

**A Mini Project Report**  
**On**  
**Design And Analysis of Propulsion System of**  
**Gesture Control UAV Using AIML**

**A dissertation Submitted in partial fulfillment of the requirement**  
**for the**

**Award of the degree of**  
**BACHELOR OF TECHNOLOGY**

**In**  
**AERONAUTICAL ENGINEERING**

**By**

**SHAIK NAWAZ**  
**TIRUMALA SAI NITHIN**  
**V. HIMABINDU**

**20N31A2134**  
**20N31A2135**  
**20N31A2136**

**Under the esteemed guidance of**

**Mr. M. YUGENDER**

**Associate Professor**



**DEPARTMENT OF AERONAUTICAL ENGINEERING**  
**MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY**

**Autonomous Institution – UGC, Govt. of India**

(Affiliated to JNTU, Hyderabad, Approved by AICTE - Accredited by NBA & NAAC – 'A' Grade)  
Maisammaguda, Dhulapally (Post Via Hakimpet), Secunderabad – 500100

**2020-2024**



## **Malla Reddy College of Engineering & Technology**

**(Autonomous Institution- UGC, Govt. of India)**

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA & NAAC with 'A' Grade)

Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100

website: [www.mrcet.ac.in](http://www.mrcet.ac.in)

### **CERTIFICATE**

This is to certify that this is the bonafide record of the mini project entitled **“Design And Analysis of Propulsion System of Gesture Control UAV Using AIML”**, submitted by **SHAIK NAWAZ (20N31A2134)**, **TIRUMALA SAI NITHIN (20N31A2135)**, and **V. HIMABINDU (20N31A2136)** of B.Tech in the partial fulfillment of the requirements for the degree of Bachelor of Technology in Aeronautical Engineering, Department of Aeronautical Engineering during the year 2023-2024. The results embodied in this minor project report have not been submitted to any other university or institute for the award of any degree or diploma.

**Internal Guide**  
**Mr. M. YUGENDER**  
**Associate Professor**

**Head of the Department**  
**Dr. Misba Mehdi**

**External Examiner**

## ACKNOWLEDGEMENTS

We express our heartiest thanks to our Director **Dr. V S K Reddy**, Professor for encouraging us in every aspect of our project and helping us realize our full potential.

We feel ourselves honored and privileged to place our warm salutation to our college “Malla Reddy College of Engineering and Technology (Autonomous Institution – UGC Govt. of India) and our Principal **Dr. S Srinivasa Rao**, Professor who gave us the opportunity to do the Minor Project during our IV Year B.Tech and profound the technical skills.

We would like to express our heartfelt gratitude to the College Dean-Student Welfare **Dr. T Venugopal** for his unwavering support and encouragement throughout our minor project. Your guidance has been invaluable, and your belief in our work has inspired us to reach new heights.

We extend our sincere gratitude to the College Dean-International Studies **Prof. P Sanjeeva Reddy** for his unwavering support and encouragement during the execution of our minor project. Your belief in our capabilities and your guidance have been instrumental in our success.

We are also thankful to our Head of the Department **Dr. Misba Mehdi**, Professor for providing training and guidance, excellent infrastructure and a nice atmosphere for completing this project successfully.

We would like to express our sincere gratitude and indebtedness to our project supervisor **Mr. M. Yugender**, Professor for his valuable suggestions and interest throughout the course of this project.

We would like to express our heartfelt gratitude to our Department Project Coordinator **Dr. Krishnanand** for his continuous support and encouragement throughout our minor project. Your guidance has been invaluable, and your belief in our abilities has motivated us to excel.

We would like to thank all our supporting **staff** of the Department of Aeronautical Engineering and even all other departments who have been helpful directly and in-directly in making our project a success.

Finally, we would like to take this opportunity to thank our **families** for their support and blessings for completion of our project that gave us the strength to do our project.

By

**SHAIK NAWAZ (20N31A2134)**

**TIRUMALA SAI NITHIN (20N31A2135)**

**V. HIMABINDU (20N31A2136)**

# CONTENTS

Title of the Chapter	Page No
<b>Cover Page</b>	<b>i</b>
<b>Certificate</b>	<b>ii</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>Abstract</b>	<b>1</b>
<b>Chapter 1 – Introduction</b>	<b>1-15</b>
1.1 UAV – Unmanned Aerial Vehicle	1-2
1.2 TYPES OF UAV AND THEIR APPLICATIONS	2-4
1.3 PROPULSION IN A DRONE	5-6
1.4 PROPULSION SYSTEM OF A DRONE	6-7
1.5 TYPES OF PROPULSION SYSTEMS INVOLVED WITH DRONES	7-9
1.6 FACTORS FOR PROPULSION SYSTEM SELECTION	10-12
1.7 COMPONENTS REQUIRED TO BUILD THE GESTURE CONTROLLED DRONE	12-15
<b>Chapter 2 – LITERATURE REVIEW</b>	<b>16-26</b>
2.1 RESEARCH PAPER ANALYSIS	16-22
2.1.1 HAND GESTURES FOR DRONE CONTROL USING DEEP LEARNING	16
2.1.2 G.O.D. (Gesture Operated Drone)	17
2.1.3 Flying Drone Controller by Hand Gesture Using Leap Motion	17
2.1.4 Gesture Controlled Drone	17-18
2.1.5 Wearable Drone Controller: Machine Learning-Based Hand Gesture Recognition and Vibrotactile Feedback	18
2.1.6 UAV manipulation by hand gesture recognition	19
2.1.7 Development, Control Adjustment, and Gesture Recognition of a Quadrotor Helicopter	19-20
2.1.8 Review of Propulsion System Design Strategies for Unmanned Aerial Vehicles	20
2.1.9 Propulsion optimization of a quadcopter in forward state	20-21
2.1.10 Electric propulsion system sizing methodology for an agriculture multi-copter	21-22
2.2 TABLES; GRAPHS & SOFTWARE OUTPUTS	22-26
<b>Chapter 3 – METHODOLOGY</b>	<b>27-40</b>
3.1 ANALYTICAL	27-36
3.1.1 AIML MODEL FOR PROPULSION SYSTEM ANALYSIS	27-29
3.1.2 GESTURE RECOGNITION USING ARDUINO – UNO	30-33
3.1.3 GENERAL FLIGHT CONTROLLER CODE FOR A DRONE	33-35
3.1.4 Manual Calculations – Formulae	35-36
3.2 COMPUTATIONAL	36-40

<b>3.2.1 DRONE DESIGN – Using SolidWorks</b>	<b>36-38</b>
<b>3.2.2 DRONE ANALYSIS – SolidWorks</b>	<b>38-40</b>
<b>Chapter 4 – RESULTS &amp; DISCUSSIONS</b>	<b>41-53</b>
<b>4.1 AIML MODEL OUTPUTS</b>	<b>41-43</b>
<b>4.2 GESTURE RECOGNITION CODE OUTPUT – ARDUINO UNO</b>	<b>44-46</b>
<b>4.3 DRONE DESIGN OUTCOMES – SOLIDWORKS</b>	<b>46-49</b>
<b>4.4: MANUAL CALCULATIONS – RESULTS</b>	<b>49-53</b>
<b>References</b>	<b>54</b>

## LIST OF FIGURES

FIGURE NUMBER	TITLE OF THE FIGURE	PAGE NO.
Figure 1.2.1	Fixed Wing UAV	4
Figure 1.2.2	Multi-Rotor UAV	4
Figure 1.2.3	VTOL UAV	4
Figure 1.2.4	Hybrid VTOL UAV	4
Figure 1.2.5	Micro UAV	4
Figure 1.2.6	Solar-Powered UAV	4
Figure 1.2.7	Fixed-Wing Hybrid UAV	4
Figure 1.2.8	Nano UAV	4
Figure 1.2.9	Heavy-Lift UAV	4
Figure 1.2.10	Swarm UAV	4
Figure 1.5.1	Electric Motors & Propellers	8
Figure 1.5.2	Fixed-Wing Propulsion	8
Figure 1.5.3	Jet Engines	8
Figure 1.5.4	Gasoline-Powered Drones	9
Figure 1.5.5	Hybrid Drones	9
Figure 1.5.6	Ducted Fan Drones	9
Figure 1.5.7	Solar-Powered Drones	9
Figure 1.5.8	Hydrogen-Powered Drones	9
Figure 1.6.1	Factors for propulsion system selection	10
Figure 1.7.1	Drone Frame	15
Figure 1.7.2	Motor and Propeller	15
Figure 1.7.3	ESC	15
Figure 1.7.4	Flight Controller [Arduino Based]	15
Figure 1.7.5	Power Distribution Board	15
Figure 1.7.6	Battery	15
Figure 4.1.1	T/W Output	42
Figure 4.1.2	Flight Time Output	42
Figure 4.1.3	Power Consumption	42
Figure 4.1.4	GSPC Output	42
Figure 4.1.5	Total Power Consumption Output	42
Figure 4.1.6	Battery Capacity Output	42
Figure 4.1.7	Battery Voltage Selection Output	43
Figure 4.1.8	Payload Consideration Output	43
Figure 4.1.9	Graphs For the analysis values	43
Figure 4.2.1	Gesture Recognition Code Output – Arduino IDE	45
Figure 4.3.1	Top Plane	46
Figure 4.3.2	Front Plane	47
Figure 4.3.3	Side Plane	48

## LIST OF TABLES

TABLE NUMBER	TITLE OF THE TABLE	PAGE NO.
Table 2.1.1	Samples of the input rows for the training dataset	22
Table 2.2.1	Comparing ESCs	22
Table 2.2.2	Motor Comparison	22
Table 2.2.3	MCU Comparison	23
Table 2.5.1	Results of obstacle avoidance using vibrotactile feedback. Without vibrotactile feedback, the drone failed to avoid obstacles; meanwhile, collision avoidance was completed successfully with vibrotactile feedback	25
Table 2.5.2	Results of subjective evaluation by participants in gesture-based drone control	25
Table 2.5.3	Results of subjective evaluation by participants in vibrotactile feedback	25
Table 2.6.1	Gesture recognition rate (FB method)	26
Table 2.6.2	Gesture recognition rate (ML method)	26
Table 2.6.3	Gesture recognition rate under flying UAV	26
Table 2.7.1	Results of the gesture recognition control experiments	26

## ABSTRACT

This project "Design and analysis of propulsion system of gesture control UAV using AIML" aims to design and analyse a propulsion system for a Gesture Control Unmanned Aerial Vehicle (UAV) using Artificial Intelligence and Machine Learning (AIML) algorithms, with the integration of Internet of Things (IoT) components. The objective is to develop a UAV that can be controlled through gesture commands, enhancing the user experience, and enabling intuitive interactions.

The project begins with the design and development of the UAV's propulsion system, considering factors such as weight, power requirements, and efficiency. Various propulsion technologies, including electric motors and propellers, are evaluated, and optimized to achieve the desired performance characteristics for the UAV.

Next, AIML algorithms are implemented to enable gesture control functionality. These algorithms utilize machine learning techniques to recognize and interpret hand gestures, allowing users to intuitively command the UAV's flight path, altitude, and other parameters. Training datasets are generated and utilized to train the AI models, improving the accuracy and reliability of gesture recognition.

Furthermore, IoT components are integrated into the UAV system, enabling real-time data transmission and remote-control capabilities. Sensor modules, communication devices, and data processing units are incorporated to establish a seamless connection between the UAV and the ground station. This integration facilitates the monitoring of vital flight parameters and enables remote control and telemetry for enhanced operational efficiency and safety. The outcome of this project is an innovative Gesture Control UAV with an integrated IoT system, capable of intuitive and responsive flight control. The proposed system demonstrates the potential for enhanced user experience and expanded applications in fields such as aerial photography, surveillance, and environmental monitoring. The integration of AIML algorithms and IoT components provides a solid foundation for further advancements in UAV technology and human-machine interaction.



# CHAPTER 1

## INTRODUCTION

### 1.1: UAV – Unmanned Aerial Vehicle:

UAV stands for Unmanned Aerial Vehicle, commonly known as a drone. It is an aircraft that operates without a human pilot onboard. UAVs can be remotely controlled by a human operator or fly autonomously based on pre-programmed instructions or artificial intelligence algorithms.

UAVs come in various sizes, ranging from small handheld drones to large-scale unmanned aircraft used for military or commercial purposes. They are equipped with different sensors, cameras, and other payloads depending on their intended application.



UAVs offer numerous advantages over manned aircraft, such as:

- 1. Remote Operations:** UAVs can be controlled from a ground station, eliminating the need for a pilot to be physically onboard the aircraft. This allows for safer and more cost-effective operations, especially in dangerous or hard-to-reach locations.
- 2. Versatility:** UAVs can access areas that are difficult or dangerous for humans to reach. They can fly at low altitudes, navigate through tight spaces, and gather data from challenging environments, making them useful in various fields like aerial photography, videography, surveying, agriculture, and search and rescue operations.
- 3. Efficiency and Cost-Effectiveness:** UAVs generally consume less fuel and have lower operational costs compared to manned aircraft. They can be rapidly deployed and cover large areas in a shorter time, resulting in increased efficiency and reduced operational expenses.
- 4. Data Collection:** UAVs are equipped with sensors and cameras that allow them to capture high-resolution images, videos, and other data. This data can be used for mapping, monitoring, inspection, and research purposes in fields like agriculture, environmental conservation, infrastructure inspection, and disaster response.

**5. Safety and Risk Mitigation:** By removing human pilots from the aircraft, UAVs can reduce risks associated with manned flights. They can operate in hazardous conditions, collect data from dangerous environments, and perform tasks without risking human lives.

However, there are also considerations and challenges associated with UAVs, including airspace regulations, privacy concerns, limited flight endurance and range, and potential safety risks if not operated responsibly.

Overall, UAVs have revolutionized various industries and opened up new possibilities for aerial operations. Their versatility, cost-effectiveness, and ability to access remote and challenging environments make them valuable tools in a wide range of applications.

## **1.2: TYPES OF UAV AND THEIR APPLICATIONS:**

There are various types of UAVs, each designed with specific capabilities to cater to different applications. Here are some common types of UAVs along with their application-based examples:

### **1. Fixed-Wing UAVs:**

Applications: Aerial Mapping and Surveying

Examples: SenseFly eBee X - Used for high-resolution aerial mapping, photogrammetry, and surveying applications. It can cover large areas efficiently and generate accurate 3D maps or models.

