AeroTHON 2024

UNCREWED AIRCRAFT SYSTEM (UAS) DESIGN, BUILD AND FLY CONTEST

TEAM GARUDA || ID - AT2024061

Phase1: Design Report and Presentation

INSTITUTE OF ENGINEERING & MANAGEMENT, KOLKATA, WEST BENGAL

INSTITUTE OF ENGINEERING & MANAGEMENT
Good Education, Good Jobs

Under the sincere guidance of

Dr. Rahul Baidya

Faculty Coordinator

Department of Mechanical Engineering

STATEMENT OF COMPLIANCE

Certification of Qualification

Team Name: TEAM GARUDA

University/Institute: INSTITUTE OF ENGINEERING AND MANAGEMENT

Faculty Advisor: Dr. RAHUL BAIDYA

Faculty Advisor's Email: rahul.baidya@iem.edu.in

Statement of Compliance

As Faculty Advisor, I certify that the registered team members are enrolled in collegiate courses. This team has designed the UAS for the SAE AEROTHON 2024 contest, without direct assistance from professional engineers, R/C model experts or pilots, or related professionals.

Signature of Faculty Advisor

07.06.2024
Date

Team Captain Information:

Team Captain's Name: NIRBAN ROY

Team Captain's E-mail: nirbanroy04@gmail.com

Team Captain's Phone: 8910518175

Table of Contents

1.INTRODUCTION

The report discusses the detailed design of out drone **"JASIG".** It covers the design, analysis, and manufacturing processes of a quadcopter drone featuring one pair of clockwise and counter-clockwise propellers and an innovative dropping mechanism. This drone can carry and deliver a payload of 200 grams. The center of gravity has been precisely calculated to ensure the drone's stability and efficiency. Additionally, it is equipped with a live camera capable of 300-degree rotation, providing continuous flight footage. The drone can operate both manually and autonomously.

Figure 1: JASIG

1.1 Objective

The objective behind designing this drone is package delivery, both in manual and autonomous mode. Its purpose is to lift a load from one particular point, carry it across all middle way obstacles and successfully delivering it to the destination point. The payload that it will deliver is 200 g.

1.2 Problem Statement and Requirements

The problem statement suggests to design, build and fly a multirotor UAV that can deliver a specified payload to a target area by manual as well as autonomous operations. The payload given will be of following dimensions:

Table 1: Parameters and their requirements

S. No.	Parameter	Requirement/Limitation
1.	UAS Type	Multirotor
2.	UAS Category	Micro UAS (i.e., Take-off weight < 2kg)
3.	Payload Capacity	200 Grams
4.	Propulsion Type	Electric
5.	Communication System Range	At least 1 km

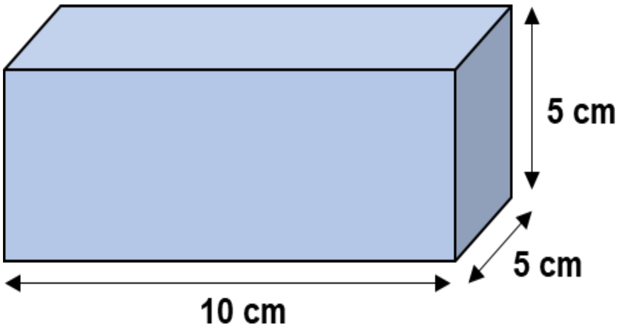


Figure 2: Payload Dimensions

2. TECHNICAL CONTENTS

2.1 Conceptual Design

Our drone consists of a central hub to which 4 rotor arms are attached. We have 4 motor mounts which are attached at the ends of the rotor arms. We are using one pair of 12-inch clockwise propellers and another pair of 12-inch anti-clockwise propeller. Below each rotor arm, we have a landing gear attached. Below the central hub, we have a payload dropping mechanism and a fixed camera.
We have electronic parts in their respective locations as mentioned below.

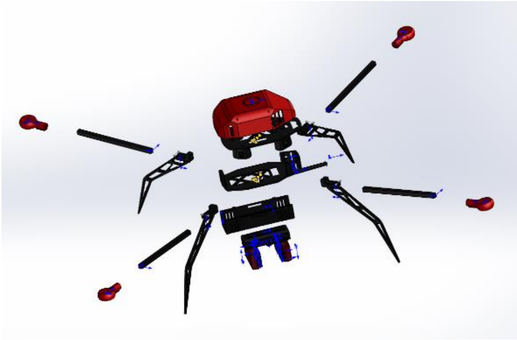


Figure 3: Exploded view of the UAV

Table 2: Components and their arrangements

Physical Components	Arrangements
Hub	At the center of the drone
Rotor arms	Attached to the central hub symmetrically
Motor mounts	Attached at the end of the rotor arms
Propellers	Attached to the motors
Landing Gears	Attached below the rotor arms
Payload Dropping Mechanism	Attached below the hub

Table 3: Components and their locations

Electronic Components	Locations
Brushless DC Motor	Inside the motor mounts
Electronic Speed Controllers	Inside the rotor arms
Flight Controller	Inside the hub
Raspberry Pi	Inside the hub
Sensors	Inside the Hub
Battery	Inside the Hub
Receiver	Inside the Hub
Camera	Attached to the body

2.2 Detailed Design - Preliminary Weight Estimation

Estimation of preliminary weight = drone mass + payload mass

= 1800 gm + 200 gm
= 2000 gm

Here are the weights of the individual components: -

Table 4: Weights of components

Components	No. of Units	Total Weight (Wt × Units) (in g)
Motor	4	272
Electronics Speed Controller (ESC)	4	144
Servo Motor	1	13.4
FPV Camera	1	9.5
On board camera (Arducam)	1	13
Propeller	4	70
Carbon Fiber Rod	4	80
Payload Box	1	236
Payload	1	200
Pixhawk Hex	1	73
Raspberry Pi	1	60
Video Transmitter	1	15.4
Radio Receiver	1	16.6
Telemetry	1	16.5
GPS	1	33
Battery	1	645
Total		1897.4

2.3 Thrust required Estimation

With Payload:

We are keeping, Thrust : weight ratio = 3:1

Thus, thrust required to uplift the drone = 3 x 2000 gm = 6000 gm = 6 kg
As we are making a quadcopter,
per motor thrust = mass(uplift) (kg)/no. of motors x 9.8 N
Thus, Thrust = $\frac{6}{4} \times 9.8$ N = 1.5 x 9.8 N = 14.7 N/motor

Without payload:

Calculated weight of the drone
= 1600 + 200 gm = 1800 gm
We are keeping, Thrust : weight ratio = 3:1
Thus, thrust required to uplift the drone = 3 x 1800 gm = 5400 gm = 5.4 kg

Thrust for 1 motor = $\frac{5.4}{4} \times 9.8$ N = 13.23 N/motor

2.4 Selection of Propulsion System

The most important task in the phase of making a design of an UAV is to properly select its propulsion system.
We have chosen following models in accordance to our calculations to match the flight time of the mission of 10 mins.

➢ Brushless DC Motor:

Model	Required ESC	Required Lipo Batteries	Maximum Thrust	Reason	Image
SunnySky V4006 740KV	40A	4S	1667g	The chosen motor provides more than the required thrust	

➢ ESC:

Model	Current	Compatible Battery	Reason	Image
Hobbywing SkyWalker	40A	3S – 4S	The chosen ESCs are compatible with the chosen motors	

➢ Battery:

Model	Capacity and Discharge rate	Voltage	Reason	Image
Gaoneng Lipo Battery	8000 mAh 150C	14.8V	The given battery is sufficient to power up all the components	

➢ Propellers:

Model	Dimension	Reason	Image
SunnySky EOLO	12×5 inch	This pair of propellers provide the required thrust	

2.5 UAV Sizing

Table 5: Components and their size

Component	Size (mm)
Wheelbase	566
Rotor Arm	214.5
Motor Mount (diameter)	21.62
Central Hub	198×128×89
Propeller Clearance	38.1
Landing Gear	200
Payload Dropping Box	114×88×85

2.6 UAS Performance

➢ **Power Required Estimation:**

$$(thrust)^3 = \frac{\pi}{2} \times (diameter)^2 \times \rho_{air} \times (mechanical\ output\ power)^2$$

$$(Mechanical\ output\ power)^2 = \frac{(thrust)^3}{\frac{\pi}{2}} \times d^2 \times \rho_{air}$$

Mechanical power output= $\left(\frac{(11.76)^3}{\frac{\pi}{2}} \times (0.3048)^2 \times 1.164\right)^{\frac{1}{2}} = 97.874\ W$

➢ **Current Drawn Calculation**

Input power by 1 motor (from datasheet) = 297.48 W

For 4 motors, total input power = $4 \times Power_{input}$

= 4 × 297.48 W = 1189.92 W

Voltage provided by the battery (according to it's specification) = 14.8 V

Power = $Voltage_{(supplied\ by\ 1\ battery)} \times current$

Current = power (input by 1 motor)/voltage = 297.48/14.8 A = 20.1 A

➢ **Kv calculation**

Thrust = 11.76 N/motor

ρ_{air} = 1.164 kg/m^3

Diameter of the propeller, D = 12 inches = 0.3048 m

pitch = 5 inches = 0.127 m

speed in rps = $\sqrt{\frac{thrust}{0.5 \times \rho_{air} \times \pi \times \sqrt{diameter} \times pitch^2}}$

$$= \sqrt{\frac{11.76}{0.5 \times 1.164 \times \pi \times \sqrt{0.3048} \times 0.127^2}}$$

$$= 131.074\ rps$$

speed in rpm = $rps \times \frac{60}{2} \times \pi = 131.074 \times \frac{60}{2} \times \pi = 1252.30\ rpm$

➢ **Reaction Force**

Reaction force = Total Weight = 1.6 × 9.8 N

Using Bernoulli's form :-
Bending Moment,
B.M = Reaction force × length of arm
BM = 1.6 × 9.8 × 0.2145 = 3.36 N-m

Length of drone arm = 0.2145 m
Modulus of elasticity of material = 33 msi (228 GPa)
Ultimate tensile strength (UTS) = 500 Ksi (3.5 GPa)
Outer Diameter(OD) = 10 mm
Inner Diameter(ID) = 8 mm

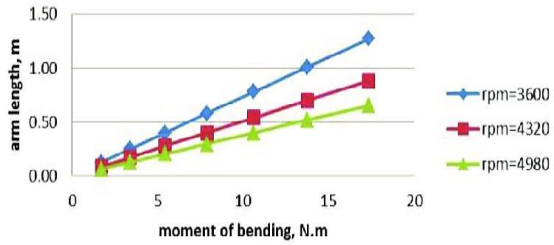


Figure 4: Rotor arm length vs bending moment

Moment of Inertia (I) :-

Ixx $= \frac{10\times8^3-8\times10^3}{12} = 240\ mm^4$

Iyy $= \frac{10\times8^3-8\times10^3}{12} = 240\ mm^4$

➢ **Battery & Endurance Calculations**

Battery:
Capacity = 6200mAH = 6.2AH
Discharge Rating = 90C

Endurance Estimation:
We are considering a maximum discharge limit of 90% for our battery. Therefore, the maximum usable capacity of the battery = 6.2 x 0.9 = 5.58AH

Case 1: Flight time with minimum thrust(hovering)
Weight of Drone = 2kg x 9.81 m/s^2 = 19.62 kg-f
Therefore, for hovering minimum thrust required per motor = 19.62/4
= 4.905 kg-f
= 490.5 gf
≈ 500 gf
Current Draw per motor at 500 gf thrust = 3.6 A (from Manufacturer Datasheet)
Total Current Draw by 4 motors = 3.6 x 4 = 14.4 A
Considering ESC efficiency 80%
Current Draw by Propulsion System = 14.4/0.8 = 18 A
Therefore, Flight time = 5.58/18 = 18.6 min
Flight Time = 18.6 min (with minimum thrust for sustaining hover)

