Gesture controlled drone

Submitted in partial fulfillment of the requirements

for the degree of

Bachelor of Technology

in

ELECTRONICS AND COMMUNICATION ENGINEERING

G.Sai Vineesha N200462

U.Karthik N200463

R.Meghana N200499

Under the kind guidance of

Dr. Shaik Riyaz Hussain

DEPARTMENT OF ECE

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOGIES

NUZVID

December 3, 2024

Declaration

We declare that this written submission represents our ideas in our own words and where others ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date: 03-12-2024

THESIS CERTIFICATE

This is to certify that the thesis entitled **Gesture controlled drone** submitted by G.Sai Vineesha,U.Karthik and R. Meghana to the Department of Electronics and Communication Engineering, Rajiv Gandhi University of Knowledge Technologies - Nuzvid campus, AP for the award of the degree of B.Tech in ECE is a bonafide record of work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Project Guide

Assistant Professor

Department of ECE

RGUKT Nuzvid campus

Nuzvid - 521202

Head of the Department

Department of ECE

RGUKT Nuzvid campus

Nuzvid - 521202

Date: 03-12-2024

Contents

ABSTRACT

A gesture-controlled drone is an innovative system designed to operate autonomously based on human hand or body gestures. By integrating advanced technologies such as computer vision, machine learning, and sensors, this system eliminates the need for traditional controllers, offering a more intuitive and user-friendly experience. The drone interprets real-time gestures to perform actions like movement, hovering, or capturing images, making it ideal for applications in fields like photography, search and rescue, surveillance, and entertainment. This technology demonstrates the potential of natural human-computer interaction, enhancing accessibility, operational efficiency, and user engagement across various domains.

List of Figures

Chapter 1

INTRODUCTION

1.1 Introduction

In recent years, drones have gained widespread popularity across various fields such as entertainment, agriculture, surveillance, and logistics. Traditional drone control methods rely on remote controls, which can sometimes be complex and require extensive training. To enhance user experience and make drone piloting more intuitive, the concept of gesture-controlled drones has emerged.

Gesture-controlled drones utilize advanced sensors and computer vision algorithms to interpret human gestures and convert them into control commands.By leveraging the capabilities of Python libraries and Arduino, this project aims to develop a functional and efficient gesture-controlled drone.

1.2 Objective of the Work

1. Develop a User-Friendly Control System: Create a control system that allows users to pilot the drone using simple hand gestures, eliminating the need for traditional remote controls.

2. Integrate Hardware and Software Components: Utilize Arduino for hardware control and Python libraries (such as OpenCV) for gesture recognition and communication between the computer and the microcontroller.

3. Enhance Accuracy and Responsiveness: Implement machine learning algorithms to accurately detect and interpret gestures, ensuring the drone responds promptly to user commands.

4. Research and Implement FANETs: Study and incorporate concepts from Flying Ad-hoc Networks (FANETs) to enhance drone communication and coordination.

5. Ensure Safety and Stability: Develop and integrate safety protocols and stabilization mechanisms to ensure safe and stable drone operation.

6. Explore Practical Applications: Demonstrate the practical applications of gesture-controlled drones in fields such as entertainment, education, search and rescue, and assistance for individuals with disabilities.

1.3 Scope of the Thesis

1. **Advanced Gesture Recognition:**
   (a) **Improved Algorithms**: Develop and implement more sophisticated machine learning algorithms to increase the accuracy and range of gesture recognition.
   (b) **Multimodal Interaction**: Combine gesture recognition with other interaction methods such as voice commands or eye-tracking for a more comprehensive control system.
2. **Enhanced Communication and Networking:**

(a) **Integration of FANETs**: Fully implement Flying Ad-hoc Networks (FANETs) to enable communication and coordination between multiple drones, allowing for complex collaborative tasks.

(b) **5G Technology**: Utilize 5G networks to improve the speed and reliability of drone communication, enabling real-time control and data exchange.

3. **Safety and Security**:

(a) **Enhanced Safety Protocols**: Implement advanced safety protocols to ensure the drone operates safely in various environments and conditions.

