Flipkart GRiD 2.0

# Autonomous Indoor Drone - Round 3 Template

**Team Name: RuntimeTerror**

**Institute Name: Indian Institute of Technology, Kanpur**

# Team Member Details

Experience of each team member - Any rewards, accolades achieved in robotics/ drones

**International Micro Aerial Vehicle Competition (IMAV), Madrid, Spain 2019 (imav2019.org)**

- *Participated by : All members*
- Amongst the **Top 15** teams selected globally for the Outdoor Challenge
- Developed a fleet of custom-made MAVs capable of detecting mailboxes and delivering packages into them across an area of 30,000 sq. m

**D.R.D.O.'s SASE UAV Fleet Challenge (Inter IIT TechMeet 8.0), IIT Roorkee**

- *Participated by : All members*
- Secured the **1st Position** and the Gold Medal, with a perfect score of 400/400
- Developed a fleet of MAVs capable of collaboratively surveying a 1600 sq. m. grass field to locate green-coloured boxes of interest

# Technical Specifications

| Parameter | Values |
|---|---|
| Type | Hexacopter |
| Frame Material | PA66 & 30GF High strength plastic |
| Diagonal Wheelbase | 550 mm |
| Weight (excluding payload) | 2.2 Kg |
| Battery | 6000 mAh LiPo 4S |
| Flight Controller | Pixhawk Cube |
| Flight Time | 4-6 min |
| Computational unit | Odroid H2+ |
| Payload capacity | 2.7 kg |

# Aerodynamic & Payload Calculations

| Component | Weight(gm) |
|---|---|
| Frame | 478 |
| Motors | 384 |
| ESC | 160 |
| Battery | 623 |
| Compt. Unit | 320 |
| Flight Controller | 40 |
| Camera | 30 |
| Propellers | 30 |
| Other | 100 |
| **Total** | **2165** |

Thrust per motor (using 9” propellers) = 880 gm

*From Motor datasheet*

Total Thrust (from 6 motors) = 5280 gm

Total Effective Thrust = 5000 gm (5% thrust loss)

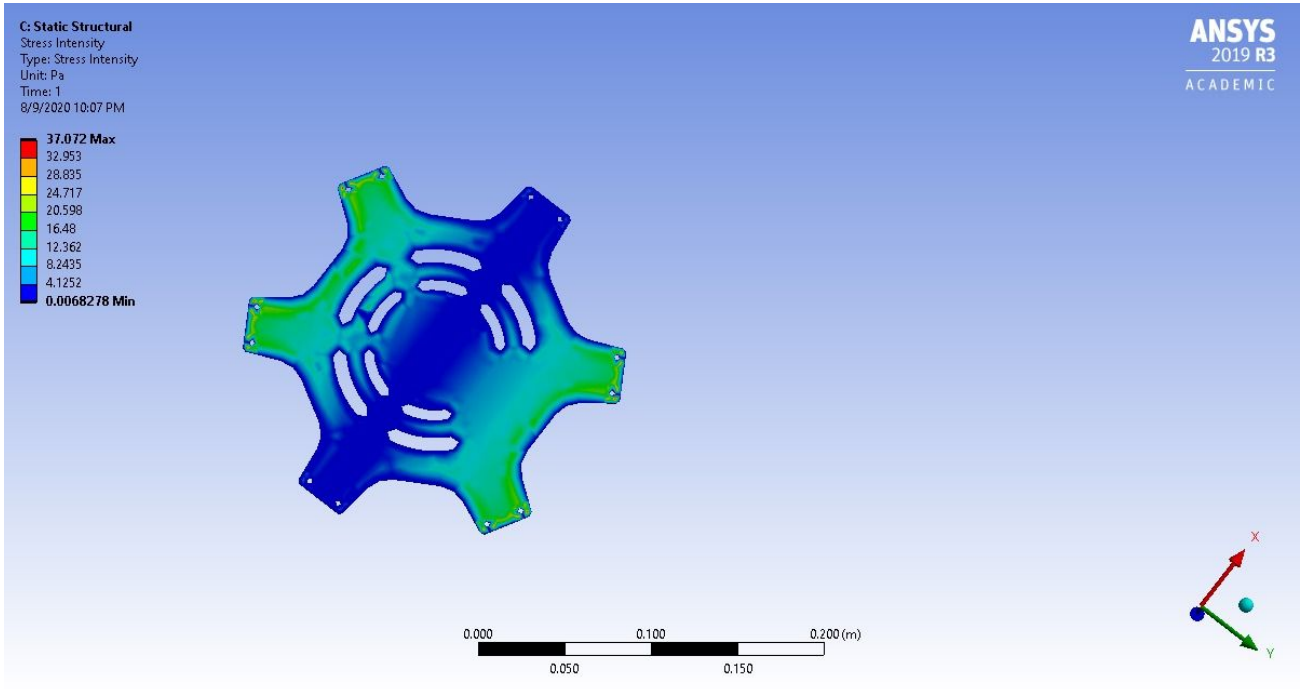
Total Current draw from motors = 65 Amps

