DEPARTMENT OF AEROSPACE ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS
CHENNAI – 600036

Development of a Vision Guided Aerially Launched Short Range Missile

A Thesis

Submitted by

APPILI VAMSI KRISHNA

For the award of the degree

Of

MASTER OF TECHNOLOGY

May 2022

DEPARTMENT OF AEROSPACE ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS
CHENNAI – 600036

Development of a Vision Guided Aerially Launched Short Range Missile

A Thesis

Submitted by

APPILI VAMSI KRISHNA

For the award of the degree

Of

MASTER OF TECHNOLOGY

May 2022

There’s power in your purpose!

– Joe Braxton

I dedicate this thesis to my parents, teachers for inspiring me, guiding me, thus, shaping a beautiful life to me

THESIS CERTIFICATE

This is to undertake that the Thesis titled **DEVELOPMENT OF A VISION GUIDED AERIALLY LAUNCHED SHORT RANGE MISSILE**, submitted by me to the Indian Institute of Technology Madras, for the award of **Master of Technology**, is a bona fide record of the research work done by me under the supervision of **Dr. S R Chakravarthy**. The contents of this Thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Chennai 600036 **Appili Vamsi Krishna**

Date: May 2022

Dr. Chakravarthy S R
Research Guide
Professor
Department of Aerospace Engineering
IIT Madras

Dr. Murthy H S N
Head of Department
Department of Aerospace Engineering
IIT Madras

ACKNOWLEDGEMENTS

I would like to express my gratitude towards my guide Prof.S.R.Chakravarthy. It was a great pleasure learning and to work under his guidance and support which were invaluable for me in completing the MTech Project. The discussions with him during the course of this project work have been extremely helpful in understanding the complexities of systems in precision-guided munition(PGM), which helped me in design of missile architecture, subsystems, intuitive simulations, innovative fabrication. His ingenious conversations and ideas have continuously inspired me at every step to achieve good progress in the ultimate goal in our attempt to reduce collateral damage in battle field operations.

I am thankful to our Department Head, Prof. H.S.N.Murthy and Professors of Aerospace Engineering Department for sharing their valuable knowledge.

I would like to mention the selfless help, advice and knowledge support lent to me by my friends, Bapu Gayathri Maddisetty, Sugeerthi Gurumoorthy, Madhavan, Adhith M, Aadhithya, Robin, Siva Vignesh, Rahul, Santhosh, Ethihas, Emmanual Jose.

I would like to thank my parents for their unwavering support during the pandemic.

ABSTRACT

When the knowledge of Aerospace engineering proved its use in possibilities of enhancement in defense and security, providing supremacy for a nation in the global community, with ensuring land safety and country sovereignty had lead to the interest over years in development of weapons like UCAVs, Missiles.

Now a days, there is high tension in every country if a country announces their advanced defense capabilities in terms of missile technology, namely nuclear missiles, ICBMs, Interceptor systems. Missiles are highly used in today war (or) to prove strength, so other countries respects our interests by treating their equals.

India is exporting Brahmos missiles to Vietnam, Philippines, these missiles have unit cost in order of Millions of USD. Since each missile is so costly, we cant afford to lose it due to interception by enemy during operation, Thus missile wont be used until it is an inevitable situation

The Idea of the thesis is to develop a low cost air to ground missile which is capable of attacking target that is 2 km far, when launched via an aerial platform (fixed wing UCAV or Hexarotor UCAV) from an altitude of 500 to 1000 meters from mean sea level. Aim is to launch a swarm of missiles over a target autonomously, So, we need not worry about any interceptors in mid-way. Also, we can save many of our valuable soldier lives, since we use autonomous, human-pilot-less systems, thus reducing majority of the possible collateral damage during attack initiation. This technology is need of the hour since India received potential suside bomber threats from Al-Qaeda in the Subcontinent (AQIS) on June 6, 2022. This missile is portable and is capable of launch from the hand launcher.

This thesis details the design of rocket motor and 4 versions of missile with their archetecture, subsystem standards and specifications and overall development of Precision Guided Munition (PGM) Missile 5 kg

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

AC	Aerodynamic Centre.
Al	Aluminium.
AOA	Angle of attack.
AP	Ammonium perchlorate.
ASCII	American Standard Code For Information Exchange.
ASI	Aircraft Stores Interface.
BC	Bus Controllet.
BEM	Balastic Evaluation Motor.
BIT	Built-in-Test.
BLOSOA	Backup-Line-of-sight-Obstacle-Avoidance-Trajectory.
CEA	Chemical Equilibrium Analysis.
CEP	Circular Error Probability.
COTS	Commercially Off the Shelf.
CPU	Central Processing Unit.
DARPA	Defense Advanced Research Projects Agency.
DOA	Dioctyl adipate.
DOF	Degree of Freedom.
FCC	Flight Control Computer.
FCS	Flight Control System.
FPS	Frame Per Second.
GPS	Global Positioning System.

HTPB	Hydroxyl-Terminated PolyButadiene.
I2C	Inter Integrated Circuit.
IITM	Indian Institute of Technology, Madras.
IMU	Inertial Measurement Init.
IPDI	Isophorone diisocyanate.
Lat-Long	Latitude and Longitude.
LOS-OA	Line of Sight Obstacle Avoidance.
LOS-OA-G	Line of Sight Obstacle Avoidance Guided.
MC	Mission Computer.
NCCRD	National Centre for Combustion Research and Development.
ORB	Oriented FAST and Rotated BRIEF.
PCB	Printed Circuit Board.
RAM	Random Access Memory.
ROM	Read Only Memory.
RT	Remote Terminal.
SCADA	Supervisory Control and Data Acquisition.
SCB	The Single Board Computer.
SIFT	Scale-invariant feature transform.
SMP	Stores Management Processor.
SMS	Stores Management System.
SRM	Solid Rocket Motor.
TCP/IP	transmission Control Protocol / Internet protocol.
TTL	Transistor Transistor Logic.
UAV	Unmanned aerial Vehicle.

UCAV	Unmanned Combat Aerial Vehicle.
VTOL	Vertical Take-Off and Landing.
WP	Warhead Payload.

NOTATION

V	Free stream Velocity
g	Acceleration due to gravity
ϕ	Roll angle
θ	Pitch angle
C^*	Propellant Characteristics Velocity
C_D	Coefficient of Drag
I_{sp}	Specific Impulse
α	Angle of attack
β	Side slip angle
γ	Flight path angle
ψ	Yaw angle
ρ	density of substance
C_L	Coefficient of Lift
I_{xx}	Mass moment of inertia
t_b	Propellant burnout time
C_M	Coefficient of pitching moment
r_0	Propellant burn rate
L	Rolling moment
M	Pitching moment

N	Yaw moment
P	Pressure
p	Roll rate
q	Pitch rate
r	Yaw rate

CHAPTER 1

INTRODUCTION

Defense Advanced Research Projects Agency (DARPA) is working on utilising Unmanned Combat Aerial Vehicles (UCAV)s for front-line defense where there is high chance of collateral damage, Refer [1]. The target application of these missions was to attack the enemy location without loss of our own soldiers. DARPA's X-45 program produced a UAV with tactical aircraft performance, which demonstrated formation flight, manned/unmanned teaming, and a coordinated strike against simulated radars and missile launchers. X-45 could be seen as the proof-of-concept vehicle for the unmanned combat air vehicle (UCAV) concept. These systems developed by DARPA is quite expensive, but efficient in their operation. There is need of indigenous development of classified payloads, for the specific UCAVs that can be developed indigeniously. This idea has been originated in 2019, by Squadron Leader Ravi Raj SR, and began the development of project with the core idea of cost-effective, lessen collateral damage, equipment that is fully indigenous designed & developed.

1.1 PROBLEM STATEMENT

Design a missile of Vision Guided Aerially Launched (from altitude of 500 meter to 1000 meter) Short Range Missile (with range of 2 km) with overall structural mass of 5 kg, which includes the payload, propellant, ignition system, battery, electrical equipment, gimbal system, structural components. Also, Design the accessory systems ie., ground launcher, hand launcher, weapon release system.

1.2 AIM

To indigenously develop, cost effective, 5 kg short range missile with payload of minimum 2 kg, which can be attached to UCAV such as indigenously developed VTOL drone or Hexa-rotar UAV to carry out tatical attack operations in hazardous area, and reduce collateral damage.

1.3 OBJECTIVE

1. To Design the architecture of possible version of missile, with input-output pair of execution of mission
2. To Design and Develop optimised Solid-Rocket Engine with on-par sub-systems of missile
3. To Design and analyse accessory systems of missile launch
4. Stability and Trajectory analysis, with a study of further improvement on accuracy using data obtained from launches using deep learning and machine learning models

1.4 PREVIOUS WORK

* Squadron leader Ravi Raj during 2019-20 : Solid Rocket motor for the range of 1500 m with $\Delta\theta < 18$ deg with thrust of 1476 N for 1.5 s burnout is designed with propellant composition shown in table 1.1, It is referred as Solid Rocket Motor (SRM) Version 1.0. It uses 68 mm Rocket Warhead manufactured by Indian Ordnance Factory, It is referred in the project as Weapon Payload (WP) version 1.0. This SRM V1.0 is attached to WP V1.0 and launched aerially from UCAV from alitude of 1000 m. [3]
* Niharika during 2020-21 : Literature Review on missile guidance system is studied, using OpenCV tracking implimentation algorithems like BOOSTING, MIL, KCF, CSRT, MedianFlow, TLD, MOSSE. [4]
* Squadron leader Piyush Raj during 2019-20 : Integration of guided weapon on indigeously developed UAV is carried out, with design of Stores Management System (SMS), Avionic Bus Architecture Design with RS-485 Bus using MODBUS ASCII Protocol, with finalisation on Safty Critical system system integration. [2]

Ingredient	function	**Wt**%
HTPB	Fuel, Binder	10%
DOA	Plasticizer	3%
Ambilink	Cross-linking agent to $\uparrow$ tensile property	0.12%
PBNA	Improves aging characteristics	0.1%
TDI	Curing agent	0.73%
Al_2O_3	Fuel, Catalyst	0.1%
AP(course)	Oxidizer	51.5%
AP(Fine)	Oxidizer	34.4%
ACR	Modify Burn Rate	0.05%

Table 1.1: SRP Version 1.0 : Composite Propellant Composition

1.5 CURRENT RESEARCH AND WORKFLOW

1. Design of Solid Rocket motor V2.0 for the range of 2000 m with $\Delta\theta < 45$ deg with thrust of 1000 N for 2.5 s burnout

2. 2500 gram Warhead payload profile, named as WP V2.0 and 2100 gram Warhead payload profile, named as WP V3.0

3. Optimising structural weight without compromising on performance : Carbon composite material selected for Missile frame, to provide same strength as that of aluminum alloy, with 50 percent less weight than aluminum alloy

4. Hydrotest for the material selected to validate the maximum pressure Rocket Motor case can handle

5. Selection of battery and rocket motor igniter

6. Trust calculation using the Pressure data obtained in firing of propellant using BEM

7. Canard Design

8. Canard mechanism design

9. Design of custom 2 axis gimbal for missile
10. Stability analysis of Missile
11. Design of Ground Launcher, Hand Launcher, Weapon Release System A and Weapon Release System B
12. Study on FPGA Architecture
13. Study on Sensors, Single Board Computers (SBC), csrt tracking with pan-tilt servo configuration
14. Trajectory Simulation and Prediction

The target of this project is to develop Vision Guided Aerially Launched Short Range Missile. Since the target missile development have many key variables which are to be in validated certainly to ensure precision, safety in its operation. The version control methodology is adopted, which in each stage gives conclusive data on the key variables tested in the version. Also, each version can be developed as a product with the limitation of version targeted capabilities, after certain iterations in design from the live-field data obtained during validation of certain version missile.

CHAPTER 2

ARCHITECTURE & DESIGN OF SYSTEM

2.1 DEVELOPMENT PROCESS

The missile development needs validation of various key variables, which is possible when we develop the final version through stages[9] (named versions), and achieving each version validates a major key variable that determines the trajectory of final version. [7]

In this chapter, Aerial platform refers to UCAV which is hosting the missile for launch from known lat-long, altitude, attitude.

2.2 VERSION 1

2.2.1 Input

1. Target Latitude - Longitude
2. Latitude, Longitude, Attitude, Altitude of Aerial Launcher

2.2.2 Logic

As we know the initial position and final position of the required trajectory, using Thrust vs Time data, considering drag, gravity,[5] we can calculate the required depression angle, with which the missile is to launched so as to follow the LOS trajectory. This is analogous to the propelled bullet, fired targeting the aim.

2.2.3 Details

S.No	Component	Calculated Mass (in gram)
1	Ignitor, CD Nozzle, Insulation	386
2	Electronics	152
3	Propellant	1200
4	Payload	2500
5	Carbonfiber Frame	519
6	Gimbal Shape	47
Total		4804

Table 2.1: Version 1 : Mass Distribution

1. No Gimbal, No Camera
2. Canards are fixed at zero angle of attack, act like stablisers. (These are placed so as the trajectory, field data collected during testing can be used for validation, and used in development of upcoming versions)
3. This has payload mass of 2500 gram and propellant mass of 1200 gram
4. It has battery and electronics necessary for the ignition, thus, the electronics mass calculated to be 152 gram
5. This Missile Version use SRM Version 2.0, and WP Version 2.0
6. Weapon Release System A (WRS-A) is used here which is estimated about 2.4 kg, thus, altogether mass of missile and WRS-A is about 7.2 kg
7. This is perfect to launch from hexarotor drone (UCAV) with minimum payload capacity of 8 kg

2.2.4 Validated Key Variable

1. This Launch validates Thrust Vs Time graph
2. This Launch validates our trajectory calculation, and the live data collected in each launch can be used for refining & validating our angle of depression required

at launch for the targeted range, using deep learning, thus having developing the trajectory prediction machine learning code

3. Validates Weapon Release System used

4. Validates UCAV Stability during operation of Launch

2.2.5 Output & Expected Trajectory

1. Fire and Forget

2. Target Lock before Launch

3. Launched at required angle of depression, for the targeted range via LOS trajectory

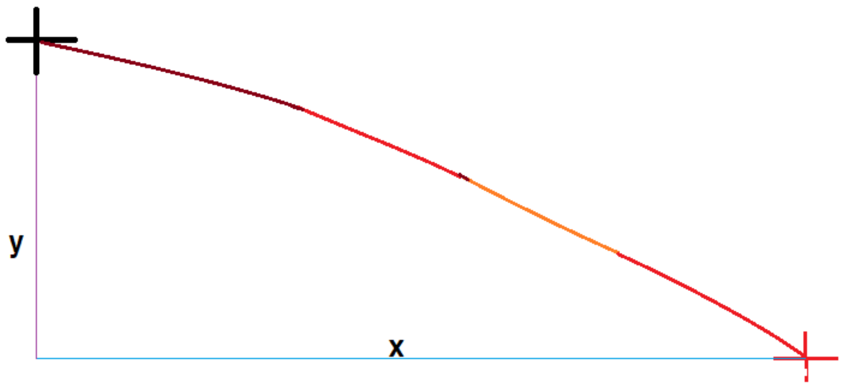


Figure 2.1: Expected LOS Trajectory

2.2.6 Architecture

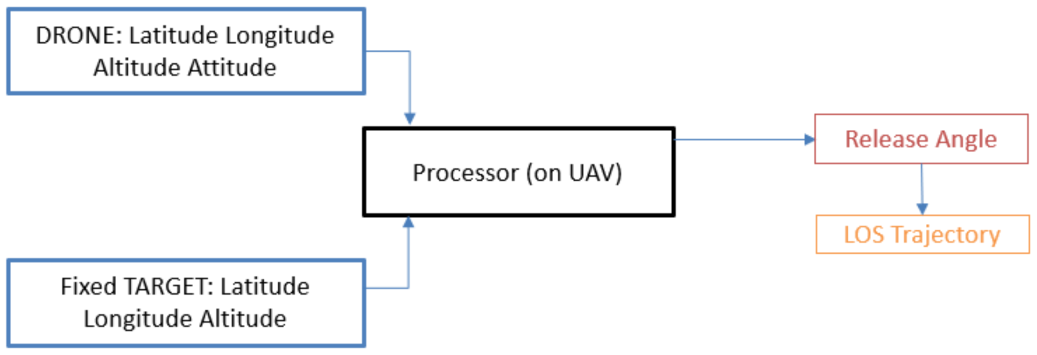

DRONE: Latitude Longitude Altitude Attitude
Processor (on UAV)
Fixed TARGET: Latitude Longitude Altitude
Release Angle
LOS Trajectory

Figure 2.2: Architecture : Version 1

2.3 VERSION 2

2.3.1 Input

1. Target Latitude - Longitude
2. Latitude, Longitude, Attitude, Altitude of Aerial Launcher
3. Obstacle Latitude - Longitude, Height, Trajectory safe height required from obstacle height, Radius of obstacle, Safe Radial distance needed from obstacle

2.3.2 Logic

We know the initial position, final position and the obstacle location on the way, with its height, and respective required safe distances. Thus, with our precise Thrust vs Time data (obtained by cleaning the data using machine learning using live-field data obtained in testing of version 1), considering gravity, drag, canard control force and moment, we can calculate deflection angle required for canard during canard operation for duration of 0.5 second pre and post obstacle, so as to pitch up missile and pitch down missile, so as to escape the obstacle. The initial angle of depression of launch from aerial platform(Hexarotar UAV or fixed wing - VTOL) will be determined such a way that, missile reach the safe distance height coordinates of obstacle and can be able to perform turn maneuver.

Here, our trajectory is determined and pre-set before launch.[6] Thus, the PWM signals needed for the canard over duration of flight is determined before launch, and the data is fed to missile processor. Hence it is a open-loop control system. The trajectory path is said as Line of Sight - Obstacle Avoidance (LOS-OA) Trajectory

2.3.3 Details

S.No	Component	Calculated Mass (in gram)
1	Ignitor, CD Nozzle, Insulation	386
2	Electronics	191
3	Propellant	1200
4	Payload	2500
5	Carbonfiber Frame	519
6	Gimbal Shape	47
7	Servo Mechanism	148
Total		4991

Table 2.2: Version 2 : Mass Distribution

1. No Gimbal, No Camera
2. Servo mechanism attached to canards, and they take precise angle of attack, with accuracy of 0.1 degree by pairing it with our PWM servo controller
3. This has payload mass of 2500 gram and propellant mass of 1200 gram
4. The electronics constitutes of battery, servo controller, raspberry pi constitutes mass of 191 gram
5. This Missile Version use SRM Version 2.0, and WP Version 2.0
6. Weapon Release System A (WRS-A) is used here which is estimated about 2.4 kg, thus, altogether mass of missile and WRS-A is about 7.4 kg
7. This is perfect to launch from hexarotor drone (UCAV) with minimum payload capacity of 8 kg

2.3.4 Validated Key Variable

1. This launch validates the simulated values of aerodynamic properties of canard (C_L vs α & C_D vs α) and helps to re-calibrate using machine learning the data obtained in field tests. This helps to accurately control the trajectory using canards, thus enabling us to go for closed loop feedback control in the next version of missile

2. With the validation of our calculations with real-time data obtained in this version, we can program to extend this version to be able to avoid multiple obstacles in its path.

2.3.5 Output & Expected Trajectory

1. Fire and Forget

2. Target Lock before Launch

3. Launched at required angle of depression, with a pre-set path for the targeted range, by travelling above obstacles via LOS-OA trajectory

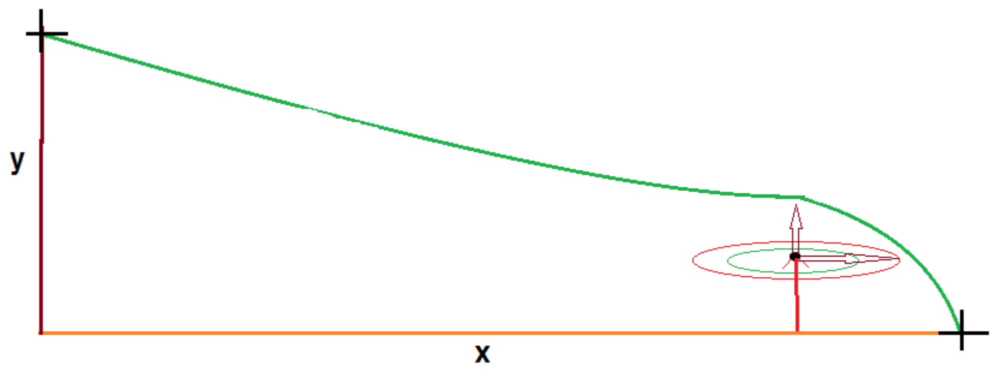


Figure 2.3: Expected LOS-OA Trajectory

2.3.6 Architecture

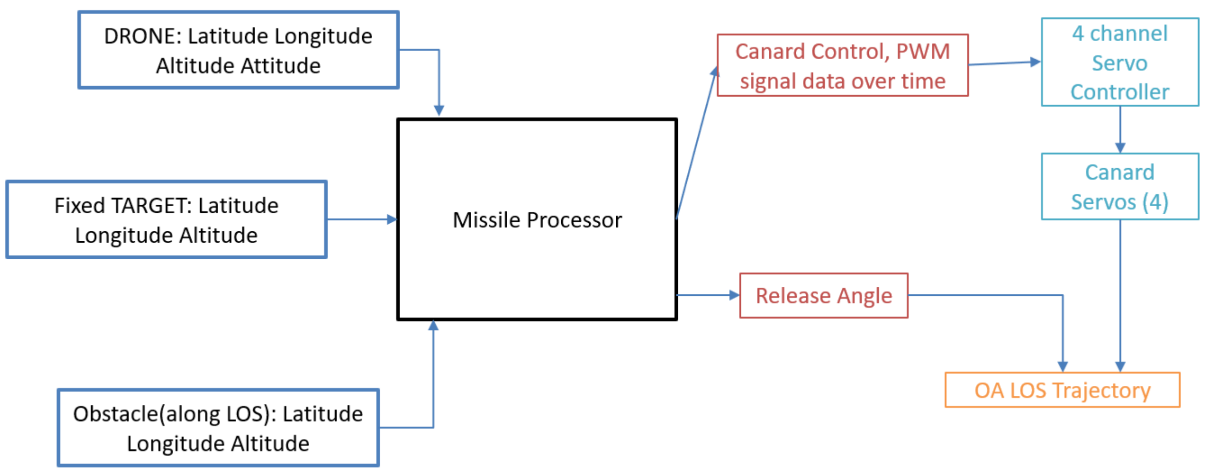

DRONE: Latitude Longitude Altitude Attitude
Fixed TARGET: Latitude Longitude Altitude
Obstacle(along LOS): Latitude Longitude Altitude
Missile Processor
Canard Control, PWM signal data over time
4 channel Servo Controller
Canard Servos (4)
Release Angle
OA LOS Trajectory

Figure 2.4: Architecture : Version 2

2.4 VERSION 3

2.4.1 Input

1. Latitude, Longitude, Attitude, Altitude of Aerial Launcher
2. Obstacles Latitude - Longitude, Height, Trajectory safe height required from obstacle height, Radius of obstacle, Safe Radial distance needed from obstacle
3. Exact location(Lat-Long) of **stationary target** at the time of firing or launch is needed

2.4.2 Logic

We know the stationary target, lat-long, and obstacles in the mid way, we can now provide sofware with minimum "y" above which missile path has to be planned. Here, we consider the refined precise data of instantaneous thrust, drag component, gravity effect, with possible disturbances in freestream wind, we can calculate best feasible trajectory and provide flight-path angle over time required to reach the target to missile processor on-board. This trajectory that is generated is called as Backup-Line-of-sight-Obstacle-Avoidance-Trajectory (BLOSOA Trajectory). [10] The missile is launched with calculated initial angle of depression in the intent to travel in BLOSOA path, however we have a feedback system, an IMU attached to a stablised gimbal (locked in the attitude of target, before launch). The instaneous IMU signal and instaneous PWM signal is fed to Trajectory Processor.[8] This feedback system, validates the current path with intended path, and send PWM signals to canard control surfaces, to achieve closed loop processing of real-time trajectory correction to approach the known stationary target with minimal miss-distance. Thus with use of the closed feedback system, to deal with real-time disturbances, we get into more realistic trajectory called as Line of sight Obstacle Avoidance Guided (LOS-OA-G) Trajectory.