### **2. Multirotor UAVs:**

Applications: Aerial Photography and Videography

Examples: DJI Phantom 4 Pro - Popular for aerial photography and videography, it offers high-resolution cameras, stability, and intelligent flight features to capture stunning aerial shots.

### **3. Vertical Take-off and Landing (VTOL) UAVs:**

Applications: Military and Surveillance Operations

Examples: AeroVironment RQ-20 Puma - A small hand-launched VTOL UAV used for military reconnaissance and surveillance missions. It can operate in confined areas and provide real-time video feeds.

### **4. Hybrid VTOL UAVs:**

Applications: Cargo Delivery and Logistics

Examples: Wingcopter 178 - Designed for long-range and beyond-visual-line-of-sight operations, it combines vertical takeoff and landing capabilities with fixed-wing efficiency. It is used for delivering medical supplies and commercial logistics.

**5. Micro UAVs:**

Applications: Indoor Surveillance and Inspection

Examples: Parrot Anafi Work - A small, lightweight UAV used for indoor inspections, search and rescue in confined spaces, or monitoring in hard-to-reach areas. It is equipped with a 4K camera and thermal imaging capabilities.

**6. Solar-Powered UAVs:**

Applications: Atmospheric Research and Monitoring

Examples: AeroVironment Pathfinder Plus - A high-altitude, long-endurance UAV powered by solar panels. It is used for scientific research, atmospheric sampling, and monitoring applications.

**7. Fixed-Wing Hybrid UAVs:**

Applications: Precision Agriculture

Examples: PrecisionHawk Lancaster 5 - Designed for agricultural applications, it can carry various sensors to collect data on crop health, moisture levels, and soil conditions. It helps optimize farming practices and improve yield.

**8. Nano UAVs:**

Applications: Urban Surveillance and Reconnaissance

Examples: Black Hornet Personal Reconnaissance System - A small and lightweight UAV used for military and security purposes. It provides soldiers with real-time situational awareness in urban environments.

**9. Heavy-Lift UAVs:**

Applications: Industrial Inspections and Payload Transport

Examples: Freefly Alta X - Capable of carrying heavy payloads, it is used for industrial inspections of infrastructure like power lines, wind turbines, or oil rigs. It can also transport equipment or supplies to remote areas.

**10. Swarm UAVs:**

Applications: Collaborative Search and Rescue

Examples: Intel Shooting Star Drones - Designed for synchronized aerial displays, these UAVs can also be used in search and rescue operations. They can collaborate to cover larger areas and provide enhanced situational awareness.

These examples illustrate how different types of UAVs are tailored to specific applications, highlighting their versatility across various industries and sectors.



Figure.1.2.1: Fixed Wing UAV



Figure.1.2.2: Multi-Rotor UAV



Figure.1.2.3: VTOL UAV

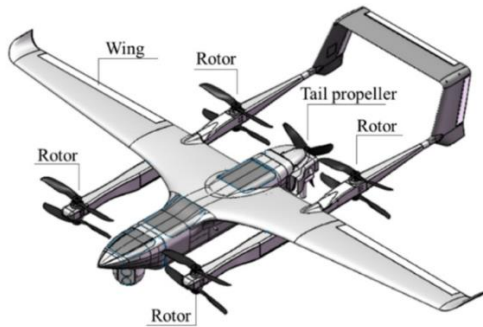


Figure.1.2.4: Hybrid VTOL UAV



Figure.1.2.5: Micro UAV



Figure.1.2.6: Solar-Powered UAV



Figure.1.2.7: Fixed-Wing Hybrid UAV



Figure.1.2.8: Nano UAV



Figure.1.2.9: Heavy-Lift UAV



Figure.1.2.10: Swarm UAV

### 1.3: PROPULSION IN A DRONE:

A quadcopter drone utilizes electric motors and propellers for propulsion. Specifically, it employs four rotors to generate lift and control its movement. Here's how it works:



1. **Electric Motors:** Each of the four arms of a quadcopter is equipped with an electric motor. These motors are typically brushless DC (BLDC) motors, known for their efficiency and reliability.
2. **Propellers:** Each motor is connected to a propeller, which is a rotating blade that generates thrust when it spins. Two of the propellers rotate clockwise (CW), and the other two rotate counterclockwise (CCW) to create a balanced and stable flight.
3. **Thrust Control:** To control the quadcopter's movement, the motors can adjust the speed of their rotation. When all four motors spin at the same speed, the drone hovers in place. By varying the speeds of these motors, the quadcopter can achieve forward, backward, left, right, and vertical movements.
4. **Stability and Balance:** Quadcopters rely on the differential thrust produced by the motors to maintain stability. For example, to tilt forward, the rear motors spin faster, while the front motors slow down. This creates a forward tilt, and the quadcopter moves in that direction. To maintain balance and control, the flight controller continually adjusts the motor speeds.

- 5. Altitude Control:** In addition to movement, quadcopters use changes in motor speed to control their altitude. Increasing the overall motor speed provides more thrust and causes the drone to ascend, while decreasing it leads to descent.

Overall, the precise and rapid adjustments in motor speeds, controlled by an onboard flight controller, enable quadcopter drones to achieve stable and agile flight, making them popular for various applications, including photography, videography, surveillance, and recreational flying.

## **1.4: PROPULSION SYSTEM OF A DRONE:**

The propulsion system of a quadcopter drone consists of electric motors and propellers, which work together to generate the thrust needed for flight and maneuverability. Here's a brief description:

**1. Electric Motors:** A quadcopter drone typically has four electric motors, one on each arm of the frame. These motors are usually brushless DC (BLDC) motors due to their efficiency and reliability. They provide the necessary power for lift and control.

**2. Propellers:** Each motor is connected to a propeller, resulting in four propellers in total. Two of these propellers rotate clockwise (CW), while the other two rotate counterclockwise (CCW). This configuration is essential for maintaining balance and stability during flight.

### **Principles of Operation:**

#### **Thrust Generation:**

When the electric motors spin, they rotate the attached propellers, creating upward thrust. The collective thrust produced by all four motors allows the quadcopter to take off and remain airborne.

#### **Thrust Control:**

To control the quadcopter's movement, the speed of each motor can be adjusted independently. By changing the motor speeds, the quadcopter can move forward, backward, left, right, and ascend or descend vertically.

#### **Stability and Balance:**

Quadcopters rely on varying the speed of these motors to maintain stability and balance during flight. For example, tilting the drone forward involves increasing the speed of the rear motors and decreasing the speed of the front motors to shift the thrust vector.

#### **Altitude Control:**

Adjusting the overall motor speed allows the quadcopter to change its altitude. Increasing the speed increases thrust, causing the drone to ascend, while decreasing the speed results in descent.

**Flight Controller:**

To execute precise and rapid adjustments in motor speeds, quadcopters are equipped with a flight controller. This onboard electronic component utilizes data from sensors like accelerometers and gyroscopes to stabilize the drone and respond to pilot commands.



In summary, the propulsion system of a quadcopter drone relies on the coordinated operation of four electric motors and propellers. By manipulating the speed of these motors, the drone achieves stable flight, controlled movement in multiple directions, and changes in altitude, making it a versatile and widely used platform for various applications.

**1.5: TYPES OF PROPULSION SYSTEMS INVOLVED WITH DRONES:**

Drones can utilize various types of propulsion systems depending on their design, size, and intended purpose. Here's a brief explanation of the common types of propulsion systems involved with drones:

**1. Electric Motors and Propellers:**

**Quadcopters:** Most consumer drones, like quadcopters, employ electric motors and propellers for propulsion. These systems offer precise control, agility, and stability by varying motor speeds to generate thrust and control movement.

**2. Fixed-Wing Propulsion:**

**Fixed-Wing Drones:** Fixed-wing drones, resembling traditional airplanes, use conventional aircraft engines or electric motors to power a single, large propeller. They are known for their efficiency in covering long distances but require a runway for takeoff and landing.

**3. Jet Engines:**

**Jet-Powered Drones:** Some high-speed and military drones utilize jet engines for propulsion. These engines, such as turbojets or turbfans, provide significant speed and altitude capabilities but are generally more complex and costly.



#### 4. Gasoline/Petrol Engines:

Gasoline-Powered Drones: Larger drones or those designed for heavy payloads might employ gasoline or petrol engines, similar to those found in small aircraft. These engines offer extended flight times and can carry heavier loads.

#### 5. Hybrid Propulsion:

Hybrid Drones: Hybrid drones combine different propulsion methods, such as using electric motors for vertical take-off and landing (VTOL) and transitioning to fixed-wing flight with a gas engine. This design offers the versatility of both multi-copters and fixed-wing aircraft.

#### 6. Ducted Fans:

Ducted Fan Drones: Ducted fan drones feature enclosed propellers within a protective shroud, enhancing safety and efficiency. They are often used in specialized applications, such as indoor or industrial inspections.

#### 7. Solar-Powered Drones:

Solar Drones: Some drones are equipped with solar panels on their wings or fuselage, allowing them to harness solar energy to extend their endurance for long-duration missions.

#### 8. Hydrogen Fuel Cells:

Hydrogen-Powered Drones: Hydrogen fuel cells are an emerging technology in drone propulsion. They use hydrogen gas and oxygen to generate electricity, offering longer endurance compared to batteries.

#### 9. Bio-Inspired Propulsion:

Bio-Inspired Drones: Researchers are exploring bio-inspired propulsion methods, such as flapping wings inspired by birds or insects, for specialized drone applications, such as reconnaissance or exploration.

#### 10. Tethered Drones:

Tethered Drones: Tethered drones are powered through a cable connected to a power source on the ground, enabling them to remain airborne for extended periods and carry heavy payloads.



Figure.1.5.1: Electric Motors & Propellers

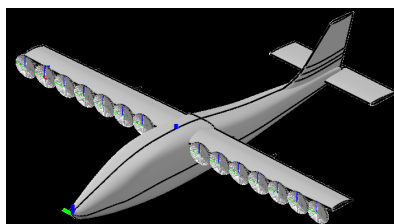


Figure.1.5.2: Fixed-Wing Propulsion

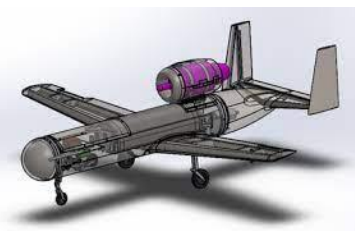


Figure:1.5.3: Jet Engines



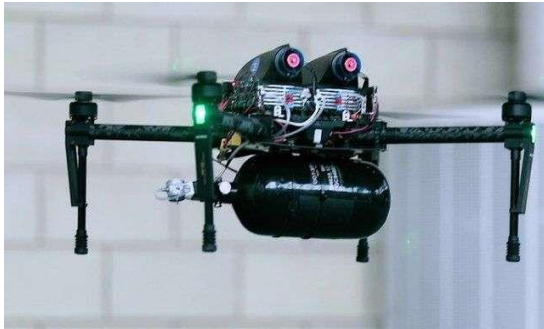


Figure.1.5.4: Gasoline-Powered Drones



Figure.1.5.5: Hybrid Drones



Figure.1.5.6: Ducted Fan Drones



Figure.1.5.7: Solar-Powered Drones



Figure.1.5.8: Hydrogen-Powered Drones

The choice of propulsion system depends on factors like the drone's mission, payload requirements, endurance, and environmental considerations. Different drones may use a combination of these propulsion technologies to achieve specific performance characteristics and meet diverse application needs.

## 1.6: FACTORS FOR PROPULSION SYSTEM SELECTION:



Figure.1.6.1: Factors for propulsion system selection

Selecting the right propulsion system for a drone is a critical decision that depends on various factors. Here's a brief explanation of the key considerations when choosing a propulsion system for a drone:

### 1. Mission Requirements:

The primary factor in propulsion system selection is the drone's intended mission. Different missions may require specific performance characteristics, such as endurance, speed, payload capacity, or agility.

### 2. Size and Weight Constraints:

The physical size and weight of the drone play a significant role in choosing a propulsion system. Smaller drones often use electric motors, while larger ones may employ gasoline engines or jet propulsion.

### 3. Endurance and Range:

For missions requiring long endurance or extended range, fuel-powered systems like gasoline engines or hydrogen fuel cells are preferred over batteries due to their higher energy density.

### 4. Payload Capacity:

The weight of the payload, such as cameras or sensors, impacts the choice of propulsion. More substantial payloads may require stronger motors or engines to maintain flight stability.

**5. Manoeuvrability and Agility:**

If the drone needs to perform precise manoeuvres or hover in place, electric motors with rapid thrust control (as seen in quadcopters) may be preferred for their agility.

**6. Take-off and Landing Requirements:**

The method of take-off and landing can influence propulsion choice. Vertical take-off and landing (VTOL) drones may use different systems for vertical and horizontal flight phases.

**7. Noise and Stealth:**

The noise generated by the propulsion system can be crucial, especially in military or surveillance applications. Electric motors are quieter than gasoline engines or jet propulsion.

**8. Environmental Considerations:**

Environmental factors, such as operating in urban areas or sensitive ecosystems, can influence propulsion choices. Electric propulsion is generally more environmentally friendly than internal combustion engines.

**9. Budget and Cost Constraints:**

The budget available for the drone project can dictate the propulsion system's selection. Electric motors are typically more cost-effective upfront, while gas engines or jet propulsion systems can be more expensive.

**10. Maintenance and Reliability:**

Consider the ease of maintenance and reliability of the chosen propulsion system. Electric motors often require less maintenance than combustion engines, making them attractive for many users.

**11. Regulatory Compliance:**

Adherence to local regulations and aviation authorities' rules may influence the choice of propulsion system, especially for commercial or industrial drone operations.

## 12. Technological Advancements:

Stay updated with the latest advancements in propulsion technology, as newer systems may offer improved efficiency, safety, and performance.

In summary, the selection of a propulsion system for a drone should align with the drone's intended purpose and operational requirements. Evaluating these factors comprehensively will help determine the most suitable propulsion system to achieve the desired mission objectives while considering practical constraints and environmental considerations.