*From Motor datasheet*

**Payload = 2000-3000 gm**

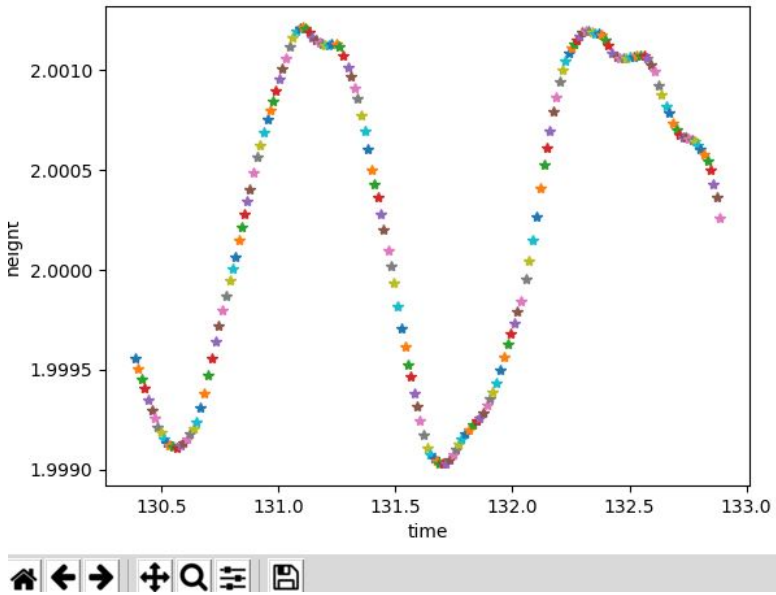
**Flight time = 4-6 min**

# Structural Analysis of Base Plate

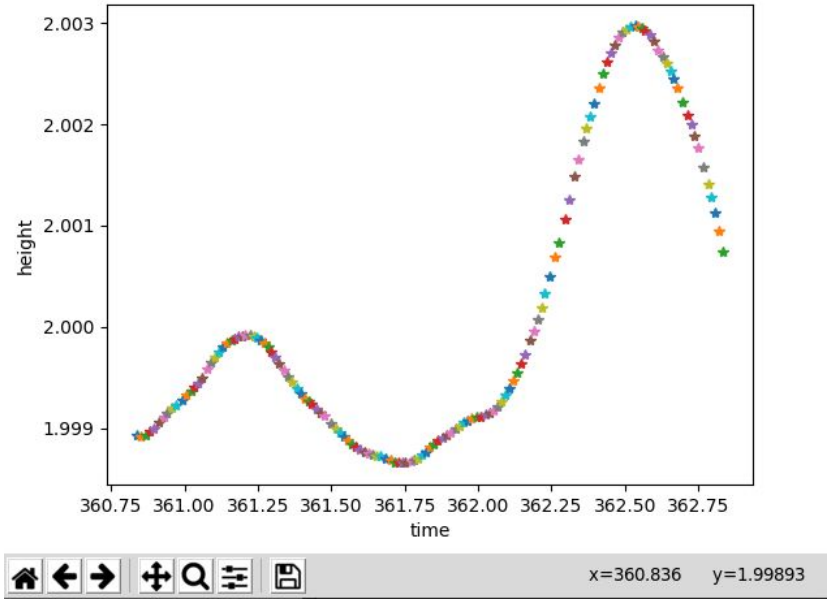


# Stability Analysis (with & without payload)
# Controller z response to payload disturbance

## Hovering & Zero Payload

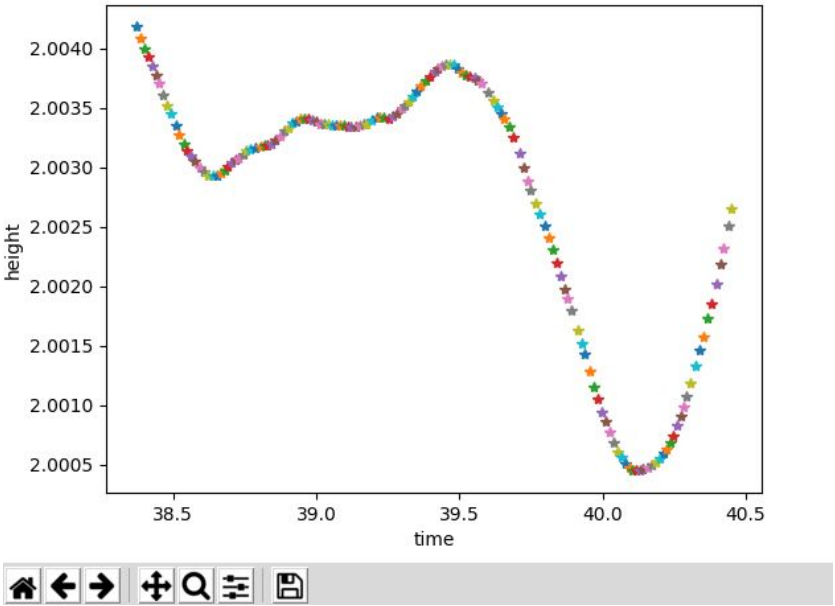



## Hovering & 3kg Payload

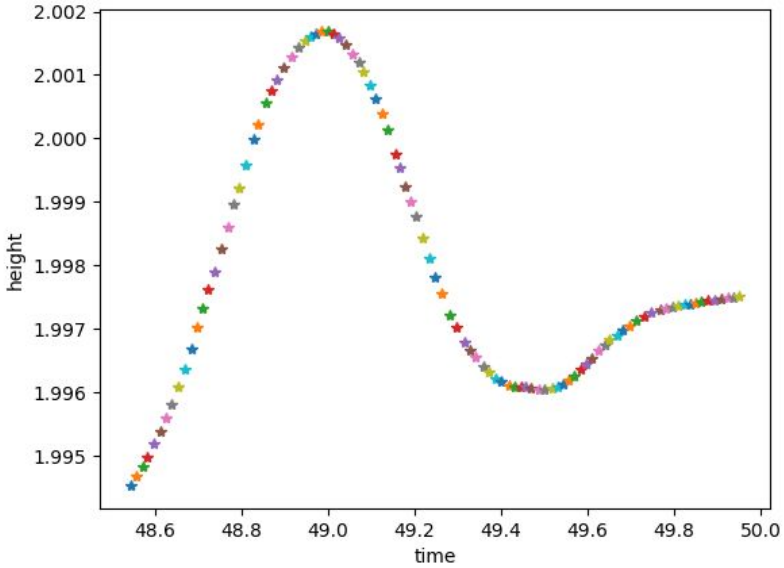


# Stability Analysis (Contd.)

## Movement & 3kg Payload




## Movement & Zero Payload



# CAD File with Payload & Drone Components

Base Frame CAD:

https://drive.google.com/drive/folders/1T4WBn6gRzy-3VtYS-BZAM_MWKNoz1Wb8?usp=sharing

The presence of other hardware components is not required for simulation, even so the relevant CAD models are available online.