2.4.3 Details

S.No	Component	Calculated Mass (in gram)
1	Ignitor, CD Nozzle, Insulation	386
2	Electronics	468
3	Propellant	1200
4	Payload	2100
5	Carbonfiber Frame	519
6	Gimbal + IMU	66
7	Servo Mechanism	148
Total		4887

Table 2.3: Version 3 : Mass Distribution

1. Gimbal is present, with IMU, without camera module
2. Servo mechanism attached to canards, and they take precise angle of attack, with accuracy of 0.1 degree by pairing it with our PWM servo controller
3. This has payload mass of 2100 gram and propellant mass of 1200 gram
4. The electronics constitutes of battery, servo controller, raspberry pi constitutes mass of 191 gram, and the servo mechanism system adds up to 148 grams

5. This Missile Version use SRM Version 2.0, and WP Version 3.0

6. Weapon Release System A (WRS-A) is used here which is estimated about 2.4 kg, thus, altogether mass of missile and WRS-A is about 7.4 kg

7. This is perfect to launch from hexarotor drone (UCAV) with minimum payload capacity of 8 kg

2.4.4 Validated Key Variable

1. Validation of Trajectory Processor, & efficiency of canards has to be validated

2. Closed feedback loop of Gimbal Stabilization board has to be tested and results are validated

3. Launch with angle of elevation has to be tested with validation

4. The switching from LOS-OA-G Trajectory to BLOSOA Trajectory in the emergency scenario (sensor blackout or) of sensor errors, has to be tested & validated

2.4.5 Output & Expected Trajectory

1. Fire and Forget

2. Target Lock before Launch

3. Launched at required angle of depression, for the targeted range via LOS-OA-G trajectory

4. Since it is closed loop guided, for a stationary target, we can have launch with angle of elevation attack a target at extended range, This will use LOS-OA-G trajectory system in descend phase of trajectory

5. This Version aims at stationary lat-long of target at the time of firing or launch, and also achieves less miss distance compared to previous versions

2.4.6 Architecture

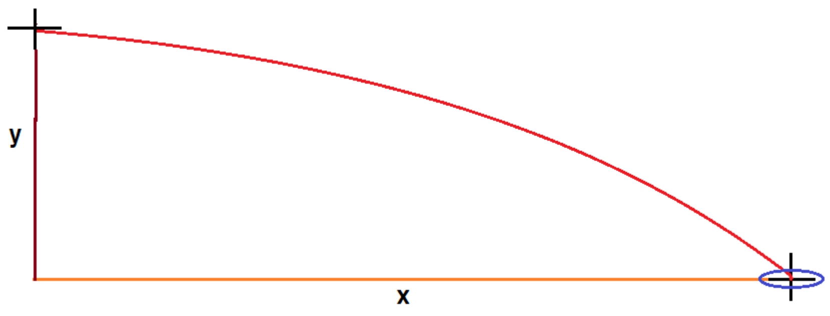


Figure 2.5: Expected LOS-OA-G Trajectory (launched with known angle of depression)

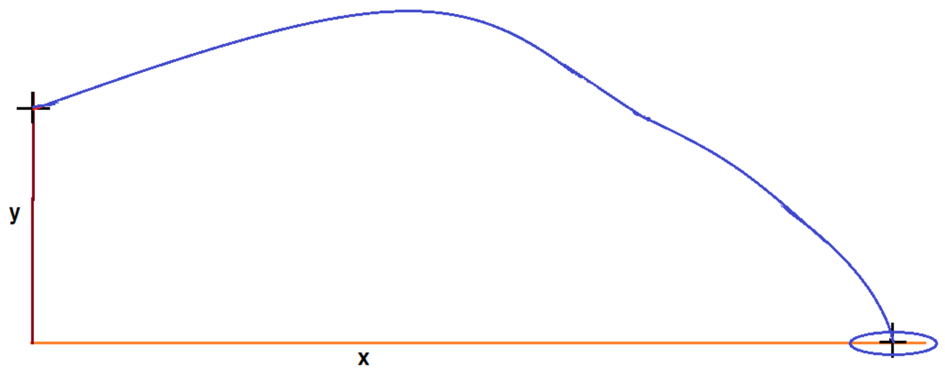


Figure 2.6: Expected LOS-OA-G Trajectory (launched with known angle of elevation)

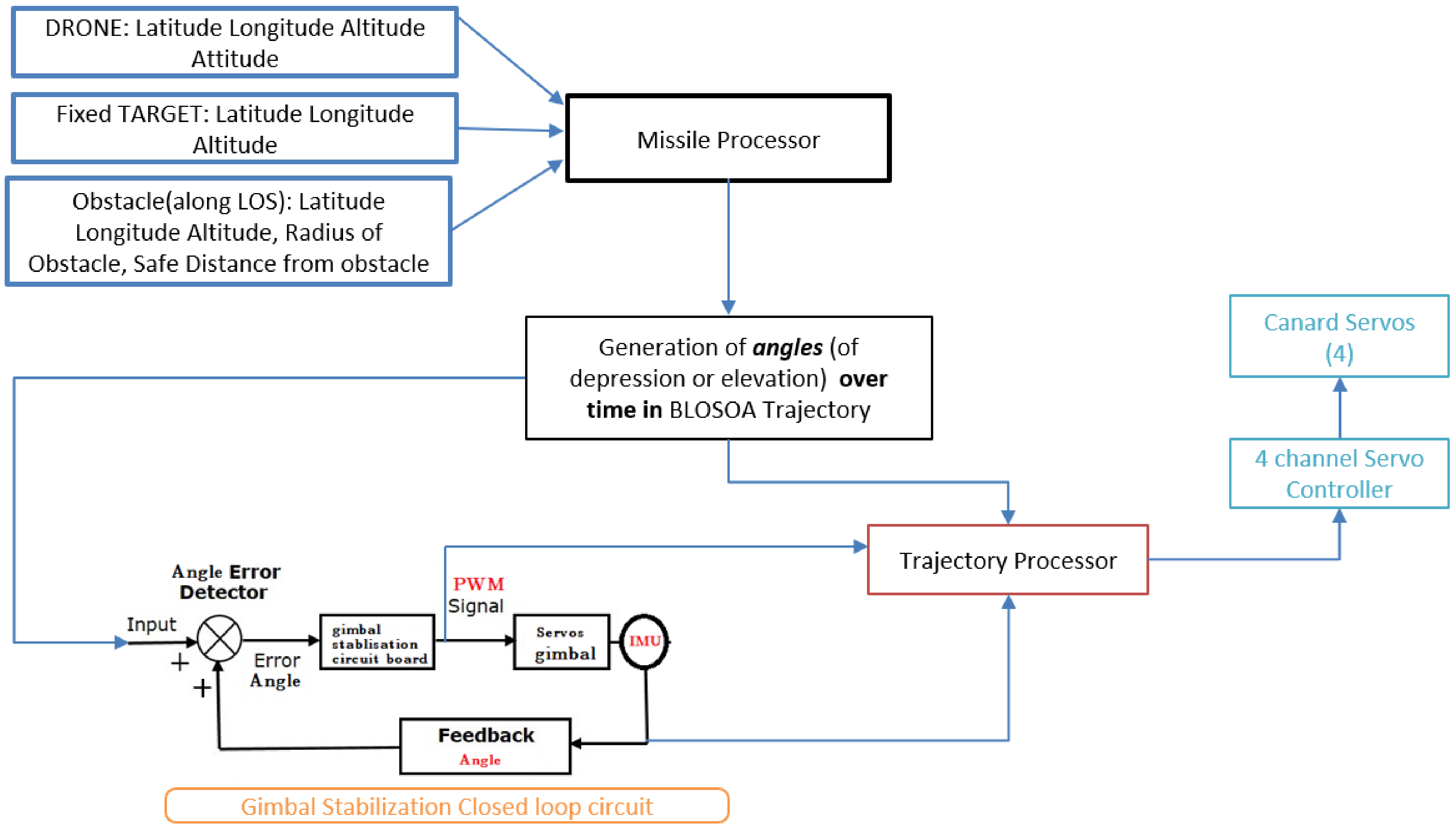

DRONE: Latitude Longitude Altitude Attitude
Fixed TARGET: Latitude Longitude Altitude
Obstacle(along LOS): Latitude Longitude Altitude, Radius of Obstacle, Safe Distance from obstacle
Missile Processor
Generation of ***angles*** (of depression or elevation) **over time in** BLOSOA Trajectory
Canard Servos (4)
4 channel Servo Controller
Trajectory Processor
Angle Error Detector
Input
+
+
Error Angle
gimbal stablisation circuit board
PWM Signal
Servos gimbal
IMU
Feedback Angle
Gimbal Stabilization Closed loop circuit

Figure 2.7: Architecture : Version 3

2.5 VERSION 4

This Version works for both stationary and moving targets.

2.5.1 Input

1. Latitude, Longitude, Attitude, Altitude of Aerial Launcher
2. Obstacle Latitude - Longitude, Height, Trajectory safe height required from obstacle height, Radius of obstacle, Safe Radial distance needed from obstacle
3. To Lock the Target, we need to select target image pixels in the camera view. Also we need to provide exact location(lat-long) of the target at the time of firing

2.5.2 Logic

We know the Initial Lat-long of the target, and its image pixels. We aim the missile camera towards the target during target lock, i.e., before firing the missile.

The Missile Camera which is mounted on 2 axis gimbal, will form a closed loop tracking system with the help of tracking processer and gimbal stabliser processor. The working principle of this tracking is, the PWM signal is sent to gimbal servos, such a way that, position of the target is centered in the frame view of the camera. This is achieved[12] by dividing the frame in to 4 quadrants, and with the use of openCV csrt algorithm, we can get the object coordinates (x,y), and tracking[13] processor communicates (x,y) data to gimabal stabliser (which has servo controller in it) processes and give correction PWM signal to servo 1 (pitch, yaw), servo 2 (roll), such that (x,y) tend to (0,0). This Feedback loop between camera, and gimbal servos is a closed loop (say CL-1) independent other systems of missile.

Thus, camera always aligned, and pointed towards the target, thus giving us orientation parameter, ie., the vector of the target, to which the missile has to align to. If we try to get this orientation data via gimbal tracking system, there is lag in this communication to trajectory processor. Thus, we are getting this data signal (say "A") from IMU (say IMU-1) attached physically to camera, this way, we get data much faster speed[14], so the trajectory processor, compares the data signal (say "B"), with body fixed IMU (say, IMU-2), and calculates the PWM signal needed to send to canards, so as to reduce this error between IMU-1 and IMU-2. This Feedback loop between IMU-1, IMU-2, to make $(A - B) \rightarrow 0$, forms a independent closed loop system (say CL-2).

The criteria for convergence of this control system is, In a operation cycle(in order of 0.01 second or lower), The orientation signal (delta angle change) of IMU-1 has **to be higher** than the the same of IMU-2. It implies that the system has functional response, which is greater than the required response, [11]in missile orientation change during the operation. This system alters the missile path, and guides in such a way that the miss distance is least, compared to previous versions of missile, Thus we get Optical Guided Precision (OGP) Trajectory

If there is a camera signal blackout or the target is not detected in the image frames captured by camera, then, The missile[15] follows the BLOSOA Trajectory, so as to hit the area near to target, having a considerable miss distance.[16]

2.5.3 Details

S.No	Component	Calculated Mass (in gram)
1	Ignitor, CD Nozzle, Insulation	386
2	Electronics	534
3	Propellant	1200
4	Payload	2100
5	Carbonfiber Frame	519
6	Gimbal + IMU + Camera	97
7	Servo Mechanism	148
Total		4984

Table 2.4: Version 4 : Mass Distribution

1. Gimbal is present, with IMU and camera module. IMU is physically fixed behind camera.
2. Servo mechanism attached to canards, and they take precise angle of attack, with accuracy of 0.1 degree by pairing it with our PWM servo controller
3. This has payload mass of 2100 gram and propellant mass of 1200 gram
4. The electronics constitutes of battery, servo controller, raspberry pi constitutes mass of 191 gram, and the servo mechanism system adds up to 148 grams
5. This Missile Version use SRM Version 2.0, and WP Version 3.0
6. Weapon Release System A (WRS-A) is used here which is estimated about 2.4 kg, thus, altogether mass of missile and WRS-A is about 7.4 kg
7. This is perfect to launch from hexarotor drone (UCAV) with minimum payload capacity of 8 kg

2.5.4 Validated Key Variable

1. Tracking speed is validated in real time operation
2. Real time closed loop control system convergence is tested & validated

3. Miss distance is obtained and is verified with previous versions, to validate the least miss distance achievable.

4. This also validates our BLOSOA Trajectory effectiveness as it is entirely based on machine learning models of missile theoretical data, and field data. Also, The trend of value miss distance should decrease in each version after consecutive tests as our machine learning model evolves efficient over time, as we provide it with many instance field data.

2.5.5 Output & Expected Trajectory

1. Fire and Forget

2. Target Lock before Launch

3. Launched at required angle of depression, for the targeted range via LOS trajectory

2.5.6 Architecture

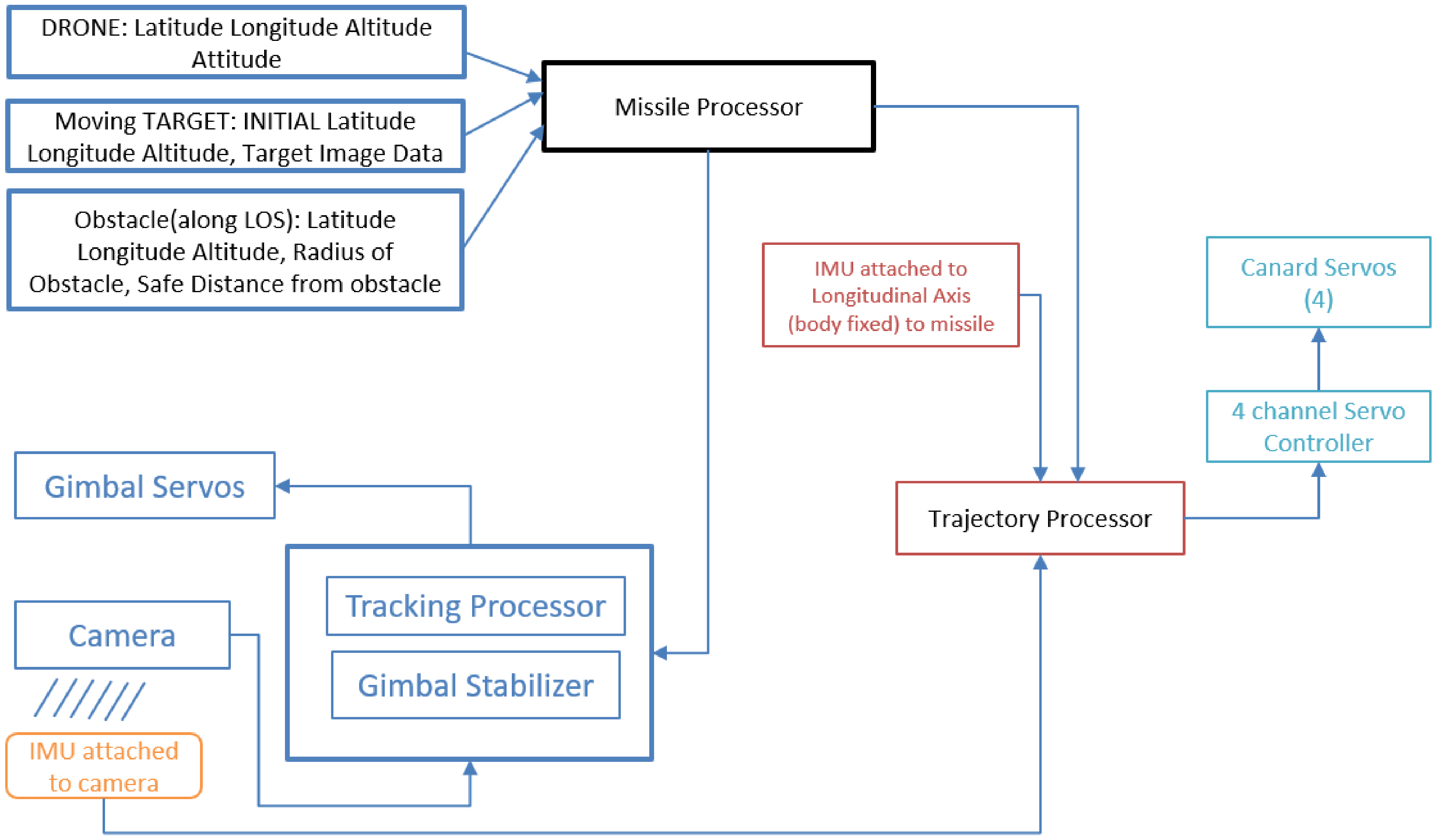

DRONE: Latitude Longitude Altitude Attitude
Moving TARGET: INITIAL Latitude Longitude Altitude, Target Image Data
Obstacle(along LOS): Latitude Longitude Altitude, Radius of Obstacle, Safe Distance from obstacle
Missile Processor
IMU attached to Longitudinal Axis (body fixed) to missile
Canard Servos (4)
4 channel Servo Controller
Trajectory Processor
Gimbal Servos
Tracking Processor
Gimbal Stabilizer
Camera
IMU attached to camera

Figure 2.8: Architecture : Version 4

2.6 FRAME MATERIAL SELECTION

Constraints Options

* **Nose** : Material at the Nose need to strong enough to sustain in the duration of flight path. Nose need to break when impulse hit to ground, i.e., Nose should not absorb greater impulse of sock at hit with ground or target. Thus we can choose 3D printing method, of most general material which suits the requirement, i.e., ABS plastic.[17]

* **Frame** : Missile frame need to be light weight with higher strength compared with any alloy of metal. Thus, we can use Epoxy Carbon Composite with manufacturing process suiting the needs. [18] [19] [20]

* **Rocket Engine** : Rocket Motor is a pressure vessel, should withstand high temperature of combustion chamber, of course with insulation, for a time of 15 seconds atleast, Also should be able to withstand pressure of 45 bar atleast. Thus, we can use Phenolic Carbon Composite. [21] [22]

* **CD Nozzle** : Graphite can be used in initial tests and validation. Once the design is proved, CD Nozzle can be made from Phenolic Carbon Composite, as we get same properties with light weight. [23] [24]

CHAPTER 3

DESIGN AND ANALYSIS OF MISSILE

3.1 ENGINE OF MISSILE (SOLID ROCKET DESIGN)

A solid rocket motor is a rocket engine that use solid as a propellant (fuel/oxidiser). Since solid-fuel rockets can remain in storage for a long time without much propellant degradation, and the fact that they almost always launch reliably, they have been frequently[25] used in military applications

The design parameters are interdependent, choice of propellant is dependent on thrust required, and thrust required is dependent [26]on range requirement. The structural mass is dependent on propellant mass, chamber pressure (p_c) and temperature (t_c). Thrust available depends on exit gas velocity (V_e) , V_e is dependent on nozzle design, nozzle design is dependent on choice of propellant. [27] [28]

To design the solid rocket motor, the following steps are followed :

1. Calculate Required Thrust and Burn-time for the requirement of mission. Also, Fix the targeted mass of propellant
2. Fix a propellant composition
3. Fix a chamber pressure and nozzle area ratio
4. Putting all these design criteria in CEA software we get values of various parameters for both Equilibrium flow and Frozen Flow (if applicable)
5. Since the flow in the nozzle is neither equilibrium nor frozen therefore we take the average of all the required quantities obtained from CEA. (C_F, P_e, T_c, C^*, γ)
6. Using all these values find out different parameters like U_e, T, A_t, A_e etc.,
7. Choose a type of nozzle and calculate its geometric properties

8. Calculate length of convergent portion
9. Then find out mass flow requirements for the given of solid rocket motor thrust for both fuel and oxidizer
10. Estimate mass flow rate, density, burn rate, burn surface area and all other relevant parameters
11. Choose a grain geometry for your motor and calculate the required parameters
12. Select suitable Material for chamber casing and calculate Wall thickness, volume of casing, mass of the motor, length, diameter etc
13. Select the Thermal insulation material and design for nozzle throat insulation / cooling [29] [30]

3.1.1 Composition of Propellant

In this iterative process, after referring database of the various propellant composition combinations in the in-house propellants developed by National Centre for Combustion Research and Development (NCCRD), the following propellant ingredients is finally considered optimally for the range of 2000 m with Launch depression angle less than 45 deg with thrust of 1000 N for 2.5 s burnout time. The propellant selected is a Composite Propellant because of its higher densities,[31] specific impulses, and a wider range of burn rates properties.[32]

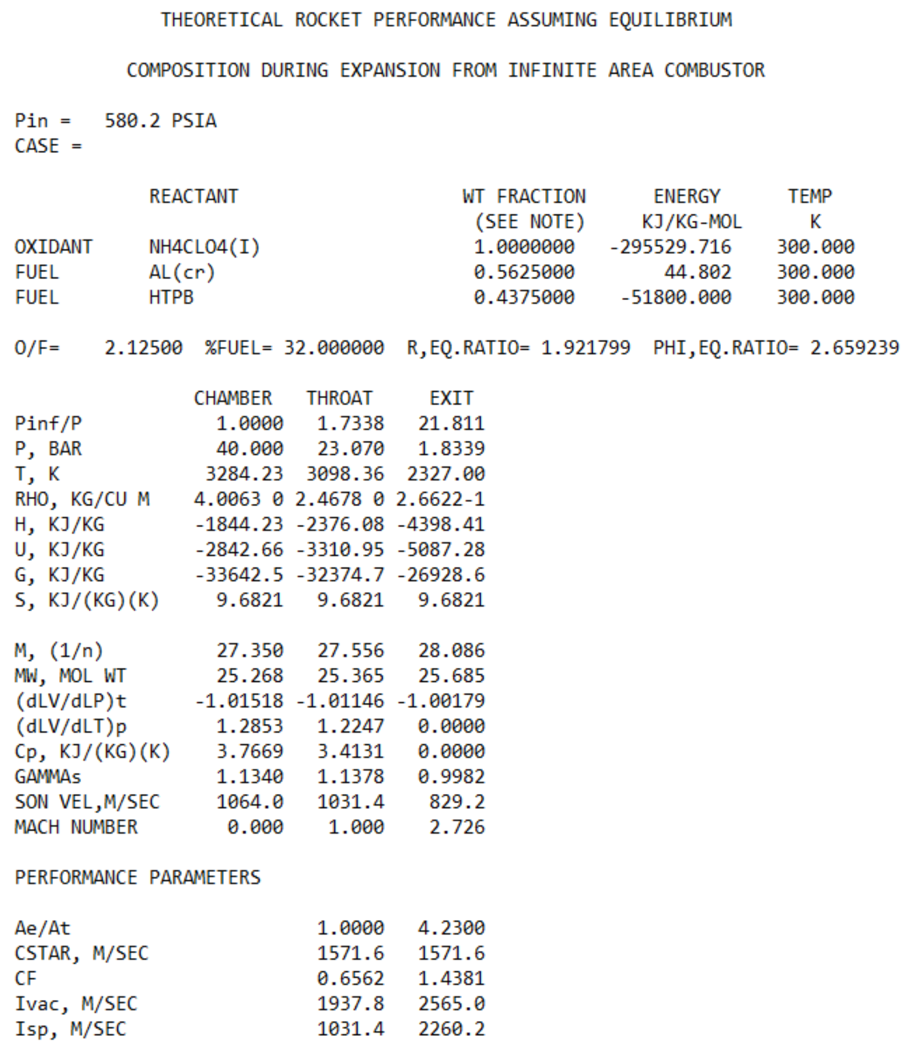
Ingredient	function	**Wt**%
HTPB	Fuel,Binder	10%
DOA	Plasticizer	3%
Ambilink	Cross-linking agent to ↑ tensile property	0.28%
IPDI	Curing agent	0.72%
Al	Fuel	18%
AP(course)	Oxidizer	40.8%
AP(Fine)	Oxidizer	27.2%

Table 3.1: SRP Version 2.0 : Composite Propellant Composition

SRM Version 2.0 is the Ammonium perchlorate composite propellant (APCP). The composition of APCP can vary significantly depending on the application, intended burn characteristics, and constraints such as nozzle thermal limitations or specific impulse (I_{sp}). The composition of APCP chosen is 68 % Ammonium Perchlorate (AP), 14 % Hydroxyl Terminated Polybutadiene (HTPB) (which includes plasticizer, cross-linking agent, curing agent) and 18 % Aluminium by mass. AP acts as the oxidizer, Al as the fuel and HTPB as a binder and fuel.

3.1.2 CEA Analysis

The chamber pressure is equal to 40 bar and Design Mach number of CD Nozzle is 3, thus nozzle area ratio (Ae/At) is 4.23.



```
                THEORETICAL ROCKET PERFORMANCE ASSUMING EQUILIBRIUM

             COMPOSITION DURING EXPANSION FROM INFINITE AREA COMBUSTOR

 Pin =   580.2 PSIA
 CASE =

             REACTANT                    WT FRACTION      ENERGY      TEMP
                                          (SEE NOTE)     KJ/KG-MOL      K
 OXIDANT     NH4CLO4(I)                   1.0000000   -295529.716   300.000
 FUEL        AL(cr)                       0.5625000        44.802   300.000
 FUEL        HTPB                         0.4375000    -51800.000   300.000

 O/F=    2.12500  %FUEL= 32.000000  R,EQ.RATIO= 1.921799  PHI,EQ.RATIO= 2.659239

                 CHAMBER   THROAT     EXIT
 Pinf/P            1.0000   1.7338   21.811
 P, BAR            40.000   23.070   1.8339
 T, K             3284.23  3098.36  2327.00
 RHO, KG/CU M    4.0063 0 2.4678 0 2.6622-1
 H, KJ/KG        -1844.23 -2376.08 -4398.41
 U, KJ/KG        -2842.66 -3310.95 -5087.28
 G, KJ/KG        -33642.5 -32374.7 -26928.6
 S, KJ/(KG)(K)     9.6821   9.6821   9.6821

 M, (1/n)          27.350   27.556   28.086
 MW, MOL WT        25.268   25.365   25.685
 (dLV/dLP)t      -1.01518 -1.01146 -1.00179
 (dLV/dLT)p        1.2853   1.2247   0.0000
 Cp, KJ/(KG)(K)    3.7669   3.4131   0.0000
 GAMMAs            1.1340   1.1378   0.9982
 SON VEL,M/SEC     1064.0   1031.4    829.2
 MACH NUMBER        0.000    1.000    2.726

 PERFORMANCE PARAMETERS

 Ae/At                      1.0000   4.2300
 CSTAR, M/SEC               1571.6   1571.6
 CF                         0.6562   1.4381
 Ivac, M/SEC                1937.8   2565.0
 Isp, M/SEC                 1031.4   2260.2
```

Figure 3.1: CEA Analysis

The performance analysis assuming frozen flow is not shown here. It is due to the fact that addition of Aluminium to the propellant makes it a two phase problem due to presence of Aluminium in gases and so the above parameters cannot be obtained by Chemical Equilibrium Analysis software assuming frozen flow. Hence the further analysis is being done assuming equilibrium flow in the nozzle, though in reality the flow is neither equilibrium nor frozen.