Case 2: Flight Time with Peak Performance:
Peak Thrust per motor = 1667 gf
Peak Current per motor = 20.1 A
Net Peak Current by 4 motors = 20.1 A x 4 = 80.4 A
Considering ESC efficiency = 80%
Current draw by propulsion system = 80.4/0.8 = 100.5 A
Therefore, Flight time = 5.58/100.5 = 3.33 min
Flight Time = 3.33 min (at peak performance)

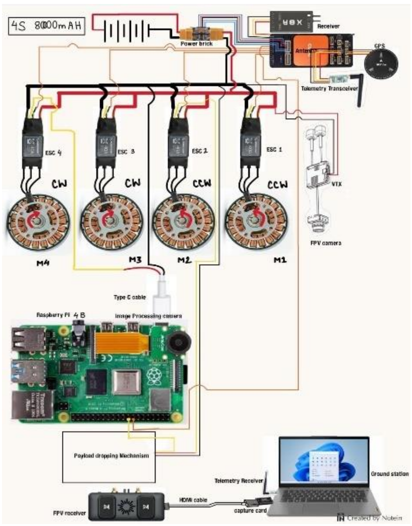


Figure 5: Circuit diagram

2.7 Material Selection

Rotor Arms

Table 6: Comparison between materials for rotor arms

Material	Density (g/cm^3)	Strength (MPa)	Durability	Machinability	Cost
Carbon Fibre	1.7 - 1.8	3000 - 5000	Extremely High	Moderate-Difficult	Moderate-High
Aluminium Alloy	2.7 - 2.8	70 - 600	High	Easy	Moderate
ABS	1.0 - 1.4	40 - 60	Moderate	Easy	Very low
Glass Fibre	1.5 - 2.0	2000 - 3500	Very High	Moderate	Moderate-High
PETG (Polyethylene Terephthalate Glycol)	1.2 - 1.4	40 - 80	Moderate	Easy	Low
PLA+ (Polylactic Acid Plus)	1.2 - 1.4	55 - 75	Moderate	Easy	Low

After considering all the factors, carbon fibre was selected over others for its Strength-to-Weight Ratio, Stiffness ,Durability, Impact Resistance and Design Flexibilty for high efficiency and performance.

Frame and Central Hub

Table 7: Comparison between materials for main frame

Material	Strength (MPa)	Durability	Density	Flexibility	Manufacturing Complexity	Water Resistance
CFRP	3000-5000	Very High	1.5 –1.6	Low	High	Variable
LW PLA	30 - 60	Low	1.2-1.3	Moderate	Low	Low to Moderate
Aluminium	20-120	High	2.7	Low	Moderate	High
ABS	40 - 60	Moderate	1.0 –1.2	High	Low	Moderate
PLA PRO+	55 - 85	Low	1.2 - 1.3	Moderate	Low	Low to Moderate
PETG	40 - 80	Moderate	1.2 - 1.4	High	Moderate	High

PLA PRO+ is chosen for making the frame and central hub of the drone due to its lightweight yet durable nature, crucial for light efficiency.

Landing Gear

Table 8: Comparison between materials for landing gear

Material	Density (g/cm³)	Strength	Weight	Durability	Flexibility	Manufacturing Complexity
CF-ABS	1.5-1.8	Moderate-high	Moderate	Highest	Extremely high	Highest
Aluminium	2.7	High	Moderate	Moderate-High	Low	Moderate
ABS	1.0 - 1.4	Moderate	Lightest	Moderate	Moderate	Lowest

CF-ABS (Carbon Fibre Reinforced ABS) is chosen for landing gear due to its superior strength-to-weight ratio, high impact resistance, and excellent fatigue durability.

Propeller

Table 9: Comparison between materials of propeller

Material	Density (g/cm³)	Strength (MPa)	Stiffness (GPa)	Weight	Water Resistance	Durability
Polypropylene	0.9	10-50	0.1-0.2	Lightest	Very High	Low
ABS	1.0-1.4	50-70	2-3	Light	Moderate	Moderate
Fibreglass	1.5-2.0	200-600	10-60	Heaviest	High	Moderate
CFRN	1.1-1.4	100-350	150-250	Moderate	High	Very High

CF-Nylon (Carbon Fibre Reinforced Nylon) is preferred for drone propellers due to its lightweight yet strong carbon fibre reinforcement, ensuring durability under high stress.

Motor Mount

Table 10: Comparison between materials for motor mount

Material	Density (g/cm³)	Strength (in MPa)	Stiffness (in GPa)	Durability	Heat Resistance	Manufacturing Complexity	Cost
Carbon fibre	1.7 – 1.8	250-650	200-700	Very High	Excellent	High	High
PLA	1.2 – 1.3	45 - 85	2.7 – 4.0	Moderate	Moderate	Moderate	Moderate
ABS Pro	1.0 – 1.4	60-90	1.9-2.6	High	High	Moderate-High	Moderate

ABS Pro is chosen for motor mounts due to its properties like cost-effectiveness, ease of manufacturing using injection moulding ,its balance of toughness, impact resistance and temperature stability.

2.8 Subsystem Selection

SENSORS

➢ GPS

Name	Reason	Image
CUAV NEO 3 Pro GPS Module | GNSSU-BLOX M9N CAN BUS	Compatible with Hex Pixhawk cube	

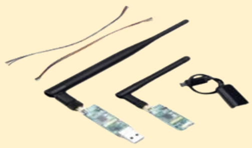
➢ Receiver and Transmitter

Name	Reason	Image
VTX (TRANSMITTER): WALKSNAIL HD PRO VTX	digital hd 1080p video transmission at avg 22ms latency	
VRX (RECEIVER): WALKSNAIL AVATAR VRX (comes along with camera kit)	reliable, hd video reception, ensuring real-time monitoring of the FPV feed	
WALKSNAIL MINI HDMI CABLE FOR VRX (comes with kit)	Mini hdmi to hdmi output converter cable	
HDMI CAPTURE CARD	capture video signals from the VRX via HDMI	

➢ Flight Controller

Name	Reason	Image
Hex Pixhawk Cube+Flight Controller Autopilot	Has all the required sensors and is a good product for autopilot	

➢ Telemetry

Name	Reason	Image
3DR Single TTL MINI Radio Telemetry 433MHz 500mWfor PIXHAWK and APM FC	433 MHz is the standard frequency for radio communication	

➢ **Camera Models**

Name	Specifications	Reason	Image
(On board Camera) Arducam	12MP IMX708 HDR 120°(H) Wide Angle Camera Module with M12 Lens	High resolution, wide angle coverage, excellent dynamic range for HDR imaging enables precise object detection, facilitates accurate payload deployment	
(Fpv camera) WALKSNAIL AVATAR HD PRO	1080P 120fps compatibility; 22ms low latency; 4km range; 8mp Sony Starvis sensor; Light weight design; FOV160°	HD digital video transmission ensures accurate image processing for object detection	

2.9 C.G. Estimation & Stability Analysis

CG of preliminary CAD Model

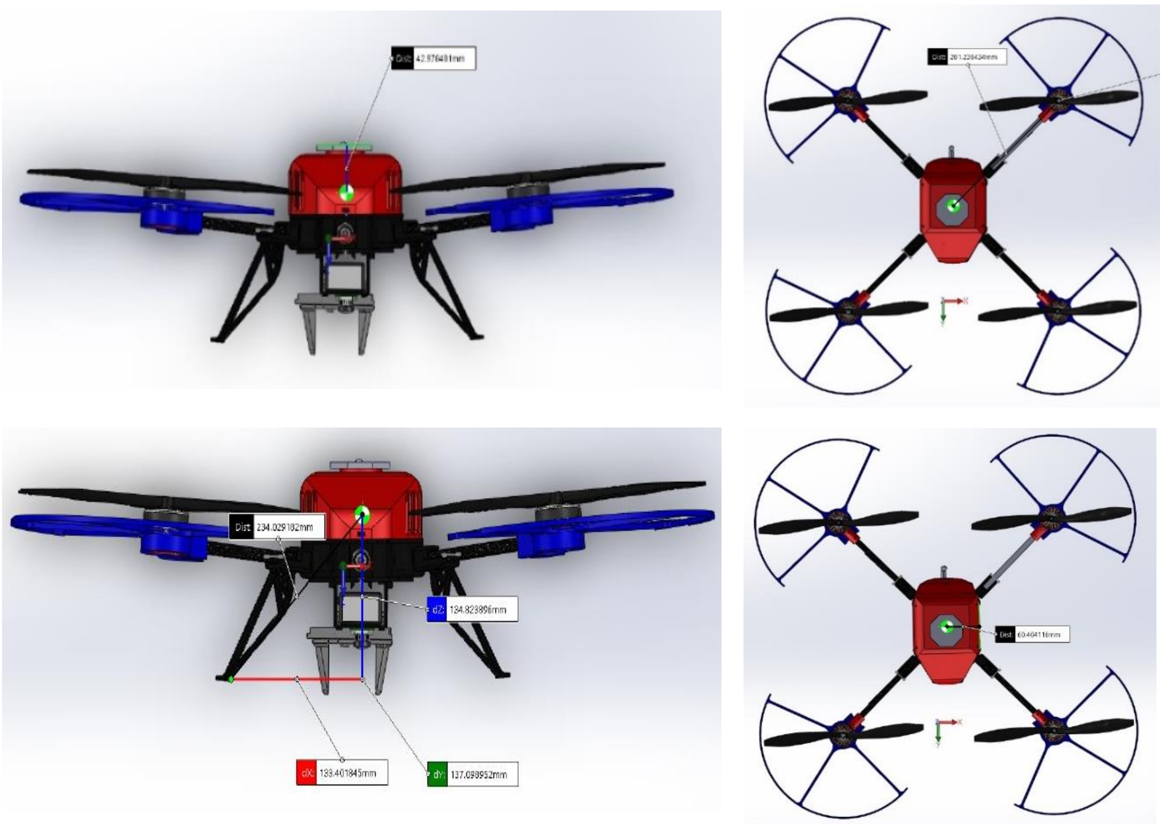


Figure 6: Location of C.G. in the Preliminary CAD Model

Stability Analysis

1. Yaw dynamics equation

$$I_{zz} \times \frac{\delta^2 \phi}{\delta t^2} = Distance_{CG\ to\ motor} \times \left(Thrust_{motor_1} - Thrust_{motor_3}\right) - drag_{coefficient} \times \frac{\delta \phi}{\delta x}$$

2. Pitch dynamics equation

$$I_{yy} \times \frac{\delta^2 \phi}{\delta t^2} = -\ drag_{coefficient} \times \frac{\delta \phi}{\delta x}$$

3. Roll dynamics equation

$$I_{xx} \times \frac{\delta^2 \phi}{\delta t^2} = -\ 2 \times Distance_{CG\ to\ motor} \times Total\ thrust \times \frac{\delta \phi}{\delta x} \times drag_{coefficient} \times \frac{\delta \phi}{\delta x}$$

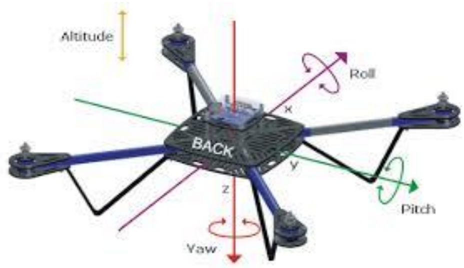


Figure 7: Yaw,Pitch,Roll of quadcopter

Mathematical equation for stability analysis

a. Longitudinal stability analysis:

Pitching motion of quadcopter (equation)

Moment about pitch axis :

I_x – Moment of Inertia about x axis

I_y – Moment of Inertia about y axis

I_z – Moment of Inertia about z axis

t - time

q – pitch rate

m - mass of drone

g – acceleration due to gravity

d - distance from centre of mass to the propeller

θ - Roll angle

ϕ - pitch angle

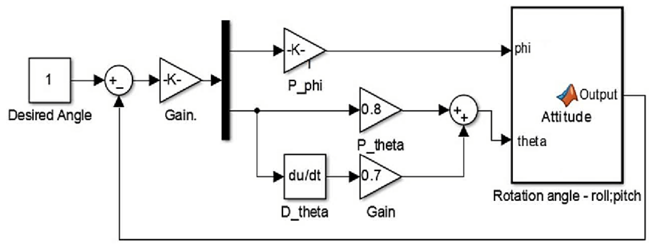


Figure 8: Simulink Block Diagram of PID Control System

$$M\phi = I_x \times \frac{\delta^2 \phi}{\delta t^2} + \left(I_x - I_y\right) \times (q \times r) - mgd \sin \theta$$

b. Lateral stability analysis:

Rolling motion equation:

Motion about θ axis

p - roll rate

r - yaw rate

$$M\theta = I_z \times \frac{\delta^2 \theta}{\delta t^2} + \left(I_y - I_z\right) \times (p \times r) \times mgd \sin \phi \times \cos \theta$$

Yawing motion equation:

Motion about yaw axis:

p – roll rate

q – pitch rate

$$M\phi = I_z \times \frac{\delta^2 \phi}{\delta t^2} + \left(I_y - I_z\right) \times (p \times q) + (mgd \times \cos \phi \times \cos \theta \times \sin \phi)$$

2.10 Preliminary Computer Aided Design Model

We have utilized SolidWorks for the CAD. A key focus was the application of topological optimization to enhance the structural efficiency and performance of the landing gear. Topological optimization is an advanced computational method that optimizes the material layout within a given design space, for a given set of loads, boundary conditions, and constraints with the goal of maximizing performance while minimizing weight.