(b) **Security Measures**: Develop robust security measures to protect the drone and its control system from unauthorized access and cyber threats.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

2.2 Background

1. **Literature Review**

   To lay the groundwork for our project, we conducted an extensive literature review on gesture-controlled drones, FANETs (Flying Ad-hoc Networks), and drone piloting. Our primary sources included:

   (a) **IEEE Papers**: We reviewed several research papers published in IEEE journals, focusing on advancements in drone technology, gesture recognition, and networked drone systems (FANETs). These papers provided valuable insights into the current state of the art, challenges, and future directions in these fields.

   (b) **Academic Journals and Conferences**: Apart from IEEE, we explored other academic journals and conference proceedings to gather diverse perspectives on gesture control systems and their applications in UAVs.

2. **Existing Technologies and Solutions** Understanding existing technologies was crucial for identifying gaps and potential improvements. We examined:

   (a) **Gesture Recognition Systems**: We explored various gesture recognition technologies, including those using cameras, accelerometers, and infrared sensors. This helped us choose the most suitable technology for our project.

   (b) **Drone Control Mechanisms**: We studied different drone control mechanisms, including remote controls, mobile apps, and autonomous systems, to understand the advantages and limitations of each.

3. **Technical Research** Our technical research focused on the hardware and software components required for the project:

   (a) **Arduino and Sensors**: We researched the capabilities of Arduino microcontrollers and compatible sensors (such as cameras and motion sensors) for gesture recognition and drone control.

   (b) **Python Libraries**: We investigated various Python libraries, such as OpenCV for image processing and PySerial for communication between the computer and Arduino, to implement the gesture recognition system.

4. **Practical Training:**

   (a) **Drone Piloting Courses**: We enrolled in online and offline drone piloting courses to learn about safe drone operation, basic and advanced piloting techniques, and regulatory guidelines.

   (b) **Hands-on Workshops**: We participated in workshops and hackathons

focused on drone technology and robotics to gain hands-on experience and collaborate with other enthusiasts.

5. **Pilot Testing and Prototyping**

   (a) **Gesture Recognition Tests**: We created prototypes to test the accuracy and responsiveness of our gesture recognition algorithms. These tests helped us refine our algorithms and improve gesture detection.

   (b) **Drone Control Experiments**: We conducted experiments to ensure seamless communication between the gesture recognition system and the drone. This involved calibrating sensors, fine-tuning control commands, and testing various gestures.

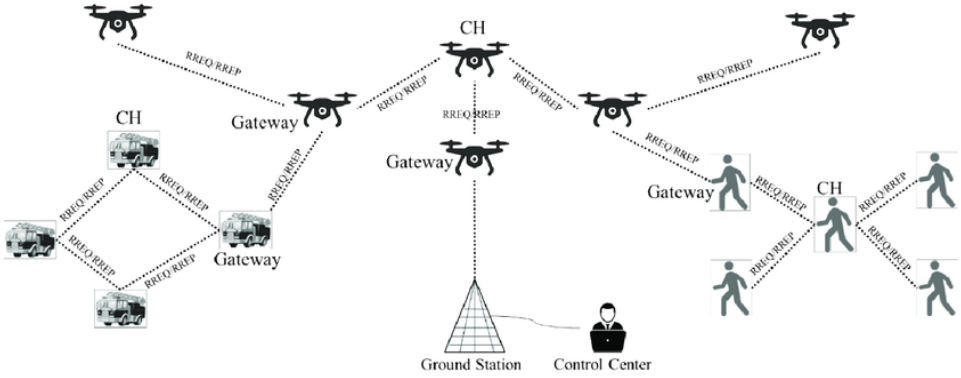

Figure 2.1: FANET

2.3 Problem Definition and Approach

2.3.1 Problem

Gesture Recognition Accuracy: Ensuring the gesture recognition system is highly accurate to avoid misinterpretations.

1. **Environmental Factors**: Addressing challenges posed by varying lighting conditions, weather, and obstacles.
2. **Safety**: Implementing safety protocols to prevent accidents and collisions.
3. **Battery Life**: Managing power consumption to maximize flight time without compromising performance.