Thrust required is $F = 1000N$ for the trajectory of 2000 m

$$\mathbf{F} = \eta \cdot \lambda \cdot P_c A_t C_F \tag{3.1}$$

Where η-efficiency factor (0.95),

λ - divergence loss factor,

$$\lambda = \frac{1 + \cos\alpha}{2} = 0.9531$$

(for $\alpha = 25$ deg)

Since,

$$C_F = 1.43$$

$$P_c = 4MPa$$

Throat Area,

$$\mathrm{A}_t = \frac{F}{\eta \cdot \lambda \cdot P_{c^* C_F}} = 0.00019 \text{ m}^2$$

Throat diameter,

$$d_t = \sqrt{\frac{\mathrm{A}_t}{\pi/4}} = 0.0156 \text{ m} = 1.56 \text{ cm}$$

Since

$$\frac{A_e}{A_t} = 4.23$$

Exit Area,

$$\mathrm{A}_e = 0.00082 \text{ m}^2$$

Exit diameter,

$$d_e = \sqrt{\frac{\mathrm{A}_e}{\pi/4}} = 0.0322\ \mathrm{m} = 3.22\ \mathrm{cm}$$

Mass flow rate,

$$\dot{\mathrm{m}} = \frac{P_c At}{c^*} = 0.4914\ \mathrm{kg/s}$$

Thrust equation gives

$$\mathbf{F} = \boldsymbol{\eta} * \lambda * (\dot{\mathrm{m}}\mathbf{U}_e + \mathbf{A}_e\,(\mathbf{P}_e - \mathbf{P}_a)$$

Exhaust velocity,

$$\mathrm{U}_e = I_{sp} - \frac{\mathrm{A}_e}{\dot{\mathrm{m}}}\,(\mathrm{P}_e - \mathrm{P}_a) = 2124.4\ \mathrm{m/s}$$

Vacuum I_{sp},

$$I_{vac} = 2565\ \ \mathrm{Ns/kg}$$

Vacuum thrust,

$$\mathrm{F_{vac}} = \eta \cdot \lambda \cdot (\dot{\mathrm{m}} \cdot I_{vac}) = 1141\ \ \mathrm{N}$$

3.1.3 Method of Characteristics

Design of CD Nozzle diverging portion is done using method of characteristics.

$$\text{Pressure Ratio } = \frac{\text{Outside Pressure}}{\text{Chamber Pressure}}$$

$$\text{Temp Ratio} = (\text{Pressure Ratio})^{\frac{\gamma-1}{\gamma}}$$

$$\text{Critical (Throat) Temp} = \frac{(2 \cdot \gamma \cdot R \cdot \text{Chamber\_Temp})}{\gamma - 1}$$

$$\text{Critical (Throat) Pressure} = \left(\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{\gamma-1}}\right).\text{Chamber\_Pressure}$$

$$\text{Critical (Throat) Velocity} = \sqrt{\frac{(2 \cdot \gamma \cdot R \cdot \text{Chamber\_Temp})}{\gamma+1}}$$

$$\text{Exit Velocity} = \sqrt{\text{Throat Temp}\,(1 - \text{Temp Ratio})}$$

$$\text{Exit Temp} = (\text{Chamber } T) \cdot \left(\frac{\text{Pexit}}{\text{Pchamber}}\right)^{\frac{\gamma-1}{\gamma}}$$

$$\text{Exit Sound Speed} = \sqrt{\gamma \cdot R \cdot \text{Exit Temp}}$$

$$\text{Exit Mach} = \frac{\text{Exit Velocity}}{\text{Exit Sound Speed}}$$

$$PM = \sqrt{\frac{\gamma+1}{\gamma-1}} \cdot \operatorname{atan}\sqrt{\frac{\gamma-1}{\gamma+1} \cdot \left(M^2-1\right)} - \operatorname{atan}\sqrt{M^2-1}$$

$$\text{Max Angle} = \frac{1}{2} \cdot \{PM(\text{Exit Mach})\}$$

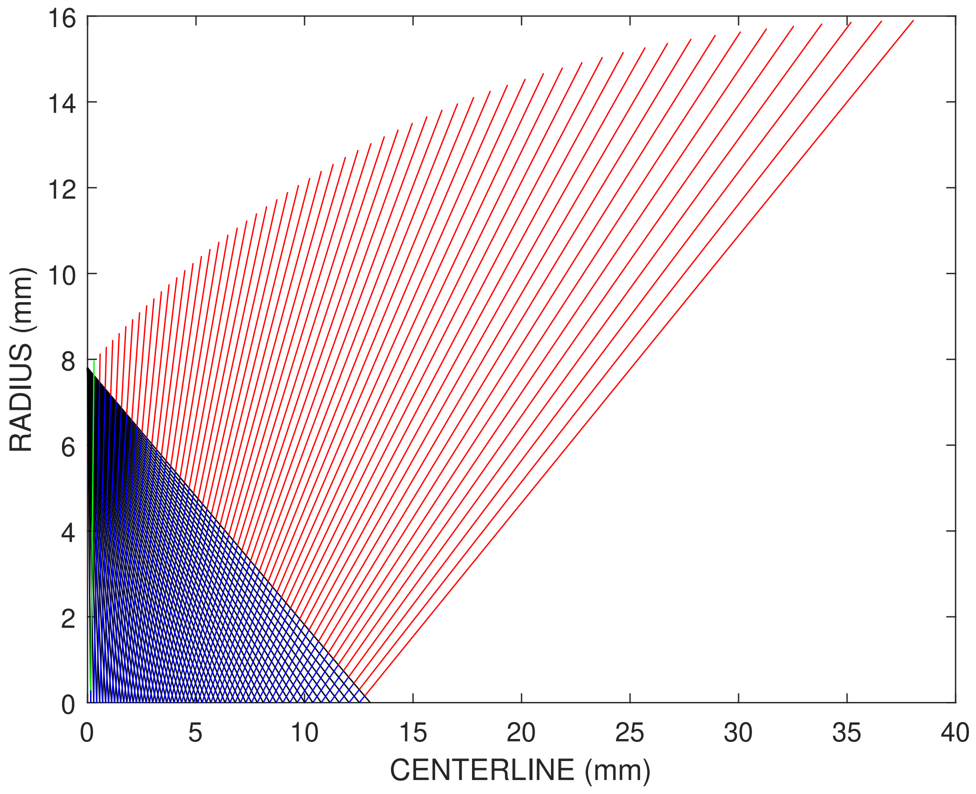


Figure 3.2: CD Nozzle : Diverging portion

$$\text{Angle Increment } = 2 \cdot ((90 - \text{Max Angle }) - \text{fix}(90 - \text{Max Angle }))$$

Matlab code is attached in Appendix B.1

3.1.4 Characteristics of Propellant

Density of propellant

$$\rho_p = \frac{1}{\frac{f_a}{\rho_a} + \frac{f_b}{\rho_b} + \frac{f_c}{\rho_c}} \tag{3.2}$$

Where a, b and c represents the AP, HTPB and Al respectively

Propellant constituent	Ammonium Perchlorate	HTPB	Aluminium (crystalline)
Density $(\mathbf{kg}/\mathrm{m}^3)$	1950	920	2700
Mass fraction	0.68	0.14	0.18

Table 3.2: Density and mass fraction of oxidizer and fuel

$$\rho_p = 1761.932 \text{ kg/m}^3$$

Mixture ratio (O/F ratio) obtained from CEA is equal to 2.125

$$\dot{\mathrm{m}}_{\mathrm{ox}}/\dot{\mathrm{m}}_{\mathrm{f}} = 2.125$$

$$\dot{\mathrm{m}} = \dot{\mathrm{m}}_{\mathrm{xx}} + \dot{\mathrm{m}}_{\mathrm{f}} = 0.4914 \text{ kg/s}$$

Solving above equations, we get

$$\dot{\mathrm{m}}_{\mathrm{f}} = 0.157 \text{ kg/s}$$

$$\dot{\mathrm{m}}_{\mathrm{ox}} = 0.334 \text{ kg/s}$$

where, $\dot{\mathrm{m}}_{\mathrm{ox}}-$ mass flow rate of oxidizer

$\dot{\mathrm{m}}_{\mathrm{f}} -$ mass flow rate of fuel

Burn rate is given by, [32]

$$\dot{r} = a{P_c}^n$$

The values of a and n are obtained as 6.7 mm/s(atm)^n and 0.42 respectively.

The a and n values obtained are calculated at 70 bar critical pressure and so the above relation modifies to the following form [32]

$$\dot{\boldsymbol{r}} = a_{70}\left(\frac{P_c}{70}\right)^n = 5.29 \text{ mm/s}$$

Burn area is calculated from the following relation

$$A_b = \frac{\dot{\text{m}}}{\rho_p \dot{r}} = 0.0526 \text{ m}^2$$

3.1.5 Grain Design

Neutral burning propellant configuration would be a preferred configuration since the pressure-time variation shows a constant pressure. Hence a star configuration has been selected to achieve neutral burning of the propellant.

No. of star points = 6

Angle between two star arms, $\theta = \theta_{n,\text{ neutral}} = 67^\circ$

Area of port,

$$A_p = nl^2 * \frac{1}{\frac{1}{\tan\left(\frac{\pi}{n}\right)} + \frac{1}{\tan\left(-\frac{\pi}{n}+\frac{\theta}{2}\right)}} \tag{3.3}$$

$$A_p = 0.3318l^2$$

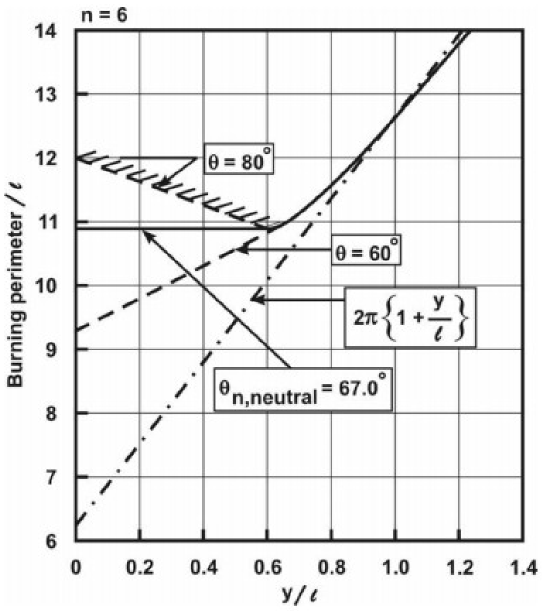


Figure 3.3: The normalized perimeter with burning distance for n = 6 [32]

The extent upto which neutral burning can be achieved is given by the relation

$$\frac{y}{l} = 0.6$$

Where, l is the length of the star point from the centre of star grain geometry

y is the web thickness (minimum thickness of grain between port and wall)

If the ratio goes more than 0.6, progressive burning comes into action replacing neutral burning.

From the above plot, at $y/l = 0.6$,

burning perimeter(P) $/l = 10.9$

$P = 10.9l$

Length of grain,

$$\text{L} = \frac{A_b}{P} = \frac{0.00483}{l}$$

Diameter of grain,

$$\text{D} = 2 \cdot (y + l) = 3.2l$$

Mass of propellant

$$m_p = \dot{m} \cdot t_b = 1.2285 \text{ kg}$$

Assuming 6% sliver loss (as star configuration has sliver loss of 4-8%),

$$m_p = 1.06 \cdot 1.2285 = 1.3 \text{ kg}$$

Volume of Propellant,

$V_p = \frac{m_p}{\rho_p} = 0.000739$

The material selected is carbon composite with 110 MPa yield stress, which is fabricated using compression mould process.

Thickness of wall can be calculated from circumferential stress relation

$$t = \frac{p d_i}{2\sigma} \tag{3.4}$$

Here factor of safety used is equal to 1.4 and so permissible stress, $\sigma = \dfrac{\sigma_y}{1.4}$

Diameter of Grain,

$$D = 64.5 \text{ mm}$$

Length of grain,

L = length of grain = 243.46 mm

thickness,

$$t = \frac{40 \cdot 100000 \cdot 64.5 \cdot 10^{-3}}{2 \cdot \frac{110 \cdot 10^6}{1.4}}$$

$$t = 0.00164 \text{ m} = 1.64 \text{ mm} \tag{3.5}$$

Liner

Liner serves an important function to ensure good bond interface between the propellant and insulation at wall. It also inhibits burning so that propellant only burns at the unlined surfaces. In general, the liner material is the bonding matrix or fuel material in the propellant. This insures a good propellant to liner bond.

The liner material which can be used is HTPB binder cured with Isophorone Diisocyanate (IPDI). In general, thinner liners produce greater bond strengths and increase the propellant loading fractions. So, the thickness of thin liner we use is around (0.127 mm).

Stress Release Boots

Propellant grains are cast at elevated temperatures and allowed to cool and polymerize. The resultant shrinkage develops residual stresses in the propellant charge that are not relieved by plastic deformation. These stresses can result in bond failure between propellant and liner or case to liner. If liner or the bond is having high strength, the propellant charge will probably fail by cracking down the base of the star point. The result can be a catastrophic failure due to the increased burning surface area.

A technique to overcome this weakness is called the "stress release boot". The boot is fabricated as two individual items and bonded out at the maximum diameter area. Each piece is made from a rubbery flexible material.

3.1.6 Propellant mixing

A plastic beaker of 250 ml is taken and placed on a weighing balance. The balance is then reset to zero. HTPB is poured slowly into the beaker up to the required weight. The weighing balance is again reset to zero with beaker still on the balance tray. Then DOA is poured drop by drop using a dropper and followed by TDI using a glass stirrer rod. The ingredients are well mixed with hands by means of a spatula. The beaker is again kept on the weighing balance with the wetted spatula and the balance is again set to zero. Ambilink and Nonax-D are added to the mixture followed by the addition of coarse and fine AP powder as per the formulations above. The mixture is again hand mixed for 15 minutes and then poured into the propellant mixer. The mixer, shown in figure 3.4, is developed by IIT madras and has a capacity of 200 grams. An evacuation pump is connected to the mixer and the propellant is mixed for an hour within the mixer. Then the propellant is appeared as smooth and glossy, which indicates complete wetting of all the solid surfaces by the binder.

3.1.7 Propellant casting

The propellant slurry is taken out from the mixer and poured into the Teflon casting cup. A cylindrical shaped[36] casting cup (figure 3.7) with one end closed and having a mandrel in the centre is used. Before the propellant is poured to the casting cup, very thin layer of silicon oil is applied on the inner surface of the casting cup and mandrel so

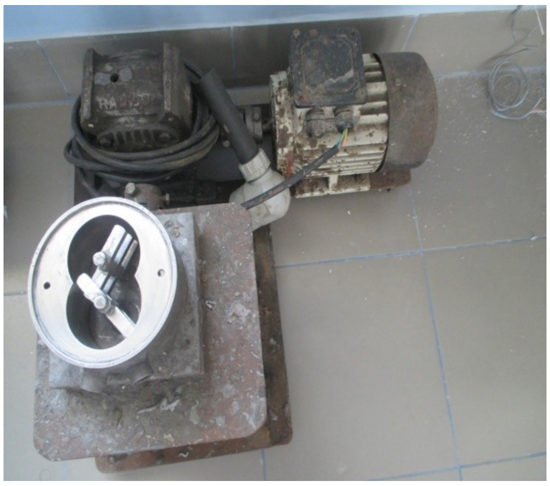


Figure 3.4: Propellant Mixer

that when the cast grain needs to be removed, it would come out easily. The casting cup is then inserted into the casting chamber. The schematic drawing of the casting chamber along with the actual casting chamber is shown in figure 3.5 and 3.6 respectively. The outside diameter of the casting cup is the same as of the internal diameter of the casting chamber. A vacuum pump is connected to the casting chamber in order to remove the air tapped in the propellant. Hot water at 50 °C is circulated in the annular region of the casting chamber to decrease the viscosity[34] of the propellant during casting. An air-tight piston connected to the plunger is inserted into the chamber and the top cap is screwed to the chamber at the top end. The hot water circulation in a jacket surrounding the chamber and vacuum in the chamber are maintained for about 30 minutes. After half an hour of evacuation, the plunger is pressed and the propellant slurry is compressed. A mass of 2-3 kgs is kept on the top of the plunger for one day. After one day, the bottom plate is removed and the cast propellant along with the casting cup is taken out.[35] This is cured in an oven at a temperature of 60 ºC for 7 days. After 7 days, the propellant grain is taken out of the casting cup. the propellant grain and it's dimensions are shown in figure 3.8 and 3.9 respectively.[37]

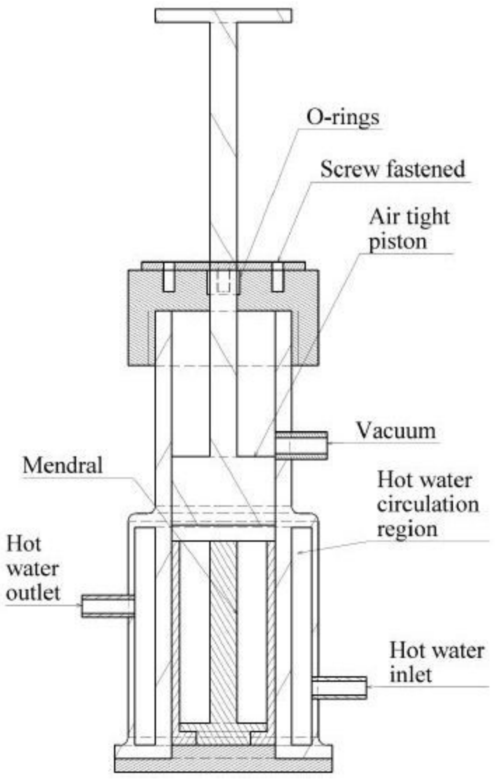


Figure 3.5: schematic drawing of casting chamber

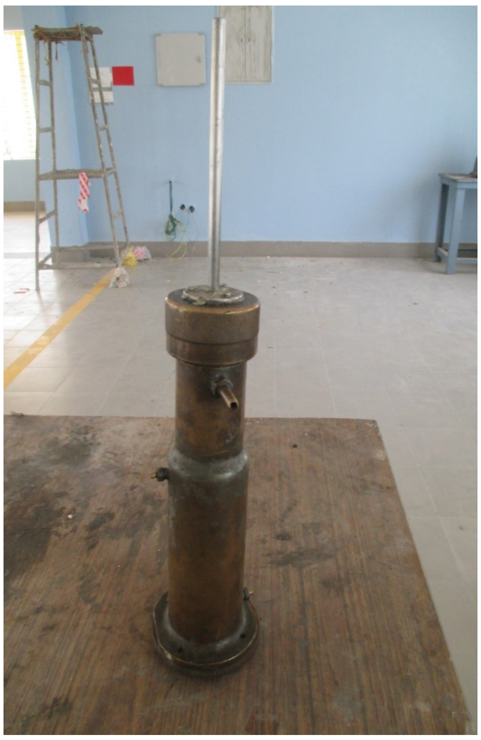


Figure 3.6: casting camber

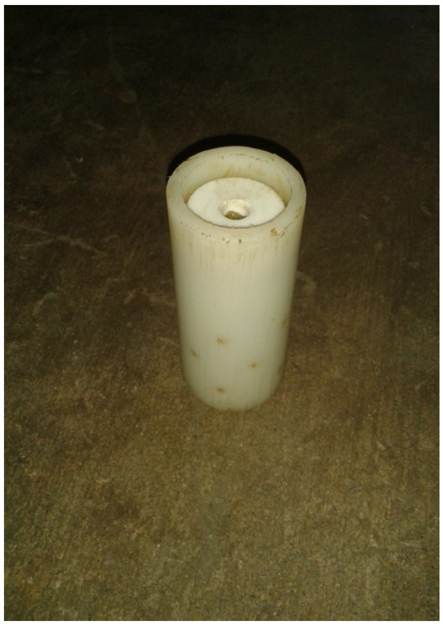


Figure 3.7: propellant in casting cup

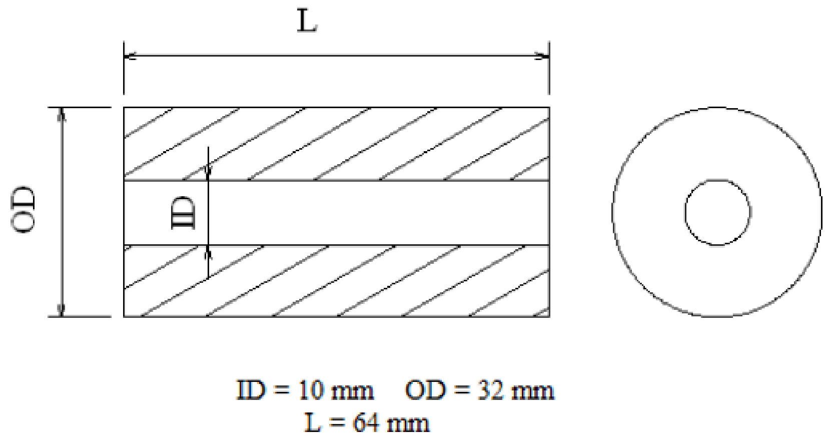


Figure 3.8: propellant dimensions

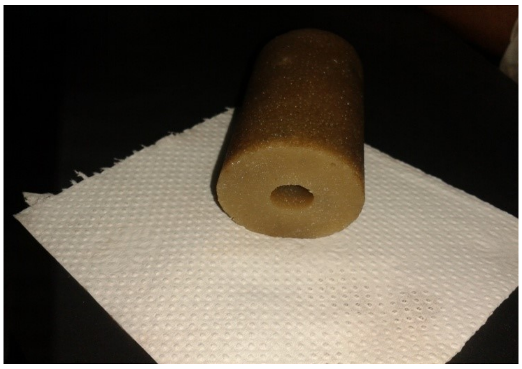


Figure 3.9: Propellant Grain

3.1.8 Ballistic evaluation motor (BEM)

The ballistic evaluation motor (BEM) test is to measure the motor performance parameters under actual operating conditions. The complete assembly drawing of the motor is shown in figure 3.10 below.

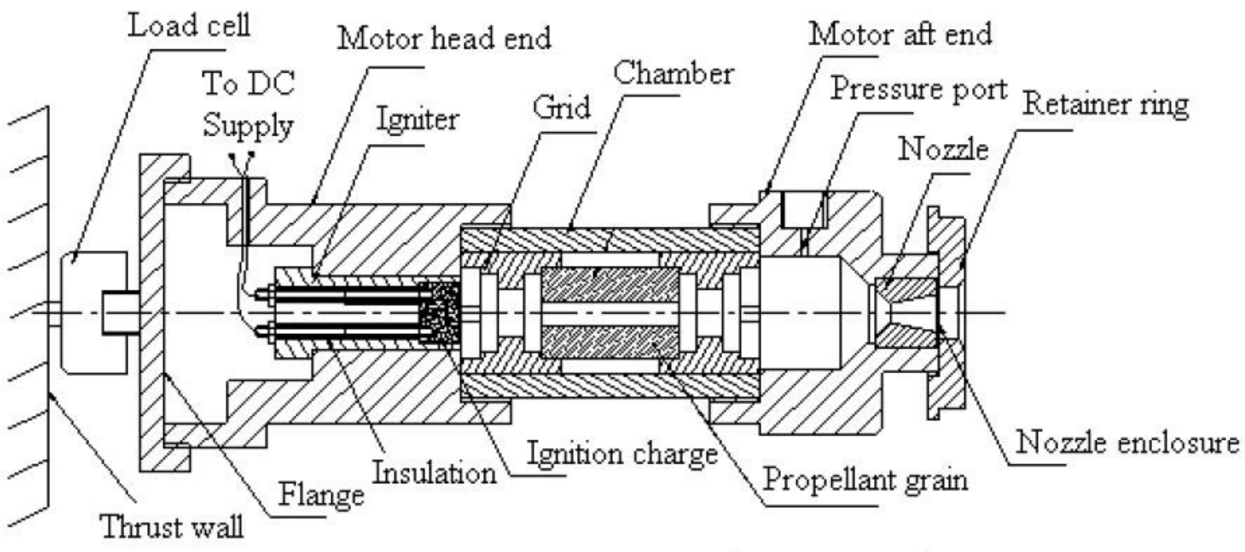


Figure 3.10: Ballistic evaluation motor assembly drawing

The main motor parts and their functions are tabulated in table 3.3

S. No.	Motor part	Function
1	Load cell	Thrust measurement
2	Motor head end	Holds igniter, connection between combustion chamber and load cell
3	Combustion chamber	Holds propellant, provides space for combustion
4	Grid	Holds the propellant in horizontal axis in combustion chamber
5	Motor aft end	Provides space for pressure port and nozzle
6	Nozzle	Expansion of combustion products
7	Retainer ring	Secures nozzle

Table 3.3: Motor parts and their functions

3.1.9 Ignition system

The igniter consists of a metallic body through which two tungsten electrodes passed inside covered with hylum insulation. Gunpowder is used as the ignition charge. This is a mixture of potassium nitrate, sulphur and carbon in the ratio of 80:15:5 by mass. Ignition charge is placed in a small paper packet and nichrome wire is passed through it.[38] The ends of the nichrome wire are connected to the electrodes of the igniter. The other end of the igniter is connected to a 24V DC power supply. When the power is supplied to the igniter, the nichrome wire gets heated and thus the ignition charge is fired. Figure 3.11 shows the igniter used in BEM.