Figure 9: Preliminary CAD model of UAV

Through the application of topological optimization, we were able to achieve a significant weight reduction of approximately 80% in the landing gear. This was accomplished by systematically removing under-utilized material from the structure, resulting in an optimized design that maintains the necessary strength and stiffness requirements while significantly reducing the overall weight.

2.11 Computational Analysis

➢ **FULL BODY CAE:**

We have performed the full body CAE simulations of our drone in Ansys 2019 R2.
Given below are some results that we got after the simulation: -

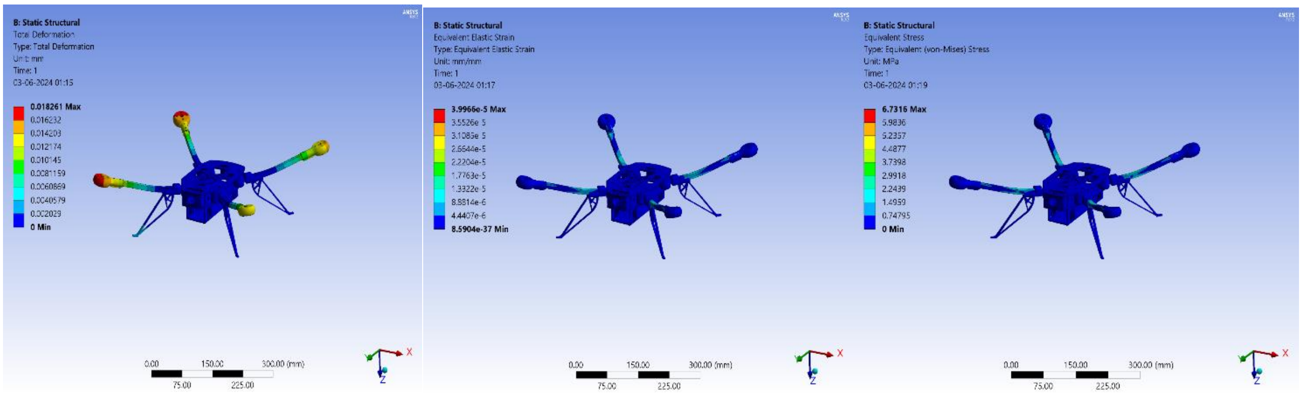


Figure 10: Structural Integrity Test (Left:Deformation,Middle:Elastic Strain,Right:Von Mises Stress)

We have applied thrust of 3.30469146375928lbf on each arm and body weight of 4.40625528501237lbf on the center body.
The result outcomes under following boundary conditions are:

- Maximum deformation: 0.18261in
- Maximum Equivalent Von Mises Stress: 3.9966e-5lbf
- Maximum stress:6.7316lbf

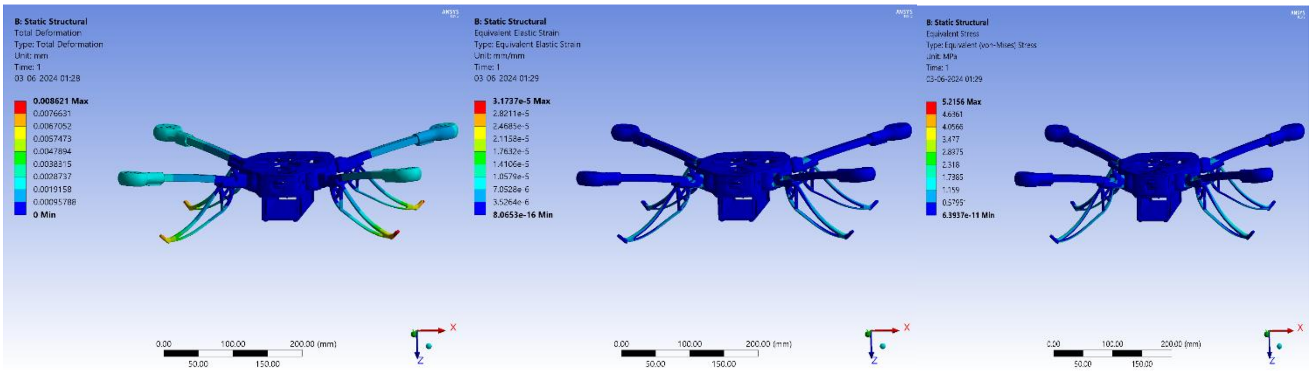


Figure 11: Structural Drop Test (Left:Deformation,Middle:Elastic Strain,Right:Von Mises Stress)

We have applied thrust of 4.40625528501237lbf on each landing gear and kept the body fixed

The result outcomes under following boundary conditions are:

- Maximum deformation: 0.008621in
- Maximum Equivalent Von Mises Stress: 3.1737e-5lbf
- Maximum stress:5.2156lbf

➢ FULL BODY CFD:

We have performed the full body Computational Fluid Dynamics (CFD) of our drone in SolidWorks.

Given below are some results that we got after the simulation: -

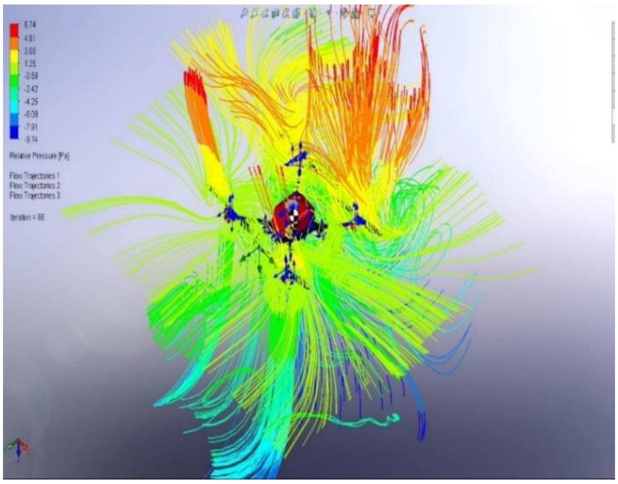


Figure 12: CFD simulation of relative pressure

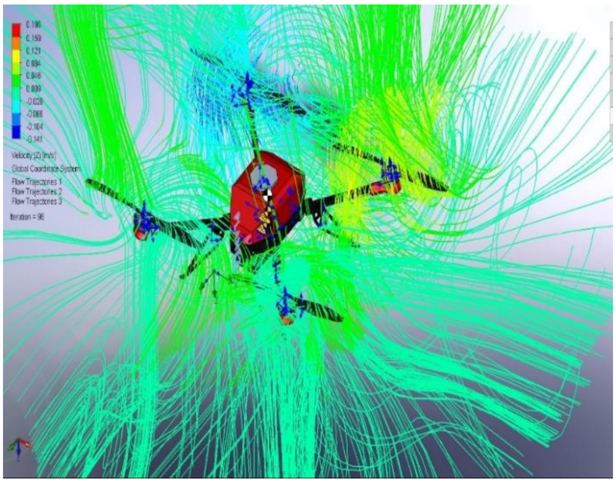


Figure 13: CFD simulation of air velocity along z axis

2.12 Optimised Final Design

In the final design of the drone, we have extended the length of the landing gears and enhanced the payload dropping mechanism.

Figure 14: Optimised Final CAD design of our model

2.13 Detailed Weight Breakdown & C.G. of final UAS Design

2.14 UAS Performance Recalculation

- **Thrust Recalculation :**
  Calculated weight of the drone(with payload) = 1600 + 200 gm = 1800 gm
  We are keeping, Thrust : weight ratio = 3:1
  Thus, thrust required to uplift the drone = 3 x 1800 gm = 5400 gm = 5.4 kg
  Thrust for 1 motor = $\frac{5.4}{4} \times 9.8$ N = 13.23 N/motor

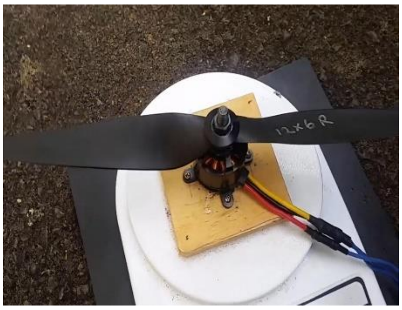


Figure 15: Setup for thrust test

- **Power Recalculation :**

$$(Mechanical\ output\ power)^2 = \frac{(thrust)^3}{\frac{\pi}{2}} \times d^2 \times \rho_{air}$$

$$Mechanical\ output\ power = \sqrt{\frac{(13.23)^3}{\frac{\pi}{2}} \times 0.3048^2 \times 1.164}$$

$$= 116.788\ W$$

Considering the efficiency of the motor to be 80%, then

$$Power\ input = 116.788 \times \frac{100}{80} W = 145.985\ W$$

- **Current Drawn Recalculation**
  Total input power of 4 motors = 4 x Power(input) = 4 x 145.985 W = 583.94 W
  Voltage provided by the battery (according to it's specification) = 14.8 V
  Current = power (input by 1 motor)/voltage = $\frac{583.94}{14.8}$ A = 39.45 A
  Current, per motor = $\frac{39.45}{4}$ A = 9.86 A

Figure 16: Thrust test of the motor

- **Kv Recalculation**
  Thrust = 13.23 N/motor
  ρ_{air} = 1.164 kg/m^3
  Diameter of the propeller, D = 12 inches = 0.3048 m
  pitch = 5 inches = 0.127 m
  speed in rps = $\sqrt{\frac{thrust}{0.5 \times \rho_{air} \times \pi \times \sqrt{diameter} \times pitch^2}}$

$$= \sqrt{\frac{13.23}{0.5 \times 1.164 \times \pi \times \sqrt{0.3048} \times 0.127^2}}$$

$$= 137.477\ rps$$

  speed in rpm = $rps \times \frac{60}{2} \times \pi = 137.477 \times \frac{60}{2} \times \pi = 1313.480\ rpm$

- **Reaction Force Recalculation**
  Reaction force = Total Weight = 1.8 $\times$ 9.8 N

Using Bernoulli's form :-
Bending Moment,
B.M = Reaction force × length of arm
BM = 1.8 × 9.8 × 0.2145 = 3.78 N-m

Length of drone arm = 0.2145 m
Modulus of elasticity of material = 33 msi (228 GPa)
Ultimate tensile strength (UTS) = 500 Ksi (3.5 GPa)
Outer Diameter(OD) = 10 mm
Inner Diameter(ID) = 8 mm

Moment of Inertia (I) :-
Ixx $= \frac{10\times8^3-8\times10^3}{12} = 240\ mm^4$
Iyy $= \frac{10\times8^3-8\times10^3}{12} = 240\ mm^4$

- **Final Battery Capacity Recalculation**

The battery chosen (assumed) for initial calculations give a 14.5 min flight time just for hover, which is not enough considering the motors will draw more power based on its flight path while liftoff and changing direction. So, the practical flight time will be not be enough.