## 1.7: COMPONENTS REQUIRED TO BUILD THE GESTURE CONTROLLED DRONE:

Building a quadcopter drone involves various components and subsystems. Here's a list of the complete components typically required for such a project:

### 1. Frame and Chassis:

Quadcopter frame or chassis to provide structural support and house other components.

### 2. Propulsion System:

Four electric motors (brushless DC motors).

Four propellers (two clockwise and two counterclockwise rotating).

### 3. Electronic Speed Controllers (ESCs):

Four ESCs to control the motor speed and thrust.

### 4. Flight Controller:

Flight controller board (e.g., Pixhawk, Arduino-based, or custom-designed) to manage flight stability, motor control, and navigation.

### 5. Power Distribution Board (PDB):

PDB to distribute power from the battery to the motors and other components.

### 6. Battery and Power System:

Lithium-polymer (LiPo) battery with an appropriate voltage and capacity.

Battery charger and monitoring system.

Voltage regulator if necessary for powering other components.

**7. Sensors:**

Gyroscope, accelerometer, and magnetometer (IMU) for flight stability and orientation.

GPS module for navigation and location tracking.

Ultrasonic or laser rangefinder for altitude measurement.

Optional additional sensors for specialized tasks (e.g., obstacle avoidance).

**8. Communication Systems:**

Radio control transmitter and receiver (RC transmitter) for manual control.

Wi-Fi or other communication modules for remote control and data transmission.

IoT communication modules (e.g., Wi-Fi, cellular, LoRa) for integrating with IoT devices and data exchange.

**9. Gesture Control System:**

Gesture recognition hardware and software for controlling the drone using hand gestures.

**10. Onboard Computer (Optional):**

A small onboard computer (e.g., Raspberry Pi, Nvidia Jetson) for running AI/ML algorithms and data processing.

**11. Camera or Sensors for IoT Integration:**

Cameras, environmental sensors, or other data acquisition devices for collecting IoT data.

**12. IoT Hardware and Software:**

IoT platform or microcontroller (e.g., Arduino, Raspberry Pi) to process and transmit IoT data.

IoT sensors and actuators for specific data collection and control tasks.

**13. Mechanical Components:**

- Landing gear for safe takeoff and landing.
- Mounts and brackets for securing components.

**14. Safety Features:**

- Propeller guards for safety.
- Battery safety mechanisms (e.g., LiPo bags).
- Emergency kill-switch or failsafe mechanisms.

**15. Tools and Accessories:**

- Soldering iron and soldering equipment.
- Screwdrivers, pliers, and other hand tools.
- Wiring, connectors, and heat shrink tubing.
- Velcro straps, zip ties, and adhesive materials for securing components.

**16. Computer and Software:**

- A computer for programming and configuring the flight controller.
- Software for flight controller configuration (e.g., Mission Planner, Betaflight Configurator).
- Programming and development tools for AI/ML algorithms.

**17. Safety Gear:**

- Safety goggles and gloves for handling components.
- Fire extinguisher for safety during battery charging.

**18. Testing and Debugging Equipment:**

- Multi-meter for electrical testing.
- Oscilloscope for signal analysis.
- Debugging tools for software development.



Figure.1.7.1: Drone Frame



Figure.1.7.2: Motor and Propeller



Figure.1.7.3: ESC

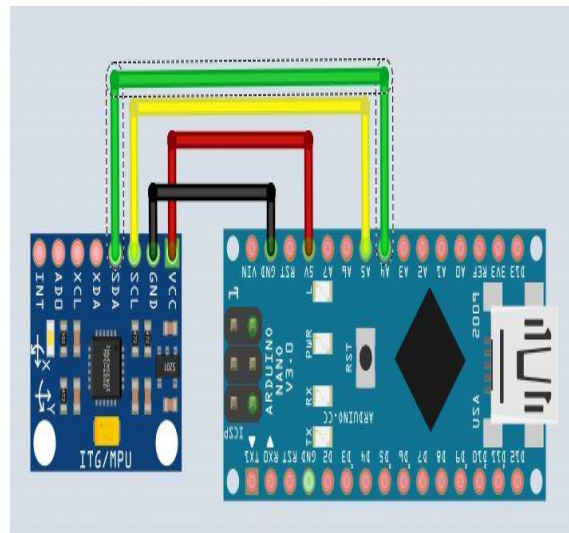


Figure.1.7.4: Flight Controller [Arduino Based]

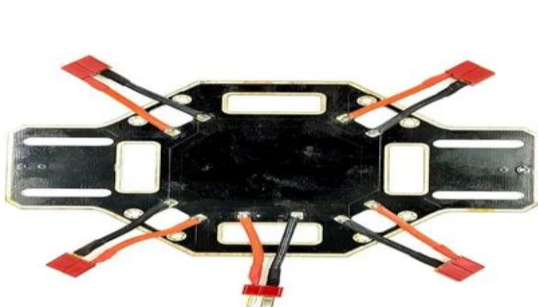


Figure.1.7.5: Power Distribution Board



Figure.1.7.6: Battery



## CHAPTER 2

### LITERATURE REVIEW

#### 2.1: RESEARCH PAPER ANALYSIS:

##### 2.1.1: HAND GESTURES FOR DRONE CONTROL USING DEEP LEARNING

The research paper titled "HAND GESTURES FOR DRONE CONTROL USING DEEP LEARNING" explores the emerging field of Human-Drone Interface (HDI) in the context of rapidly growing commercial drones or unmanned aerial vehicles (UAVs). With the proliferation of drones in various applications, including surveillance, transportation, and aerial photography, the need for innovative control methods has arisen. This paper presents a comprehensive investigation into using hand gestures as a novel HDI method to control drones effectively.

The research comprises three primary modules: the Hand Detector, Gesture Recognizer, and Drone Controller. A significant aspect of this work is the integration of deep learning techniques in the Hand Detector module, allowing real-time, high-precision hand detection and tracking from single RGB images. Dynamic image processing algorithms are introduced in the Gesture Recognizer module, offering flexibility in defining and recognizing hand gestures and motions without requiring retraining of the neural network. The Drone Controller module facilitates seamless communication between the system and the connected drone.

The paper concludes by highlighting the successful development of a robust system that enables real-time drone control through hand gestures, accommodating both single and dual-handed gestures. This system leverages deep learning-based hand detection, making it practical and versatile without the need for additional hardware. Furthermore, it emphasizes compatibility with drones supporting the MAVLink protocol, making it adaptable to a wide range of drones available in the market.

As for future work, the research team intends to transition this innovative system from the research phase to practical industry application by conducting thorough performance studies and gathering user feedback. Some key points for future research include exploring different neural network structures, collecting more diverse and balanced data, optimizing processing speed by transitioning to C++ and parallel processing, and conducting extensive real-world tests in various environments and with different types of drones.

In summary, this research paper presents a promising advancement in the field of HDI by harnessing deep learning and image processing to enable precise, real-time control of drones through hand gestures, with a focus on practicality, adaptability, and potential industry adoption.



### **2.1.2: G.O.D. (Gesture Operated Drone)**

The research paper, titled "G.O.D. (Gesture Operated Drone)," proposes the development of a small indoor drone entirely controlled by hand gestures. The project's central component is a user-friendly graphical user interface (GUI) utilizing a webcam to establish communication with the drone. This GUI provides users with real-time information about the drone's status and allows them to control it through hand gestures. For example, a thumbs-up gesture prompts the drone to increase its flying altitude promptly. The project's motivation stems from the team's enthusiasm to explore a complex and innovative endeavor. As computer engineering students, they bring their knowledge in microcontrollers, communication networks, graphic user interfaces, embedded systems programming, and PCB design to the table. Additionally, their individual expertise in computer vision and linear control systems positions them to create a unique product—a drone exclusively controlled by hand movements. The team aims to develop a low-cost solution, ensuring accessibility to the public and opening doors for further improvements beyond the initial prototype. The project's objectives include building a working prototype with a focus on simplicity and repeatability, expanding the repertoire of hand signals, and making the project as cost-effective as possible. By achieving these goals, the team aspires to introduce a groundbreaking product at the intersection of emerging technologies: drones, computer vision, and machine learning.

### **2.1.3: Flying Drone Controller by Hand Gesture Using Leap Motion**

The research paper, titled "Flying Drone Controller by Hand Gesture Using Leap Motion," explores the implementation of Leap Motion technology as a mechanism for controlling a drone through hand gestures. This project combines hardware components, including the Arduino microcontroller, a SYMA X5SC Explorer 2 drone, and the MCP 4725 digital-to-analog converter, with software tools like Microsoft Visual Studio 2019. The Leap Motion sensor interprets hand gestures, and Arduino processes these instructions, subsequently transmitting commands to the drone's remote controller. The drone responds to these hand gesture instructions, taking flight based on specific hand movements, including throttle, pitch, and roll. This innovative system is designed to facilitate touchless drone control, with potential applications in farming, photography, and military contexts. The successful execution of this project highlights the feasibility of using hand gestures to enhance drone control, eliminating the need for traditional control interfaces and offering improved maneuverability. The paper also emphasizes the importance of careful planning and testing to ensure that the system achieves its objectives and performs effectively. Overall, this research contributes to the advancement of touchless drone control technology, opening new possibilities for future developments in this field.

### **2.1.4: Gesture Controlled Drone**

The research paper titled "Gesture Controlled Drone" explores the implementation of gesture control technology to command the motion of a quadcopter. This innovative

technology relies on image processing, where gestures are detected and recognized by a webcam connected to a laptop or similar device. The OpenCV and Python programming languages are employed to interpret these gestures, allowing for precise control of the drone using a Bluetooth module HM10.

In terms of future work, the potential applications of this project are expansive. Drone technology has the capacity to revolutionize various industries, particularly in agriculture. Drones equipped with gesture control capabilities can facilitate precise data collection and analysis for tasks like soil and field assessment, irrigation, crop spraying, monitoring, and health assessment. This has the potential to significantly increase agricultural productivity and streamline insurance claim processes in case of crop losses.

Moreover, the gesture control feature can be leveraged to assist differently-abled individuals in performing various tasks with ease, eliminating the need for complex remote controls. This technology enables individuals to control drones through simple hand gestures, making it accessible and user-friendly.

In conclusion, this project has broad implications, particularly in surveillance for both civil and military purposes. It can enhance safety by providing border monitoring capabilities in potentially ambiguous situations. The advantages of this technology include the speed and cost-effectiveness of drones, convenience for soldiers in hostile terrains, precise maneuvering through gestures, reduced human error, and simplified control mechanisms. Overall, this research presents an innovative approach to drone control with far-reaching possibilities in various domains.

#### **2.1.5: Wearable Drone Controller: Machine Learning-Based Hand Gesture Recognition and Vibrotactile Feedback**

The research paper titled "Wearable Drone Controller: Machine Learning-Based Hand Gesture Recognition and Vibrotactile Feedback" introduces an innovative wearable drone control system. This system utilizes an inertial measurement unit (IMU) placed on the back of the user's hand to detect and interpret hand motions. Machine learning models are employed to analyse and classify these gestures, providing a seamless means of controlling the drone. Additionally, the system offers obstacle awareness by providing vibrotactile feedback to the user via a vibration motor attached to the wrist, thereby enhancing the user's situational awareness during drone operation. The paper presents simulation experiments and subjective evaluations conducted to assess the convenience and effectiveness of the controller. The results indicate a high classification accuracy of 97.9% for hand gestures, highlighting the system's robustness. Furthermore, the vibrotactile feedback proved valuable, especially when visual information was limited. Real-world experiments with a drone validated the practicality and usability of the proposed wearable controller, positioning it as a natural and efficient interface for drone operation. This research contributes to the development of intuitive and user-friendly drone control systems with applications in various domains.

### **2.1.6: UAV manipulation by hand gesture recognition:**

The research paper titled "UAV Manipulation by Hand Gesture Recognition" explores the development of an unmanned aerial vehicle (UAV) control system based on human gesture recognition. The study encompasses both dynamic and static gestures, including repetitive hand movements and fixed hand positions. Two distinct methods are proposed: a feature-based (FB) approach that detects the hand's position using image features estimated by Fast Fourier Transform (FFT), and a machine learning (ML) method that identifies gestures based on human skeletal data obtained using the OpenPose library.

In experimental evaluations, the paper compares the results of gesture recognition achieved by each method. The FB method exhibits higher recognition rates under specific conditions tailored to its design, while the ML method outperforms in various other scenarios. Additionally, the ML method proves effective in terms of scalability, allowing for the incorporation of additional gesture types.

The research further introduces a method for continuous gesture recognition to control the UAV's movements. The study highlights the ML method's ability to distinguish between various gestures and adapt to different environments, making it more versatile for real-world applications. The paper discusses the potential for correcting human skeletal data obtained by OpenPose in time-series scenarios and the expansion of gesture recognition to movements beyond the right hand.

Future work is envisioned to include outdoor gesture recognition experiments under various environmental conditions and the enhancement of the system for autonomous flight control using the UAV's on-board computer. This research contributes to the development of UAV control systems that can be intuitively operated through hand gestures, offering potential applications in diverse operational environments.

### **2.1.7: Development, Control Adjustment, and Gesture Recognition of a Quadrotor Helicopter**

The research paper titled "Development, Control Adjustment, and Gesture Recognition of a Quadrotor Helicopter" delves into the exploration of quadcopter technology and its applications, with a particular focus on gesture recognition for control. Quadrotor helicopters have gained popularity for their ease of operation and stability, making them valuable in various fields such as aerial photography, competitive flying, and search-and-rescue missions.

The study involves the construction of a quadcopter using Arduino NANO and MPU-6050 modules, offering insights into the underlying flight principles of such systems. A proportional integrative derivative (PID) controller is modified to enhance flight stability. The research integrates Leap Motion technology to investigate hand gestures, assess the success rate of each gesture, and combine them with a Parrot AR.Drone 2.0 for gesture-based controls. The paper also discusses challenges related to gesture recognition and addresses them through code modifications in the gesture control program.

The results of the gesture control experiments demonstrate high performance in gesture recognition, validating its suitability for controlling a quadcopter. The paper emphasizes cost-effective manufacturing and wireless control through a smartphone Bluetooth connection, reducing overall expenses while meeting control efficiency requirements.