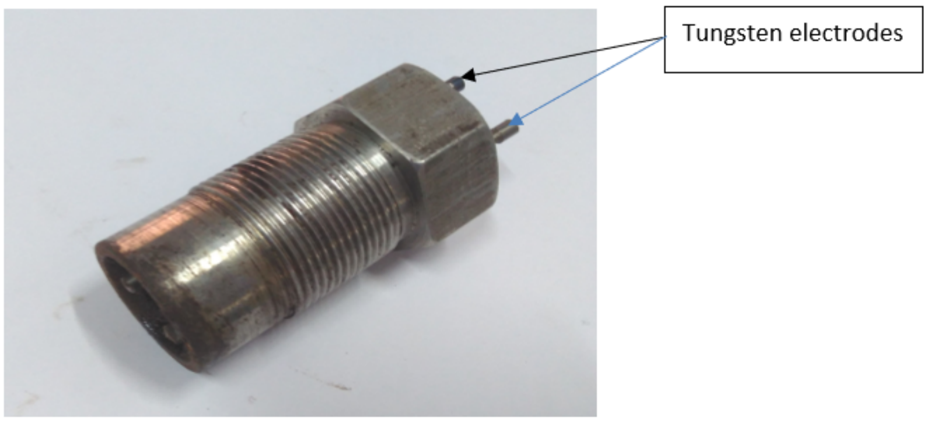


Figure 3.11: Igniter for BEM

3.1.10 Testing of BEM

Once the motor is ignited, the flame and hot gas are produced due to the firing of the igniter charge. This flame spreads on the propellant surface, ignites the propellant in close proximity and propellant combustion is commenced. A piezo-resistive transducer connected to the aft end of the motor measures the pressure rise during the combustion. The thrust developed is recorded by the load cell, which gets compressive load in between the thrust wall and motor aft end [39]during firing. Initial port diameter of the propellant and nozzle throat diameter are 10 mm and 9 mm respectively, so the area ratio 1.24. The transducer, load cell and other accessories are connected to the system through data acquisition system. The assembled BEM is shown in Figure 3.12

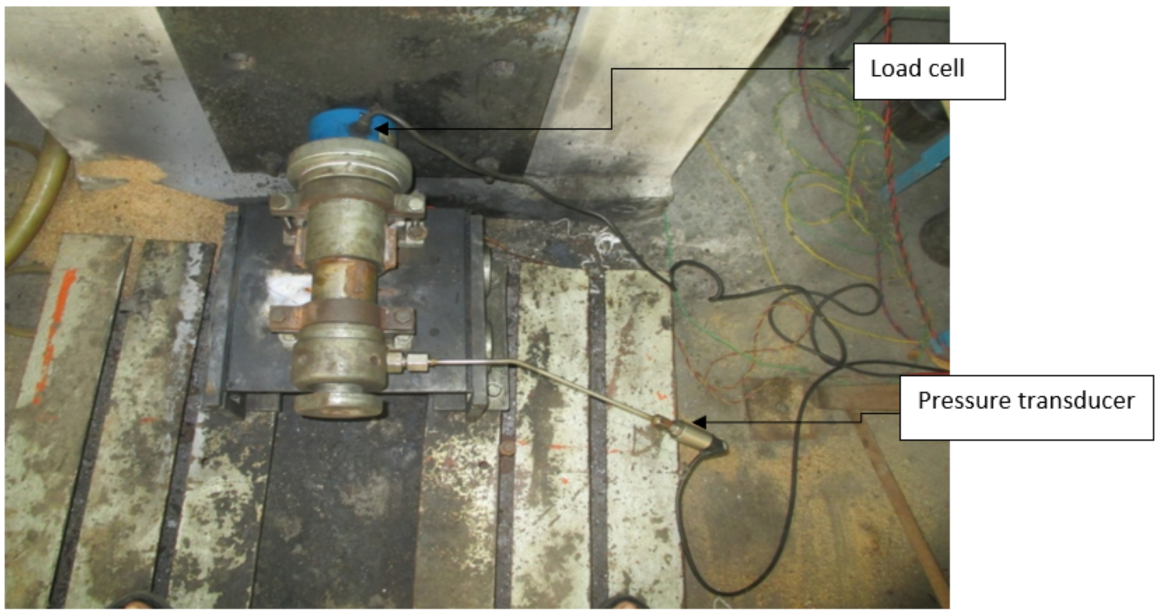


Figure 3.12: Assembled BEM

3.1.11 BEM Chamber design

The motor chamber is a hollow cylinder made up of stainless steel grade 316. The inner diameter of the chamber is 50 mm and the wall thickness provided is 10 mm. Assuming the maximum motor pressure to be 200 bar, the minimum chamber thickness is calculated as follows: [39]

The hoop stress formula a thin walled pressure vessel is given as,

$$\sigma_\theta = \frac{P_c \times D_c}{2 \times t_{\text{wall}}} \tag{3.6}$$

where

P_c = chamber pressure,

D_c = diameter of the chamber,

t_{wall} = wall thickness.

The yield strength of steel is 200 MPa and the minimum wall thickness for a 20 MPa, 50 mm inner diameter of the chamber with a factor of safety of 3 is calculated as,

$$t_{\text{wall}} = \frac{(200 \times 10^6) \times (50 \times 10^{-3})}{2 \times 300 \times 10^6} \times 3 = 5 \times 10^{-3} \text{ m} \tag{3.7}$$

Since the provided wall thickness is 10 mm, which is more than the required thickness, the design is safe.

3.1.12 BEM Thread Calculation

External threads of 14 turns per inch (tpi) are provided at both ends of the chamber, and these threaded joints connect the motor head end and aft end. They are made up of mild steel of yield strength 200 MPa, and the force exerted by chamber pressure on these threads is found as

$$F_p = \frac{\pi}{4} \times D_c^2 \times P_c \tag{3.8}$$

The shear force that the threads can withstand is,

$$F_{\text{shear}} = \pi \times d \times m \times \frac{p}{2} \times \sigma_{\text{shear}} \tag{3.9}$$

where,

d = pitch circle diameter of thread

m= number of threads

$$\sigma_{\text{shear}} = \frac{\sigma_\theta}{2} \tag{3.10}$$

Equating the pressure force and shear force, we get $F_p = F_{\text{shear}}$

$$m = \frac{P_c \times D_c^2}{4 \times d \times \frac{p}{2} \times \sigma_{\text{shear}}} \tag{3.11}$$

$$m = \frac{200 \times 10^6 \times \left(50 \times 10^{-3}\right)^2}{4 \times 68.8218 \times \left(\frac{1.814}{2} \times 10^{-3}\right) \times 33.33 \times 10^2} = 6.002 \tag{3.12}$$

That is, there should be approximately 6 threads, and since the chamber is provided with 11 threads, the design is safe.

3.1.13 Burn rate calculation

Let the propellant burn by a distance y from all the exposed surfaces in time t. The burnt volume of propellant grain is indicated by the shaded area in Figure 3.13.

OD = outer diameter of the propellant grain,

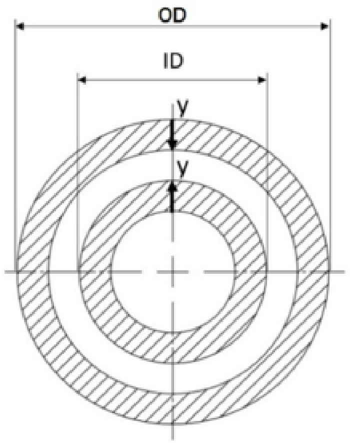
ID = inner diameter of the propellant grain

L = length of the propellant grain.

Then, the present exposed area is calculated as follows. The end surface area is given by,

$$Ab(y) \text{ end } = 2\left(\frac{\pi}{4}\right)\left[(OD - 2y)^2 - (ID + 2y)^2\right] \tag{3.13}$$

$$Ab(y) \text{ lateral } = \pi(L - 2y)[(OD - 2y) + (ID + 2y)] \tag{3.14}$$



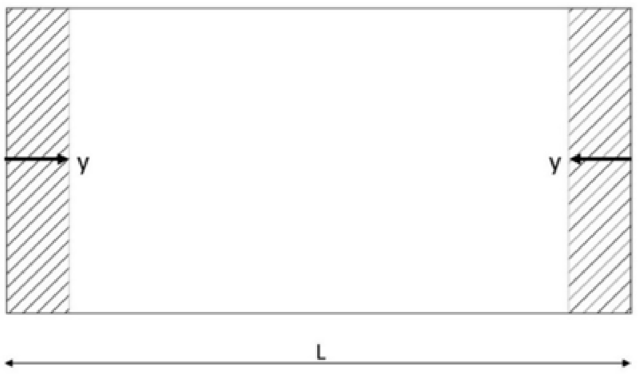


Figure 3.13: Propellant grain burning profile

$$Ab(y) = 2\left(\frac{\pi}{4}\right)\left[(OD - 2y)^2 - (ID + 2y)^2\right] + \pi(L - 2y)[(OD - 2y) + (ID + 2y)] \tag{3.15}$$

$$= \left(\frac{\pi}{2}\right)(OD + ID)(OD - ID + 2L - 8y)$$

The remaining volume of the propellant with respect to burnt distance y is given by

$$
\begin{aligned}
Vp(y) &= \left(\frac{\pi}{4}\right)\left[(OD-2y)^2-(ID+2y)^2\right](L-2y)\\
&= \left(\frac{\pi}{4}\right)(OD+ID)(OD-ID-4y)(L-2y)
\end{aligned}
$$

The volume of the propellant consumed up to time t is given by

$$\Delta V_p = \frac{\int_0^t (P_c \times A_t \times dt)}{\rho_p \times C*} \tag{3.16}$$

A_t = nozzle throat area,

ρ_p = propellant density,

C^* = characteristic velocity,

$$C^* = \frac{\int_0^{t_b} (P_c \times A_t \times dt)}{m_p} \tag{3.17}$$

Where t_b = total burning time.

The total volume of the propellant at any time = volume consumed + volume remaining

$$V = \Delta V_p + V_p(y) \tag{3.18}$$

Consider the mass balance equation

$$\rho_p \times A_b \times r_0 = P_c \times \frac{A_t}{C^*} + \frac{d}{dt}(\rho_0 V_0) \tag{3.19}$$

For steady state

$$\frac{d}{dt}(\rho_0 V_0) = 0 \tag{3.20}$$

Thus the mass balance equation can be rewrite as,

$$\rho_p \times A_b \times r_0 = P_c \times \frac{A_t}{C^*} \tag{3.21}$$

Instantaneous burning rate

$$r_0 = \frac{P_c \times A_t}{(C^* \times \rho_p \times A_b)} \tag{3.22}$$

3.1.14 Mandrel for Full Scale Solid Rocket Motor

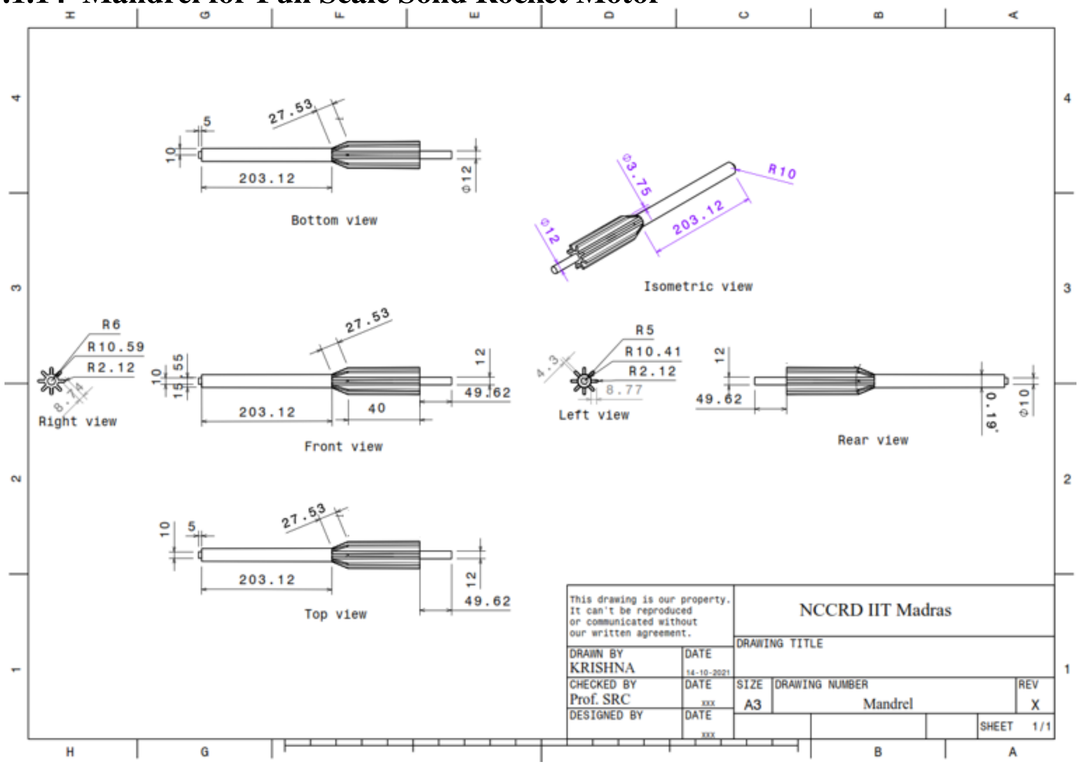


Figure 3.14: Full Scale Mandrel

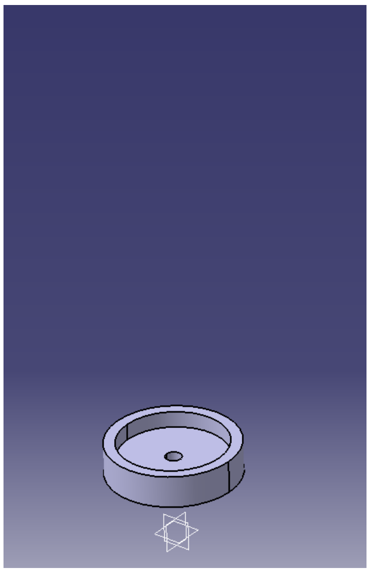


Figure 3.15: Full Scale Mandrel : Base Plate

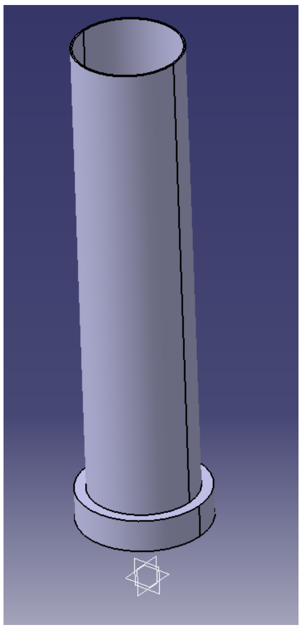


Figure 3.16: Mandrel Assembly

3.1.15 Nozzle design for BEM to test SRP V3

The nozzle is designed using Chemical Equilibrium Analysis (CEA) software, in order to expand the propellant combustion gas isentropically from a chamber pressure of 20 bar to atmosphere. The amounts of fuel and oxidizer and pressure ratio are fed in as input for the analysis. The input and output files of CEA analysis are attached as appendix. The important parameters obtained is tabulated in table 3.4 and the fabricated graphite nozzle is shown in fig.5. The nozzle throat is calculated from equation (3.22) as the propellant burning rate is known from window bomb analysis.

Parameter	Chamber	Throat	Exit
Temperature T(K)	3253.85	3071.25	2327.00
Pressure P(bar)	20.000	11.529	1.0000
Mach number	0.000	1.000	2.687
Area ratio Ae/At	-	1.0000	3.9467

Table 3.4: BEM Nozzle Dimensions

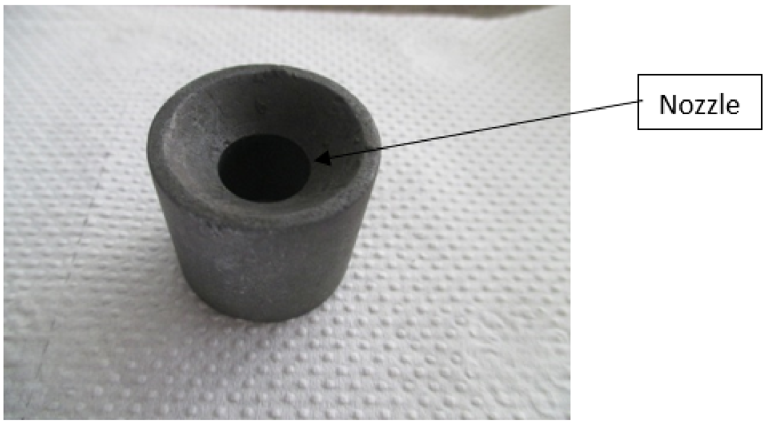


Figure 3.17: Graphite Nozzle

3.1.16 Grain Dimensions

Two different grain geometry are used in the present study. Their dimensions are given by table 3.5. Grain 1 is a cylindrical grain without inhibition. The burning unrestricted and the profile is regressive. Whereas the grain 2 is a bigger grain and the outer surface and end surfaces are inhibited. Thus the burning is inside out only and the profile is a progressive one. A mixture of HTPB, TDI and carbon black in the ratio 71.43:7.14:21.43

respectively is used as the inhibitor. The mixture is applied on propellant surface about 1.5 mm thick and cured for one day. An aluminized and non-aluminized grain of geometry 1 is shown in fig.3.18 and an inhibited grain 2 inside the motor chamber is shown in fig. 3.19

Dimensions	Grain
ID (mm)	10
OD (mm)	32
Length (mm)	64
Weight (g)	80

Table 3.5: Grain Dimensions

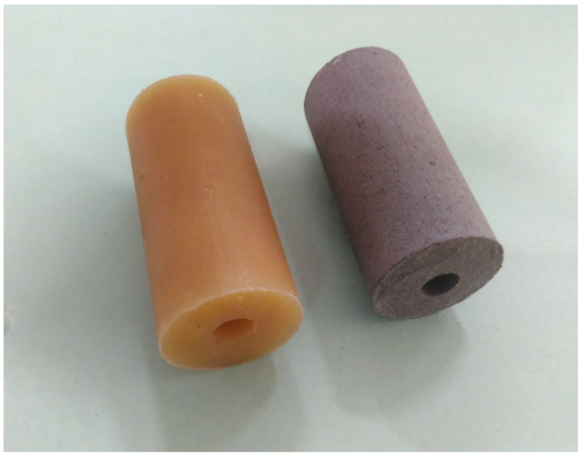


Figure 3.18: Non-aluminized (left) and aluminized (right) propellant grain 1

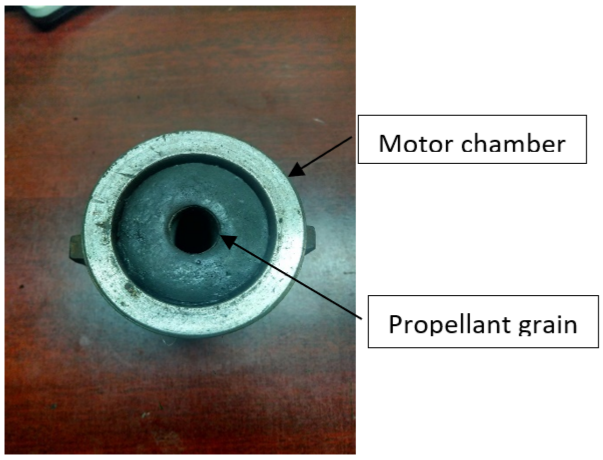


Figure 3.19: Inhibited propellant grain 2

3.1.17 Firing & P vs t graph

The composition of Solid Rocket Propellant V 3.0 chosen is 70 % Ammonium Perchlorate (AP), 15 % Hydroxyl Terminated Polybutadiene (HTPB) (which includes plasticizer, cross-linking agent, curing agent) and 15 % Aluminium by mass. AP acts as the oxidizer, Al as the fuel and HTPB as a binder and fuel.

Ingredient	function	**Wt%**
HTPB	Fuel,Binder	11%
DOA	Plasticizer	2.8%
Ambilink	Cross-linking agent to ↑ tensile property	0.38%
IPDI	Curing agent	0.82%
Al	Fuel	15%
AP(course)	Oxidizer	41.3%
AP(Fine)	Oxidizer	28.7%

Table 3.6: SRP Version 3.0 : Composite Propellant Composition

BEM firing was performed for propellant combination shown in the table 3.6. The Result data is shown below in figure 3.20

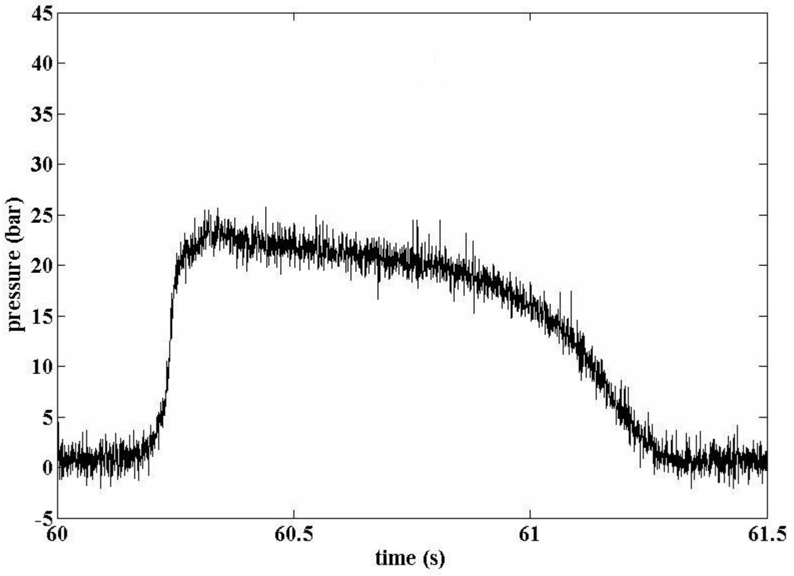


Figure 3.20: P vs T graph

3.1.18 Thrust Calculation

We can calculate Thrust vs time graph, provided with Pressure vs time graph.

From P vs t graph, we can get chamber pressure at a time step, we know the back pressure is always equal to atmospheric pressure i.e., 101325 Pa. We also know the CD Nozzle Area ratio ie., A/A_{star} through which the flow passing through.

Thus, For a given Pressure Ratio (Chamber Pressure / Back Pressure), and Area Ratio (Exit Area / Throat Area), (having known the temperature at chamber, nozzle exit) . There is only one possible flow. This flow can be calculated at each time step, using a Matlab code, for the calculation of thrust at each time step, code is given in appendix

Working of Thrust Calculator code

1. For the given Area Ratio, Code calculates subsonic mach number, supersonic mach number. These Mach number gives pressure ratio, as both cases are isoentropic.

When Input the back pressure, we get the range of chamber pressures, below which we get subsonic flow, above which we get under-expanded flow

2. The intermediate chamber pressure, that forms NS at exit plane is calculated, using normal shock at exit plane case

3. Thus we have pressure divisions as follows in table 3.7. Using which each case, we find exit mach number and exit pressure using gas dynamics

ID	Range of Chamber Pressure	Case
1	0 to P01	Subsonic
	P01	Chocking point
2	P01 to P0i	Normal Shock in Diverging portion
	P0i	Normal Shock at diverging portion of CD Nozzle
3	P0i to P02	Oblique Shock, Over Expanded, (Pe <Pb)
	P02	Perfectly expanded, Design Mach number case (exit mach number = design super sonic mach number)
4	P02 to infinity	Under expanded (Pe >Pb)

Table 3.7: Pressure Calculation

From CEA analysis of the propellant composition, we can know the temperature at exit, chamber temperature, molecular weight (which can be used to calculate gas constant). Applying these data at each instant in the Thrust Formula, we get Thrust at an instant.