Battery Capacity Calculation (for hover)

Considered Flight Time = 20 mins = 1/3 H
Net current draw for hovering = 23.067 A
Required Battery Capacity = Current Draw x Flight time
= 23.067 x 1/3
= 7.689 AH = ~7.7 AH = 7700mAh
(This is 90% of the battery capacity)
Therefore, required battery capacity = 8.556 AH ≈8500mAh

Nearest Commercially Available Battery Capacity ≈ 8000mAh
Therefore, available capacity = 8000 x 0.9 = 7200mAh

- **Flight Time Re-calculation with chosen Battery Capacity**

Case 1: Flight time with minimum thrust(hovering):

Weight of Drone = 2kg x 9.81 m/s^2 = 19.62 kg-f
Therefore, for hovering minimum thrust required per motor = 19.62/4
= 4.905 kg-f
= 490.5 gf
≈ 500 gf
Current Draw per motor at 500 gf thrust = 3.6 A (from Manufacturer Datasheet)
Total Current Draw by 4 motors = 3.6 x 4 = 14.4 A
Considering ESC efficiency 80%
Current Draw by Propulsion System = 14.4/0.8 = 18 A
Current Draw by Remaining Electronics = 5.067 A
Net Current draw = 23.067 A
Flight time = 7.2AH/23.067A = 0.3121 H = 18.726 min
Flight time = 18.72 mins

This is a theoretical flight time and only includes hovering of the drone.
The practical flight time will decrease as the drone will consume more power while lift-off and changing direction based on its flight path.

Case 2: Flight Time with Peak Performance:

Peak Thrust per motor = 1667 gf
Peak Current per motor = 20.1 A
Net Peak Current by 4 motors = 20.1 A x 4 = 80.4 A
Considering ESC efficiency = 80%
Current draw by propulsion system = 80.4/0.8 = 100.5 A
Current draw by remaining electronics = 5.067 A
Net current draw = 105.567 A
Therefore, Flight time = 7.2/105.567 = 0.0682 H = 4.098 min
Flight Time ≈ 4.1 min (at peak performance)

3. FINAL UAS SPECIFICATIONS & BILLS OF MATERIALS

Table 11: Bills of materials

Item Name	Part Name	Supplier	Price	Qty	Total
Motor	SunySky V4006 740KV	SunnySky	₹ 4426	4	₹ 17704
Propeller	EOLO 12”x5”	SunnySky	₹ 2505/pair	2	₹ 5010
Battery	8000mAH 4S	Robu	₹ 11193	1	₹ 11193
ESC	Hobby Wing 40A	Robu	₹ 1750	4	₹ 7000
Flight Controller	Hex Pixhawk Cube+	Robu	₹ 34,999	1	₹ 34,999
Transmitter Receiver	FlySky FS-i6	Robu	₹ 4,459	1	₹ 4,459
GPS	CUAV Neo 3 Pro	Robu	₹ 17,335	1	₹ 17,335
Telemetry	3DR Single TTL Mini 433MHz	Robu	₹ 7,999	1	₹ 7,999
Flight Computer	Raspberry Pi 4B (8GB)	Robu	₹ 8,248	1	₹ 8,248
Image Processing Camera	ArduCAM IMX708 12MP	Robu	₹ 3,899	1	₹ 3,899
FPV Camera	Walksnail Avatar HD Pro	CaddxFPV	₹ 5429	1	₹ 5429
VTX (with 2 antennas & cable)	Walksnail HD Pro VTX	CaddxFPV	₹ 10775	1	₹ 10775

Item Name	Part Name	Supplier	Price	Qty	Total
HDMI Capture Card	Sounce 4K HDMI Video Capture Card	Amazon	Rs 499	1	₹ 499
VRX	Walksnail Avatar VRX	CaddxFPV	₹ 18293.	1	₹ 18293.
PLA Pro+			₹ 848	1	₹ 848
ABS Pro			₹ 1469	1	₹ 1469
CF-ABS			₹ 3000	1	₹ 3000
Carbon fiber rods			₹ 559	1	₹ 559
Servo Motor	TowerPro MG90S	Robu	₹ 119	1	₹ 119

4. SYSTEM DESIGN FOR CAPTURING THE SURVEY DATA

Setup for Mission 1 and 2

Mission 1: Manual Control
In Mission 1, the UAV is equipped with a high-resolution camera and connected to a ground station laptop running YOLO v9 via a wireless RF link. The UAV streams live video to the laptop, where YOLO v9 processes each frame to detect hotspots and objects and classify them in real-time, highlighting them with bounding boxes and labels on the video feed. The operator navigates the UAV manually using this visual feedback to survey the field, identify objects, and detect hotspots. When objects or hotspots are detected, they are logged and the counter raises by 1 also a picture is taken along with a timestamp for later analysis.

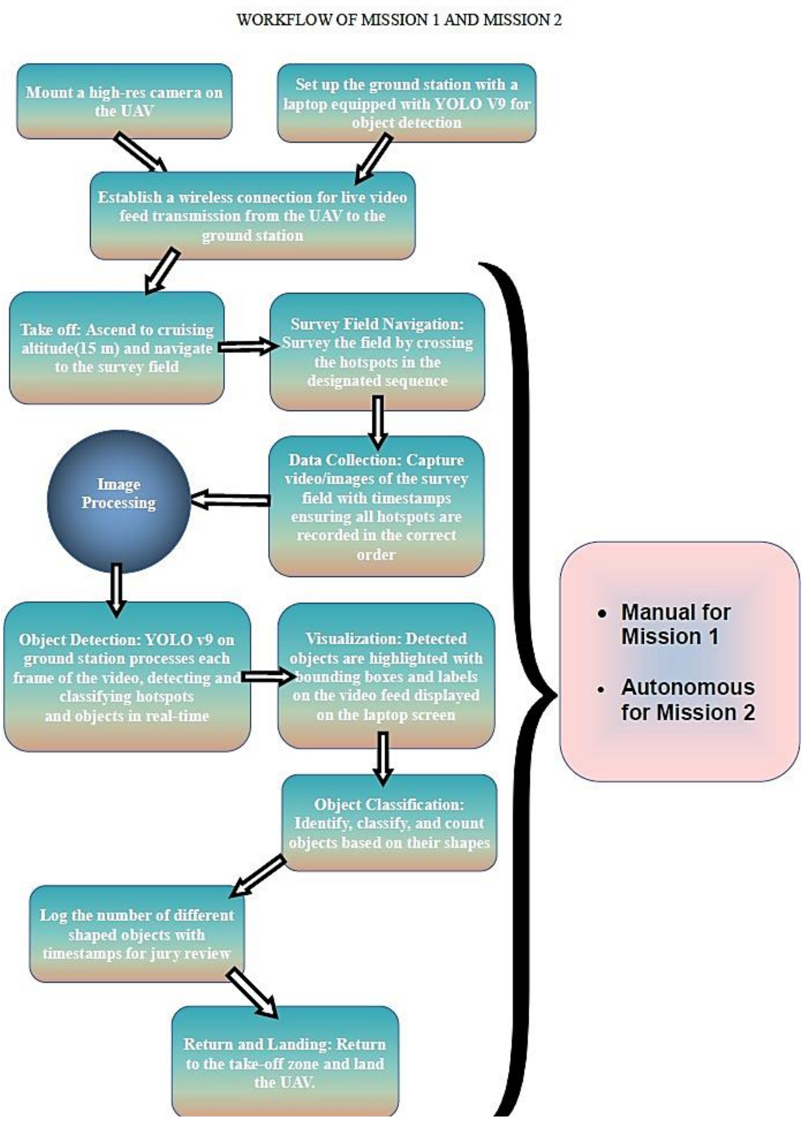


Figure 17: Workflow of Mission 1 and 2

Mission 2: Autonomous Operation
In Mission 2, the UAV, equipped with a high-resolution camera and an autopilot system compatible with Mission Planner, autonomously navigates a predefined flight path. The ground station laptop, connected to the UAV's telemetry system, processes the live video

feed using YOLO v9 to detect and classify objects and hotspots in real-time. The UAV follows waypoints set in Mission Planner, and when an object or hotspot of interest is detected, a picture is sent to the ground station and the counter is raised by one.This allows for real-time adjustments and autonomous identification of objects during the mission and at the end of the mission the drone returns to the take off zone and lands safely.

Setup for Mission 3

In Mission 3, the UAV is manually navigated through a maze using an onboard high-resolution camera that streams a live video feed to a ground station laptop. As the UAV maneuvers
through the maze, the operator relies on their skills and visual feedback to avoid obstacles and navigate successfully. The ground station laptop equipped with YOLO v9 processes the live
video feed to detect hotspots in the environment. These hotspots are highlighted in the video
feed, assisting the operator in manually identifying the correct targets. Upon confirming the
target visually, the operator manually controls the UAV to descend from 15m cruise altitude
to 5m dropping altitude and drop the payload accurately on the identified target. This mission combines manual navigation skills with advanced hotspot detection to ensure precise payload delivery.

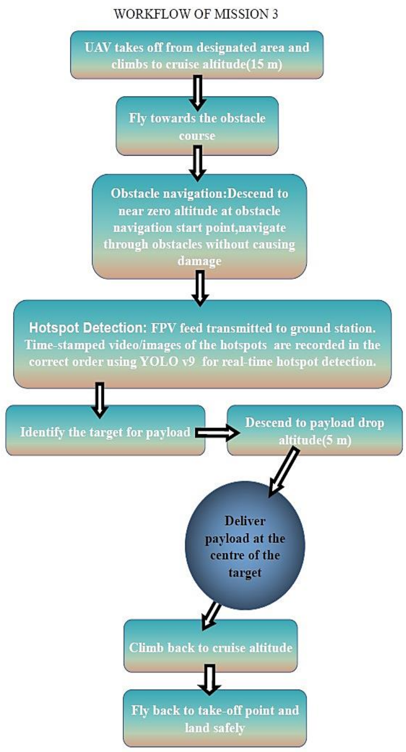


Figure 18: Workflow of Mission 3

Setup for Mission 4

Mission 4 involves conducting an autonomous raster search, detecting real and fake targets, accurately dropping payloads, and using YOLO v9 on the ground station for hotspot detection. The UAV is equipped with a high-resolution FPV camera, a Raspberry Pi with MobileNet SSD v2 and TensorFlow Lite for onboard target detection, a payload release mechanism, and a telemetry system for data transmission. The ground station, featuring a laptop with Mission Planner for flight path planning and YOLO v9 for hotspot detection, is also equipped to receive and analyze data from the UAV. The flight path planning in Mission Planner starts with defining a raster search pattern for the designated search area, setting the UAV's cruise height to 15 meters for the search operation. Waypoints are set for the search grid, descending to 5 meters for payload drop upon target detection, and finally for the UAV to return to the takeoff zone after mission completion.

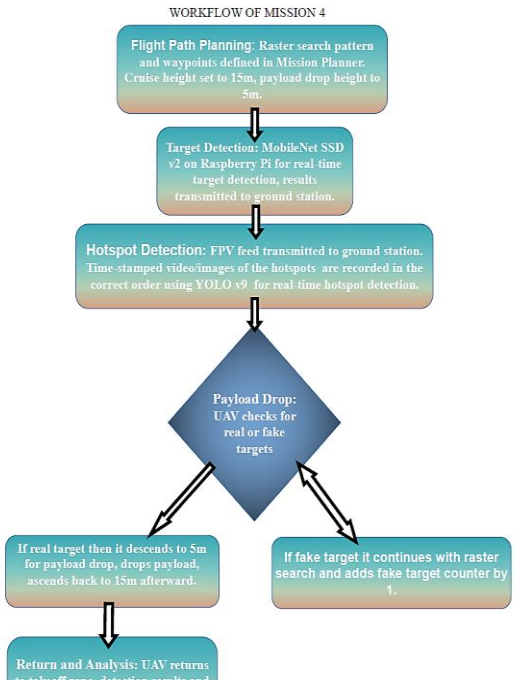


Figure 19: Workflow of Mission 4

Autonomous Raster Search with Target Detection and Payload Drop

UAV Equipment:

- High-resolution FPV camera.