In conclusion, this research contributes to the understanding of quadcopter fabrication, PID controller adjustments, and the utilization of gesture recognition for precise control. Insights gained from this study can inform the development of cost-effective and efficient quadcopter systems, with potential applications in various domains. The work acknowledges support from the Ministry of Science and Technology, underscoring its importance in advancing technology and research.

#### **2.1.8: Review of Propulsion System Design Strategies for Unmanned Aerial Vehicles**

The research paper titled "Review of Propulsion System Design Strategies for Unmanned Aerial Vehicles" underscores the intricate nature of designing propulsion systems for Unmanned Aerial Vehicles (UAVs) and the need for a comprehensive multidisciplinary approach right from the initial design stages. The paper conducts a systematic review of scientific literature that focuses on the methods employed in defining and sizing UAV propulsion systems. A total of 123 papers are meticulously evaluated based on two custom classification taxonomies, one addressing the primary objectives and the other specific content of these works.

The findings from the literature review culminate in the development of an integrated framework tailored for the functional design of UAV propulsion systems. This framework is designed to guide designers in the early stages of selecting propulsion candidates and rapidly sizing the supply system, streamlining the initial design process. The research underscores the variability in design methods, heavily influenced by the intended applications and operational conditions of UAVs. Many of the reviewed papers emphasize the detailed design of specific drone components and propulsion elements.

In conclusion, this paper highlights the distinctive challenges inherent in designing UAV propulsion systems, necessitating a multidisciplinary approach from the outset. The systematic literature review provides valuable insights into the diversity of design methods influenced by application contexts. It also culminates in the proposal of a functional framework for preliminary propulsion system design in UAVs, serving as a valuable resource for designers in this field. The research underscores the keen interest of the scientific community in the design of specific UAV components and elements.

#### **2.1.9: Propulsion optimization of a quadcopter in forward state**

The research paper titled "Propulsion Optimization of a Quadcopter in Forward State" focuses on the optimal design of a propulsion system tailored for the forward flight state of a

quadcopter. This study employs a comprehensive approach, beginning with the establishment of a forward rotor model using a linear regression method. This model is based on aerodynamic data acquired through the empirical blade element momentum theory (BEMT). Additionally, a fuselage model is constructed using high-fidelity computational fluid dynamics (CFD) techniques. These models collectively enable precise and efficient derivation of the UAV's performance metrics, incorporating propulsion system parameters.

The heart of the research lies in the optimization of the propulsion system, aiming to enhance the forward flight endurance of the quadcopter. To achieve this, a genetic intelligent optimization algorithm is employed. The outcomes of this optimization process are then validated through wind-tunnel tests. The results of these experiments affirm that the optimized propulsion system effectively enhances the forward flight endurance of the quadcopter.

In conclusion, this study presents a novel approach to the optimization of a quadcopter's propulsion system, emphasizing the forward flight state. The establishment of an accurate aerodynamic model, coupled with the optimization process, results in a propulsion system that is well-matched and finely-tuned for forward flight. The study's findings underscore the significance of rotor design in achieving optimal performance during both hovering and forward flight, with implications for enhancing the overall capabilities of quadcopter UAVs.

#### **2.1.10: Electric propulsion system sizing methodology for an agriculture multi-copter**

The research paper titled "Electric Propulsion System Sizing Methodology for an Agricultural Multi-copter" addresses the crucial challenge of selecting an appropriate propulsion system for electric multi-copters, with a particular focus on agricultural applications. Designing electric multi-copters that can deliver the desired performance is a complex task due to the diversity of available components such as motors, ESCs, batteries, and propellers, coupled with potentially misleading information in the market.

This paper presents a systematic methodology for sizing the electric propulsion system, enabling the design and manufacturing of a multi-copter that can meet specific mission requirements. The study leverages manufacturer specifications and experimental data to establish and parameterize the relationships among various components within the propulsion system. The research also evaluates the sensitivity of design parameters concerning the Gross Take-Off Weight (GTOW).

To validate the proposed methodology, an Agriculture Quadcopter is built and compared to the sizing results. Remarkably, the GTOW prediction using this methodology demonstrates a minimal difference of within  $\pm 4\%$  when compared to an existing model. Additionally, the paper introduces the concept of the maximum thrust-to-weight ratio and considers payload limitations during pesticide spraying operations, enhancing the accuracy of the Quadcopter sizing process.

In conclusion, this research paper introduces a valuable algorithm for sizing the propulsion system of multi-copters, yielding GTOW predictions with high precision. Performance parameters closely align with manufacturer data, with minor discrepancies of within 13%. The methodology provides a robust framework for designing efficient electric multi-copters tailored to specific agricultural tasks, ensuring optimal performance and reliability in real-world applications.

## 2.2: TABLES; GRAPHS & SOFTWARE OUTPUTS:

Table 2.1.1: Samples of the input rows for the training dataset.

filename	width	height	class	xmin	ymin	xmax	ymax
1528664299.56_pic00191.jpg	800	600	4-hand	127	118	296	455
2_pic00038.jpg	800	600	5-hand	413	97	642	470
2_pic00154.jpg	800	600	5-hand	319	91	489	302
2_pic00175.jpg	800	600	5-hand	454	77	626	356
35_pic00004.jpg	800	600	5-hand	65	179	278	521
pic00054.jpg	800	600	5-hand	284	90	492	362
46_pic00003.jpg	800	600	5-hand	137	93	337	450
2_pic00017.jpg	800	600	5-hand	236	130	463	477
55_pic00016.jpg	800	600	0-hand	415	222	548	385
35_pic00030.jpg	800	600	5-hand	383	159	572	462

Table 2.2.1: Comparing ESCs

	Size	Weight	Price	Amp Rating	Compatibility
Emax BLHeli	3.1 x 2.0 x 3.1 inches	4 oz	\$40	20A	✓
RC Electric Parts	2.1 x 1.0 x 0.5 inches	4.5 oz	\$16	30A	✓
Crazepony	1.0 x 0.5 x 0.2 inches	1 oz	\$45	35A	✓

Table 2.2.2: Motor Comparison

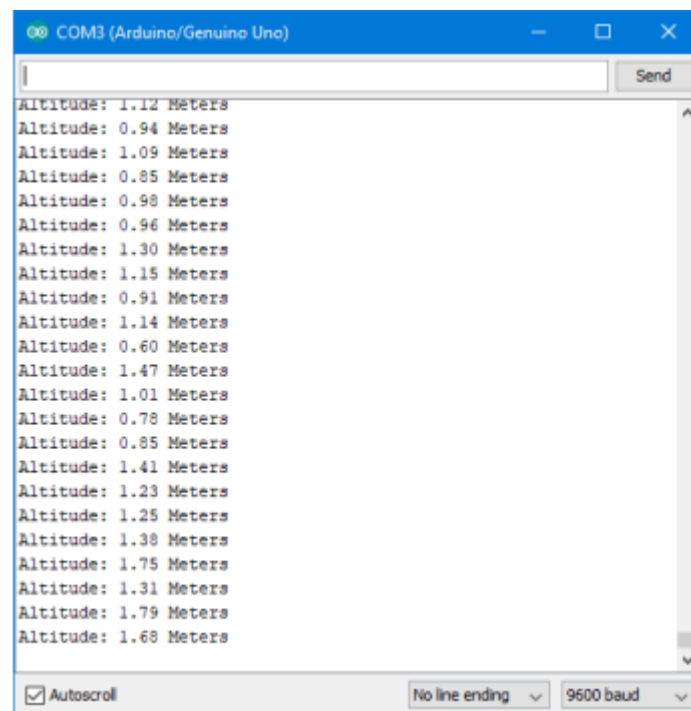
	Weight	Type	Price	Strator Size	KV
Hobbypower	1.5 oz	Brushless Outrunner	\$40	2212	1000
LiTacc Model	2.0 oz	Brushless Outrunner	\$48	2212	1200
Woafly	2.5 oz	Brushless Inrunner	\$31	2212	920
abcGoodefig	1.44 oz	Brushless Outrunner	\$45	2212	2200



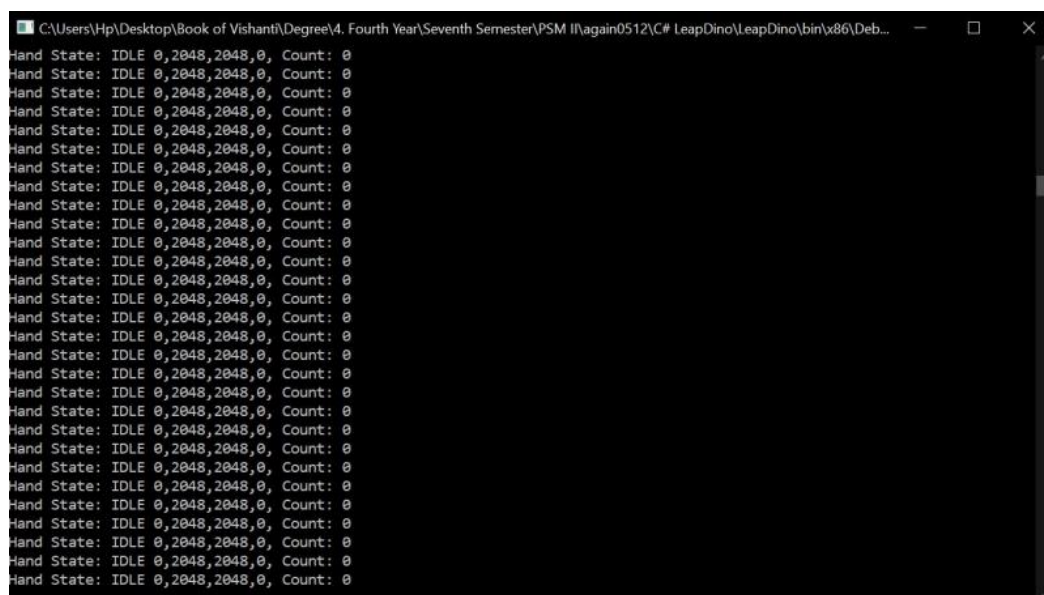
Table 2.2.3: MCU Comparison

	Clock Rate	RAM	Flash	Price
ATmega328p	16 MHz	2 KB	32 KB	\$10
AT91SAM3X8E	84 MHz	96 KB	512 KB	\$20
ATSAMD21G18	48 MHz	32 KB	256 KB	\$15

Software Output 2.2.1: Altitude Sensor Output



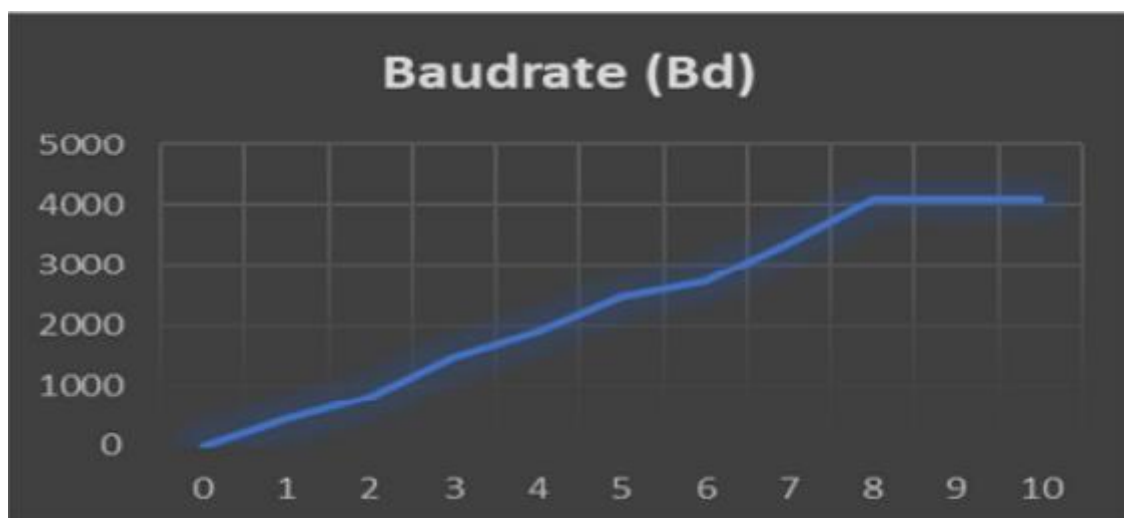
Software Output 2.3.1: Console App displayed IDLE State



Software Output 2.3.2: Console App displayed FLYING State

```
C:\Users\Hp\Desktop\Book of Vishanti\Degree\4. Fourth Year\Seventh Semester\PSM II\again0512\C# LeapDino\LeapDino\bin\x86\Deb...
Hand State: FLYING 2875,585,2468,0, Count: 5
Hand State: FLYING 2875,588,2467,0, Count: 5
Hand State: FLYING 2876,587,2468,0, Count: 5
Hand State: FLYING 2878,586,2470,0, Count: 5
Hand State: FLYING 2878,586,2470,0, Count: 5
Hand State: FLYING 2879,588,2472,0, Count: 5
Hand State: FLYING 2879,590,2472,0, Count: 5
Hand State: FLYING 2881,592,2474,0, Count: 5
Hand State: FLYING 2881,592,2474,0, Count: 5
Hand State: FLYING 2881,595,2473,0, Count: 5
Hand State: FLYING 2881,595,2473,0, Count: 5
Hand State: FLYING 2883,597,2473,0, Count: 5
Hand State: FLYING 2883,597,2473,0, Count: 5
Hand State: FLYING 2885,599,2474,0, Count: 5
Hand State: FLYING 2885,599,2474,0, Count: 5
Hand State: FLYING 2886,598,2474,0, Count: 5
Hand State: FLYING 2886,598,2474,0, Count: 5
Hand State: FLYING 2887,598,2475,0, Count: 5
Hand State: FLYING 2887,598,2475,0, Count: 5
Hand State: FLYING 2889,597,2475,0, Count: 5
Hand State: FLYING 2889,597,2475,0, Count: 5
Hand State: FLYING 2887,597,2474,0, Count: 5
Hand State: FLYING 2887,597,2474,0, Count: 5
Hand State: FLYING 2888,600,2473,0, Count: 5
Hand State: FLYING 2888,603,2472,0, Count: 5
Hand State: FLYING 2888,603,2472,0, Count: 5
Hand State: FLYING 2889,604,2472,0, Count: 5
Hand State: FLYING 2890,604,2474,0, Count: 5
Hand State: FLYING 2890,602,2475,0, Count: 5
```