Example Pressure vs Time graph obtained from Propulsion Lab shown in figure 3.21, to find Thrust vs Time graph. The Matlab code runs, once data is fed, and we get Thrust vs Time graph as shown in figure 3.22

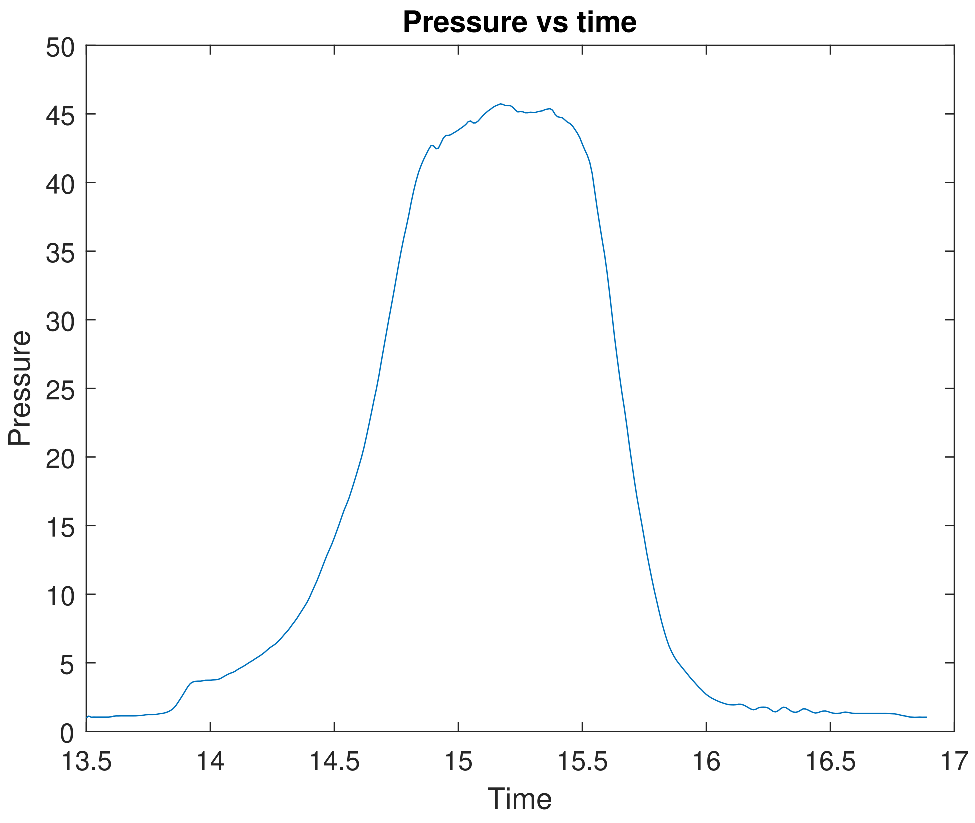


Figure 3.21: Input : Pressure Vs Time

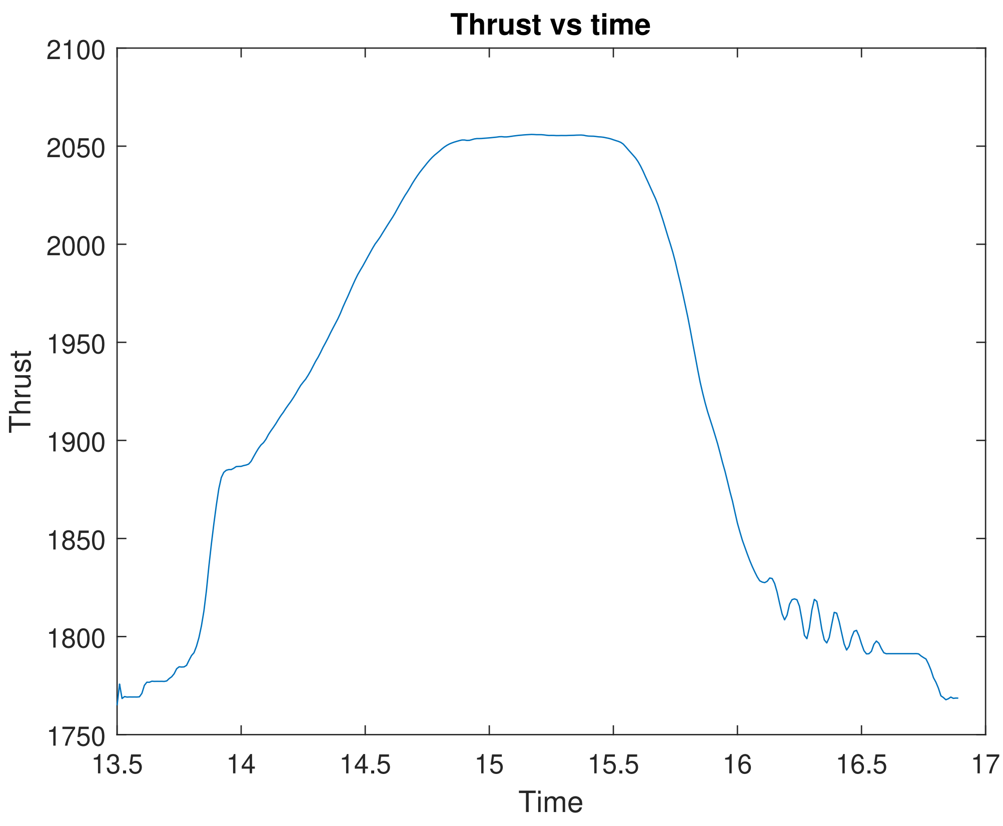


Figure 3.22: Output : Thrust Vs Time

3.1.19 Pyrotechnic Igniter & Battery

The solid rocket which[41] we made is of having 1.2 kg propellant and is small rocket motor. Pyrotechnic Igniters are best for small rocket motors. The propulsive force of a solid propellant motor is derived from the combustion of solid propellant at high temperature and pressure. The igniter induces the combustion reaction in a controlled and predictable manner by generating heat flux in the form of hot, dense[40] gases that rapidly ignite the propellant surface. The igniter also contributes towards the generation of a certain minimum pressure inside the motor that is adequate for stable and sustained combustion of the propellant.[42]

General requirements of an Igniter are:

1. The temperature of the propellant surface should be raised above its autoignition

temperature

2. Motor chamber pressure should be raised above a threshold pressure requirement for stable burning of a propellant

3. Ignition delay (time delay from application of an electrical pulse to the point when 10 per cent of motor peak pressure is achieved), should be within specified limits

4. The rate of pressure rise, dp/dt, in the motor chamber during ignition transient should not be unduly high leading to undesirable peaks or shock loads. On the other hand, it should not be too low to cause instability, hang-fires, etc.,

5. The igniter should satisfy the functional, environmental and shelf-life/storage requirements

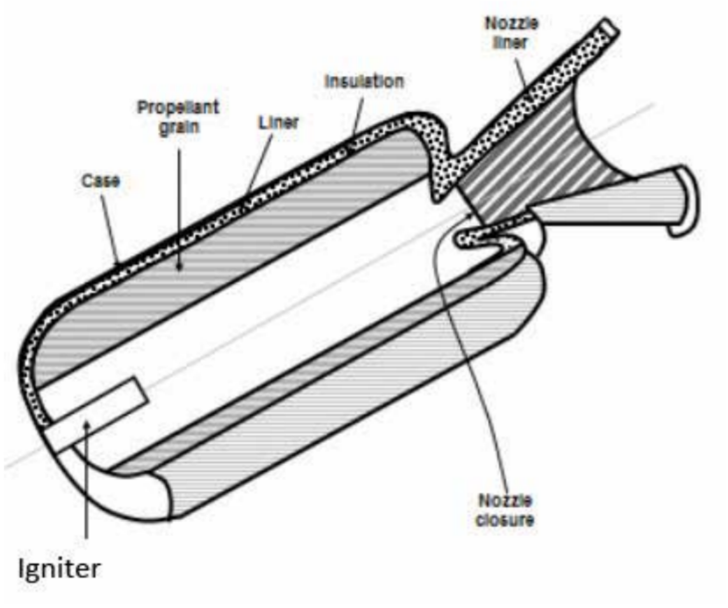


Figure 3.23: General Igniter Function

Pyrotechnic igniters are defined as igniters using pyrotechnic composition or energetic propellant like chemical formulations (usually in granule and/or small pellet form to give large burning surface area and short burn time) as a heat producing material.[43]

The constituents of igniter charge are : fuel, oxidizer, binder mainly and curing agent and other additives in some cases. Powdered Al, B, Mg are used as fuel. $KNO_3, KClO_3, KClO_4$, NH_4ClO_4, NH_4NO_3 are generally used as oxidizers.

Plastizedethyl cellulose is used as a binder which reduces sensitivity.

In ignition of solid rocket motors, a series of events takes place in a tightly timed sequence starting with:

1. Application of an electric pulse
2. Initiation of first fire element (squib or pyro-cartridge)
3. Generation of flash/flamelet
4. Amplification of flash by primer
5. Build up of flash to a strong flame by main igniter charge

Nozzle Throat Igniter

Throat-based igniters are also useful for compact rocket motors where space at nozzle divergent end cannot be used for configuring nozzle cap-based igniter due to length or diameter constraint, and therefore igniter is mounted on the throat of the motor. The material of construction is chosen to give appropriate ejection pressure and ensure smooth ejection of igniter through the throat without damaging it.[44][45][46]

1. Igniter is of ejectable type
2. Igniter hardware ensures sealing of gases at the throat using a seal
3. Charge mass consists of gunpowder, $BKNO_3$ granules and pellets, each one for its different purpose
4. O-Ring seating at the neck of the igniter provides sealing at the throat

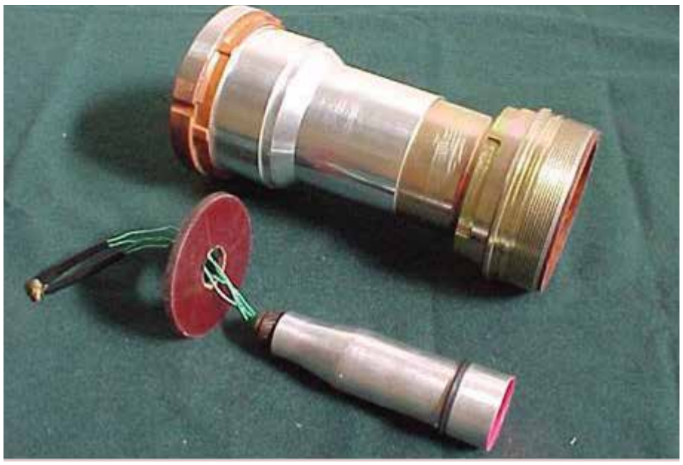


Figure 3.24: Throat-based igniter with nozzle

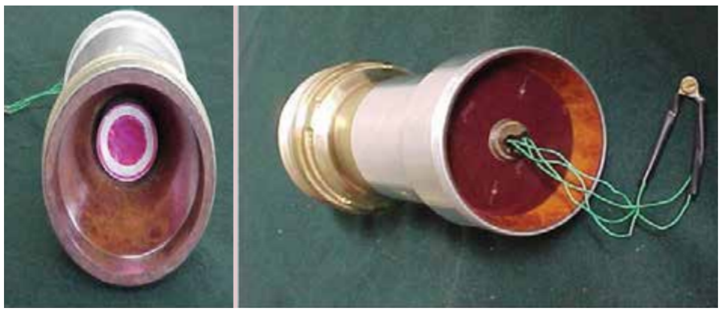


Figure 3.25: Throat-based igniter fitted inside nozzle (Throat view Divergent view)

Operational Features

Throat-based igniter is squib-based ignition system. The squib wires are taken out from the back of the igniter and is held tight at the divergent using a tapered disc butting the divergen to nozzle end. Electrically initiated squib ignites the granules, gunpowder and the pellets. Due to the pressure effect of gunpowder, burning pellets are discharged from the canister into the main combustion chamber. This ignites the propellant surface initiating sustainable ignition. Once adequate pressure is developed inside the chamber, the igniter is ejected from the throat by deforming the collar at which it was held at the throat. To ensure smooth ejection of igniter through the throat, soft material like aluminium is selected as a material of construction for igniter canister.

Throat-based igniters of this composition become very essential in high L/D motors where simultaneous ignition at the head end and nozzle end has to be assured. Similar to nozzle cap based type, this configuration also helps in increased propellant loading and simplified interfaces at the head end. These igniters are generally useful for solid propellant motors of long length.

3.1.20 Hydrotest

Hydro test or Hydrostatic test is a type of pressure test performed on piping and pressure vessels to check system integrity under pressure conditions. A hydrostatic test is performed by using water as the test medium. This test is conducted in Propulsion lab, The model design, experimental setup, results are as follows.,

Based on calculation, refer (3.5), The hydrotest setup is designed and fabricated

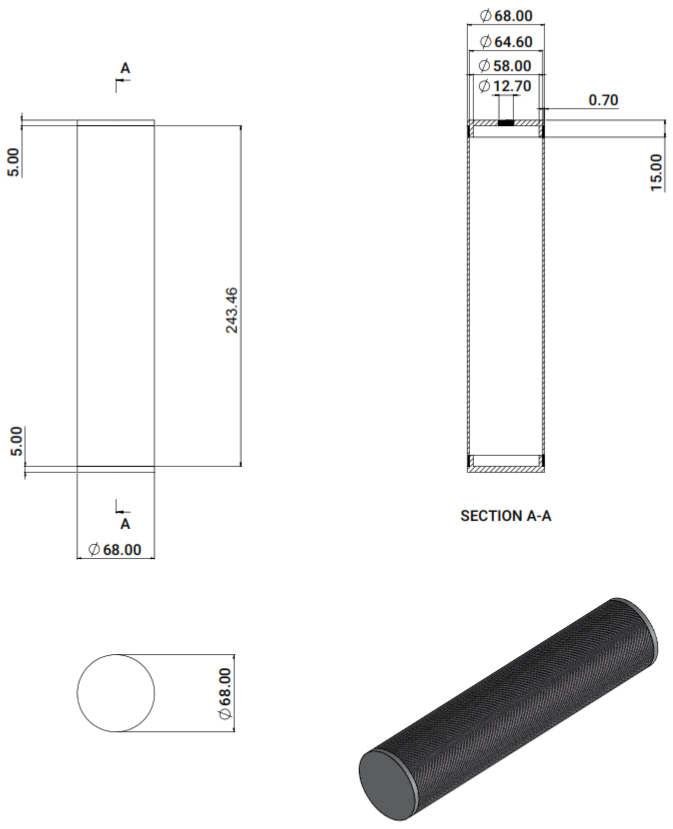


Figure 3.26: Rocket Motor Design

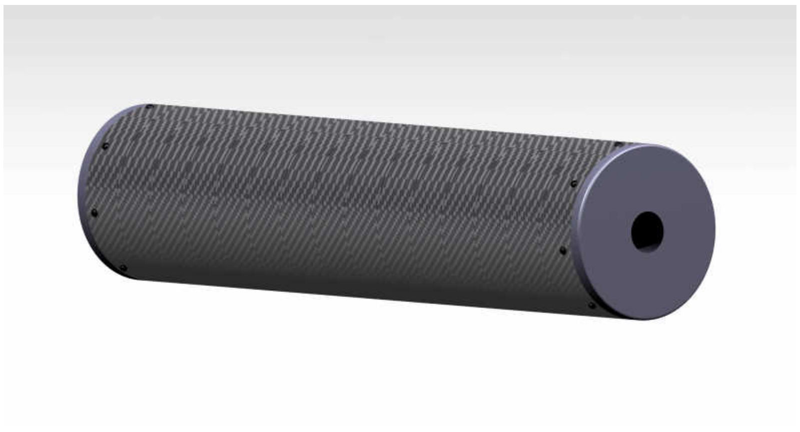
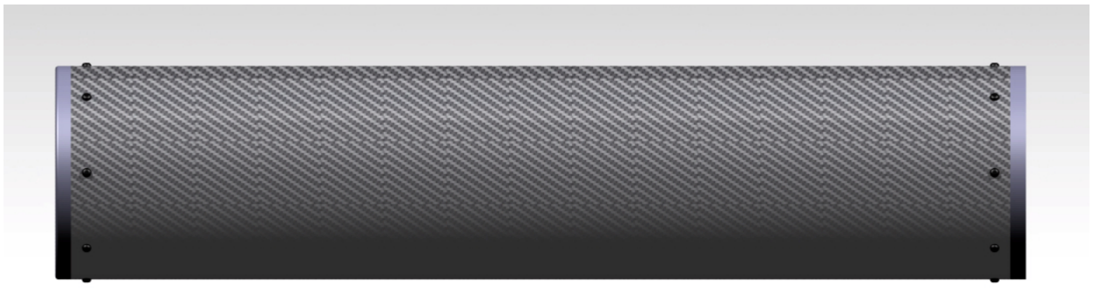
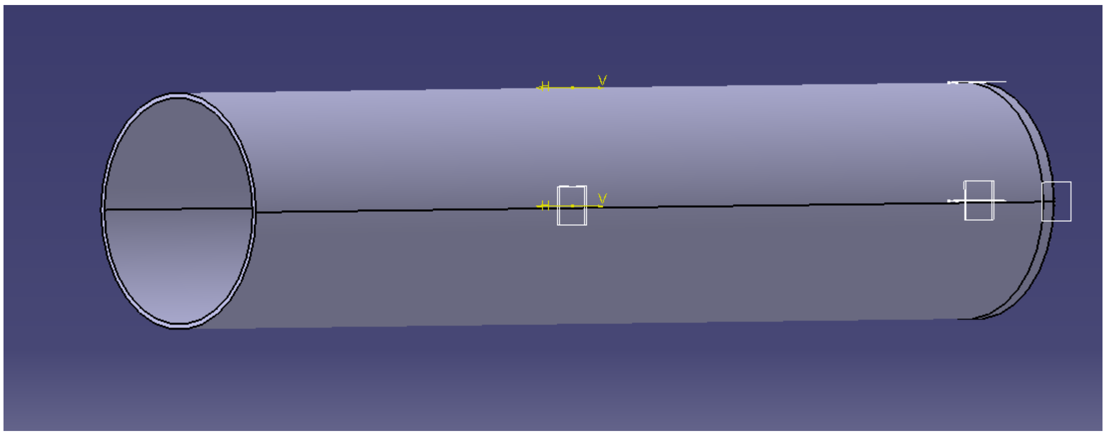
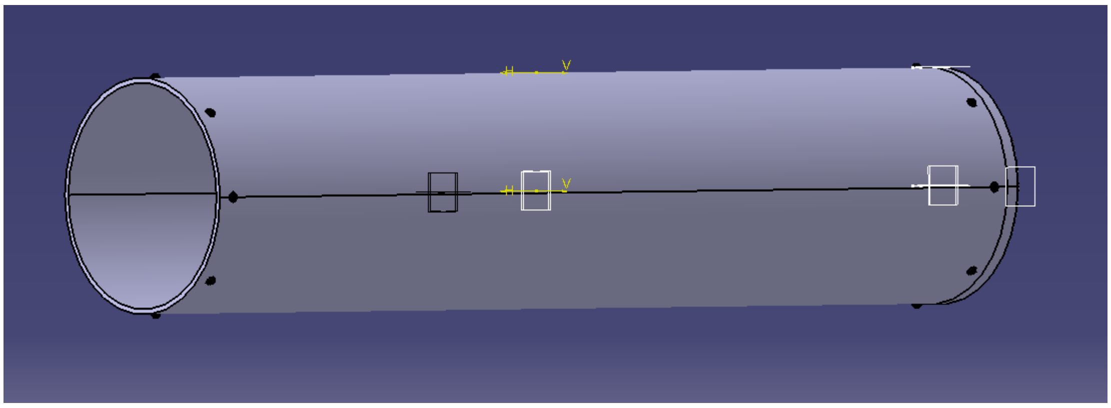


Figure 3.27: Rocket Motor CAD




H
V
H
V


H
V
H
V

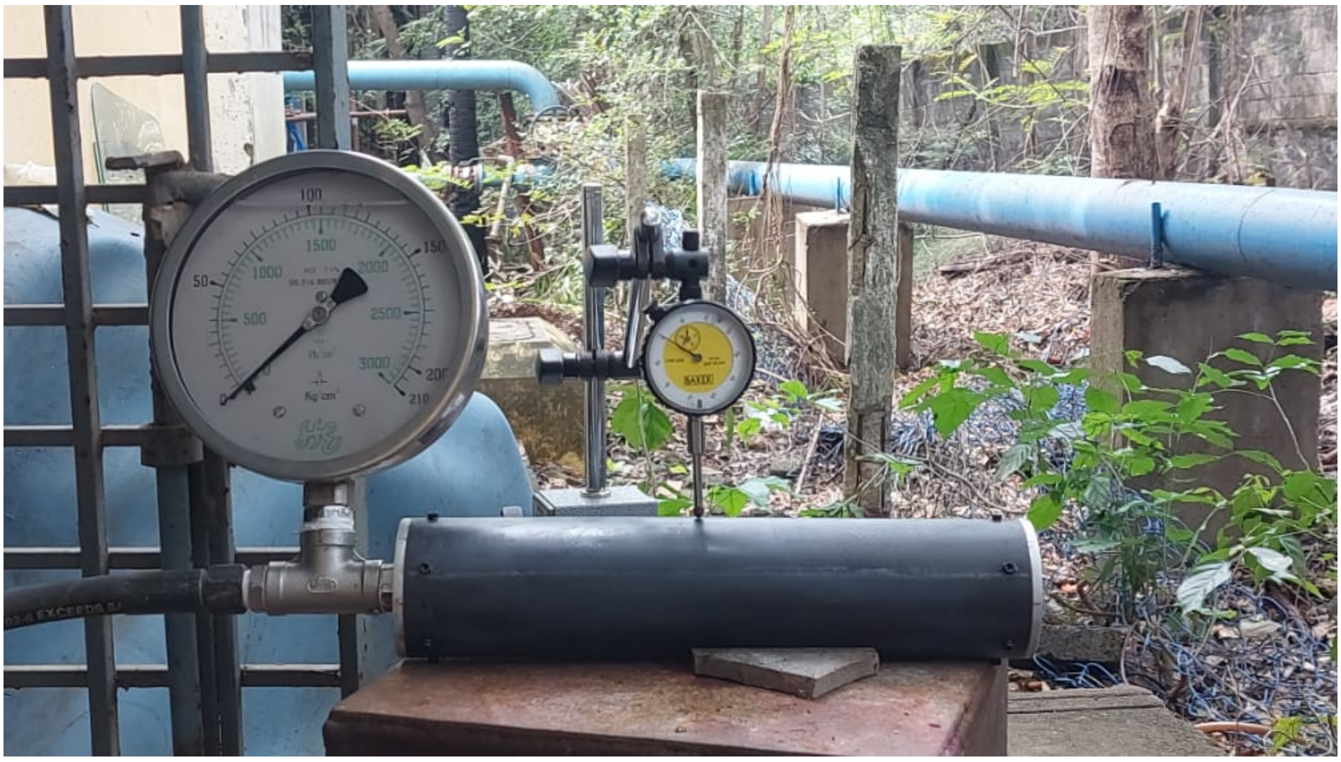


Figure 3.28: Hydrotest Setup

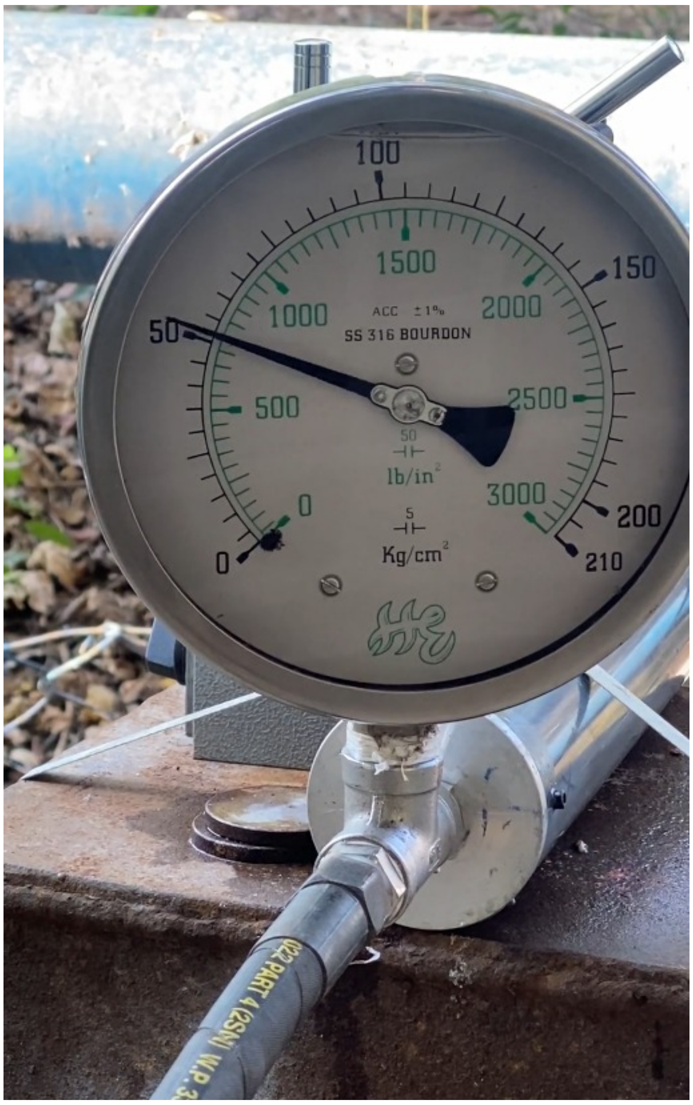


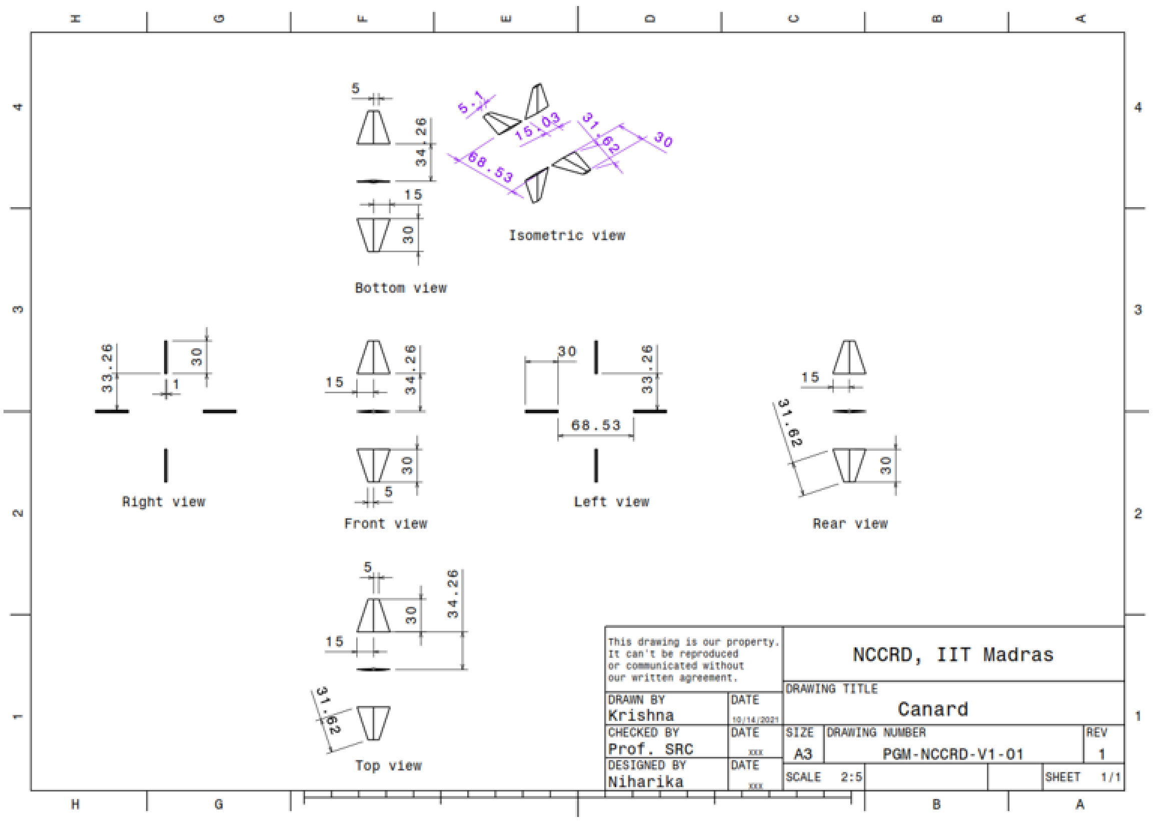
Figure 3.29: Result

Thus, the fabricated Rocket Motor successfully able to handle 50 bar pressure.