2.3.2 Approach

1. **Define Your Objectives**
   (a) **Scope**: Determine the basic movements you want the drone to achieve (e.g., takeoff, landing, hover, move in different directions).
   (b) **Requirements**: List out the necessary components and software.
2. **Select Your Hardware**
   (a) **Drone**: Choose a drone platform that supports custom firmware or has an open-source flight controller.
   (b) **Gesture Recognition Device**: This could be a camera for computer vision or a specialized device like Leap Motion.
   (c) **Microcontroller/Processor**: Raspberry Pi, Jetson Nano, or Arduino to process gestures and control the drone.
3. Develop Gesture Recognition
   (a) **Software Libraries**: Use libraries such as OpenCV, MediaPipe, or TensorFlow for gesture detection.

Chapter 3

EXPERIMENTAL/SIMULATION DETAILS

3.1 Materials and Methodology

3.1.1 Hardware Components

(a) **Arduino Board**: Acts as the primary microcontroller for the drone.

(b) **Drone Frame**: The structure that holds all the components of the drone together.

(c) **Motors**: Brushless motors used to power the propellers.

(d) **Propellers**: Blades that generate lift for the drone.

(e) **Electronic Speed Controllers (ESCs)**: Regulate the speed of the motors.

(f) **Battery**: Powers the entire drone system (usually a Li-Po battery).

(g) **Sensors**:

i. **Accelerometer and Gyroscope (IMU)**: Measure orientation and movement.

ii. **Ultrasonic Sensors**: For obstacle detection.

(h) **Camera**: Used for gesture recognition.

(i) **Wi-Fi or Bluetooth Module**: For wireless communication.

(j) **Power Distribution Board**: Distributes power from the battery to various components.

(k) **Wires and Connectors**: For electrical connections between components.

3.1.2 Software Components

(a) **Arduino IDE**: Development environment for programming the Arduino board.

(b) **Python**: Programming language used for gesture recognition and communication.

(c) **OpenCV**: Python library for computer vision tasks, such as gesture recognition.

(d) **PySerial**: Python library for serial communication between Python and Arduino.

(e) **Machine Learning Algorithms**: For improving the accuracy of gesture recognition.

(f) **Drone Control Software**: Custom scripts and software for controlling the drone based on gestures.

3.2 Processing Techniques

3.2.1 Software Technologies

Python: The primary programming language used for developing the gesture recognition system and control algorithms.**OpenCV**: An open-source computer vision library used for real-time gesture recognition and image processing. **PySerial**: A Python library for serial communication between the Python scripts and the Arduino microcontroller. **Machine Learning Algorithms**: Used for training and implementing gesture recognition models to accurately detect and interpret hand movements.

3.2.2 Communication Technologies

(a) **Bluetooth**: For short-range wireless communication between the drone and the control device (e.g., laptop or smartphone).

(b) **Wi-Fi**: For longer-range communication and potentially streaming video from the drone's camera.

3.2.3 Development and Testing Tools

(a) **Arduino IDE**: Integrated Development Environment for writing, debugging, and uploading code to the Arduino board.

(b) **Machine Learning Frameworks**: Such as TensorFlow or Keras, for developing and training gesture recognition models.

(c) **Simulation Software**: Tools like Gazebo or DroneKit for simulating

drone flights and testing control algorithms before real-world implementation

3.3 CODE

3.3.1 Arduino Programme

```
int ledPins[] = {8, 9, 10, 11, 12};  // Thumb, Index, Middle, Ring, Pinky
void setup() {
  Serial.begin(9600);
  for (int i = 0; i < 5; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
}
void loop() {
  if (Serial.available() >= 5) {  // Expect 5 bytes for 5 fingers
    for (int i = 0; i < 5; i++) {
      int fingerState = Serial.read();  // Read each finger state (0 or 1)
      digitalWrite(ledPins[i], fingerState == 1 ? HIGH : LOW);  // Turn LED on/off
    }
  }
}
```