The Payload is assumed to be a simple cardboard box with weights inside. As such it is modelled as a simple cuboid of size 20cm x 20cm x 5cm with overall mass of 3 kg. It'll be attached to the MAV using a simple hook so as to minimize additional weight.

# Autonomous Flight Algorithm Details

i) Take off and landing logic

- The MAV takes off vertically & attains a height of 3-4m (so that it is within the centres of the frames) as soon as the launch command is received from the user / pilot / ground control station.
- The MAV maintains a count of the number of gates it has passed, and also tracks the distance it has covered along the aisle. Once these variables reach the expected values, the mission terminates.

ii) Flight algorithm

- The Snake Gate Detector provides the MAV with the setpoints that it needs to travel to
- The MPC handles the motion of the MAV to each setpoint.
- The Localization module provides the odometry data with respect to which each setpoint is defined.
- The Avoidance module is to ensure that the MAV does not crash into any obstacles at all.
- All the above modules run on the onboard computer
- The flight controller provides the interface between the onboard computer and the actual hardware

# Autonomous Flight Algorithm Details (Contd.)

iii) Trip management

- The MAV flies at a nominal speed of 1m/s, hence covering the aisle (~15m) in at most a minute. Keeping this in mind the battery selected is optimized with respect to weight & budget constraints.
- All the separate modules - Controller, Detection, Localization and Avoidance are tied in together with the help of a simple finite state machine (FSM)
- The FSM is written using the Boost MSM C++ Library, and it controls the MAVs transitions across various mission states which are defined as - Takeoff, GoToGate, PassThroughGate, Land.
- The transitions are defined via a state transition diagram which models the entire basic logic of the mission - takeoff, pass through the 15 gates, and land.
- The FSM is also run on the onboard computer and serves as the main integrator of all the distinct modules.

# Frame Detection Algorithm Details

1. Using the Snake Gate Detection Algorithm for locating Frame Corners.

```
1: procedure SnakeGateDetection(image)
2:     for i = 1:maxSample do
3:         P0 = randomPoint()
4:         if isTargetColor(P0,image) then
5:             [P1,P2] = searchUpDown(P0,image)
6:             if ||P1 − P2|| > σL then
7:                 P3 = searchLeftRight(P1,image)
8:                 P4 = searchLeftRight(P2,image)
9:                 if ||P1 − P3|| > σL OR ||P2 − P4|| > σL then
10:                    [S1,S2,S3,S4] = findMinimalSquare(P1,P2,P3,P4)
11:                    detectedGate = refineCorner(S1,S2,S3,S4)
12:                    if checkColorFitness(detectedGate) > σcf then
13:                        return detectedGate
14:                    end if
15:                end if
16:            end if
17:        end if
18:    end for
19: end procedure
```

**Algorithm 1**: Snake Gate Detection

```
1: procedure searchUpandDown(P0, image)
2:     P1 = P0, P2 = P0, done = false
3:     while !done do
4:         if isTargetColor(P1.x, P1.y − 1) then
5:             P1.y = P1.y − 1
6:         else if isTargetColor(P1.x − 1, P1.y − 1) then
7:             P1.x = P1.x − 1
8:             P1.y = P1.y − 1
9:         else if isTargetColor(P1.x + 1, P1.y − 1) then
10:            P1.x = P1.x + 1
11:            P1.y = P1.y − 1
12:        else
13:            done = true
14:        end if
15:    end while
16:    done = false
17:    while !done do
18:        if isTargetColor(P2.x, P1.y + 1) then
19:            P2.y = P2.y + 1
20:        else if isTargetColor(P2.x − 1, P1.y + 1) then
21:            P2.x = P2.x − 1
22:            P2.y = P2.y + 1
23:        else if isTargetColor(P1.x + 1, P1.y + 1) then
24:            P2.x = P2.x + 1
25:            P2.y = P2.y + 1
26:        else
27:            done = true
28:        end if
29:    end while
30:    return P1, P2
31: end procedure
```