3.2 SUB-SYSTEMS OF MISSILE

3.2.1 Canard Design

The Canard Design, and Canard Data is obtained from reference of previous work of this project, done by Niharika through her duel degree thesis developing a guidance system for a low cost missile, and the same are summared below.



This drawing is our property. It can't be reproduced or communicated without our written agreement.

NCCRD, IIT Madras			
DRAWING TITLE	Canard		
DRAWN BY Krishna	DATE 10/14/2021		
CHECKED BY Prof. SRC	DATE xxx	SIZE A3	DRAWING NUMBER PGM-NCCRD-V1-01	REV 1
DESIGNED BY Niharika	DATE xxx	SCALE 2:5	SHEET 1/1

Figure 3.30: Canard Design

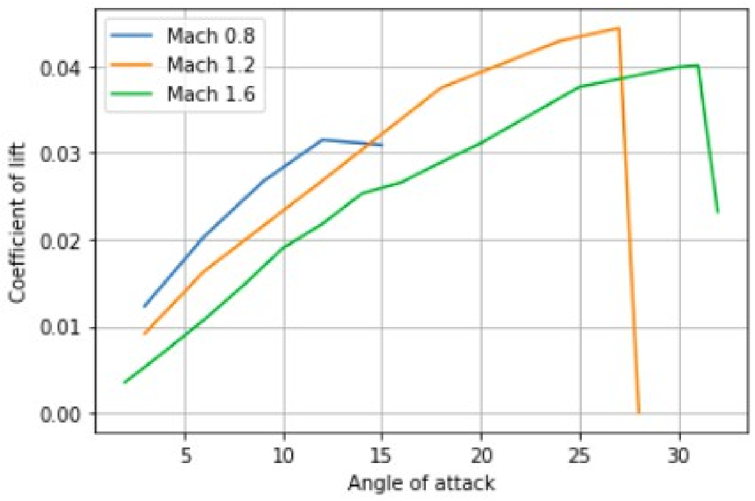


Figure 3.31: C_l vs AOA

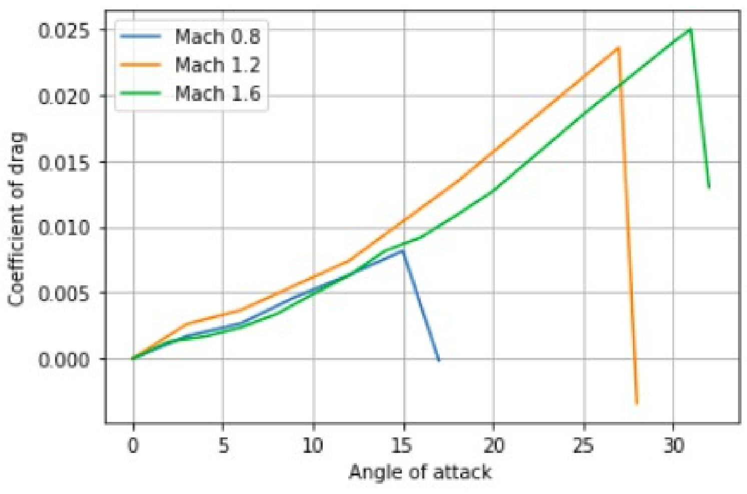


Figure 3.32: C_d vs AOA

3.2.2 Canard Mechanism

For Design of Canard Mechanism, we have taken reference of the following images from T minus arduino.

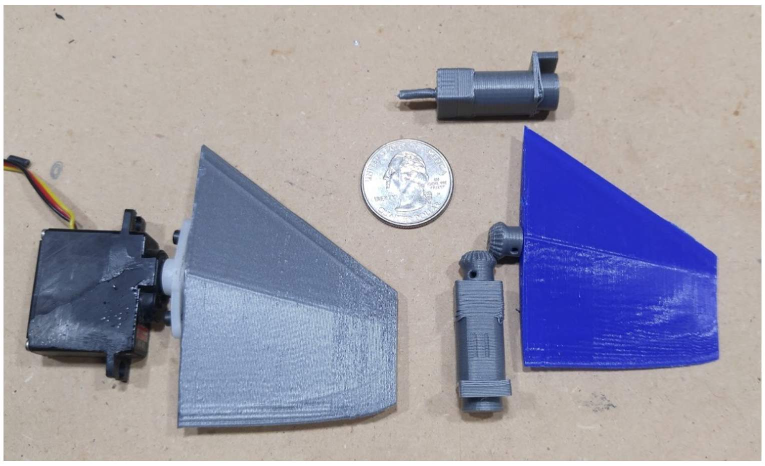
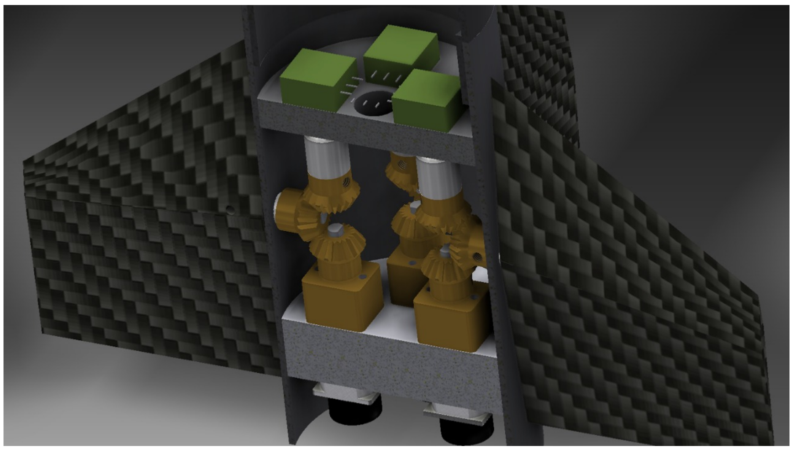
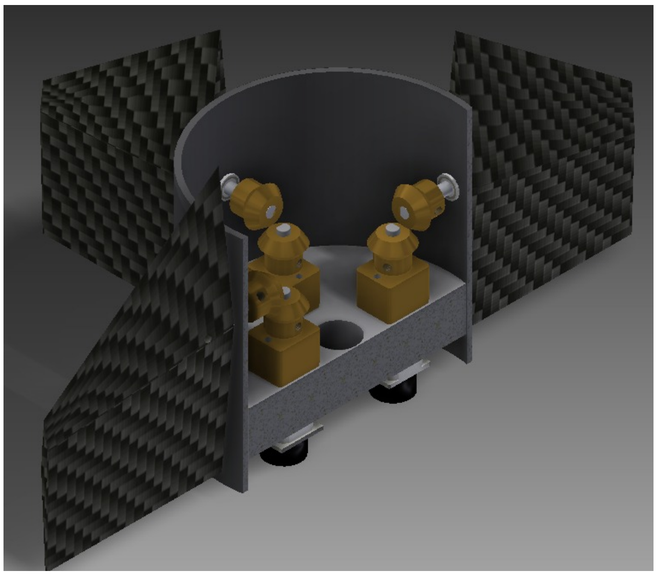
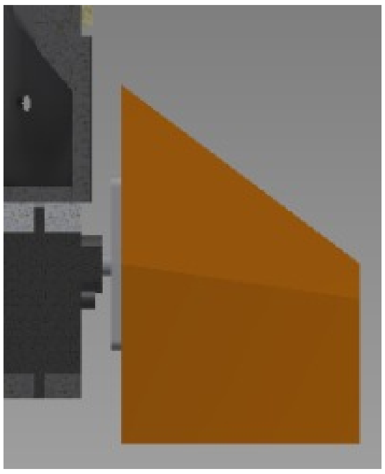
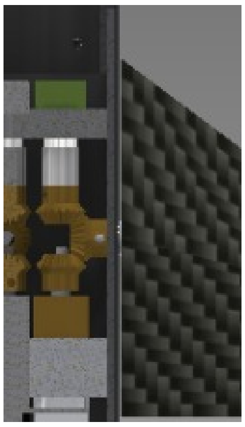


Figure 3.33: Reference : Canard Mechanism









The first design of canard mechanism, is to control the canards using servos. This design contains 2 parts.

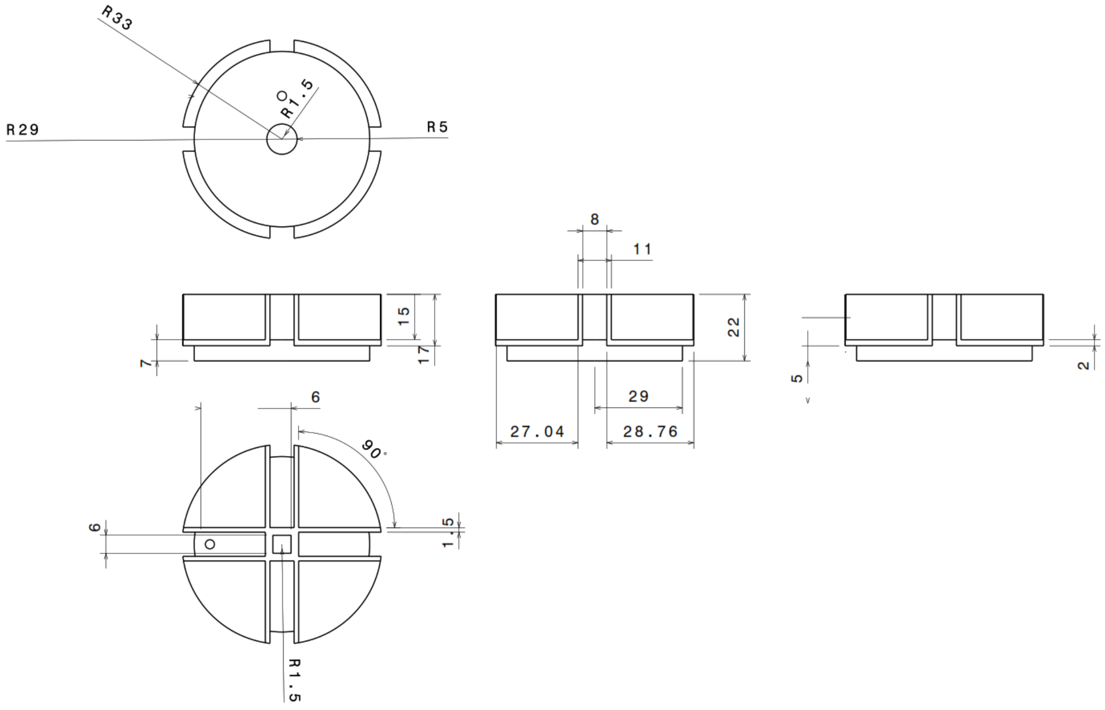
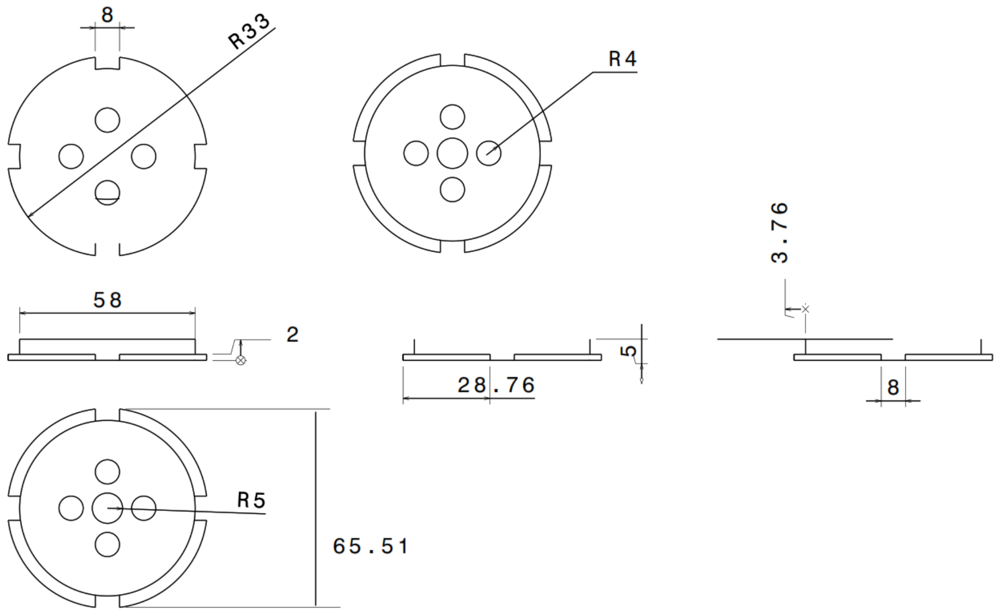


Figure 3.34: Design : Canard Mechanism 1



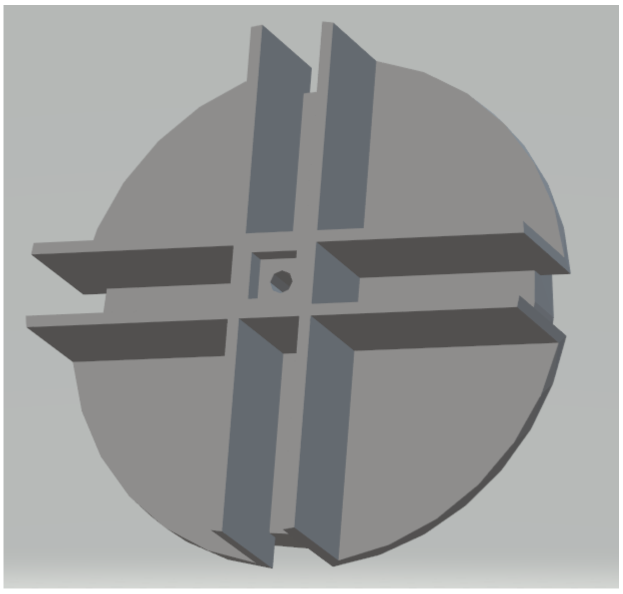
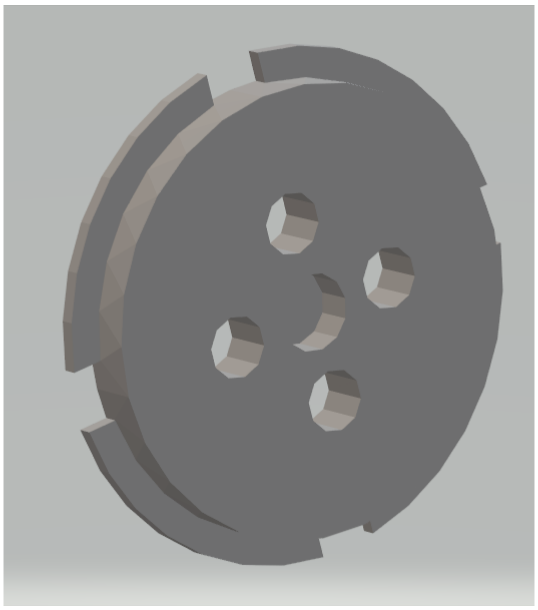


Figure 3.35: Canard Mechanism 1



The second design of canard mechanism, is made to use servos with (3.0 - 3.5 kgf.cm Torque, and Rate 0.15 / 0.13 s / 60 degrees at 5 Volts) and to make mechanism design with single part.

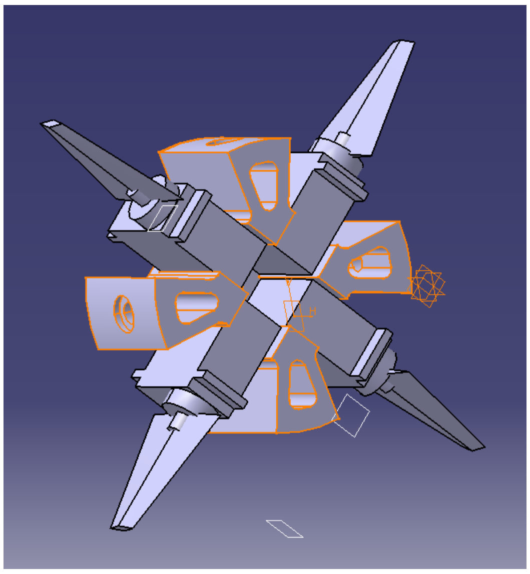


Figure 3.36: Design : Canard Mechanism 2

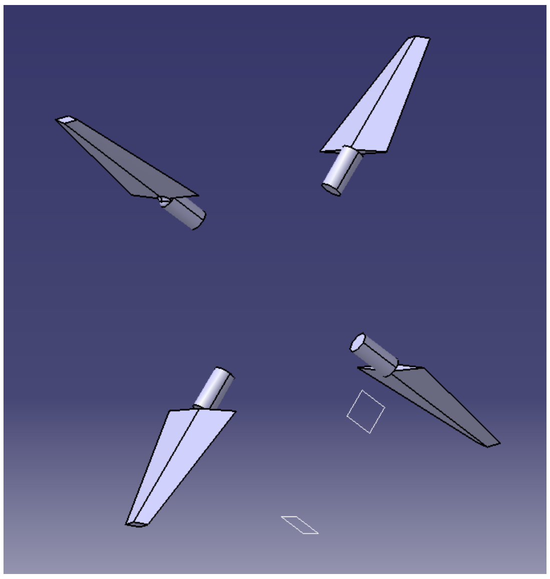


Figure 3.37: Canard Series Assembly A-1

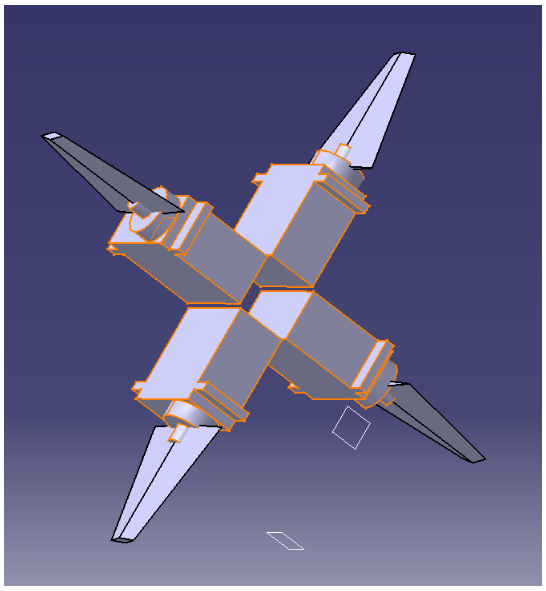


Figure 3.38: Canard Series Assembly A-2

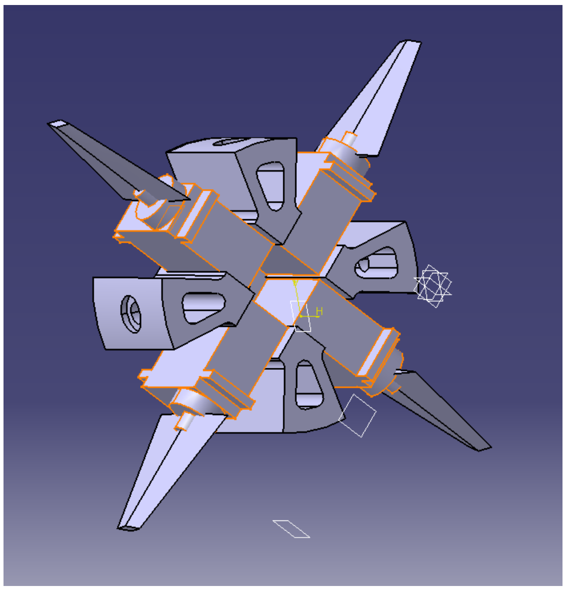


Figure 3.39: Canard Series Assembly A-3

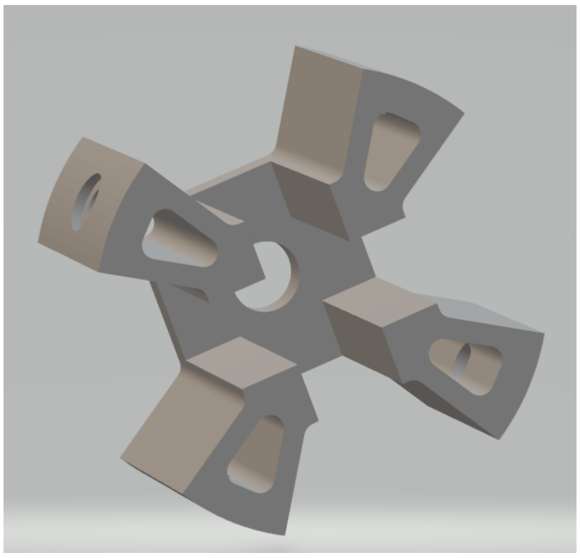
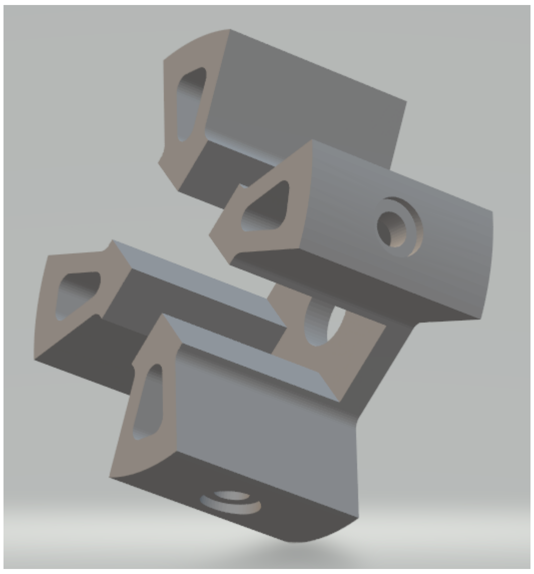
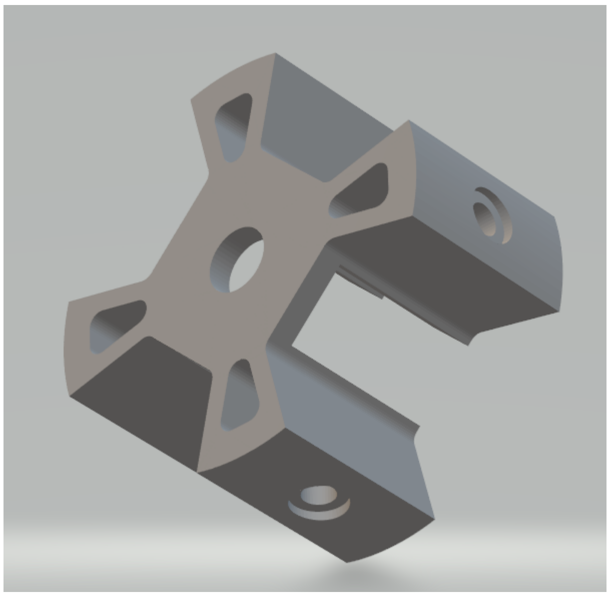


Figure 3.40: Canard Mechanism 2





3.2.3 Payload Profile

Payload is the impact fuse bomb of 2.5 kg for Missile V1, V2, and to be 2.1 kg for Missile V3, V4.