- Raspberry Pi with MobileNet SSD v2 and TensorFlow Lite for onboard target detection.
- Payload release mechanism.
- Telemetry system for data transmission.

Ground Station:

- Laptop with Mission Planner for flight path planning.
- YOLO v9 installed for hotspot detection.
- Software for receiving and analyzing data from the UAV.

Flight Path Planning:

The configuration in Mission Planner involves several key steps. First, a raster search pattern is defined for the designated search area, with the UAV's cruise height set to 15 meters for the search operation. Waypoints need to be set for the search grid and for descending to 5 meters for payload drop upon target detection. Finally, waypoints should be established for the UAV to return to the takeoff zone after completing the mission.

4.1 Data Collection method and format

Table 12: Comparison between different detection models

Model	Speed (FPS)	Accuracy (mAp)	Latency	Complexity	Robustness to Variations	Real-time Processing
YOLO v9	60+ FPS	High (75-85%)	Very low	Moderate	High	Excellent
YOLOv4	30-60 FPS	High (70-80%)	Low	Moderate	High	Excellent
Faster R-CNN	5-7 FPS	Very High (80-90%)	High	High	Moderate	Poor
SSD	22-30 FPS	Moderate (50-60%)	Low	Low	Moderate	Good
Efficient Det	25-35 FPS	High (70-80%)	Moderate	High	High	Good
Retina Net	5-10 FPS	Very High (70-80%)	High	High	Moderate	Poor

Key Matrices

FPS: Frames per second, indicating how quickly the model processes images. Higher FPS is better for real-time applications.
mAP: Mean Average Precision, a standard metric for evaluating object detection models. Higher mAP indicates better accuracy.

Conclusion

Choosing **YOLO v9** for object detection is justified due to its superior **speed (60+ FPS)** and **high accuracy (75-85% mAP)**, which are crucial for real-time applications. It has very **low latency**, ensuring timely responses, and moderate complexity, making it feasible to implement.

YOLO v9 is robust to variations in lighting and occlusions, essential for **dynamic drone feeds**, providing access to extensive resources and optimizations. Its efficiency in **real-time processing** ensures high performance in demanding scenarios, making it the ideal choice for drone-based applications.

Data Collection Format

The chosen ML model will be implemented on the video. The image at the time of detection containing the timestamp will be stored. Additionally an excel file containing the path to the image and the timestamp will also be stored in the backend.

4.2 Transmission of Data and Mechanism to retrieve it

Communication

EXPLANATION: During the first three rounds, our drone transmits live video to a main station using an FPV camera and transmitter. The data is gathered for AI-based recognition and analysis at the main station using a laptop equipped with a receiver, guaranteeing prompt processing.

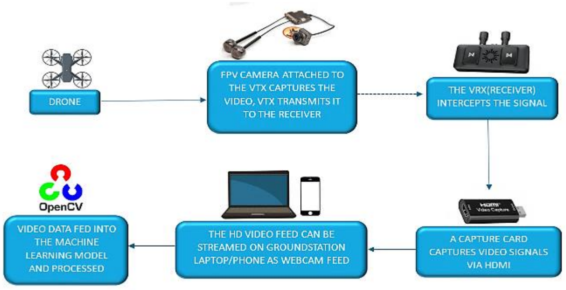


Figure 20: Flowchart of video transmission

For payload detection and dropping, we include a Raspberry Pi with a Raspicam (ARDUCAM IMX708) in the fourth iteration. Performance is enhanced by this dual-camera arrangement, which lowers latency and optimizes video feed delivery. The Raspicam manages onboard computing for accurate payload operations, while the FPV camera keeps up its real-time view of the environment. For increased precision and efficiency, this division enhances cargo management in addition to video transmission.

FPV Camera Communication

Our drone's FPV (First-Person View) system enables real-time communication through a structured process. The FPV camera captures live video, which is sent to an onboard transmitter that converts the visual data into radio frequency (RF) signals. These RF signals are broadcast to the base station, where a receiver converts them back into video data for display on a laptop. This real-time feed is crucial for observation and is integrated into our AI model for identification and evaluation, ensuring prompt visual feedback and precise control.

FPV Camera:

Digital Transmission Protocols (proprietary):

1. Digital Radio Frequencies followed (5.8 GHz frequency considered)
2. Low latency.
3. Higher video quality
4. Longer range.
5. Frequency Modulation Equipped.
6. Encryption Equipped.
7. Range extended above 1 km.

Why we did not use Standard Analog Signals:

Table 13: Difference between digital and analog signals

DIGITAL SIGNALS	ANALOG SIGNALS
Lower Latency-better video quality	Higher latency

DIGITAL SIGNALS	ANALOG SIGNALS
Reduced Inference	Higher Inference
Extended Range	Lower Range
Future Proofing	No Future Proofing

Our Requirements:

- Excel in real-time video feed.
- Low Latency.
- Quick response
- Durability, Range, and Connectivity.

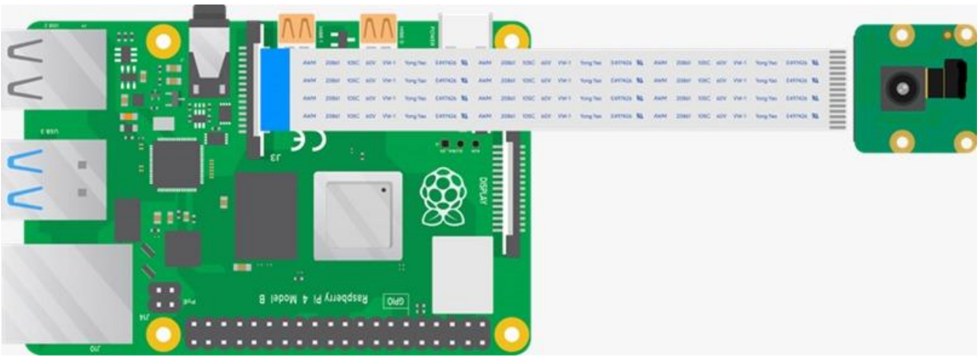


Figure 21: Connection of ARDUCAM with RASPBERRY PI

5. METHODOLOGY FOR AUTONOMOUS OPERATION

5.1 Autonomous Flight

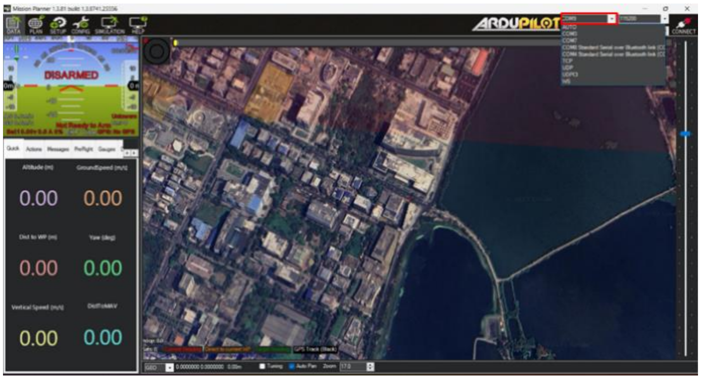


Figure 22: Select to the COM port

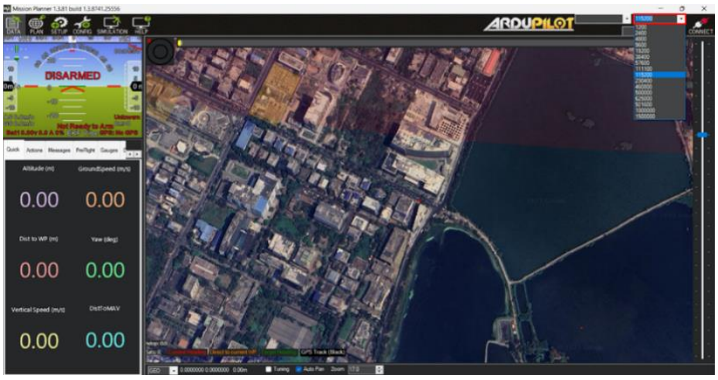


Figure 23: Select the Baud Rate

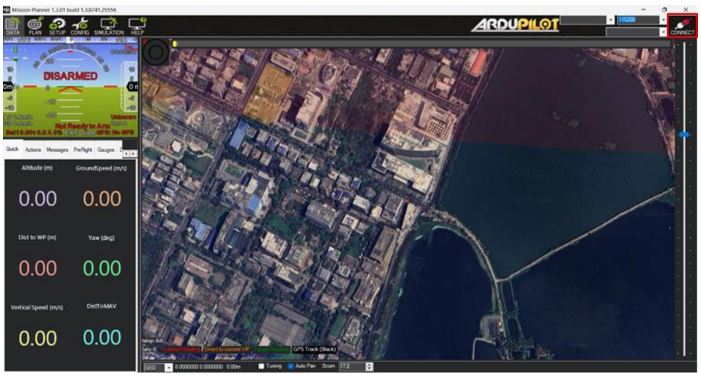


Figure 24: Connect to the Flight Controller

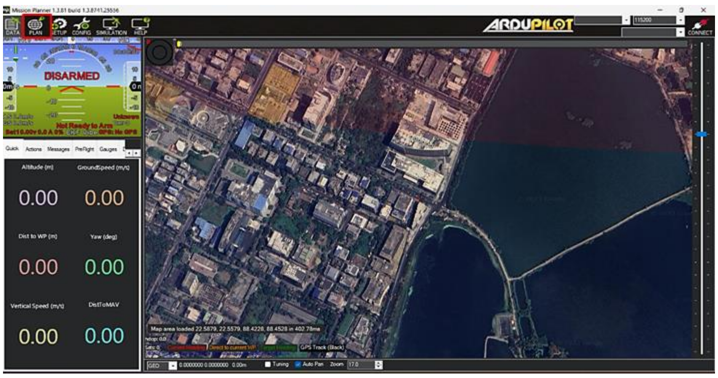


Figure 25: Select the Plan Tab

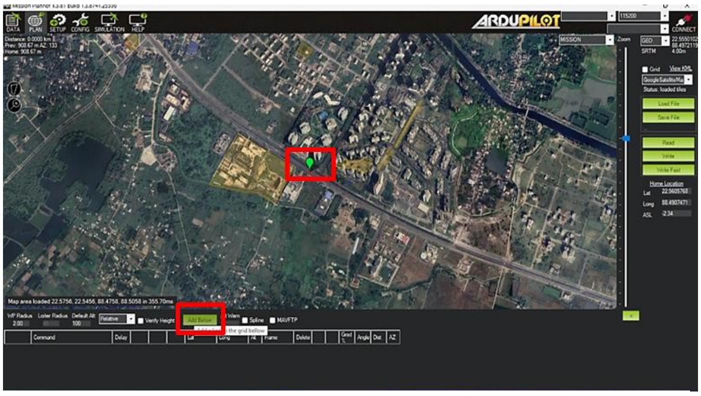


Figure 26: Select the Take-Off point and click Add

Figure 27: Select the remaining waypoints

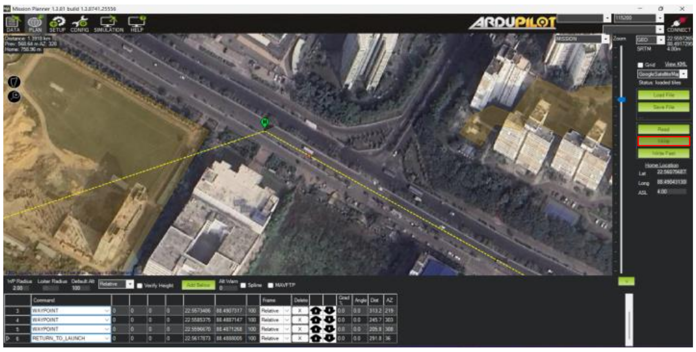


Figure 28: Press Write button to write the mission to the Flight Controller

Figure 29: From the Data Tab, in the Actions Tab, click on Arm to Arm the UAV

Figure 30: Press Do Action to Start the Mission

We used Frames brought from retailers only for testing purposes

Manual Flight Test

We used an **S500 frame** we bought from a retailer for testing to examine the functionality of our system during a manual flying test. We tested the use of a Raspberry Pi and a Pixhawk for manual monitoring and detection in a controlled mock arena with a variety of hotspots and objects. With this configuration, we were able to assess our system's performance in practical situations.