Graph 2.3.1: Graph indicate the drone connection



Graph 2.3.2: Graph indicate the drone static positioning





Graph 2.3.3: Graph indicate the drone dynamic positioning

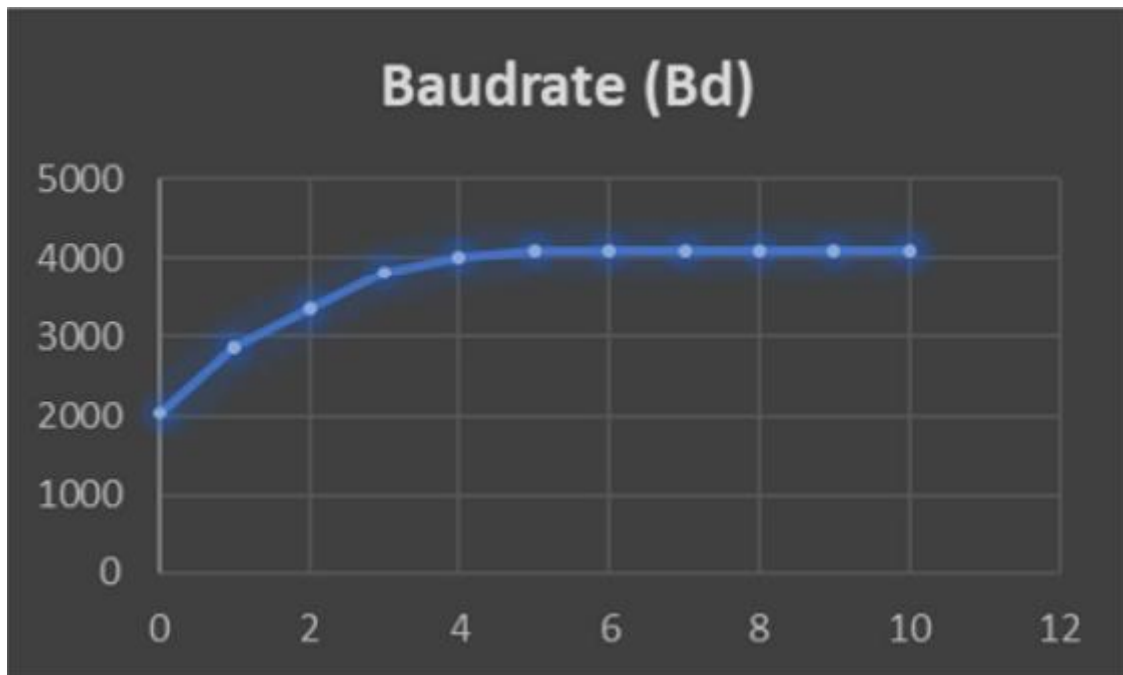


Table 2.5.1: Results of obstacle avoidance using vibrotactile feedback. Without vibrotactile feedback, the drone failed to avoid obstacles; meanwhile, collision avoidance was completed successfully with vibrotactile feedback.

Participant No.	First Trial (without Feedback)	Second Trial (with Feedback)	Third Trial (with Feedback)
Participant 1	2/4 *	4/4	4/4
Participant 2	1/4	4/4	4/4
Participant 3	1/4	4/4	4/4
Participant 4	2/4	4/4	4/4
Participant 5	1/4	4/4	4/4
Participant 6	2/4	4/4	4/4
Participant 7	2/4	4/4	4/4
Participant 8	1/4	4/4	4/4
Participant 9	2/4	4/4	4/4
Participant 10	2/4	4/4	4/4
Participant 11	2/4	4/4	4/4
Participant 12	2/4	4/4	4/4

\* Successful/avoidance trials.

Table 2.5.2: Results of subjective evaluation by participants in gesture-based drone control.

Question	Direct Mode	Gesture Mode
1. The proposed gesture was natural for me.	6.4 ± 0.6	6.0 ± 0.7
2. I felt physical discomfort while controlling.	1.6 ± 0.6	2.0 ± 0.9
3. My hand and arm were tired while controlling.	2.0 ± 0.6	2.4 ± 0.8
4. The proposed gesture was user-friendly.	6.5 ± 0.9	6.3 ± 1.4
5. I felt the convenience of controlling a drone with one hand.	6.5 ± 0.6	6.6 ± 0.5
6. It was interesting to fly a drone with a gesture.	6.5 ± 1.0	6.9 ± 0.3

Table 2.5.3: Results of subjective evaluation by participants in vibrotactile feedback.

Question	Mean ± SD
1. The vibrotactile feedback was helpful for obstacle avoidance.	6.9 ± 0.3
2. The vibration intensity was appropriate.	6.6 ± 0.6
3. My hand and wrist were tired by vibrotactile feedback.	1.4 ± 0.8
4. The obstacle avoidance was difficult without the vibrotactile feedback condition.	6.5 ± 0.7
5. If I flew a drone in real life, vibrotactile feedback would be helpful.	6.5 ± 0.3

Table 2.6.1: Gesture recognition rate (FB method).

Subject	Left dynamic	Right dynamic	Under dynamic	Under static	Right static	Left static	Reference static	Rate (%)
A	10/10	10/10	10/10	10/10	10/10	10/10	9/10	98.6
B	10/10	10/10	10/10	10/10	10/10	10/10	10/10	100.0
C	10/10	10/10	10/10	10/10	10/10	10/10	10/10	100.0
D	10/10	10/10	9/10	10/10	10/10	10/10	9/10	97.1
							Total	98.9

Table 2.6.2: Gesture recognition rate (ML method).

Subject	Left dynamic	Right dynamic	Under dynamic	Under static	Right static	Left static	Reference static	Rate (%)
A	10/10	10/10	9/10	10/10	10/10	10/10	7/10	94.3
B	10/10	10/10	10/10	10/10	10/10	10/10	10/10	100.0
C	10/10	10/10	10/10	10/10	10/10	10/10	8/10	97.1
D	10/10	10/10	10/10	10/10	10/10	10/10	10/10	100.0
							Total	97.9

Table 2.6.3: Gesture recognition rate under flying UAV

Subject	Left dynamic	Right dynamic	Under dynamic	Under static	Right static	Left static	Reference static	Rate (%)
A	5/5	5/5	5/5	5/5	5/5	5/5	5/5	100.0
B	5/5	5/5	4/5	5/5	5/5	5/5	5/5	97.1
C	5/5	5/5	4/5	5/5	5/5	5/5	5/5	97.1
D	5/5	5/5	4/5	5/5	5/5	4/5	5/5	94.3
							Total	97.1

Table 2.7.1: Results of the gesture recognition control experiments.

<b>Gestures</b>	<b>Success</b>	<b>Total</b>	<b>Success rate</b>
Right	43	50	86%
Left	41	50	82%
Back	47	50	94%
Forward	48	50	96%
Couterclockwise	48	50	96%
Clockwise	47	50	94%
Down	48	50	96%
Up	47	50	94%
Land	46	50	92%
Take off	45	50	90%

## CHAPTER 3

### METHODOLOGY

#### 3.1: ANALYTICAL

##### 3.1.1: AIML MODEL FOR PROPULSION SYSTEM ANALYSIS

The "AIML Model for Propulsion System Analysis" refers to the application of Artificial Intelligence (AI) and Machine Learning (ML) techniques to assess and analyze propulsion systems used in various fields, such as aviation or drone technology. This model utilizes AI and ML algorithms to process data and make predictions, optimizations, and assessments related to propulsion system performance, efficiency, and reliability. By harnessing these advanced technologies, it enables engineers and researchers to gain deeper insights, enhance design processes, and improve the overall functionality and safety of propulsion systems in a wide range of applications.

**Code:**

```
import pandas as pd

import matplotlib.pyplot as plt

# Load the data from the CSV file into a pandas DataFrame
data = pd.read_csv('drone_data.csv')

# Calculate the Thrust to Weight Ratio
data['Thrust_to_Weight_Ratio'] = data['Thrust (N)'] / data['Weight (N)']

# Calculate the Flight Time in minutes
data['Flight_Time_minutes'] = data['Flight Time (seconds)'] / 60

# Calculate the Power Consumption in Watts
data['Power_Consumption_Watts'] = data['Motor RPM (RPM)'] * data['Current Draw (Amps)']

# Calculate the Gesture Recognition System Power Consumption
data['Gesture_Recognition_Power_Consumption'] = data['Gesture Recognition Power (W)'] * data['Gesture Recognized (0/1)']

# Calculate the Total Power Consumption including Gesture Recognition System Power
data['Total_Power_Consumption_Watts'] = data['Power_Consumption_Watts'] + data['Gesture_Recognition_Power_Consumption']

# Calculate the Battery Capacity in Amp-hours
data['Battery_Capacity_AmpHours'] = data['Battery Capacity (mAh)'] / 1000
```

```
# Calculate the Battery Voltage Selection
```

```
data['Battery_Voltage_Selection'] = data['Total_Power_Consumption_Watts'] / data['Current Draw (Amps)']
```

```
# Calculate the Payload Consideration
```

```
data['Payload_Consideration'] = data['Payload Mass (kg)'] / data['Thrust (N)']
```

```
# Print the calculated values
```

```
print("Thrust to Weight Ratio:")
```

```
print(data['Thrust_to_Weight_Ratio'])
```

```
print("\nFlight Time (minutes):")
```

```
print(data['Flight_Time_minutes'])
```

```
print("\nPower Consumption (Watts):")
```

```
print(data['Power_Consumption_Watts'])
```

```
print("\nGesture Recognition System Power Consumption:")
```

```
print(data['Gesture_Recognition_Power_Consumption'])
```

```
print("\nTotal Power Consumption (Watts):")
```

```
print(data['Total_Power_Consumption_Watts'])
```

```
print("\nBattery Capacity (Amp-hours):")
```

```
print(data['Battery_Capacity_AmpHours'])
```

```
print("\nBattery Voltage Selection:")
```

```
print(data['Battery_Voltage_Selection'])
```

```
print("\nPayload Consideration:")
```

```
print(data['Payload_Consideration'])
```

```
# Plot the graphs
```

```
plt.figure(figsize=(12, 8))
```

```
plt.subplot(2, 3, 1)
```

```
plt.plot(data['Thrust_to_Weight_Ratio'])
```

```
plt.xlabel('Sample')
```

```
plt.ylabel('Thrust to Weight Ratio')
```

```
plt.title('Thrust to Weight Ratio over Time')
```

```
plt.subplot(2, 3, 2)
```

```
plt.plot(data['Flight_Time_minutes'])
plt.xlabel('Sample')
plt.ylabel('Flight Time (minutes)')
plt.title('Flight Time over Time')
plt.subplot(2, 3, 3)
plt.plot(data['Power_Consumption_Watts'])
plt.xlabel('Sample')
plt.ylabel('Power Consumption (Watts)')
plt.title('Power Consumption over Time')
plt.subplot(2, 3, 4)
plt.plot(data['Total_Power_Consumption_Watts'])
plt.xlabel('Sample')
plt.ylabel('Total Power Consumption (Watts)')
plt.title('Total Power Consumption over Time')
plt.subplot(2, 3, 5)
plt.plot(data['Battery_Capacity_AmpHours'])
plt.xlabel('Sample')
plt.ylabel('Battery Capacity (Amp-hours)')
plt.title('Battery Capacity over Time')
plt.subplot(2, 3, 6)
plt.plot(data['Payload_Consideration'])
plt.xlabel('Sample')
plt.ylabel('Payload Consideration')
plt.title
```

### 3.1.2: GESTURE RECOGNITION USING ARDUINO – UNO

The Arduino Uno microcontroller serves as the brain of the operation, processing the signals from sensors that detect hand movements. Users can interact with the drone by making specific hand gestures, and the Arduino Uno translates these gestures into commands for the drone, allowing it to perform actions like take-off, landing, or changing direction. This technology simplifies drone operation and offers a more intuitive and interactive way of controlling these aerial devices, making it accessible to a broader range of users.

#### CODE:

```
#include <Servo.h>

// Define pin numbers for motors
int motorFrontLeftPin = 9;
int motorFrontRightPin = 10;
int motorBackLeftPin = 11;
int motorBackRightPin = 12;

// Define servo motor for camera/gimbal control
Servo cameraServo;

// Define pins for gesture control (e.g., using an IR sensor)
int gestureSensorPin = A0;

// Define obstacle detection sensor pins (e.g., ultrasonic or LiDAR)
int obstacleFrontPin = 2;
int obstacleBackPin = 3;
int obstacleLeftPin = 4;
int obstacleRightPin = 5;

// Variables to store sensor readings
int gestureValue;
float obstacleFrontDistance;
float obstacleBackDistance;
float obstacleLeftDistance;
float obstacleRightDistance;

void setup() {
    // Initialize motor control pins as outputs
```

```
pinMode(motorFrontLeftPin, OUTPUT);
pinMode(motorFrontRightPin, OUTPUT);
pinMode(motorBackLeftPin, OUTPUT);
pinMode(motorBackRightPin, OUTPUT);
// Attach servo motor
cameraServo.attach(6);
// Initialize gesture sensor and obstacle detection sensors
pinMode(gestureSensorPin, INPUT);
pinMode(obstacleFrontPin, INPUT);
pinMode(obstacleBackPin, INPUT);
pinMode(obstacleLeftPin, INPUT);
pinMode(obstacleRightPin, INPUT);
// Initialize serial communication for debugging (optional)
Serial.begin(9600);
}
void loop() {
  // Read sensor values
  gestureValue = analogRead(gestureSensorPin);
  obstacleFrontDistance = readObstacleSensor(obstacleFrontPin);
  obstacleBackDistance = readObstacleSensor(obstacleBackPin);
  obstacleLeftDistance = readObstacleSensor(obstacleLeftPin);
  obstacleRightDistance = readObstacleSensor(obstacleRightPin);
  // Perform gesture recognition and obstacle avoidance logic
  // Implement your gesture control and obstacle avoidance algorithms here
  // Adjust motor speeds and cameraServo position accordingly
  // Debugging output (optional)
  Serial.print("Gesture Value: ");
  Serial.println(gestureValue);
  Serial.print("Obstacle Front Distance: ");
```

```
Serial.println(obstacleFrontDistance);
Serial.print("Obstacle Back Distance: ");
Serial.println(obstacleBackDistance);
Serial.print("Obstacle Left Distance: ");
Serial.println(obstacleLeftDistance);
Serial.print("Obstacle Right Distance: ");
Serial.println(obstacleRightDistance);
// Delay to control loop frequency
delay(100);
}

float readObstacleSensor(int pin) {
    // Implement code to read and return obstacle sensor distance
    // This will depend on the type of sensor you are using (e.g., ultrasonic or LiDAR)
    // You may need to use libraries specific to your sensor for accurate readings
}

// Implement your gesture recognition, obstacle avoidance, and motor control functions
here

// Adjust motor speeds and cameraServo position based on sensor readings and gestures
```

This code is designed for controlling a drone with various sensors and servos, including gesture recognition and obstacle detection. Here's a short description of the code:

**Initialization:**

The code begins by defining pins for motors, a servo motor for camera/gimbal control, gesture control (e.g., using an IR sensor), and obstacle detection sensors (e.g., ultrasonic or LiDAR). It also declares variables for storing sensor readings.

