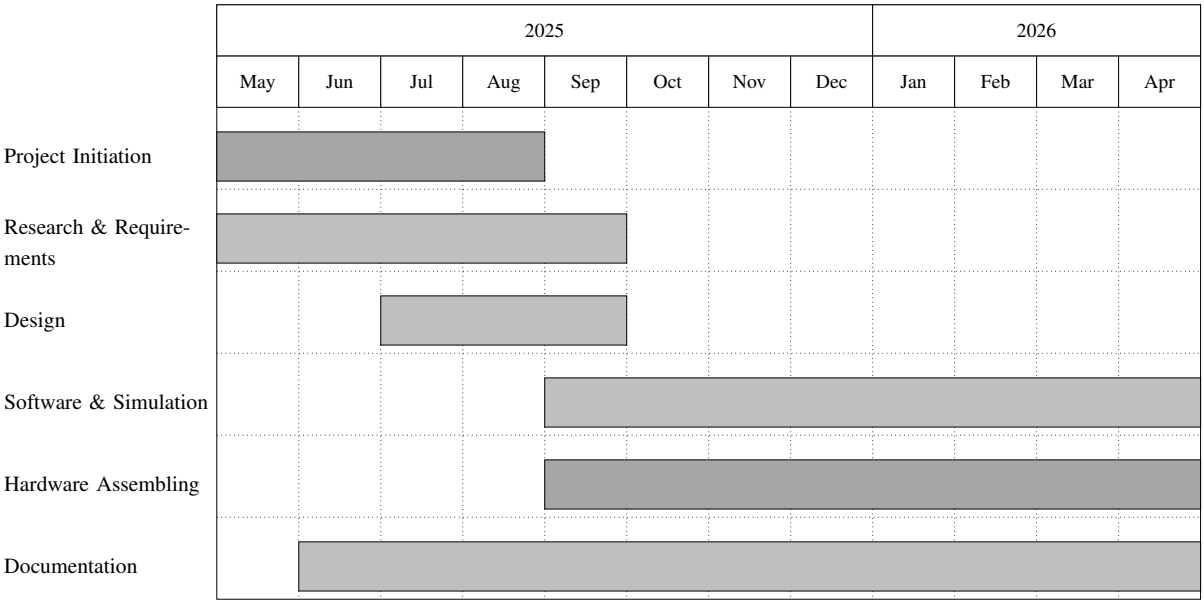
Chapter 6: Expected Output

- **Real-time Hand Motion Tracking:**
  The system will accurately track and measure hand movements using the MPU6050 sensor for orientation (±2° accuracy), flex sensors for finger bending (0° to 90° range), and tactile switches for touch detection. The combined sensor data will provide comprehensive hand position and gesture information with minimal latency (100ms).

- **Virtual Hand Replication:**
  The tracked hand movements will be replicated in real-time on a 3D hand model in Unreal Engine. The virtual hand will accurately mirror all finger movements, hand rotations, and touch interactions with smooth articulation and natural movement visualization. The system will maintain a consistent frame rate above 30 FPS to ensure fluid motion reproduction.

- **Interactive Response System:**
  The system will provide immediate feedback through the virtual hand model, responding to user inputs with less than 50ms latency. This includes accurate finger bend representation (±5° accuracy), precise hand orientation tracking, and immediate response to touch inputs with over 95% reliability. The communication system will maintain a stable 60 Hz update rate to ensure seamless interaction between the physical and virtual hands.

Gantt Chart


2025
2026
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Mar
Apr
Project Initiation
Research & Require-ments
Design
Software & Simulation
Hardware Assembling
Documentation

Cost Estimation

Components and Cost Distribution

S.N	Components	Quantity	Price	Availability
1	Momentary tactile push button	25	1000	Daraz Nepal
2	MPU 6050	5	2500	Daraz Nepal
3	ESP 8266	1	900	Daraz Nepal
4	Arduino with usb cable	1	1700	Daraz Nepal
5	Wire diameter 0.4mm ballpoint spring	10	700	Daraz Nepal
6	Flex sensor	10	7000	Daraz Nepal
7	PLA(1kg)	1	3000	Daraz Nepal
8	Gloves	2	800	Daraz Nepal
9	Superglue(vega)	3	330	Daraz Nepal
10	Enamel wire(50m)	1	1300	Daraz Nepal
11	Connecting wire	40	900	Daraz Nepal
12	Joystick controller with cable	1	1700	Daraz Nepal
13	Li-ion battery(3.7v) with charger	4	4000	Daraz Nepal
		Total Cost	**NRs. 25,830**	

Table 6.1: Components and Cost Distribution

References

1. X. Zhang, M. Liu, and Y. Zhao, “Design of angle detection system based on MPU6050,” in *Proc. Int. Conf. Intelligent Transportation, Big Data & Smart City*, Changsha, China, 2017, pp. 159–162.

2. S. J. Park and S. Y. Lee, “Real-Time Motion Recognition Using an Accelerometer,” *Sensors*, vol. 15, no. 1, pp. 135–148, 2015.

3. P. Kumar, J. Verma, and S. Prasad, ”Hand data glove: A wearable real-time device for human-computer interaction,” International Journal of Advanced Science and Technology, vol. 43, pp. 15–26, June 2012.

4. A. Khandual and A. Biswal, “IoT Based Smart Glove for Hand Gesture Recognition,” *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 5, no. 2, pp. 384–389, Mar. 2019.

5. M. Nasri et al., “Design and Implementation of a 3D Printed Hand Gesture Recognition System Using Arduino and Flex Sensors,” *IEEE Access*, vol. 7, pp. 55371–55378, 2019.

6. T. Baudel and M. Beaudouin-Lafon, “Charade: Remote Control of Objects Using Free-Hand Gestures,” *Commun. ACM*, vol. 36, no. 7, pp. 28–35, 1993.

7. S. S. Rautaray and A. Agrawal, ”Real time hand gesture recognition system for dynamic applications,” International Journal of UbiComp (IJU), vol. 3, no. 1, pp. 21–35, Jan. 2012.

8. S. Chowdhury and A. Haque, ”Animatronic hand controller,” Technical Report, Dept. of Electrical and Electronic Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh, Dec. 2013.

9. D. J. Sturman and D. Zeltzer, ”A survey of glove-based input,” IEEE Computer Graphics and Applications, vol. 14, no. 1, pp. 30–39, Jan. 1994.

10. J. Huang, ”Design of angle detection system based on MPU6050,” in Proc. 7th Int. Conf. Education, Management, Information and Computer Science (ICEMC 2017), Advances in Computer Science Research, vol. 73, pp. 1–3, 2017.