Figure 31: Manual Testing

Autonomous Flight Test

We used waypoints with telemetry and a Pixhawk flying controller to build a secure detection, evaluation, dropping, and testing arena during an outdoor autonomous flight test utilizing **F450 frame**. We tested this first, then we'll actually utilize it on a drone that is made. This configuration made it possible for our UAV to carry out a number of autonomous missions, allowing it to follow predefined paths and complete exact duties. Target area identification and assessment were made possible in real-time by the onboard systems, which included the Raspberry Pi. While our final frame is being produced, this extensive outdoor testing gave us significant information about our system's capabilities and ensured that its deployment mechanisms are precise and dependable.

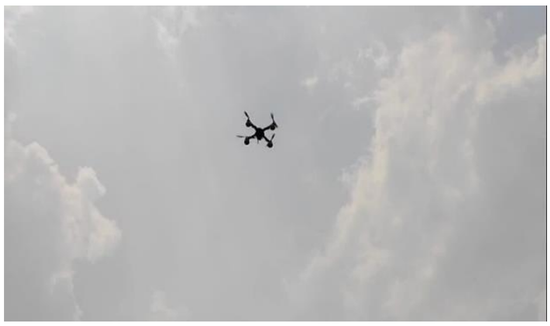


Figure 32: Autonomous Testing

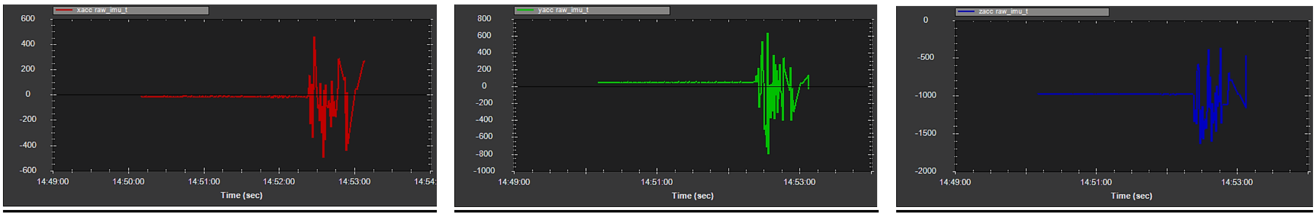


Figure 33: Accelerometer Readings from the tests along the 3 axes

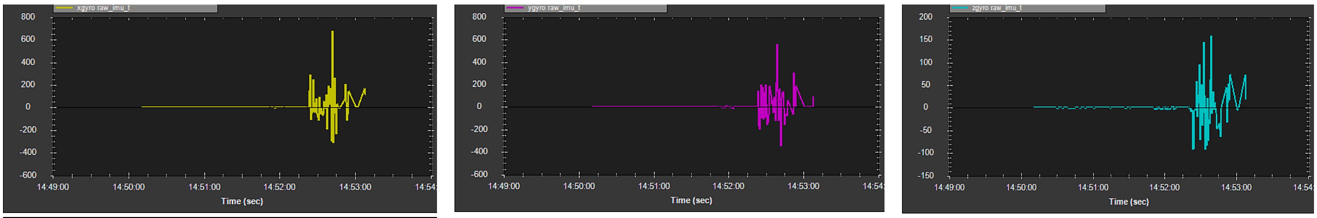


Figure 34: Gyroscope Readings from the tests along the 3 axes

5.2 Autonomous Identification of target

Comparative Analysis of Lightweight Object Detection Models to be used onboard

For deploying object detection models on a resource-constrained device like the Raspberry Pi, it is crucial to compare models based on accuracy, speed, resource usage, latency, and ease of implementation.

Detection Accuracy Comparison Chart:

Here's a comparative chart summarizing detection accuracy (mean Average Precision, mAP) and other relevant parameters for Tiny YOLO, MobileNet SSD v2, SqueezeNet, ShuffleNet, and EfficientDet Lite:

Table 14: Detection accuracy comparison chart

Model	Speed (FPS)	Accuracy (mAP)	Model Size (MB)	Ease of Use	Resource Usage	Latency (ms)
Tiny YOLO	22-25	33.1%	~33	High	Low	~40-50
MobileNet SSD v2	15-20	45.3%	~20	Very High	Medium	~50-60
SqueezeNet	20-25	34.0%	~5	High	Low	~40-50
ShuffleNet	20-25	36.7%	~5	Medium	Low	~40-50
EfficientDet Lite	10-15	49.0%	~14	Medium	Low	~60-70

Key Metrics

- **FPS:** Frames per second, indicating how quickly the model processes images. Higher FPS is better for real-time applications.
- **mAP:** Mean Average Precision, a standard metric for evaluating object detection models. Higher mAP indicates better accuracy.

Conclusion

MobileNet SSD v2 with TensorFlow Lite is ideal for the Raspberry Pi due to its balanced performance, combining moderate **speed (15-20 FPS)** and **high accuracy (45.3% mAP)**. TensorFlow Lite optimizations enable efficient use on low-power devices, crucial for the Raspberry Pi's limited resources. The model's moderate **size (~20 MB)** ensures it doesn't overburden the system. It offers significantly higher accuracy than Tiny YOLO and SqueezeNet, and is only slightly behind EfficientDet Lite but with better speed. Additionally, its ease of use, supported by extensive documentation and community support, simplifies deployment and troubleshooting

Identification of Target

Raspberry Pi 4B serves as the UAV's onboard processing unit, running MobileNet SSD v2 with TensorFlow Lite to detect real and fake targets in real-time from the onboard arducam feed. It autonomously identifies the targets, makes decisions for precise payload drops by sending commands to the release mechanism, and transmits detection results to the ground station. The Raspberry Pi 4B works in coordination with Mission Planner, which manages the UAV's flight path, allowing the UAV to perform an autonomous raster search at a cruise height of 15 meters, descend to 5 meters for payload drops upon target detection, and return to the takeoff zone upon mission completion. This setup ensures efficient target detection and payload delivery while the ground station focuses on hotspot detection using YOLO v9.

5.3 Autonomous Payload drop

The Raspberry Pi-powered UAV drone's onboard camera uses an onboard machine learning model to determine the ideal drop point when it finds the approved payload dropping region. Once this location has been established, a signal is created and sent via the GPIO pins. This activates the servo-motor of payload dropping setup, this signal guarantees that the payload is dropped precisely where it is supposed to be. Real-time signal processing and sophisticated picture recognition are seamlessly integrated, guaranteeing the UAV drone can carry out its payload delivery job with maximum efficiency and precision.

6. SUMMARY OF INNOVATION IN THE OVERALL DESIGN

This innovation centers on an unmanned air vehicle (UAV) designed to deliver payloads using a mechanism that allows for on-demand deployment. The payload can be released either by a remote operator or autonomously via the onboard computer. The primary objective is to securely hold and precisely release any irregular object as required.

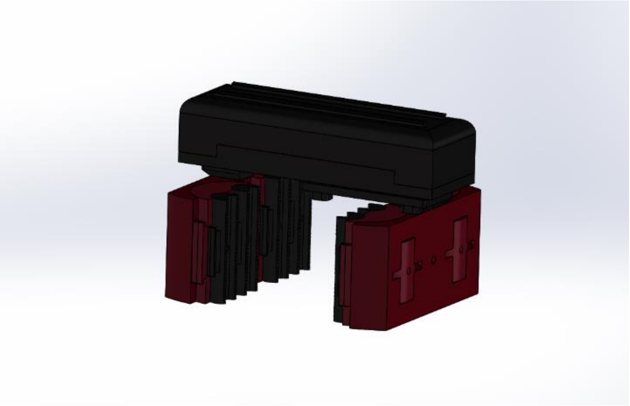


Figure 35: Payload dropping mechanism

Concept and Technical Features

The core concept of this method is the use of a fractal vice combined with a rack and pinion mechanism to securely grasp and hold irregular objects. Key technical features include:

Fractal Vice

The design of the fractal vice incorporates a fractal-inspired pattern, allowing it to conform precisely to the shape of irregular objects. This innovative approach ensures a secure hold by adapting to the unique contours of each payload, providing exceptional stability during transport. The fractal design's self-similar and repeating patterns offer significant advantages, including high precision and adaptability in grasping a wide variety of payloads. This combination of features ensures that the UAV can handle diverse and irregularly shaped objects with reliability and efficiency. The proposed mechanism uniformly distributes the grasping force to the entire body that is being grasped, irrespective of its shape.

Rack and Pinion Mechanism

The rack and pinion mechanism operates by controlling the movement of both parts of the vice, allowing them to move towards each other to secure the payload. A servo motor, controlled by a Raspberry Pi, is connected to the pinion. When a command is given, the servo rotates the pinion, causing both arms of the vice to move and push the object from both sides until it is firmly held. This mechanism ensures that the payload is securely grasped until the precise moment of release, providing high precision in the gripping and releasing process.

Raspberry Pi Integration

The Raspberry Pi serves as the control system, managing the servo motor's movements to allow for precise control over the gripping and releasing process. This integration facilitates the autonomous operation of the UAV for payload deployment, enabling both manual and automated control. The use of the Raspberry Pi enhances the system's overall functionality, making it possible to handle complex tasks with minimal human intervention, thereby increasing the efficiency and reliability of the UAV's payload delivery capabilities.

Key Features

- **Adaptability**: Capable of securely holding various irregularly shaped objects.
- **Precision Control**: Uses a rack and pinion mechanism with servo motor control for precise gripping and release.
- **Autonomous Capability**: Integrates with Raspberry Pi for automated control, enabling both manual and autonomous operation.
- **Secure Hold**: The fractal vice ensures a firm and secure grip on the payload during transport.
- **On-Demand Release**: Payload can be released exactly when needed, enhancing operational efficiency.

Conclusion

This innovative design leverages a fractal vice and a rack and pinion mechanism, controlled by a Raspberry Pi, to provide a highly adaptable, precise, and secure payload delivery system for UAVs. By ensuring a firm grip on various irregularly shaped objects and allowing for precise, on-demand release, this system significantly enhances the reliability and effectiveness of UAV-based delivery operations. The combination of adaptability, precision, and autonomous control positions this design as a cutting-edge solution in the field of UAV payload delivery.