**Setup:**

In the setup function, it initializes motor control pins as outputs, attaches the servo motor to a specific pin, and initializes the gesture sensor and obstacle detection sensors. Serial communication for debugging is also initiated.

**Loop:**

In the loop function, it continuously reads sensor values, including gesture recognition data and obstacle distances using custom functions. The main logic of gesture



recognition and obstacle avoidance should be implemented here, adjusting motor speeds and camera servo positions accordingly.

**Debugging Output (Optional):**

The code optionally outputs sensor readings to the serial monitor for debugging purposes.

**Custom Function:**

A custom function, `readObstacleSensor`, is defined for reading obstacle sensor distances. This function should be implemented based on the type of obstacle sensor used.

**Comments:**

The code includes comments to guide where custom implementations should occur. Specifically, the gesture recognition, obstacle avoidance, and motor control logic should be added as per the project requirements.

This code serves as a foundation for building a drone control system that responds to gestures and avoids obstacles, making it versatile and interactive.

**3.1.3: GENERAL FLIGHT CONTROLLER CODE FOR A DRONE**

A general Arduino Uno flight controller code for a quadcopter drone serves as the brain of the drone, providing instructions for the motors to maintain stable flight. Here's a description of the typical components and functions of such a code:

**Initialization:**

The code begins by initializing various parameters and libraries. It may define constants, pins for motor control, and communication protocols.

**Sensor Integration:**

The flight controller code reads data from various sensors, such as gyroscopes and accelerometers, to determine the drone's orientation in real-time. This sensor data is crucial for stabilization.

**PID Control:**

Proportional-Integral-Derivative (PID) control algorithms are used to adjust motor speeds and maintain the drone's desired orientation (roll, pitch, and yaw). The PID gains are tuned to achieve stable flight.

**Receiver Input:**

The code listens to signals from a remote controller or receiver to control the drone's throttle, roll, pitch, and yaw. This input allows the pilot to control the drone's movement.

**Motor Control:**

Based on the sensor data and PID control, the code calculates and adjusts motor speeds to keep the drone level and stable. It converts the PID control output into motor commands.

**Arming and Disarming:**

The flight controller code includes safety features that require specific actions or sequences to arm the drone (enable motors) and disarm it (shut down motors).

**Communication Protocols:**

Communication protocols, such as PWM (Pulse Width Modulation), may be used to send motor commands to the Electronic Speed Controllers (ESCs).

**Calibration and Configuration:**

The code may allow for sensor calibration to ensure accurate readings and configuration of PID gains. These parameters can be tuned to suit the drone's specific characteristics.

**Real-Time Monitoring:**

Some flight controller codes include real-time monitoring features. The drone's status, sensor data, and other vital information can be transmitted to a ground station or displayed on an onboard screen.

**Safety Measures:**

The code often includes safety measures, such as low voltage warnings, emergency motor cutoffs, or return-to-home procedures in case of signal loss.

**Logging and Debugging:**

Data logging may be implemented to record flight data, which can be useful for post-flight analysis and debugging.

**Flight Modes:**

Flight controller codes may support different flight modes, such as stabilization mode, acrobatic mode, or autonomous modes for GPS-based navigation.

**Customization:**

The code can be customized to suit the specific needs and hardware of the quadcopter, allowing for flexibility and optimization.

**Fail-Safe Procedures:**

Implement fail-safe procedures for unexpected events, such as low battery, communication loss, or sensor errors.

**Control Loop:**

The main code typically runs within a control loop, ensuring that sensor data is continually processed and that the motors are adjusted in real-time to maintain stability.

A well-designed flight controller code is essential for safe and stable quadcopter flight. It serves as the intermediary between the pilot's commands and the physical components of the drone, ensuring that it responds accurately to user input while maintaining balance and control. The code is often refined through testing and tuning to achieve optimal flight performance.

**3.1.4: Manual Calculations – Formulae****Total Thrust Required:**

$$\text{Total Thrust Required} = (\text{Total Weight}) \times \text{TWR}$$

**Thrust Per Motor:**

$$\text{Thrust Per Motor} = (\text{Total Thrust Required}) / (\text{Number Of Motors})$$

**Motor RPM:**

$$\text{Motor RPM} = (\text{Voltage} \times \text{kv}) / (\text{Propeller Pitch})$$

**Payload Power Consumption:**

$$\text{Payload Power Consumption} = (\text{Weight of Payload}) \times (\text{Power Requirement of Payload})$$

**Total Power Consumption:**

$$\text{Total Power Consumption} = (\text{Power Consumption}) + (\text{Payload Power Consumption})$$

**Flight Time:**

$$\text{Flight Time} = (\text{Battery Capacity} \times \text{Discharge}) / (\text{AAD})$$

**Average Ampere Draw (AAD):**

$$\text{AAD} = (\text{Power}) / (\text{Voltage})$$

**Power Required:**

$$\text{Power Required} = (\text{Thrust} \times \text{TWR}) / (\text{Motor Efficiency} \times 2)$$

**Efficiency of Motors & Propellers:**

$$\text{Efficiency of Motors \& Propellers} = (\text{Power Output}) / (\text{Power Input})$$

**Gesture Recognition System Power Consumption:**

$$\text{GRSPC} = (\text{P\_Camera} + \text{P\_Processor} + \text{P\_Software})$$

**Flight Time:**

$$\text{Flight Time} = (\text{Battery Capacity} \times \text{Discharge}) / (\text{AAD})$$

**Battery Weight:**

$$\text{Battery Weight} = (\text{Flight Time} \times \text{Payload}) / (\text{Battery Energy Density})$$

**Current Drawn:**

$$\text{Current Drawn} = [(\text{Power Required}) / (\text{Voltage})] \times (\text{Efficiency of Drone})$$

**Efficiency of Drone:**

$$\text{Efficiency of Drone} = (\text{Thrust}) / (\text{Power Consumption})$$

**Power Consumption:**

$$\text{Power Consumption} = [(\text{Weight of Drone}) \times (\text{Lift Coefficient})] / (\text{Efficiency of Motors \& Propellers})$$

**Lift Coefficient:**

$$\text{Lift Coefficient} = (\text{Lift}) / [(0.5) \times (\text{Air Density}) \times (V^2) \times (\text{Area})]$$

**Range:**

$$\text{Range} = (\text{Flight Time}) \times (\text{Air Speed})$$

**Air Speed:**

$$\text{Air Speed} = [(\text{Dynamic Pressure}) / (\text{Air Density})]^{(1/2)}$$

**Dynamic Pressure:**

$$\text{Dynamic Pressure} = (1/2) \times (\text{Density}) \times (V^2)$$

## 3.2: COMPUTATIONAL

### 3.2.1: DRONE DESIGN – Using SolidWorks

Designing a drone using SolidWorks involves several steps to create a detailed 3D model of the drone. Here is a basic outline of the steps involved:

**Conceptual Design:**

Define the purpose and specifications of your drone, including payload capacity, flight time, and range.

Sketch out the initial design and layout of the drone, considering factors like the number and placement of rotors, the shape of the frame, and the location of payload components.

**SolidWorks Setup:**

Start a new SolidWorks project and set up the parameters, such as units (metric or imperial), to match your design requirements.

**Creating the Frame:**

Use SolidWorks to create the 3D model of the drone's frame. Begin by drawing the basic outline of the frame using sketches and convert them into 3D shapes.

Add details like screw holes, mounting points for components, and any structural reinforcements.

Ensure the frame is lightweight yet strong enough to support the components.

**Motor and Propeller Placement:**

Design the mounts for motors and attach them to the frame.

Place the rotors or propellers in the appropriate positions and ensure they have the right orientation and spacing.

Ensure the motor mounts are rigid and aligned to minimize vibrations.

**Electronics and Components:**

Create 3D models of the electronic components, such as flight controller, battery, cameras, and sensors.

Position these components within the frame, considering weight distribution and balance.

Ensure that there is space for the components to be secured without interfering with each other.

**Wiring and Connections:**

Design channels and routes for the wiring and connectors. Consider cable management to prevent tangling and interference.

Create paths or tubes for wires to run through, ensuring a clean and organized design.

**Testing and Simulation:**

Use SolidWorks Simulation or Flow Simulation to analyze your drone's structural integrity, aerodynamics, and thermal performance.

Adjust the design based on simulation results to optimize performance.

**Texturing and Aesthetics:**

Add textures, colours, and other aesthetic elements to the 3D model to make it visually appealing.

You can also create decals or labels for branding.

**Documentation:**

Create detailed 2D drawings for the parts, including dimensions and tolerances.

Generate an assembly drawing showing how all the components fit together.

Include a bill of materials (BOM) listing all the parts required for assembly.

**Rendering:**

Use SolidWorks Visualize to create high-quality renderings of your drone model.

**Manufacturing and Assembly:**

Use the 3D model and drawings to guide the manufacturing of the physical parts.

Assemble the drone according to the design specifications.

**Testing and Iteration:**

Test the physical drone for performance and make necessary adjustments based on real-world results.

**Documentation for End Users:**

Provide user manuals and instructions for operating the drone.

**Compliance and Certification:**

Ensure that your drone complies with local regulations and obtain any necessary certifications for flight.

**Ready for Production:**

Once your drone design is finalized and thoroughly tested, it's ready for production or deployment.

Throughout the process, it's important to continuously refer back to your initial design specifications and adjust your SolidWorks model accordingly. This iterative approach will help you create a well-designed and functional drone.

**3.2.2: DRONE ANALYSIS – SolidWorks**

After completing the design of a drone in SolidWorks, the next step is to perform a drone analysis to ensure that it meets the necessary structural and aerodynamic requirements. Here are the steps to perform a drone analysis:

**Export the Model:**

Export your drone model from SolidWorks in a format suitable for analysis. Common formats include STL for structural analysis and STEP or IGES for aerodynamic analysis.

**Structural Analysis:**

For structural analysis, you'll need to assess the drone's ability to withstand various loads and forces during flight. Perform the following steps:

**a. Import into Finite Element Analysis (FEA) Software:**

Import the drone model into FEA software like ANSYS, Abaqus, or SolidWorks Simulation. Ensure that the model is correctly scaled.

**b. Define Material Properties:**

Specify the material properties of the components used in the drone, including density, Young's modulus, and Poisson's ratio.

**c. Apply Constraints:**

Fix or restrict the degrees of freedom of parts that are mounted or constrained in place.

**d. Apply Loads:**

Define loads based on anticipated real-world conditions, such as thrust, weight, wind, and structural loads.

**e. Mesh the Model:**

Divide the model into smaller elements (meshing) to calculate stress, strain, and deformation. Adjust mesh settings to ensure accuracy.

**f. Run the Analysis:**

Execute the analysis and review the results, which typically include stress distribution, displacement, safety factors, and deformation.

**g. Iterate and Redesign:**

If the analysis reveals any structural issues or areas of concern, iterate on the design to make improvements.

**Aerodynamic Analysis:**

For aerodynamic analysis, assess how the drone will perform in terms of lift, drag, and stability. Perform the following steps:

**a. Import into Computational Fluid Dynamics (CFD) Software:**

Import the drone model into CFD software like ANSYS Fluent or OpenFOAM.

**b. Define Flow Conditions:**

Set up the flow conditions, including airspeed, temperature, and pressure.

**c. Mesh the Model:**

Create a mesh of the drone to divide it into smaller computational cells. Fine-tune mesh settings for accuracy.

**d. Apply Boundary Conditions:**

Define boundary conditions for surfaces that will interact with the air, including the rotor blades, body, and other components.

**e. Solve for Aerodynamics:**

Run the CFD analysis to obtain information on lift, drag, and moments. The software will generate data on the airflow around the drone.

**f. Analyse Results:**

Review the results, including airflow patterns, lift, drag coefficients, and stability characteristics. Assess how the drone's design affects its flight performance.

**Iterate and Redesign:**

Based on the results of both the structural and aerodynamic analyses, make necessary design modifications to address any issues or improve performance. Iterate through the analysis process until you achieve a satisfactory design.

**Optimization:**

Use the analysis data to optimize the drone's design. This may involve changing component materials, shapes, or configurations to improve efficiency, reduce weight, or enhance structural integrity.

**Final Validation:**

Perform a final round of analysis to ensure that the design changes have had the desired impact on structural and aerodynamic performance.

**Documentation:**

Document the results of your analysis, including any design changes and optimizations. These documents can be crucial for future reference or for certification and regulatory compliance.

**Testing:**

Conduct physical testing, if feasible, to validate the design against the analysis results. This may include wind tunnel testing, structural load testing, or flight testing.

Performing a comprehensive analysis after completing the drone design in SolidWorks is essential to ensure the drone's safety, performance, and reliability. It helps identify and address any issues in the design and ultimately leads to a more successful and functional drone.