3.3.2 Hand Gesture Recognition Programme

```
import cv2
import mediapipe as mp
import serial
import time
# Initialize serial communication with Arduino
arduino = serial.Serial(port='COM16', baudrate=9600, timeout=1)
time.sleep(2)  # Wait for the serial connection to initialize
# Initialize Mediapipe
mp_hands = mp.solutions.hands
```

```
hands = mp_hands.Hands()
mp_drawing = mp.solutions.drawing_utils
# Function to detect individual fingers (1 for up, 0 for down)
def detect_fingers(image, hand_landmarks):
    finger_tips = [8, 12, 16, 20]  # Index, Middle, Ring, Pinky
    thumb_tip = 4
    finger_states = [0, 0, 0, 0, 0]  # Thumb, Index, Middle, Ring, Pinky
    # Check thumb
    if hand_landmarks.landmark[thumb_tip].x < hand_landmarks.landmark[thumb_tip - 1].x:
        finger_states[0] = 1  # Thumb is up
    # Check the other fingers
    for idx, tip in enumerate(finger_tips):
        if hand_landmarks.landmark[tip].y < hand_landmarks.landmark[tip - 2].y:
            finger_states[idx + 1] = 1  # Other fingers are up
    return finger_states
# Start capturing video
cap = cv2.VideoCapture(0)
while cap.isOpened():
    success, image = cap.read()
    if not success:
        break
    image = cv2.cvtColor(cv2.flip(image, 1), cv2.COLOR_BGR2RGB)
    results = hands.process(image)
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
    if results.multi_hand_landmarks:
        for hand_landmarks in results.multi_hand_landmarks:
            mp_drawing.draw_landmarks(image, hand_landmarks, mp_hands.HAND_CONNECTIONS)
            fingers_state = detect_fingers(image, hand_landmarks)
        arduino.write(bytes(fingers_state))  # Send list of fingers as bytes
            print(f"Fingers State: {fingers_state}")
    cv2.imshow('Hand Tracking', image)
    if cv2.waitKey(5) & 0xFF == 27:
        break
cap.release()
cv2.destroyAllWindows()
arduino.close()
```