The dimensions of the specification of impact fuse payload bomb, remain same for all 4 versions of missiles, as given in figure 3.41

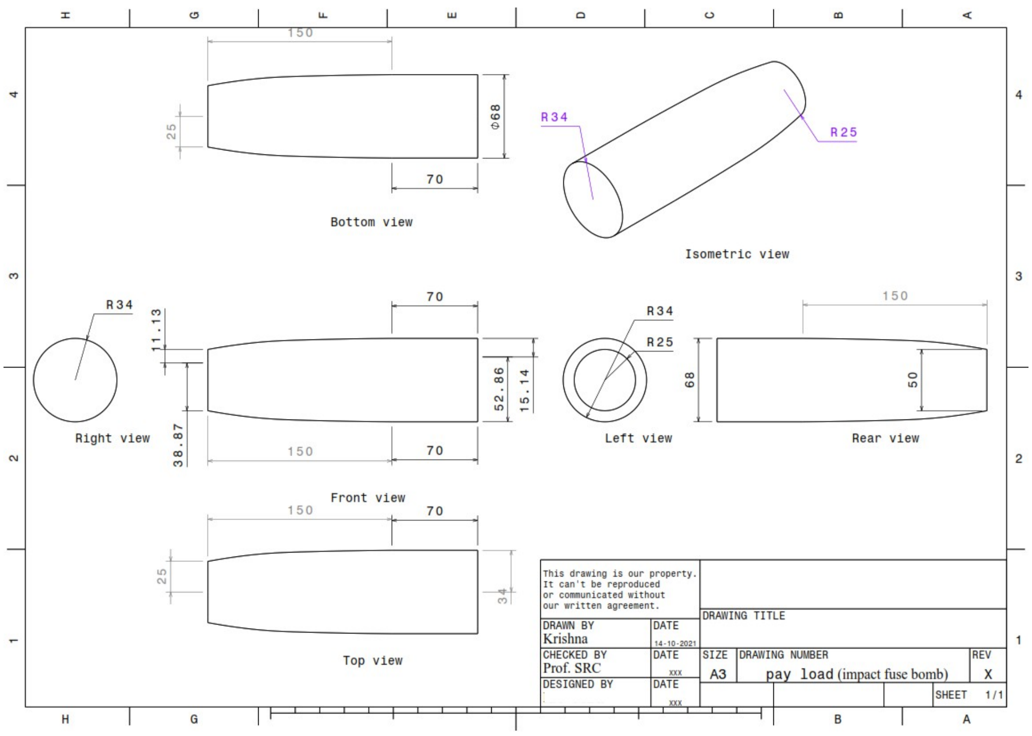


Figure 3.41: Payload Dimensions

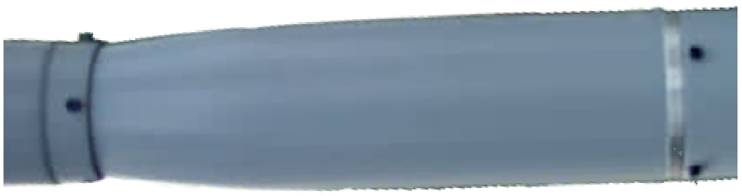


Figure 3.42: Payload

3.2.4 Gimbal Design

Initial version of gimbal is shown in 3.43

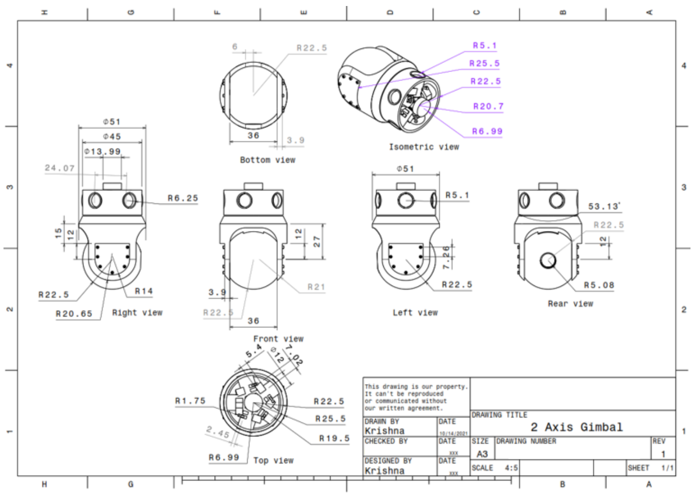


Figure 3.43: Gimbal : Version 0

The 2 Axis gimbal is as follows in figure . This Gimbal is finally designed after so many iterations of design and testing as shown in figures 3.44 , 3.46 , 3.48.

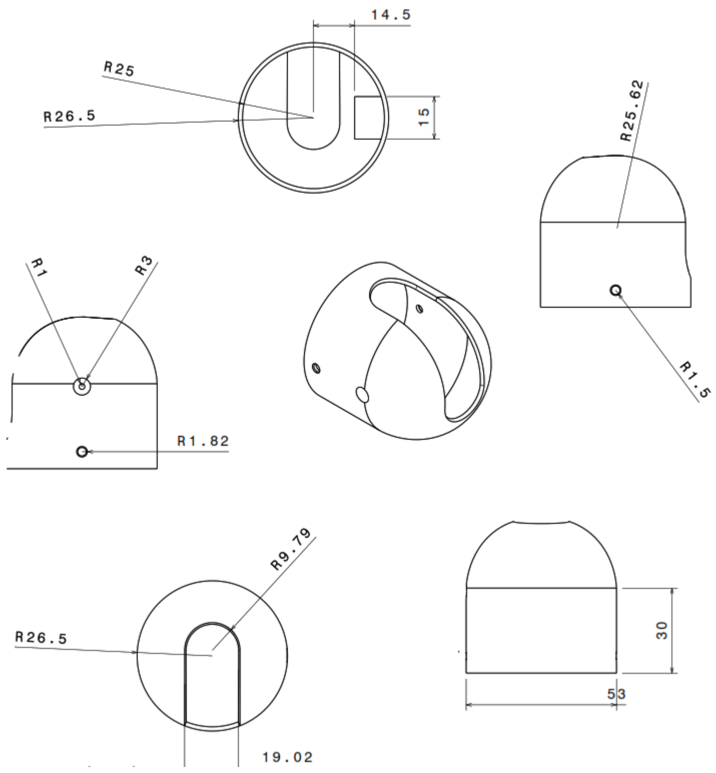


Figure 3.44: Gimbal : Master Frame

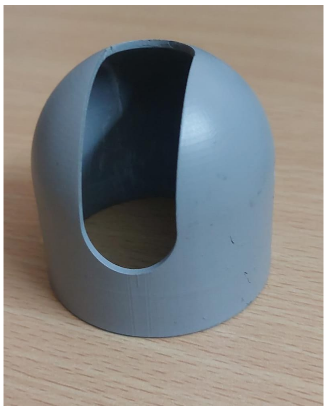
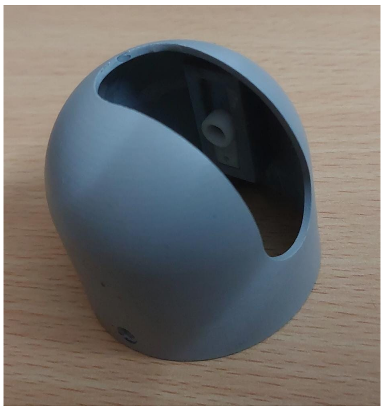


Figure 3.45: Fabricated : Gimbal : Master Frame



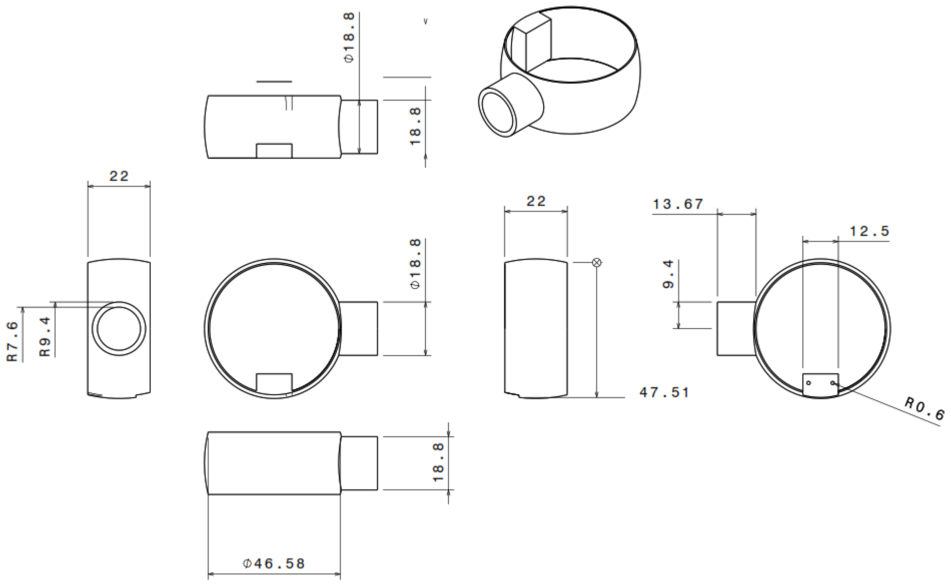


Figure 3.46: Gimbal : Rolling Structure

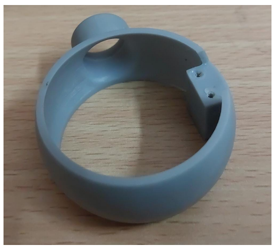
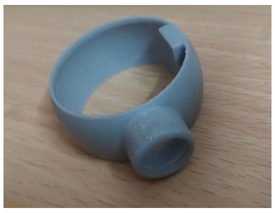


Figure 3.47: Fabricated : Gimbal : Rolling Structure



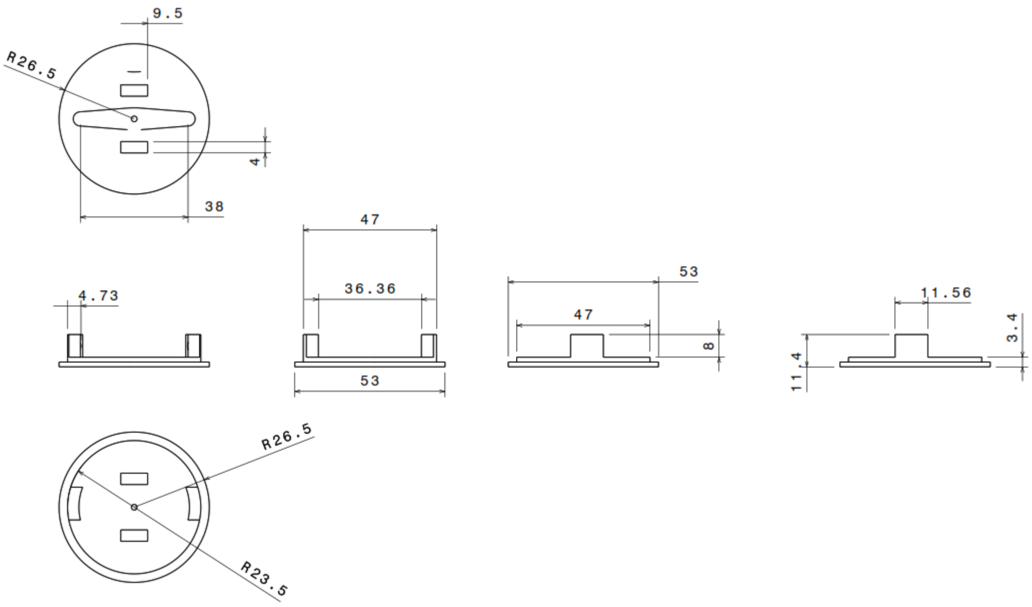


Figure 3.48: Gimbal : Base Plate

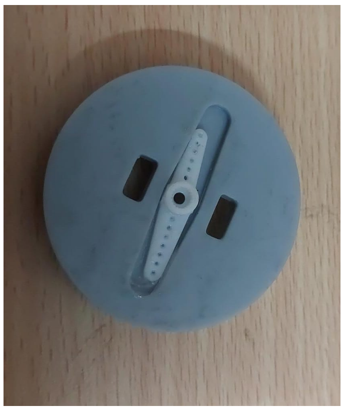
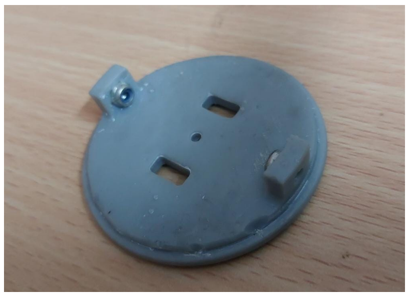


Figure 3.49: Fabricated : Gimbal : Base Plate



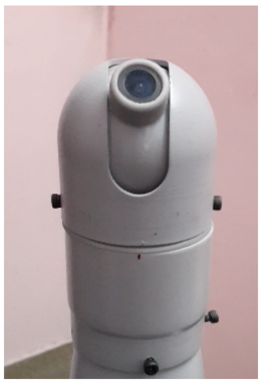


Figure 3.50: Fabricated : Assembled : Gimbal

3.2.5 Object Tracking

The camera will work with RGB color mode, which can be understood by thinking of it as all possible colors that can be made from three colored lights for red, green, and blue. This will be read as[47] Blue, Green, Red by the software designed. To detect the color to track, use code, in appendix C.1. As described with BGR, a pixel is represented by 3 parameters, blue, green, and red. Each parameter usually has a value from 0–255 (or O to FF in hexadecimal). For example, a pure blue [48]pixel on your computer screen would have a B value of 255, a G value of 0, and a R value of 0.

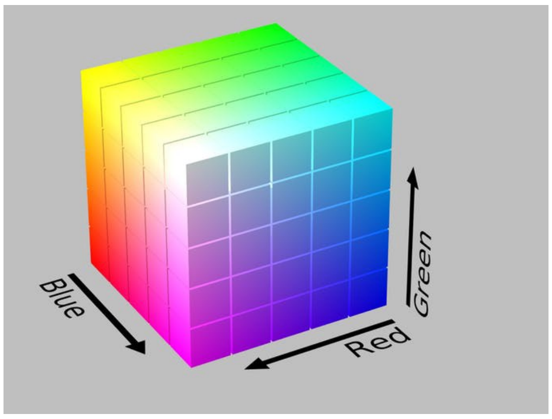


Figure 3.51: BGR Color code

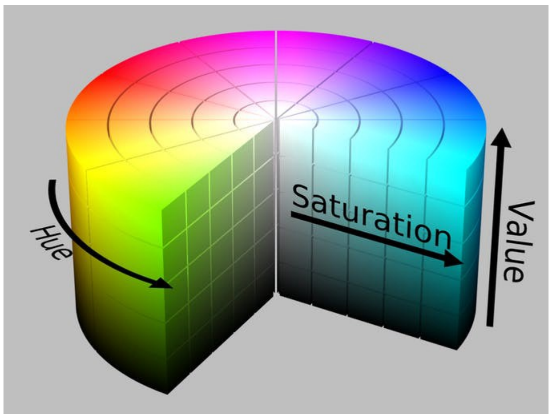
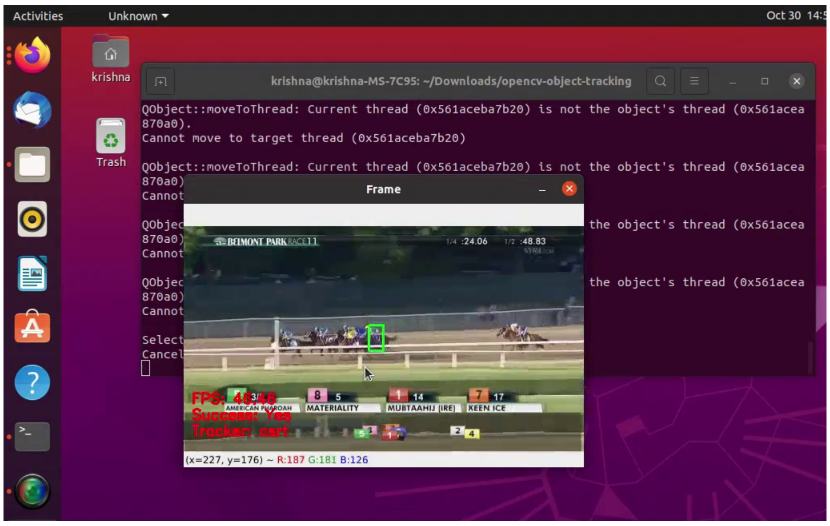


Figure 3.52: HSV Color code

OpenCV works with HSV (Hue, Saturation, Value) color model, that it is an alternative representation of the RGB color model, designed in the 1970s by computer graphics researchers[49] to more closely align with the way human vision perceives color-making attributes. To Track certian color using OpenCV, HSV model must be used. [50]

The code to convert BGR value to HSV value is attached in Appendix C.2. To get the BGR values of the interested pixels, use code in Appendix C.1

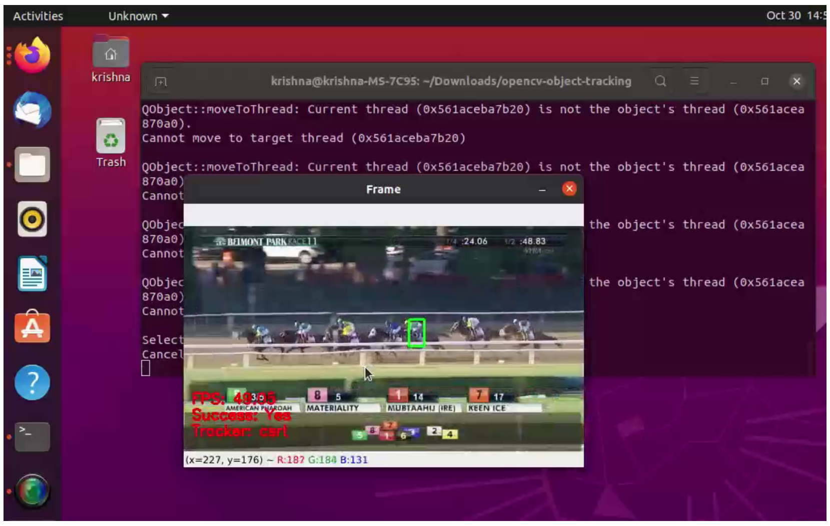
Now, The tracking of selected (Region of Interest) ROI in recorded video using the OpenCV csrt algorithm[51], code is given in appendix C.3



```
QObject::moveToThread: Current thread (0x561aceba7b20) is not the object's thread (0x561acea
870a0).
Cannot move to target thread (0x561aceba7b20)

QObject::moveToThread: Current thread (0x561aceba7b20) is not the object's thread (0x561acea
870a0)
Cannot

QObjec                                                       the object's thread (0x561acea
870a0)
Cannot

QObjec                                                       the object's thread (0x561acea
870a0)
Cannot

Select
Cancel
```

```
FPS: 46.48
Success: Yes
Tracker: csrt
(x=227, y=176) ~ R:187 G:181 B:126
```

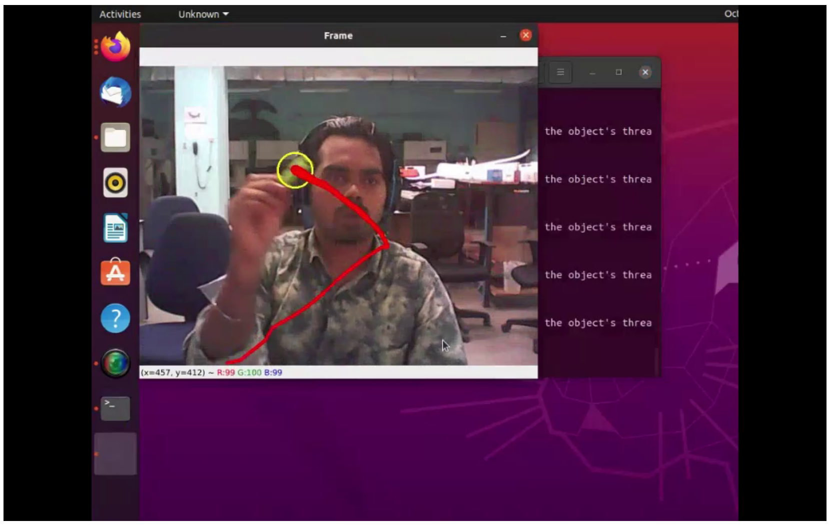
Figure 3.53: ROI Tracking : Recorded Video



```
QObject::moveToThread: Current thread (0x561aceba7b20) is not the object's thread (0x561acea
870a0).
Cannot move to target thread (0x561aceba7b20)

QObject::moveToThread: Current thread (0x561aceba7b20) is not the object's thread (0x561acea
870a0)
Cannot

QObjec                                                       the object's thread (0x561acea
870a0)
Cannot

QObjec                                                       the object's thread (0x561acea
870a0)
Cannot

Select
Cancel
```

```
FPS: 40.05
Success: Yes
Tracker: csrt
(x=227, y=176) ~ R:187 G:184 B:131
```

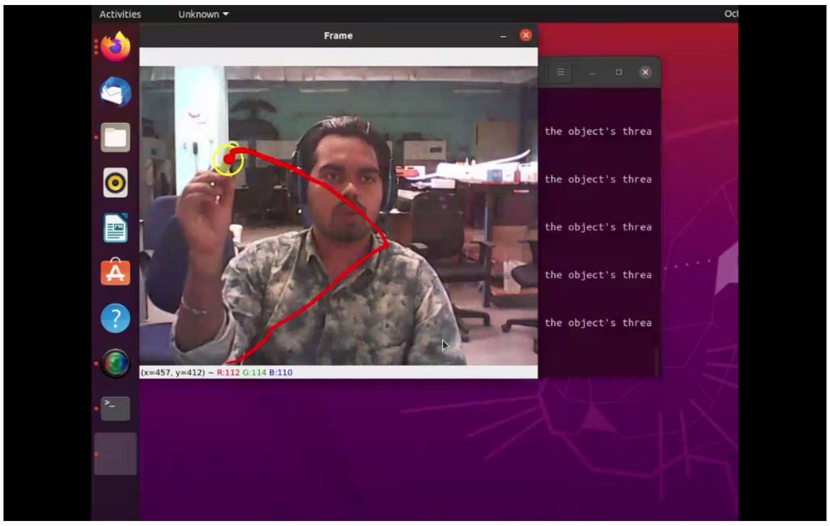
The tracking of object in the frame of live video using openCV csrt algorithm, using the code in appendix C.4



```
the object's threa

the object's threa

the object's threa

the object's threa

the object's threa
```

```
(x=457, y=412) ~ R:99 G:100 B:99
```

Figure 3.54: ROI Tracking : Live Video



```
the object's threa

the object's threa

the object's threa

the object's threa

the object's threa
```

```
(x=457, y=412) ~ R:112 G:114 B:110
```

The live position of interested object in the frame can be fetched by the python code, attached in appendix C.5

Using the code in appendix C.6, The live tracking of interested object to the center of frame is possible using the respective control of 2 axis pan tilt servos. The PWM signal to servo is controlled in such a way that the selected ROI object stays at centre of the image frame.

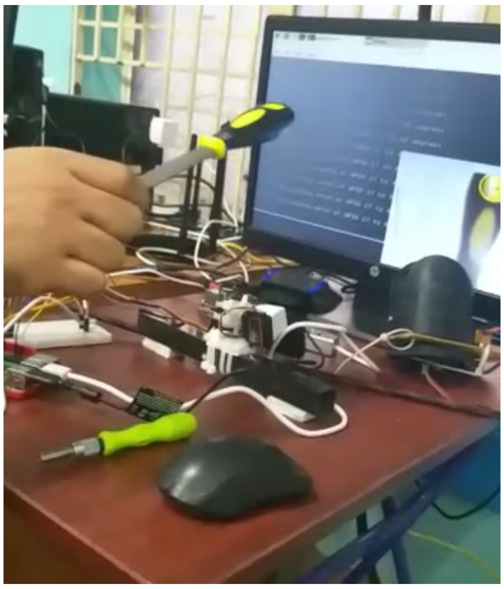
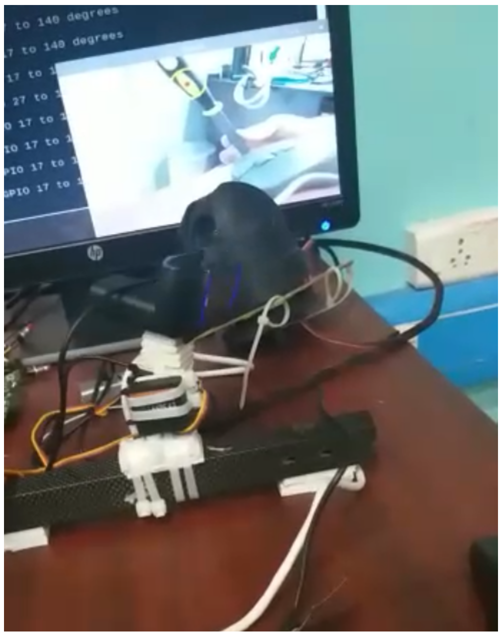


Figure 3.55: Gimbal Tracking



3.2.6 Servo Motor Controller

The Canards and Gimbal is controlled by servos, A servo controller is essential to control a servo. A servo controller is a circuit that is used to control the position of a servo motor by the input of PWM signal which is encoded with position or angle parameters.[52]

Micro Maestro 6-Channel USB Servo Controller is best servo controller for the current missile design, since it is has required 6 channels, (4 for canard servos, 2 for gimbal servos). It is light weight (4.8 grams), compact, (2.16 × 3.05 cm), upgradable firmware. [53]

Channels:	**6**
Minimum operating voltage:	5 V
Maximum operating voltage:	16 V
Supply current:	30 mA

Table 3.8: Servo Controller

Figure 3.56: Micro Maestro 6-channel USB servo controller

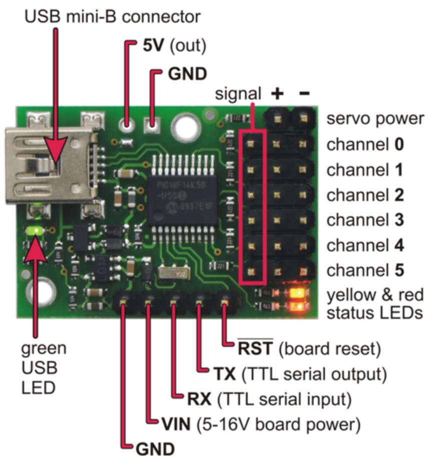


Figure 3.57: Top View : Micro Maestro 6-channel USB servo controller

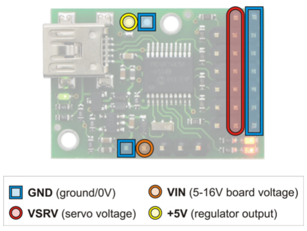


Figure 3.58: Micro Maestro Power Pins

3.2.7 Missile Computer

NVIDIA JETSON TX2 exceptionally high compute, accuracy, and power efficiency in a module the size of a credit card. Its small 50 mm x 87 mm size enables real deep learning applications in small form-factor products.

The purpose of MC is to run high-level mission planning and behavior processes and to exchange messages between processes, other vehicles, and the ground station. Furthermore, a key design goal of the system should be that it is flexible[54] in regard to the architecture and potential expansion. In the future, additional autonomous behaviors, sensors, and communication capabilities may be added. This functionality may even run across multiple processors on separate payload computers. Therefore, it is important that the software components be loosely coupled and communicate using open and flexible architectures.