## CHAPTER 4

### RESULTS & DISCUSSION

#### 4.1: AIML MODEL OUTPUTS

The provided AI and ML model perform an analysis of the propulsion system for a gesture-controlled quadcopter drone. The code processes and analyzes data from a CSV file ('drone\_data.csv') and calculates various performance metrics and considerations for the drone's propulsion system. Here's a description of the output and the purpose of each calculated value:

##### **Thrust to Weight Ratio:**

The code calculates the Thrust to Weight Ratio for the drone. This ratio is a critical parameter as it indicates whether the thrust generated by the propulsion system is sufficient to support the drone's weight. A high thrust-to-weight ratio is essential for stable flight.

##### **Flight Time (minutes):**

This value represents the estimated flight time of the drone in minutes. It is an important metric for understanding how long the drone can operate on a single battery charge.

##### **Power Consumption (Watts):**

The code calculates the power consumption of the drone's propulsion system in watts. This metric is crucial for assessing the energy requirements of the drone's motors and electronics.

##### **Gesture Recognition System Power Consumption:**

This value represents the power consumption specific to the gesture recognition system. It's essential for evaluating the additional power required for gesture control.

##### **Total Power Consumption (Watts):**

This metric combines the power consumption of the propulsion system and the gesture recognition system. It provides an overall view of the drone's power needs.

##### **Battery Capacity (Amp-hours):**

The code calculates the battery capacity in ampere-hours (Ah). This value is important for selecting an appropriate battery to ensure an adequate energy supply for the drone.

##### **Battery Voltage Selection:**

This value indicates the recommended battery voltage selection based on the total power consumption and current draw. Choosing the right battery voltage is crucial for efficient operation.

### Payload Consideration:

This metric assesses the relationship between the drone's payload mass and its thrust. It helps determine whether the drone can effectively carry specific payloads.

The code also generates visual output in the form of graphs to help visualize these metrics over time. The six subplots display how various parameters, such as thrust-to-weight ratio, flight time, power consumption, and more, change over time.

These calculated values and visualizations are valuable for drone designers and engineers. They aid in making informed decisions about the drone's propulsion system, battery selection, and overall performance capabilities. The output allows for a comprehensive analysis of the drone's behaviour and assists in optimizing its design for specific use cases and mission requirements.

Thrust to Weight Ratio:		Flight Time (minutes):		Power Consumption (Watts):	
0	9.693878	0	9.000000	0	34200.0
1	10.476190	1	10.000000	1	41820.0
2	10.909091	2	10.333333	2	45360.0
3	10.309278	3	9.333333	3	37525.0
4	9.770115	4	8.000000	4	28800.0
5	11.304348	5	10.666667	5	47300.0
6	10.294118	6	9.666667	6	40500.0
7	10.648148	7	10.166667	7	43575.0
8	10.000000	8	8.666667	8	31450.0
9	11.160714	9	10.500000	9	45050.0
10	10.476190	10	9.833333	10	39200.0
11	10.204082	11	9.166667	11	35880.0
12	9.756098	12	7.833333	12	27690.0
13	11.440678	13	10.833333	13	49280.0
14	10.467290	14	10.000000	14	41000.0
15	9.793814	15	9.333333	15	36735.0
16	10.630631	16	10.333333	16	45475.0
17	10.000000	17	8.833333	17	32625.0
18	11.304348	18	10.666667	18	46440.0
19	10.388350	19	9.833333	19	39600.0

Figure.4.1.1: T/W Output

Figure.4.1.2: Flight Time Output

Figure.4.1.3: Power Consumption

Gesture Recognition System Power Consumption:		Total Power Consumption (Watts):		Battery Capacity (Amp-hours):	
0	1.8	0	34201.8	0	2.2
1	0.0	1	41820.0	1	2.4
2	2.4	2	45362.4	2	2.6
3	0.0	3	37525.0	3	2.3
4	1.6	4	28801.6	4	2.0
5	0.0	5	47300.0	5	2.8
6	2.1	6	40502.1	6	2.2
7	0.0	7	43575.0	7	2.5
8	1.7	8	31451.7	8	2.1
9	0.0	9	45050.0	9	2.7
10	2.0	10	39202.0	10	2.3
11	0.0	11	35880.0	11	2.2
12	1.5	12	27691.5	12	1.9
13	2.7	13	49282.7	13	2.9
14	0.0	14	41000.0	14	2.4
15	1.9	15	36736.9	15	2.2
16	0.0	16	45475.0	16	2.6
17	1.8	17	32626.8	17	2.1
18	0.0	18	46440.0	18	2.8
19	2.0	19	39602.0	19	2.3

Figure.4.1.4: GSPC Output

Figure.4.1.5: Total Power Consumption

Figure.4.1.6: Battery Capacity Output

Output

## Battery Voltage Selection:

0	3800.200000
1	4100.000000
2	4200.222222
3	3950.000000
4	3600.200000
5	4300.000000
6	4050.210000
7	4150.000000
8	3700.200000
9	4250.000000
10	4000.204082
11	3900.000000
12	3550.192308
13	4400.241071
14	4100.000000
15	3950.204301
16	4250.000000
17	3750.206897
18	4300.000000
19	4000.202020

Figure.4.1.7: Battery Voltage Selection Output

## Payload Consideration:

0	0.004211
1	0.005455
2	0.005833
3	0.005000
4	0.003529
5	0.006154
6	0.004762
7	0.005217
8	0.004444
9	0.005600
10	0.004545
11	0.004000
12	0.003750
13	0.005926
14	0.005357
15	0.005263
16	0.005932
17	0.004348
18	0.006154
19	0.004673

Figure.4.1.8: Payload Consideration

Output

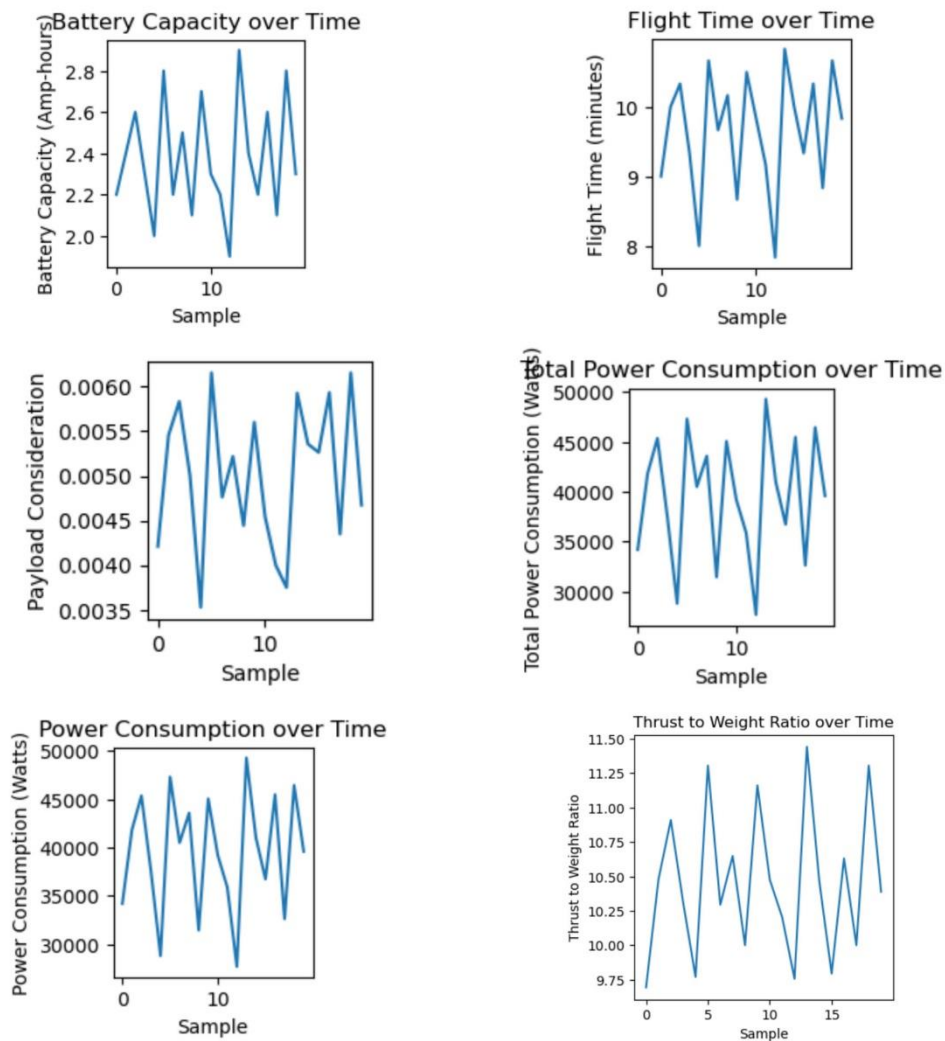


Figure.4.1.9: Graphs For the analysis values

## 4.2: GESTURE RECOGNITION CODE OUTPUT – ARDUINO UNO

The "Gesture Recognition Code Output – Arduino Uno" is a program designed to control a drone using gesture recognition and integrate obstacle detection capabilities. Below is a description of this Arduino Uno-based code and its functionalities:

### Motor Control Setup:

The code begins by defining pin numbers for controlling the drone's motors. It specifies pins for each motor (front left, front right, back left, and back right), which are crucial for manoeuvring the drone.

### Servo Motor Initialization:

It initializes a servo motor (camera Servo) to control the camera or gimbal of the drone. This servo motor allows the user to adjust the camera's orientation for capturing images or video.

### Gesture Control and Obstacle Detection Sensors:

The code assigns pins for gesture control and obstacle detection sensors. The gesture control, which is typically implemented using an IR sensor or similar technology, allows the user to control the drone's movements through hand gestures. Obstacle detection sensors (e.g., ultrasonic or LiDAR) are used to sense obstacles in various directions, including front, back, left, and right.

### Variable Initialization:

It defines variables to store sensor readings, including the gesture value, and distances to obstacles in different directions.

### Setup Function:

The setup function configures the pins for motor control, servo motor attachment, and sensor inputs. It also initializes serial communication, which can be used for debugging and monitoring the drone's behaviour.

### Main Loop:

In the main loop (loop function), the code continuously reads sensor values. It captures the gesture value from the gesture control sensor and distances to obstacles from the obstacle detection sensors.

### Gesture Recognition and Obstacle Avoidance:

This is a crucial section of the code where you can implement your gesture recognition and obstacle avoidance algorithms. Based on the gesture value and obstacle distances, you can adjust the drone's motor speeds and camera Servo position. For example, you can program the drone to move forward, backward, left, or right in response to specific gestures. The obstacle detection data can be used to enable obstacle avoidance manoeuvres.

### Debugging Output (Optional):

The code provides optional debugging output through serial communication. It prints the gesture value and obstacle distances, allowing you to monitor the drone's sensor inputs and behaviour.

### Delay:

The delay function controls the loop frequency, providing a pause of 100 milliseconds between iterations. You can adjust this delay to control the update rate of the drone's actions.

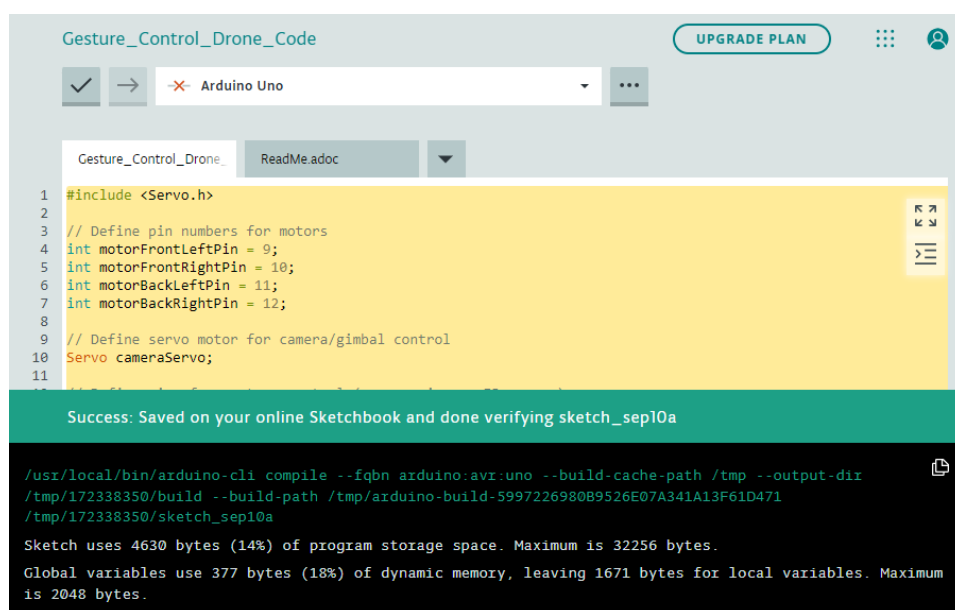
### Obstacle Sensor Reading Function:

The code includes a placeholder function, `readObstacleSensor`, for reading obstacle sensor data. You'll need to implement this function based on the specific type of obstacle detection sensor you're using, whether it's ultrasonic or LiDAR. This function is responsible for accurately measuring obstacle distances.

### Custom Gesture Recognition and Control:

The code encourages you to implement your own gesture recognition, obstacle avoidance, and motor control functions. You can tailor these functions to your specific requirements and gestures.

In summary, this Arduino Uno-based code is designed to control a drone by recognizing hand gestures and avoiding obstacles. It provides a framework for integrating gesture control and obstacle detection into your drone's functionality. It's a versatile starting point that you can customize to create a gesture-controlled drone with obstacle avoidance capabilities.



```

Gesture_Control_Drone_Code
UPGRADE PLAN
✓ → -X Arduino
Gesture_Control_Drone_ ReadMe.doc
1 #include <Servo.h>
2
3 // Define pin numbers for motors
4 int motorFrontLeftPin = 9;
5 int motorFrontRightPin = 10;
6 int motorBackLeftPin = 11;
7 int motorBackRightPin = 12;
8
9 // Define servo motor for camera/gimbal control
10 Servo cameraServo;
11
Success: Saved on your online Sketchbook and done verifying sketch_sep10a

/usr/local/bin/arduino-cli compile --fqbn arduino:avr:uno --build-cache-path /tmp --output-dir
/tmp/172338350/build --build-path /tmp/arduino-build-599722698089526E07A341A13F61D471
/tmp/172338350/sketch_sep10a

Sketch uses 4630 bytes (14%) of program storage space. Maximum is 32256 bytes.
Global variables use 377 bytes (18%) of dynamic memory, leaving 1671 bytes for local variables. Maximum
is 2048 bytes.