**Algorithm 2:** SearchUpDown Function. SearchLeftRight is very similar.

# Gazebo Simulation & Assumptions

- Adequate and consistent lighting - such that the colour of the gates as seen by the camera does not vary a lot in the HSV colorspace
- No external disturbances - Although the MPC (Model Predictive Controller) that our solution uses is robust enough, the current simulation assumes the absence of any physical disturbances such as winds.
- Distance between each gate is assumed to be 1m as per FAQs provided. The horizontal position of each gate is random, our software does not depend on these values in any way.
- The hexacopter that we are using is purely representative (it is the AscTec Firefly) and does not represent the actual hexacopter that we will be building.

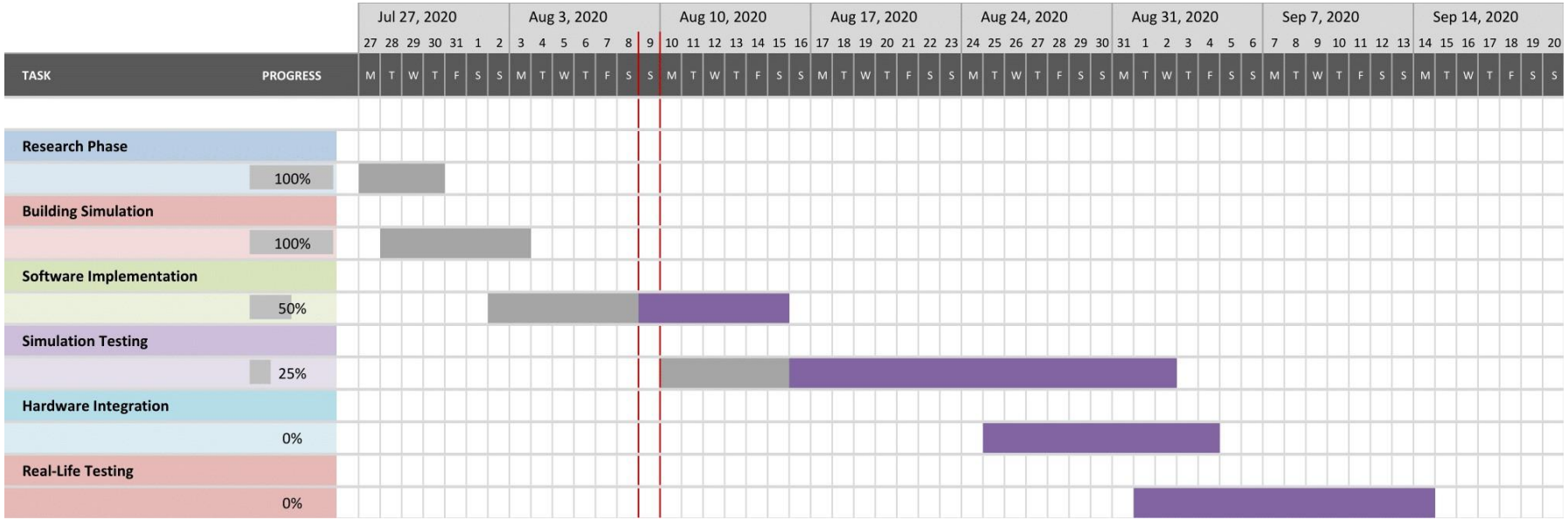
# Component Details

| Component | Specifications |
|---|---|
| Frame | DJI F550 frame: Lightweight frame made of Polyamide-Nylon Ultrastrength Material with integrated PCB Wiring. |
| Motors | KDE2315XF-965: DJI series, Industrial grade, highly efficient Brushless Motors. |
| ESC | KDEXF-UAS20LV: UAS series with refresh rate of 2000 Hz & Aluminium heatsink. |
| Propellers | 9” propellers made of GFR Polymer composite, having low vibration profile. |
| Battery | Turnigy nano-tech 4 cell LiPo battery with advanced LiCo nano-technology, having power density of 7.5 kw/kg. |
| Computational Unit | ODROID-H2+ with Intel Quad-core processor J4115, power consumption of 14W. |
| Camera | oCam: 5MP USB 3.0 Rolling Shutter Camera, with FOV of 65 degrees. |
| Flight controller | Pixhawk Cube autopilot: 32bit STM32F427 Cortex-M4F core with FPU and 3 sets of IMU. |

# Component Sources, Price & Total Cost

| Component | Unit | Price(per unit) | Source | Total |
|---|---|---|---|---|
| Frame | 1 | $26.55 | Robu | $26.55 |
| Motors | 6 | $60.95 | KDE | $365.7 |
| ESC | 6 | $40.95 | KDE | $245.7 |
| Propellers | 6 | $5.69 | HK | $34.14 |
| Battery | 1 | $66.43 | HK | $66.43 |
| Computational Unit | 1 | $119 | HardKernel | $119 |
| Camera | 1 | $96 | HardKernel | $96 |
| Flight controller | 1 | $238 | Amazon | $238 |
| ***Please Note: To ensure absolute safety of the solution, quality of motors and ESCs cannot be compromised.*** | | | | $1192 |