7. APPENDIX

Electronic Datasheets

Computational Analysis

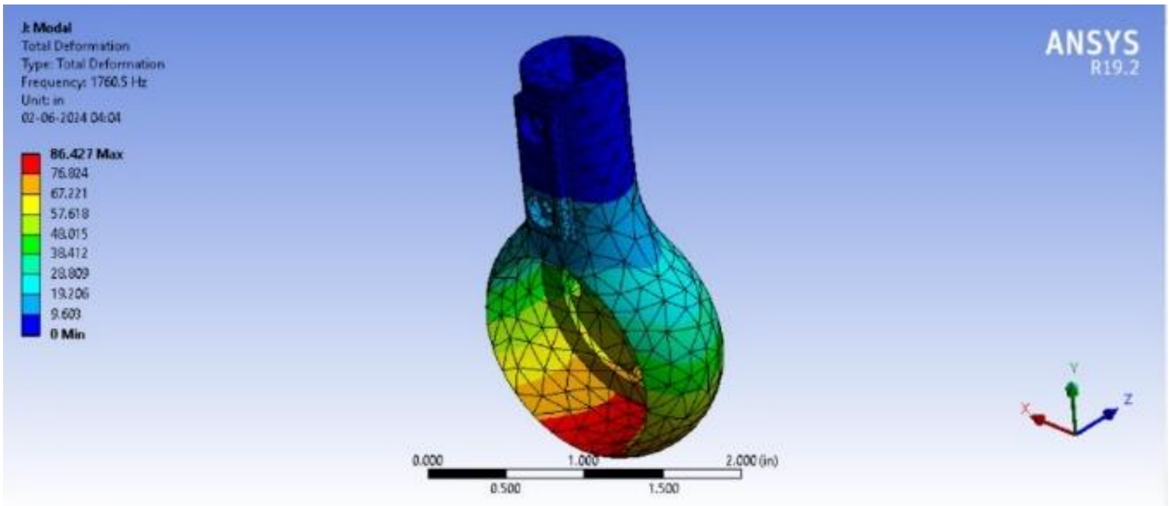


FIGURE 1: TOTAL DEFORMATION – ROTOR ARM

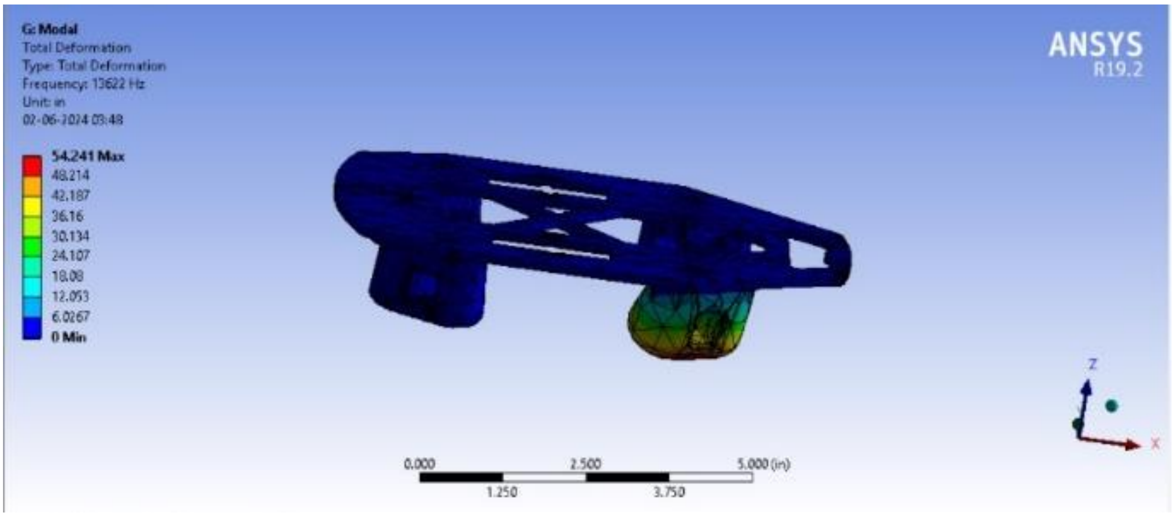


FIGURE 2: TOTAL DEFORMATION– TOP PLATE

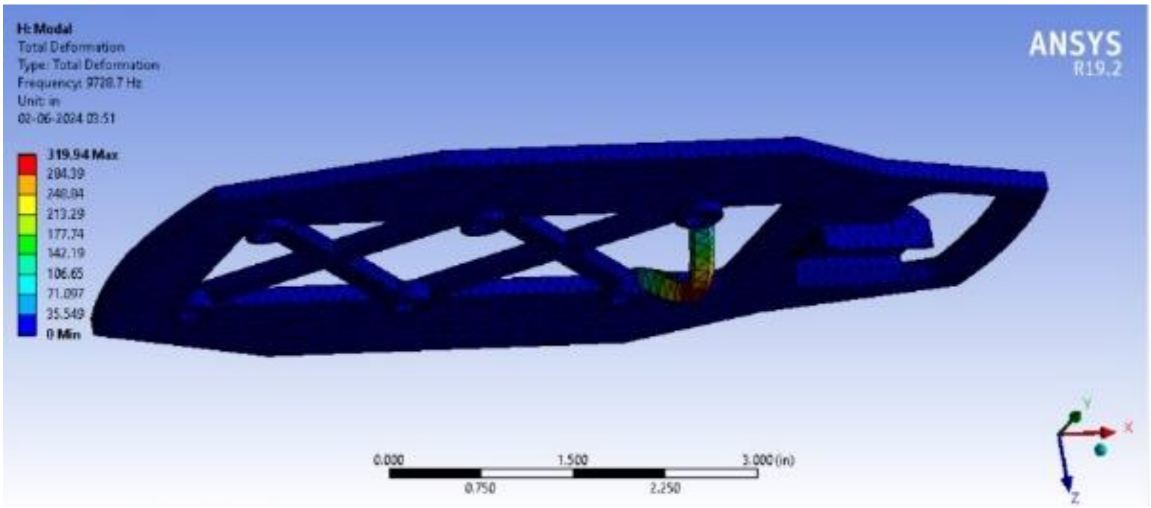


FIGURE 3: TOTAL DEFORMATION – BASE PLATE

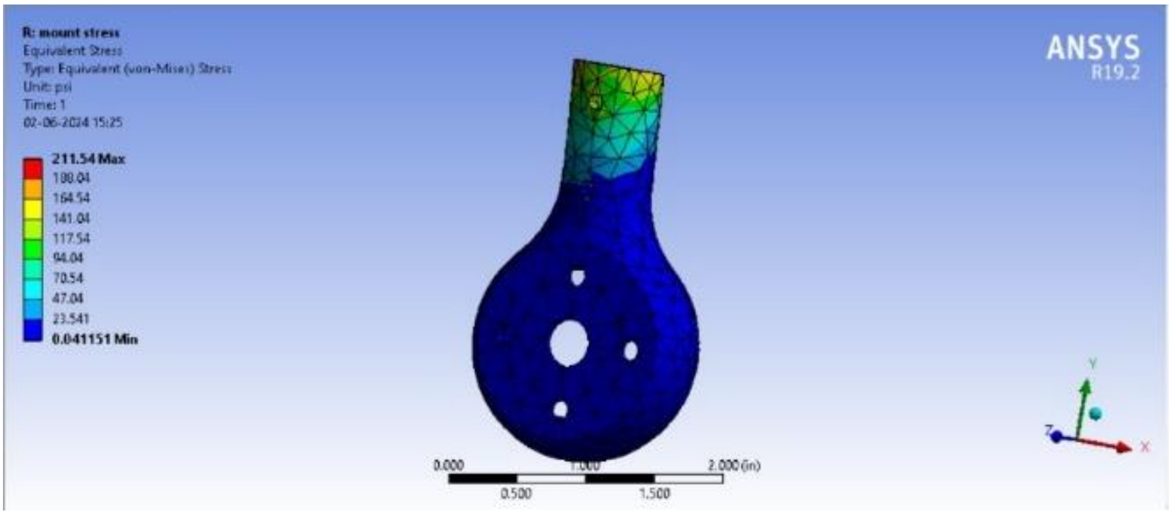


FIGURE 4: EQUIVALENT STRESS – MOTOR MOUNT

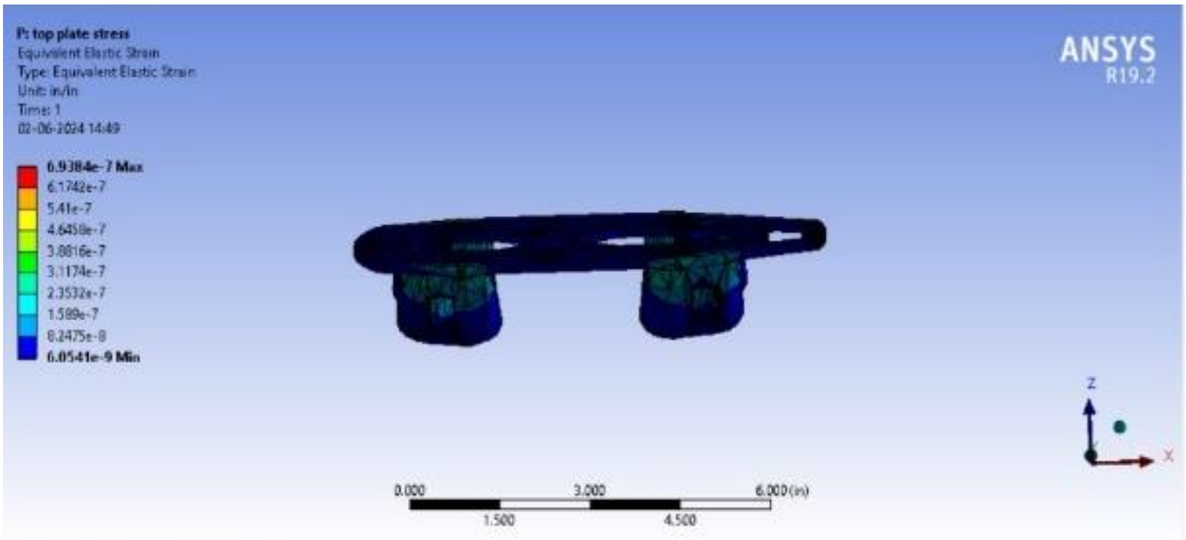


FIGURE 5: EQUIVALENT ELASTIC STRAIN – TOP PLATE

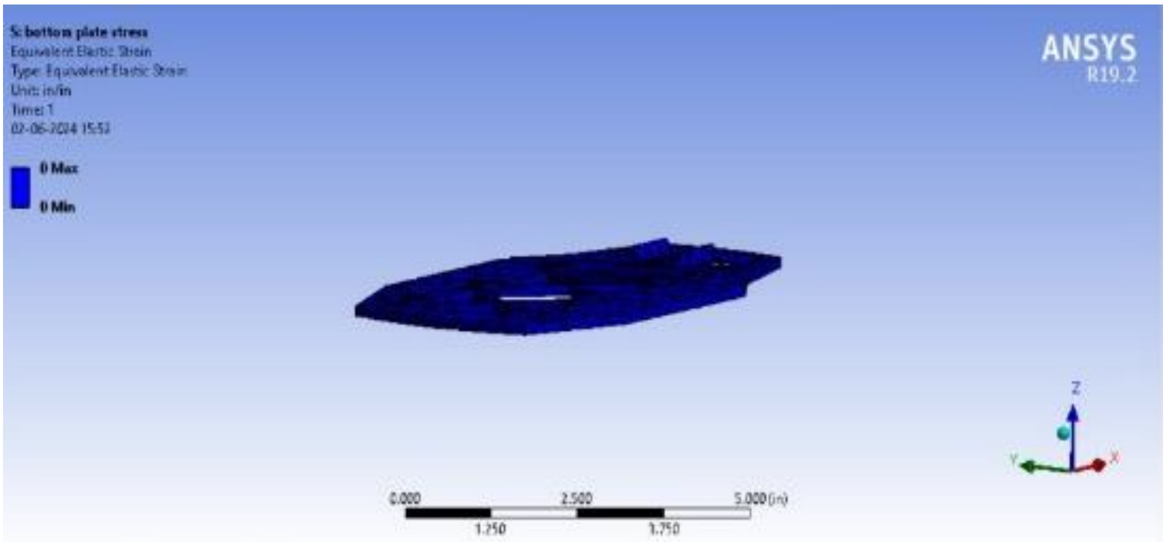


FIGURE 6: EQUIVALENT ELASTIC STRAIN – BOTTOM PLATE

Methodology For Autonomous Operation

- **Save frames of live video with timestamps**

❖ Capture Live Video: Using OpenCV to capture video from our camera

- cap = cv2.VideoCapture(0) opens the default camera.
- ret, frame = cap.read(): Captures a frame from the video stream.
- The camera will open by giving 0 as input to cv2.VideoCapture() function.

❖ Adding Timestamps: Overlaying the current timestamp on each frame.