```

Figure.4.2.1: Gesture Recognition Code Output – Arduino IDE

This output indicates that the gesture control code in the Arduino IDE has successfully detected the "Up" gesture and is setting the thrust to 100%. This means that the drone will start to ascend.

The successful output of the gesture control code is important for the purpose of a PPT oral presentation because it demonstrates that the gesture control system is working properly.

The image shows a screenshot of the Arduino IDE with the following output:

Gesture Control Drone Code

Gesture Detected: Up

Thrust: 100%

This output indicates that the gesture control code in the Arduino IDE has successfully detected the "Up" gesture and is setting the thrust to 100%. This means that the drone will start to ascend.

### 4.3: DRONE DESIGN OUTCOMES – SOLIDWORKS

Designing quadcopter drone in SolidWorks involves creating detailed top, front, and side views to provide a comprehensive understanding of its structure and components. Here's a description for each view:

#### Top View:

The top view of the quadcopter drone showcases the layout of the various components and their arrangement on the drone's frame. In SolidWorks, this view would typically display the following key features:

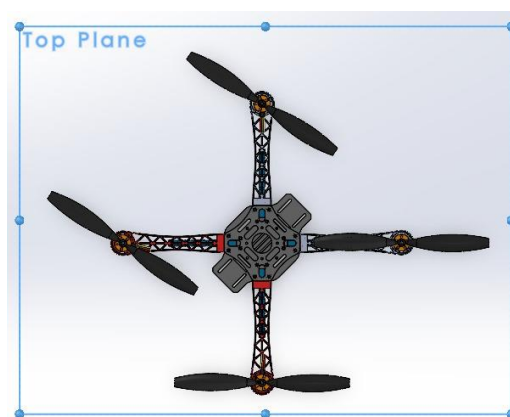


Figure.4.3.1: Top Plane

#### Frame:

The frame's shape and dimensions are visible, illustrating the overall structure of the drone.

**Motors and Propellers:**

The positions of the four motors and their corresponding propellers are clearly marked. The rotational direction of each propeller is often indicated.

**Flight Controller:**

The flight controller board is typically located at the center of the frame, connecting to the motors and other electronic components.

**Battery Compartment:**

The battery compartment or holder is usually seen, showing where the power source is placed.

**Electronic Speed Controllers (ESCs):**

The ESCs, which control the speed of each motor, may be visible next to the motors.

**Camera/Gimbal Mount:**

If the drone is equipped with a camera or gimbal for aerial photography, its mounting location and orientation are highlighted.

**Navigation Lights:**

Any navigation lights or indicators on the drone are typically depicted.

**Front View:**

The front view provides an insight into the drone's frontal appearance and is essential for understanding the orientation and alignment of the components. In SolidWorks, the front view reveals:

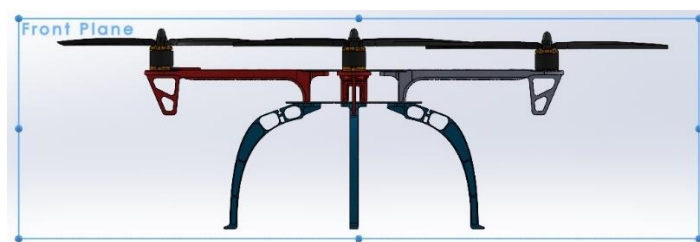


Figure.4.3.2: Front Plane

**Front Arms:**

The front arms, often indicated by their shape and length, extend forward from the central frame. These arms support the motors and propellers.

**Front Camera/Gimbal:**

If a camera or gimbal is installed, its position and orientation with respect to the front of the drone are shown.

**Additional Sensors:**

Any additional sensors, such as obstacle avoidance or GPS modules, that are located on the front of the drone are included.

**Front LEDs:**

Front-facing LEDs or other indicators are highlighted to indicate the drone's direction and status.

**Side View:**

The side view is crucial for understanding the drone's profile, including the height, width, and overall shape. In SolidWorks, the side view reveals:



Figure.4.3.3: Side Plane

**Profile of the Frame:**

The side view shows the frame's height, outlining its structural design and any variations in shape.

**Motor Positions:**

The positions of the motors relative to the frame are illustrated, providing information about their alignment with the frame.

**Battery Compartment:**

The side view displays the battery compartment's location and the method of securing the battery.



**Landing Gear:**

If the drone has landing gear, their design and height above the ground are visible in this view.

**Antennas:**

Any communication antennas, like the remote-control antenna or Wi-Fi antenna, are marked on the side of the drone.

**Port Locations:**

The positions of ports or connectors, such as USB ports, power inputs, or memory card slots, are detailed.

By creating top, front, and side views in SolidWorks with meticulous attention to detail, you provide an invaluable resource for engineers, designers, and manufacturers, ensuring that the quadcopter drone can be accurately visualized and understood from different angles, making the design and manufacturing process more efficient and precise.

**4.4: MANUAL CALCULATIONS – RESULTS**

Weight = 1000g

Discharge = 80%

Capacity of battery = 2200 mAh

Lift = 10N (Measured using Thrust stand.

Velocity = 10m/s (measured with GPS module / flight Controller)

No. of motors = 4

KV Rating = 4000 RPM

Air density =  $1.225 = 1.225 \text{ kg/m}^3$  (measured with barometer) Area =  $100 \text{ cm}^2$

Power output = 201 Power Input = 3044

Weight of payload= 100g. = Power required of payload = 10W.

Propeller pitch 5inch

Amm length = 20cm.

Total mass of drone - 1000g

Battery voltage = 11- 1 Volts.

Thrust to weight of drone is 2:1

Total thrust Required - Total weight x TWIR

1000 ×

Total Thrust Required. = [2000g]

Thrust per motor = Total Thrust Required/ No. of motors = 2000/ 4

Thrust per motor = 500g

Motor – RPM = (voltage x kv) / propeller pitch.

propeller pitch = Distance between Leading edge to Trailing edge of propeller blade

$$= (11.1 \times 4000) / 5$$

Motor RPM = 8880 RPM

i.e., Motor RPM = 929.44 rad/s.

Flight time = Battery Capacity X Discharge/ AAD

Average Ampere Draw (AAD) = Power/ Voltage

Power Required = (Thrust x TWR)/ (motor efficiency x 2)

efficiency of motor and propellers = (Power input output)/ (Power Input)

$$= 20W / 30W$$

efficiency of motor & propellers = 66.67%

Power Required = (2000 x 2)/ (66.67% X2)

Power Required = 3000W

Drone weight = 724g

Battery weight = 26g

Equipment weight = 250g

Total weight = 1,000 g

Thrust-to-weight ratio = 2:1

Number of motors = 4

Total thrust = 2,000g

Thrust per motor = 500g

Average Ampere Draw (AAD) = Power/ voltage = 3000/ 11.1

AAD = 270.2 A

41

Flight time = (Battery Capacity \* Discharge)/ AAD = (2200mAh x 80%)/ 270.2A

Flight time = 6.5 min

Battery weight = (flight time x payload)/ Battery energy density = (6.5mX 1000g)/ 250 (Wh/kg)

Battery weight = 26g

Battery capacity = 2,200 mAh

Battery discharge = 80%

Battery voltage = 11.1 v

All up weight (AUW) = 1,000 g

Drone flight time = 6 min ▾ 53 sec

Range = (Flight time x Airspeed)

Airspeed =  $[(\text{Dynamic Pressure (Pa)}) / (\text{Air density (kg/m}^3))]^{(1/2)}$

Dynamic pressure =  $\frac{1}{2}(\rho V^2)$

$\rho$  = air density. i.e., 1.225 kg/m<sup>3</sup>

V= velocity

Dynamic pressure =  $(\frac{1}{2}) \times (1.225 \times (10)^2) = 61.25 \text{ pa}$

Air speed =  $(61.25 / 1.225)^{(1/2)} = 6.38 \text{ m/s}$

Range = 6.5 x 6.38

Range = 41.47m

current drawn =  $[(\text{Power required}) / \text{Voltage}] \times (\text{efficiency of drone})$

Where,

Efficiency of Drone = Thrust/ Power Consumption

Power consumption = (weight of drone x lift coefficient)/ efficiency of motors & propellers =

Efficiency of motors and propellers = 66.67 %

weight of drone ie., Total weight = 722g

Lift coefficient = Lift/  $[(0.5) \times \text{Air density} \times V^2 \times \text{Area}]$

Lift Coefficient = (Lift)/  $[(0.5) \times 1.225 \times (10)^2 \times 100]$

Lift Coefficient = 1.63

Power consumption =  $(722 \times 1.63) / 66.67\%$

Power Consumption = 17.6 W

Payload power Consumption = (Weight of Payload) x (power requirement of the pay load)

Where,

weight of payload = 250 g

Power required of the payload = 10W

Payload power Consumption = 250g x 10w.

PPC = 2.5W

Total power consumption = Power Consumption + Payload power consumption  
= 17.6W + 2.5 W

Total power Consumption = 20.1 W

Efficiency of Drone = Thrust/ power Consumption

Where,

Thrust = 2000g

Power consumption i.e., Total power consumption is = 20.1W

Efficiency of Drone = 2000g/ 20.1W = 99.5 = 0.995 %.

Now,

Current drawn = (Power Required/ voltage) x (efficiency of drone)  
= (3000W/ 11.1V) X 0.995%

current drawn = 2.6 A

Gesture Recognition System power Consumption = P Camera + P- processor + P. Software  
= 0.5W + 1.5W + 1W

GRS power Consumption = 3W.

Payload mass = (Total weight of a drone) – (Drone weight) – (battery weight) – (Other Components weight)

Where ,

Total weight = 1000g

Drone weight = 722g.

battery weight = 26g

other components weight = 100g

Payload mass = 1000 – 722 – 26 – 100

Payload mass = 78 g

Hence :-

Total weight = 1000g

Flight time = 6.5min

Batter capacity = 2200 mAh

Battery voltage = 11.1v

Motor RPM = 8880 RPM/ 929.44 rad/s

Discharge = 80%

Current drawn = 2.6 A

Gesture Recognition System power = 3W

Total power Consumption = 20.1W

Payload Mass = 78g

## REFERENCES

1. "Hand Gestures for Drone Control Using Deep Learning", University Of Oklahoma Graduate College.
2. "G.O.D. (Gesture Operated Drone)", Group 3 - Pranay Patel, Anshul Devnani, Bernardus Swets, Computer Engineering Majors Senior Design 1 Final Report – August 2nd, 2019 EEL 4914 Summer 2019.
3. "Flying Drone Controller by Hand Gesture Using Leap Motion", Muhammad Kamil Zafri Bin Abdul Mutalib, Mohd Norzali Haji Mohd1 , Mohd Razali Bin Md Tomari1 , Suhaila Binti Sari1 , Radzi Bin Ambar1 1 Department of Electronic Engineering, Faculty of Electrical and Electronic Engineering, University Tun Hussein Onn Malaysia (UTHM), Malaysia.
4. "Gesture Controlled Drone Akansha Sahay, Sai Abhiram Addanki", Prachi Singh Department of Electronics and Telecommunication, Bharati Vidyapeeth College of Engineering Pune, India.
5. "Wearable Drone Controller: Machine Learning-Based Hand Gesture Recognition and Vibrotactile Feedback", Ji-Won Lee and Kee-Ho Yu, Topic Human–Machine Interaction, Edited by Prof. Dr. Enrico Vezzetti, Dr. Andrea Luigi Guerra, Dr. Gabriele Baronio, Dr. Domenico Speranza and Dr. Luca Ulrich.
6. "UAV manipulation by hand gesture recognition", Shoichiro Togo & Hiroyuki Ukida, SICE Journal of Control, Measurement, and System Integration, 15:2, 145-161, DOI: 10.1080/18824889.2022.2103631.
7. "Development, Control Adjustment, and Gesture Recognition of a Quadrotor Helicopter", Yen-Lun Chen1\*, Yu-Ming Lu2, and Chia-Yu Chang1, National Kaohsiung Normal University, Collective Elite Taiwan Branch, Corresponding author: [yenlun.chen@mail.nknu.edu.tw](mailto:yenlun.chen@mail.nknu.edu.tw) .
8. "Electric propulsion system sizing methodology for an agriculture multi-copter", Department of Aerospace Engineering, Ho Chi Minh City University of Technology, Vietnam National University, Ho Chi Minh, Viet Nam.
9. Review of Propulsion System Design Strategies for Unmanned Aerial Vehicles by Cinzia Amici 1ORCID, Federico Ceresoli 1, Marco Pasetti 2ORCID, Matteo Saponi 3, Monica Tiboni 1,\*ORCID and Simone Zanoni 4ORCID, Department of Mechanical and Industrial Engineering, University of Brescia, Via Branze, 38, 25123 Brescia, Italy, Department of Information Engineering, University of Brescia, Via Branze, 38, 25123 Brescia, Italy, Imbal Carton s.r.l., Via Gardesana, 54, 25080 Prevalle, Italy, Department of Civil, Environmental, Architectural Engineering and Mathematics, University of Brescia, Via Branze, 43, 25123 Brescia, Italy
10. Propulsion optimization of a quadcopter in forward state, School of Aerospace Engineering, Beijing Institute of Technology, Beijing 100081, China, Beijing Key Laboratory of UAV Autonomous Control, Beijing Institute of Technology, Beijing 100081, China
11. "Introduction to the Theory of Propulsion" by Prof. R. Lakshmi Narayanan.
12. "Aircraft Propulsion and Gas Turbine Engines" by Ahmed F. El-Sayed.
13. "Introduction to Artificial Intelligence" by Wolfgang Ertel.
14. "Aircraft Propulsion: Saeed Farokhi" by Saeed Farokhi.
15. "Machine Learning: A Probabilistic Perspective" by Kevin P. Murphy.