# Execution plan with timelines



# Current Progress

- Implemented robust Model Predictive Control (MPC) and pose command execution
- Implemented accurate localization using visual features from a monocular camera using ORB for feature detection, KF based tracking & DBoW2 for loop closing.
- Currently working on densification of the keypoint map generated during localization for robust obstacle avoidance based on Potential Field methods
- Implemented Frame Detection using the Snake Gate Detection Algorithm[3] , integrated with ROS
- Built the world and environment in Gazebo according to the provided specifications
- Will start with integrated testing using actual CAD model of frame within this week.

# References


1. Li, Shuo et al. “Visual Model-predictive Localization for Computationally Efficient Autonomous Racing of a 72-gram Drone.” J. Field Robotics 37 (2020): 667-692.
2. Foehn, Philipp et al. “AlphaPilot: Autonomous Drone Racing.” ArXiv abs/2005.12813 (2020): n. Pag.
3. Li, Shuo et al. “Autonomous drone race: A computationally efficient vision-based navigation and control strategy.” ArXiv abs/1809.05958 (2018): n. Pag.
4. Delmerico, Jeffrey A. and Davide Scaramuzza. “A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots.” 2018 IEEE International Conference on Robotics and Automation (ICRA) (2018): 2502-2509.
5. Zohaib, Muhammad et al. “Control Strategies for Mobile Robot With Obstacle Avoidance.” ArXiv abs/1306.1144 (2013): n. pag.
6. Ribeiro, M. Isabel. “Obstacle Avoidance.” (2005).
7. Tanja Baumann: Obstacle Avoidance for Drones Using a 3DVFH* Algorithm, ETHZ
8. Rosinol, Antoni. “Densifying Sparse VIO: a Mesh-based approach using Structural Regularities.” (2018)
9. Mur-Artal, Raul and Juan D. Tardós. “ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras.” IEEE Transactions on Robotics 33 (2017): 1255-1262
10. Kamel, Mina et al. “Model Predictive Control for Trajectory Tracking of Unmanned Aerial Vehicles Using Robot Operating System.” (2017)

# Links to submitted videos

Frame detection demonstration:
https://drive.google.com/file/d/1h58jHupA6QD9HQjr3l44OIPyNi1z3J6b/view?usp=sharing

Simulation demo:
https://drive.google.com/file/d/1uqsjRm-5fgd7ozKMWInTrjfmXu8z50C-/view?usp=sharing