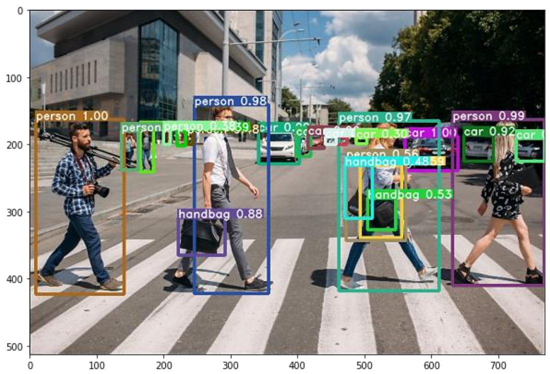
- The datetime.now() will give the current date and time put this timestamp on that frame and display the image by using the cv2.imshow() function.
- cv2.putText(): Adds the timestamp text to the frame.

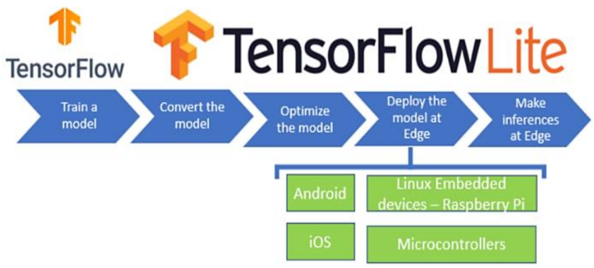
❖ Saving Frames: Saving the frames as images.

- cv2.imshow('Live Video', frame): Displays the frame with the timestamp.
- cv2.imwrite(): Saves the frame as an image file. The filename includes the current timestamp for uniqueness.

- **Define Object Detection and Classification Model**

Load a pre-trained object detection and classification model. We will use a pre-trained model like MobileNet V2 SSD with Tensorflow Lite.





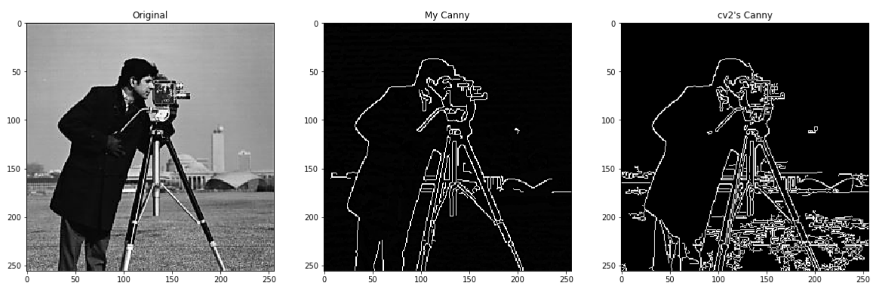
- **Canny Edge Detection**

❖ In Real-time edge detection, the image frame coming from a live video feed is continuously captured and each frame is processed by edge detection algorithms which identify and highlight these edges continuously.

❖ We will use the Canny edge detection algorithm of OpenCV to detect the edges in real-time because it produces smoother edges due to the implementation of Non-maxima suppression and thresholding.

❖ It works in four stages i.e Noise reduction, Finding the intensity gradient, Non-maximum suppression, and Hysteresis thresholding.

❖ The frame is a single image frame (numpy array) on which we will perform edge detection using the Canny edge detection method.

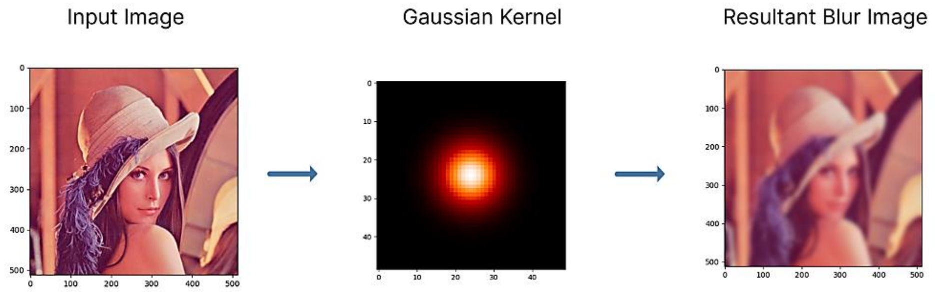


❖ **Convert the frame to grayscale for edge detection**

- gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
- This converts the input frame from the BGR colour space to grayscale. The Canny edge detection method works on grayscale images, so we first convert the frame to grayscale using cv2.cvtColor() the function.

❖ **Apply Gaussian blur to reduce noise and smoothen edges**

- blurred = cv2.GaussianBlur(src=gray, ksize=(3, 5), sigmaX=0.5)
- This applies Gaussian blur to the grayscale image. The cv2.GaussianBlur() function helps to reduce noise and smoothens the edges, which is useful for obtaining better edge detection results. The (3, 5) argument represents the size of the Gaussian kernel used for blurring and 0 indicates the standard deviation in the X and Y directions.



❖ **Perform Canny edge detection**

- edges = cv2.Canny(blurred, 70, 135)
- This performs the actual Canny edge detection on the blurred grayscale image. The cv2.Canny() the function takes three arguments: the input image (blurred), the lower threshold (70), and the upper threshold (135). Pixels with gradient magnitude below the lower threshold are considered not edges, and pixels with gradient magnitude above the upper threshold are considered strong edges. Pixels with gradient magnitude between the two thresholds are considered weak edges.
- If edges are not connected, we need to connect the edges, make them thicker & more visible using cv2.dilate

❖ **Find and Draw Contours**

- **Contours** are defined as the line joining all the points along the boundary of an image that are having the same intensity.
- OpenCV has findContour() function that helps in extracting the contours from the image. It works best on binary images, so we should first apply thresholding techniques, Sobel edges, etc.
- The *findContours()* finds the contours in the given image. The first option is the output of the canny edge detector. *RETR_EXTERNAL* tells OpenCv to only find the outermost edges. The second arguments tells OpenCv to use the simple approximation.
- The function returns two values: A list of contours found, and the hierarchy.
- The *contours* return value is a simple list that contains the number of contours found. Taking the length of it will give us number of objects found.

- convert the image into RGB from BGR
- Finally, we use the *drawContours()* function. The first argument is the image we want to draw on. The second is the contours we found in the last function. The 3rd is -1, to say that we want all contours to be drawn. The fourth is the color, and the last is the thickness.

Pseudo Code

Pseudo Code For Mission 1 To 4 (Base Station)

```
 function PROCESS_DRONE_FEED
     data_dir ← "/path/to/dataset"
    transform ← COMPOSE_TRANSFORM(RESIZE((640, 640)), TO_TENSOR(),
NORMALIZE([0.485, 0.456, 0.406], [0.229, 0.224, 0.225]))
     augmentation_transform ← COMPOSE_TRANSFORM(RANDOM_ROTATION(90),
RANDOM_HORIZONTAL_FLIP(), TO_TENSOR(), NORMALIZE([0.485, 0.456, 0.406], [0.229,
0.224, 0.225]))
    dataset ← LOAD_DATASET(data_dir, transform, augmentation_transform)
     dataloader ← DATALOADER(dataset, batch_size=32, shuffle=True)
          model ← YOLOv9(pretrained=True).cuda()
     optimizer ← Adam(model.parameters(), lr=0.001, betas=(0.9, 0.999))
    scheduler ← StepLR(optimizer, step_size=100, gamma=0.1)
     criterion ← OBJECT_DETECTION_LOSS()
     num_epochs ← 500
     for epoch in range(num_epochs) do
         model.train()
         running_loss ← 0.0
        for inputs, labels in dataloader do
             inputs, labels ← CUDA(inputs, labels)
            optimizer.zero_grad()
            outputs ← model(inputs)
            loss ← criterion(outputs, labels)
             loss.backward()
             optimizer.step()
             running_loss += loss.item()
         end for
         scheduler.step()
        PRINT("Epoch [" + (epoch + 1) + "/" + num_epochs + "], Loss: " +
(running_loss / LENGTH(dataloader)))
   end for
     PRINT("Finished Training")
     tracker ← CREATE_OBJECT_TRACKER()
     START_VIDEO_FEED("drone_fpv_camera")
    while RECEIVING_VIDEO_FEED() do
         frame ← GET_VIDEO_FRAME()
         detections ← model.detect(frame)
        timestamp ← GET_TIMESTAMP()
         tracked_objects ← tracker.update(detections)
         for tracked_object in tracked_objects do
             detection, is_new ← tracked_object
             object_type ← DETERMINE_OBJECT_TYPE(detection)
            if object_type == "HOTSPOT" or object_type == "OBJECT" then
                 SAVE_FRAME_WITH_TIMESTAMP(frame, timestamp, detection)
              if is_new then
                   INCREMENT_COUNTER(object_type, detection)
                    LOG_TO_EXCEL(timestamp, detection, object_type)
               end if
               DISPLAY_ON_LIVE_FEED(frame, detection, object_type)
            end if
```

```
        end for
    end while

     STOP_VIDEO_FEED()
 end function
function OBJECT_DETECTION_LOSS
     # Custom loss function for object detection combining classification and
localization losses
     return CUSTOM_LOSS_FUNCTION()
 end function
 function DETERMINE_OBJECT_TYPE(detection)
     # Logic to classify the detected object as a hotspot or other object
     if detection.class_id in HOTSPOT_CLASSES then
         return "HOTSPOT"
     else
        return "OBJECT"
     end if
 end function
 function SAVE_FRAME_WITH_TIMESTAMP(frame, timestamp, detection)
  # Save the frame with overlayed detection and timestamp
     SAVE_IMAGE_WITH_TIMESTAMP(frame, timestamp, detection)
 end function
 function INCREMENT_COUNTER(object_type, detection)
    # Increment the counter for the detected object type
     if object_type == "HOTSPOT" then
        HOTSPOT_COUNTER += 1
    else
        OBJECT_COUNTER += 1
    end if
 end function
 function LOG_TO_EXCEL(timestamp, detection, object_type)
    # Log the detection details into an Excel file
     WRITE_TO_EXCEL(timestamp, detection, object_type)
 end function
 function DISPLAY_ON_LIVE_FEED(frame, detection, object_type)
     # Display the detection on the live feed with a counter and timestamp
    OVERLAY_DETECTION_ON_FEED(frame, detection, object_type)
 end function
 function CREATE_OBJECT_TRACKER
     # Create and return an object tracker instance
     return OBJECT_TRACKER()
 end function
```

8. REFERENCES

Websites visited for research:

- ➢ Thrust calculation
- ❖ Aircraft Propeller Theory | AeroToolbox
- ❖ https://quadcopterproject.wordpress.com/static-thrust-calculation/
- ➢ Quadcopter Kinematics
  Drone Quadcopter (dronequadcopterx.blogspot.com)
- ➢ Quadcopter Dynamics
  Quadcopter Dynamics and Simulation - Andrew Gibiansky
- ➢ Flight Time Calculation
  https://www.grepow.com/blog/how-to-calculate-fpv-drone-flight-time.html
- ➢ C rating and battery discharging related calculation
  https://www.power-sonic.com/wp-content/uploads/2021/02/What-is-a-battery-C-rating.pdf
- ➢ Efficiency related calculation
  https://www.tytorobotics.com/blogs/articles/how-to-measure-brushless-motor-and-propeller-efficiency#:~:text=BLDC%20Motor%20Calculations&text=Electrical%20power%20(Watts)%20%3D%20Voltage,Efficiency%20%3D%20Mechanical%20power%20%2F%20Electrical%20power
- ➢ Datasheet
- ❖ https://pdf1.alldatasheet.com/datasheet-pdf/view/1132069/BOSCH/BMP280.html
- ❖ https://robu.in/wp-content/uploads/2021/01/AHT10.pdf
- ❖ https://x9gps.com/Tools/LT30TM/UBLOX_Int_GPS/NEO-M8N-FW3_HIM_(UBX-15029985).pdf
- ❖ https://pdf.lkrelec.com/366208.pdf?_gl=1*1or8o5i*_ga*MTIzNDcyOTk2Mi4xNjg1Mjc3NTI3*_ga_W60L5JM4RR*MTY4NTI3NzUyNi4xLjAuMTY4NTI3NzUyNi42MC4wLjA
- ❖ https://robu.in/wp-content/uploads/2017/10/45573-5010-750KV.png

- ➢ Websites visited for research
- ❖ nasa.gov
- ❖ byjus.com
- ❖ wikipedia.org