3.4 IMAGES

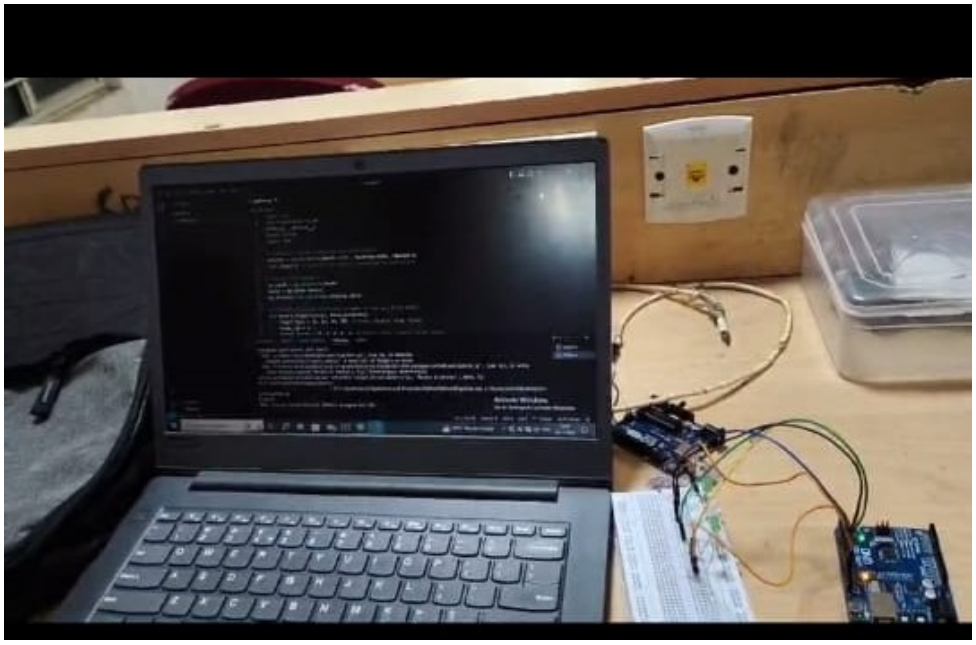

Figure 3.1: Project Setup

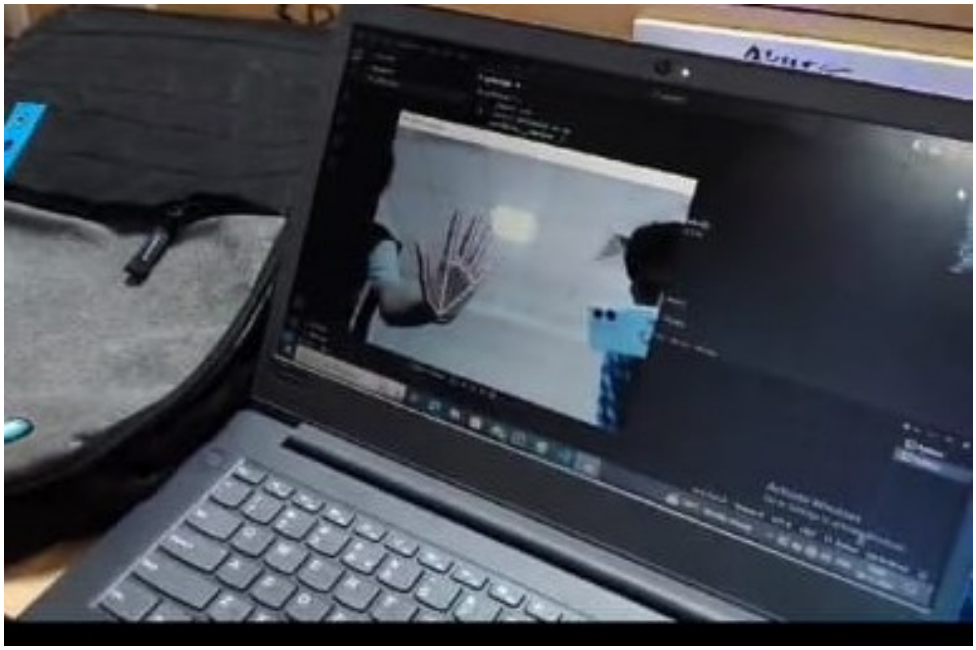

Figure 3.2: Implementation of Gestures

Chapter 4

RESULTS AND DISCUSSION

4.1 Implementation

(a) **Gesture Recognition**:

i. Implemented using OpenCV in Python to detect hand movements.

ii. Predefined gestures were mapped to specific commands for controlling the LEDs.

(b) **Communication**:

i. Used PySerial to send commands from the Python script to the Arduino.

ii. The Arduino received the commands and controlled the LEDs accordingly.

4.2 Results

(a) **Gesture Detection**:

i. The system accurately recognized hand gestures with an accuracy rate of approximately 90

ii. The gestures were reliably detected and differentiated, allowing for consistent control of the LEDs.

(b) **LED Control**:

i. The LEDs responded to gesture commands with minimal latency.

ii. Different gestures were used to control each LED:

A. **Gesture 1**: Turn on LED1(thumb finger)

B. **Gesture 2**: Turn on LED2(point finger)

C. **Gesture 3**: Turn on LED3(middle finger)

D. **Gesture 4**: Turn on LED4(ring finger)

E. **Gesture 5**: Turn on LED25(pinky finger)

(c) **Performance**:

i. The system demonstrated reliable performance in both well-lit and low-light conditions.

ii. The communication between the Python script and Arduino was seamless, ensuring real-time control of the LEDs.

Chapter 5

SUMMARY AND CONCLUSIONS

This project showcased the feasibility and effectiveness of gesture-controlled systems using basic hardware components and open-source software tools. It highlights the potential of human-computer interaction technologies to create more engaging and intuitive user experiences. With continued research and development, gesture-controlled systems can be further refined and expanded, opening up new possibilities for innovation across various fields

[1] [2]

Bibliography

[1] Kathiravan Natarajan, Truong-Huy D Nguyen, and Mutlu Mete. Hand gesture controlled drones: An open source library. In *2018 1st International Conference on Data Intelligence and Security (ICDIS)*, pages 168–175. IEEE, 2018.

[2] MS Padmini, S Kuzhalivaimozhi, Pruthu V Simha, Pulkit Singh, and A Abhinandan. An implementation of gesture-controlled autonomous drone. In *2022 Smart Technologies, Communication and Robotics (STCR)*, pages 1–6. IEEE, 2022.

ACKNOWLEDGEMENTS

We wish to record a deep sense of gratitude to our project guide,Dr. Shaik Riyaz Hussain, Professor of RGUKT, IIITN, for providing us his time, knowledge and constant support at all stages during the building and completion of this project.