The Single Board Computer (SBC) chosen for Missile Computer is NVIDIA JETSON TX2 4 GB

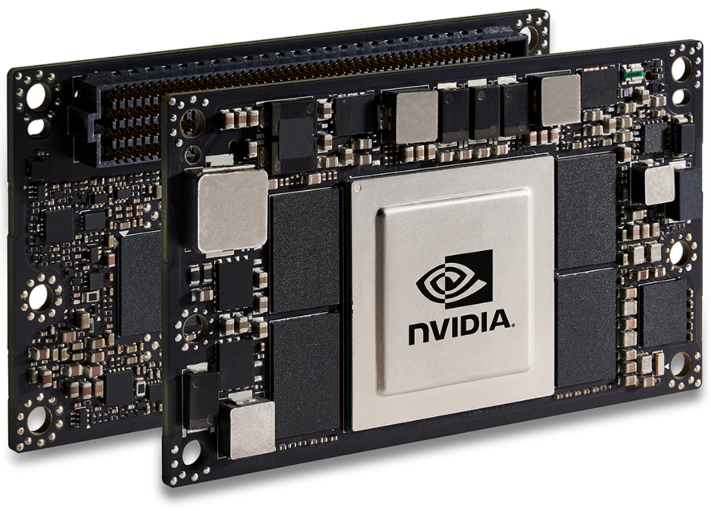


Figure 3.59: NVIDIA Jetson TX2 4 GB

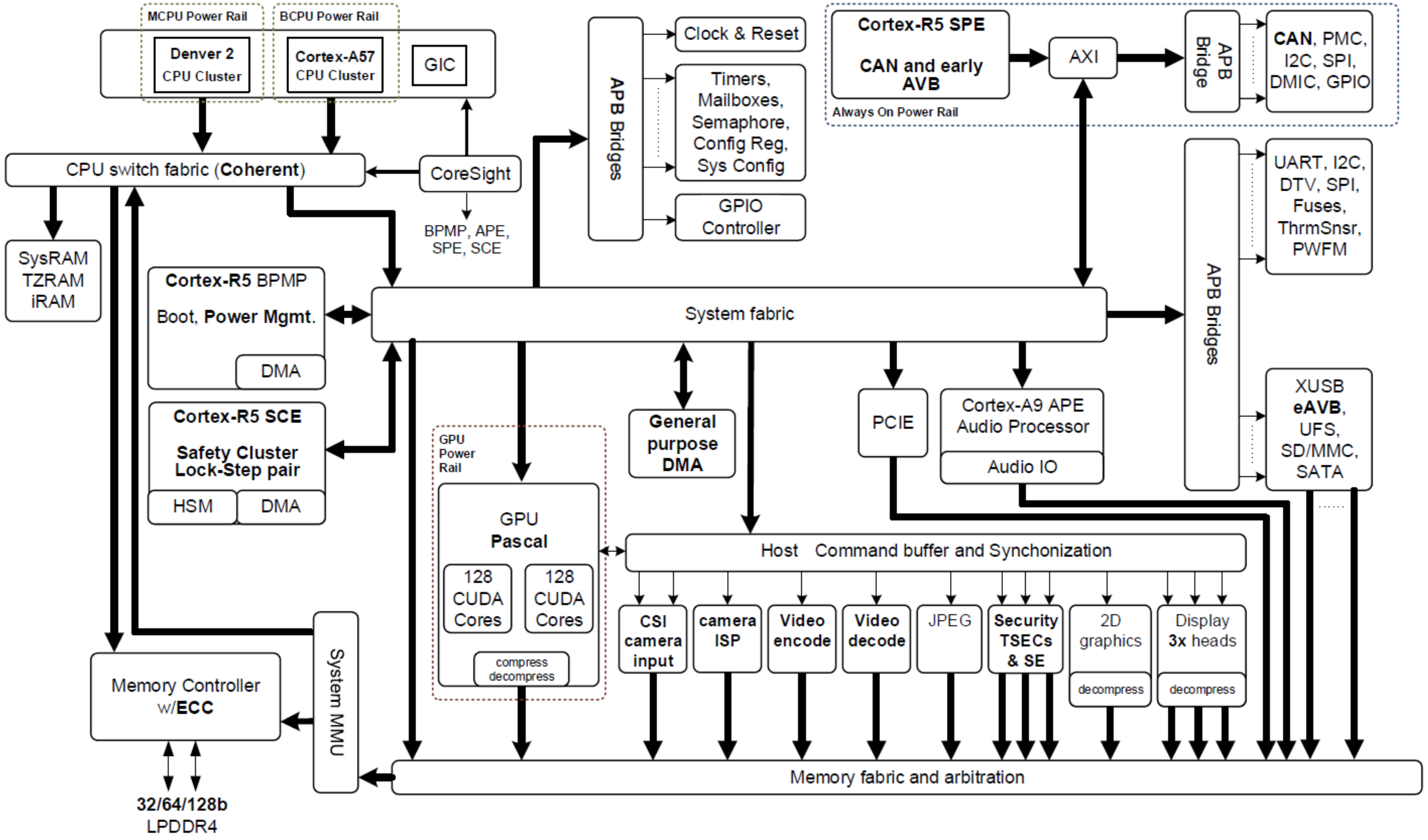


Figure 3.60: NVIDIA Jetson TX2 Architecture

Technical Specifications of Jetson TX2 4 GB	
GPU	NVIDIA Pascal™ architecture with 256 NVIDIA CUDA cores 1.3 TFLOPS (FP16)
CPU	Dual-core Denver 2 64-bit CPU and quad-core ARM A57 complex
Memory	4 GB 128-bit LPDDR4 1600 MHz - 51.2 GBs
Storage	16 GB eMMC 5.1
Video Encode	500 MP/sec 1x 4K @ 60 (HEVC) 3x 4K @ 30 (HEVC) 4x 1080p @ 60 (HEVC) 8x 1080p @ 30 (HEVC)
Video Decode	1000 MP/sec 2x 4K @ 60 (HEVC) 4x 4K @ 30 (HEVC) 7x 1080p @ 60 (HEVC) 14x 1080p @ 30 (HEVC)
Connectivity	Wi-Fi requires external chip
	10/100/1000 BASE-T Ethernet
Camera	12 lanes MIPI CSI-2, D-PHY 1.2 (30 Gbps)
Display	HDMI 2.0 / eDP 1.4 / 2x DSI / 2x DP 1.2
UPHY	Gen 2 | 1x4 + 1x1 OR 2x1 + 1x2, USB 3.0 + USB 2.0
Size	87 mm x 50 mm
Mechanical	400-pin connector with Thermal Transfer Plate (TTP)

Table 3.9: NVIDIA Jetson TX2 Specifications

3.2.8 Jetson TX2 Carrier Board : J120

The J120 carrier board turns the Jetson TX2 compute module into a super-mini-computer for desktop usage and for integration into UAVs and drones/missiles.It is very compact. It has the same height as the TX1 (50 mm) and extends to one side to make space for standard connectors for Gigabit Ethernet, two USB 3 type A and mini HDMI.

It features one M.2 type M slot for ultra fast SSDs (2280 form factor) which is connected via 4 PCle lanes to achieve a read and write performance up to 2500 Mbyte/s.

The J120 features CAN controllers which are connected via the SPI interface. A bug in the TX1 prohibits the use of spidev0.0. If it is accessed then the spidev0.1 is blocked. Please see the technical reference manual for details. So for CAN to work, the CAN controller 1 chip has to be physically removed. Only one CAN interface is supported (CAN2). So the J120 rev 3 is only equipped with one CAN interface. This is still under investigation and might be fixed in the future[55]

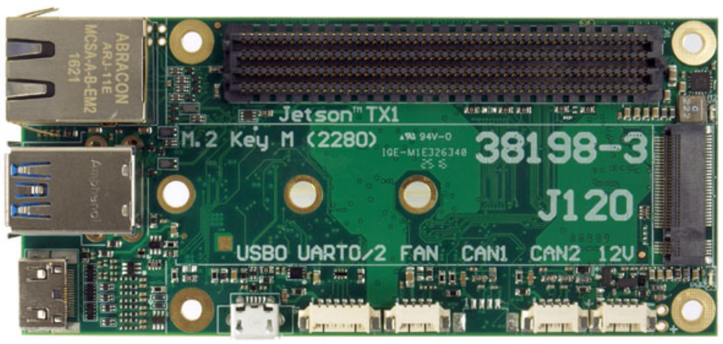


Figure 3.61: J120

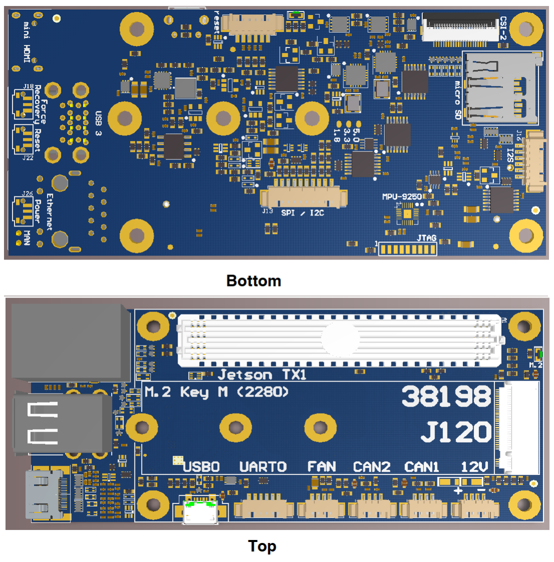


Figure 3.62: J120 Top & Bottom

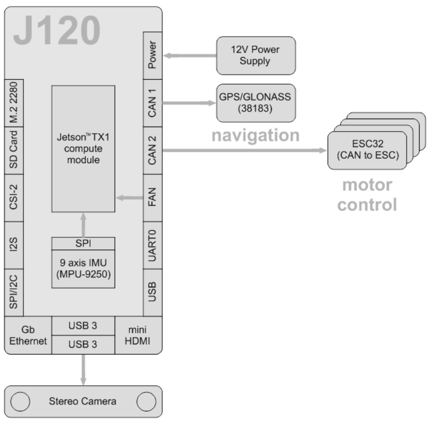


Figure 3.63: CAN centric flight controller

3.2.9 IMU

An IMU is a specific type of sensor that measures angular rate, force and sometimes magnetic field. IMUs are composed of a 3-axis accelerometer and a 3-axis gyroscope, which would be considered a 6-axis IMU. They can also include an additional 3-axis magnetometer, which would be considered a 9-axis IMU.

Sensor Name	Measures	Range & Units	Axes	Example Sensor
Accelerometer	Acceleration	± 200 g	3 axis	ADXL377 - High-G Triple-Axis Accelerometer (+-200g Analog Out)
Gyroscope	Angular velocity	± 2000 °/sec	3 axis	L3G4200D by STMicroelectronics
Magnetometer	Direction/position	± 1300 uT	3 axis	BMM150 by DFRobot Fermion

Table 3.10: 9 Axis IMU Sensor

3.2.10 Flight Control System (FCS)

Flight Control System (FCS) typically includes an autopilot, Global Positioning System (GPS), gyros, Inertial Measurement Unit (IMU), and a dedicated link to a ground station computer which can be used to send waypoint and control commands to the autopilot. The FCS also directly generates the control signals for aircrafts control surfaces and throttle. Note that the FCS should be able to operate independently and isolated from the mission computer by having one or more dedicated datalinks and power sources. This provides for a level of fault tolerance in the system, as well as enabling it to be easily extensible. In principle, the flight control system can operate as a "black box" and it does not matter where the commands originate, while the mission processor also operates as a separate entity and has little knowledge of the internal details of the flight control system. The Single Board Computer used to implement FCS is also called Flight Control Computer (FCC).

The Single Board Computer (SBC) chosen for implementing FCC is the PIXHAWK MODULE [56]

3.2.11 Dual Redundant Stores Management Processor

Stores Management Processor or SMP is a Single Board Computer implemented to control and execute weapon store related functionalities. The aim is to use a powerful, lightweight and a small form factor based computing device to implement stores management logic.[57] The SBC should support Real-Time task execution and should also have support for analog and discrete communication for interfacing with other SBCs. Various COTS SBC are available in open source market such as NVIDIA Boards, ESP32, Raspberry, Beaglebone Boards, etc. However after comparing all the above mentioned factors on each of the SBCs, we narrowed down to a Beaglebone Pocketbeagle SBC, which is extremely small and lightweight, cost effective and powerful.

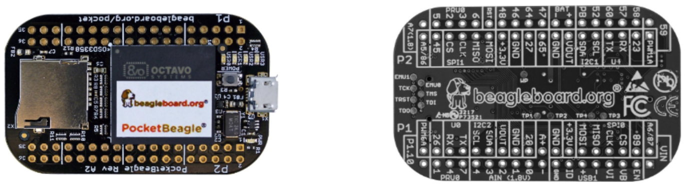


Figure 3.64: Pocketbeagle SBC from Beaglebone

PocketBeagle Technical Specifications:

* Based on new Octavo Systems OSD3358-SM 21 mm × 21 mm system-in-package that includes 512MB DDR3 RAM, 1-GHz ARM Cortex-A8 CPU, 2x 200-MHz PRUs, ARM Cortex-M3, 3D accelerator, power/battery management and EEPROM

* 72 expansion pin headers with power and battery I/Os, high-speed USB, 8 analog inputs, 44 digital I/O s and numerous digital interface peripherals

* microUSB host/client and microSD connectors

Processor: Octavo Systems OSD3358 1GHz ARM Cortex-A8

* 512MB DDR3 RAM integrated

* Integrated power management

* $2 \times 32-$ bit $200-$ MHz programmable real-time units (PRUs)

* ARM Cortex-M3

3.2.12 e-CAM132_TX2 Camera : for 2 Axis Gimbal

e-CAM132_TX2 - 13MP Autofocus Jetson ^{TM} TX2 camera board is a 4-lane MIPI CSI-2 camera for NVIDIA Jetson ^{TM} TX2. This camera is based on 1/3.2ÄR1335 CMOS image sensor with advanced 1.1μm pixel BSI technology from onsemi and an integrated high-performance image signal processor (ISP) that performs all the Auto functions (Auto White Balance, Auto Exposure control). e-CAM132_TX2 is a two board consists of a base board and a 13MP autofocus camera board. This camera board is connected to the base board using 30 cm micro-coaxial cable.

Camera module with Adaptor board - 3.5 Grams
Base board - 13.5 Grams
Coaxial cable - 4 Grams

Resolutions	**Uncompressed YUV422**
VGA	120 fps
HD (720p)	80 fps
Full HD (1080p)	80 fps
4K Ultra HD (3840x2160)	30 fps
13MP (4192 x 3120)	19 fps

Table 3.11: Maximum Image Transfer Rate

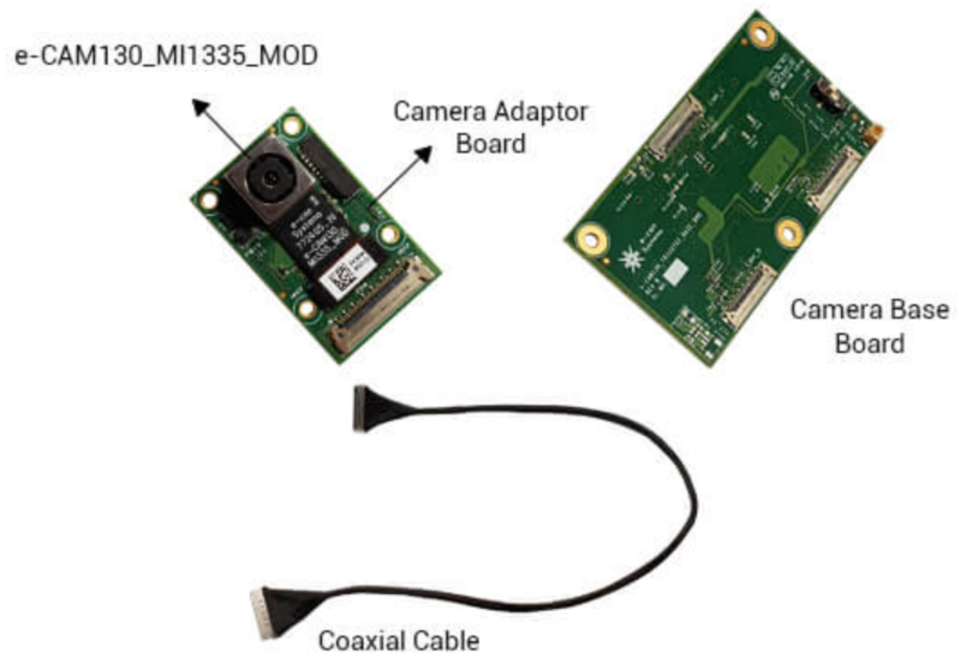


Figure 3.65: e-CAM132_TX2 Camera Module

3.2.13 FPGA Architecture

A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by a customer or a designer after manufacturing – hence the term field-programmable.[64] The FPGA configuration is generally specified using a hardware description language (HDL), similar to that used for an application-specific integrated circuit (ASIC).[60] Circuit diagrams were previously used to specify the configuration, but this is increasingly rare due to the advent of electronic design automation tools. [65]

FPGA emerged from relatively simpler technologies such as programmable read-only memory (PROM) [63]and programmable logic devices (PLDs) like PAL, PLA, or Complex PLD (CPLD).

The general FPGA architecture consists of three types of modules. They are I/O blocks or Pads, Switch Matrix/ Interconnection Wires and Configurable logic blocks (CLB). The basic FPGA architecture has two dimensional arrays of logic blocks with a means for a user to arrange the interconnection between the logic blocks. [59]

1. Configurable Logic Blocks :: which implement logic functions, includes digital logic, inputs, outputs. It implements the user logic.

2. Programmable Interconnects :: which implement routing. Interconnects provide direction between the logic blocks to implement the user logic. Depending on the logic, switch matrix provides switching between interconnects.[61]

3. Programmable I/O Blocks :: which connect with external components. These are used for the outside world to communicate with different applications.[62]

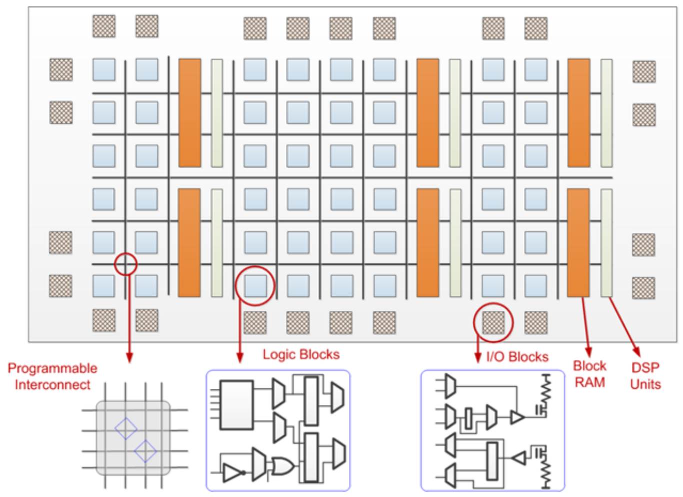


Figure 3.66: FPGA Architecture

Figure 3.67: Example FPGA Circuit board

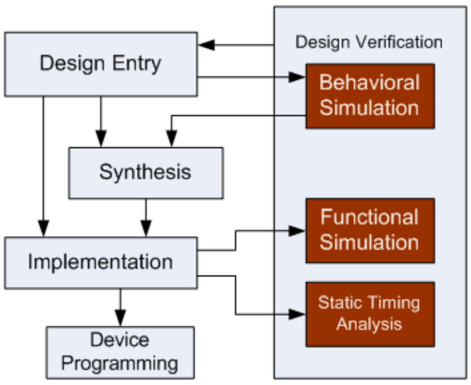


Figure 3.68: Our initial FPGA Architecture

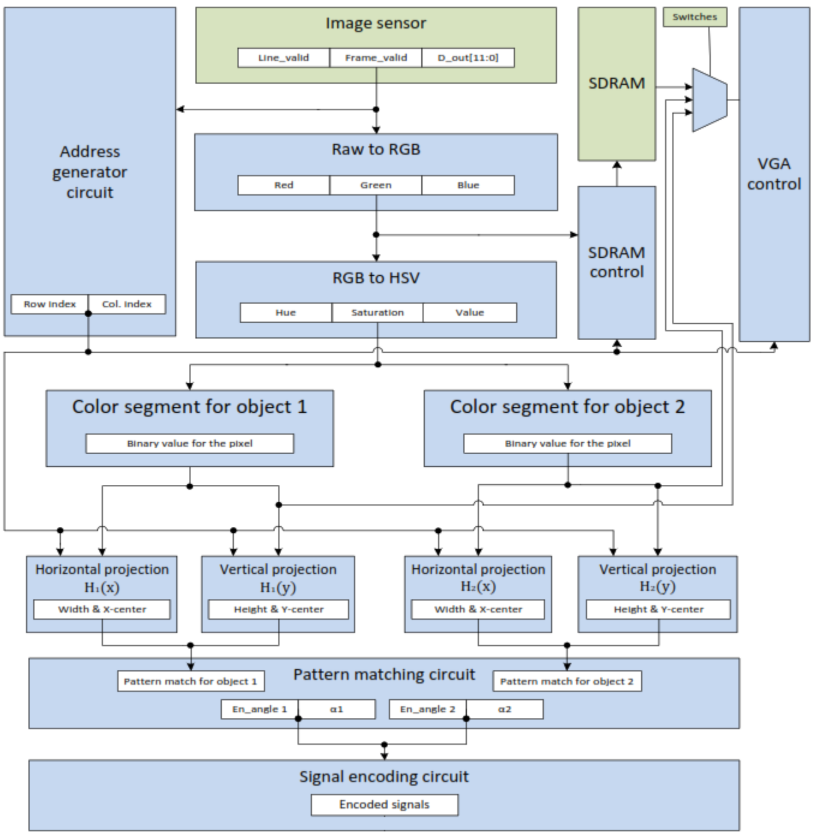


Figure 3.69: FPGA Design Workflow

3.3 DESIGN OF ACCESSORIES OF MISSILE

3.3.1 Missile Mould

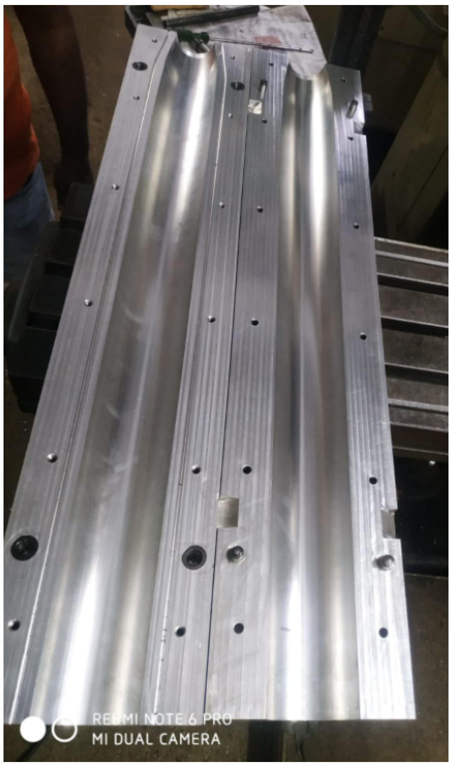
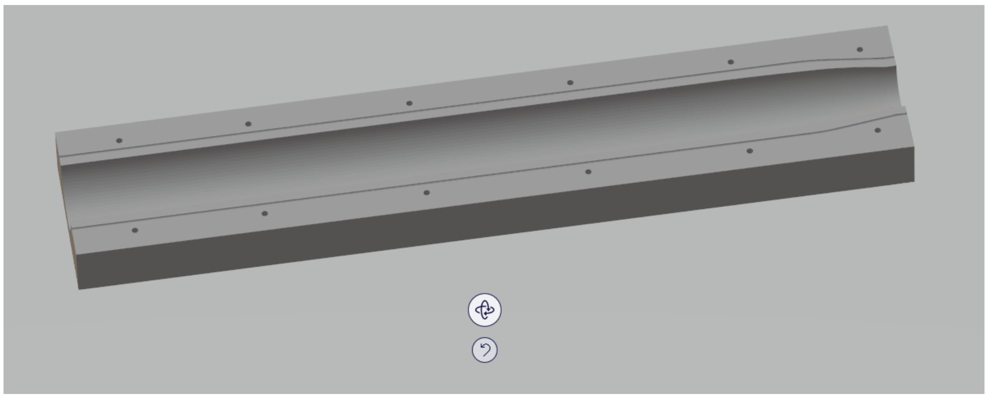
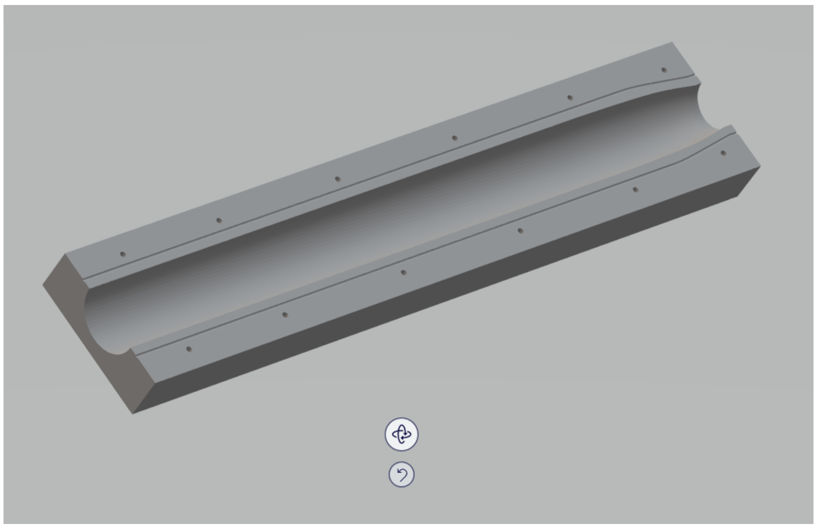


Figure 3.70: Missile Frame Mould





3.3.2 Ground Launcher

The Design of ground launcher is as follows,

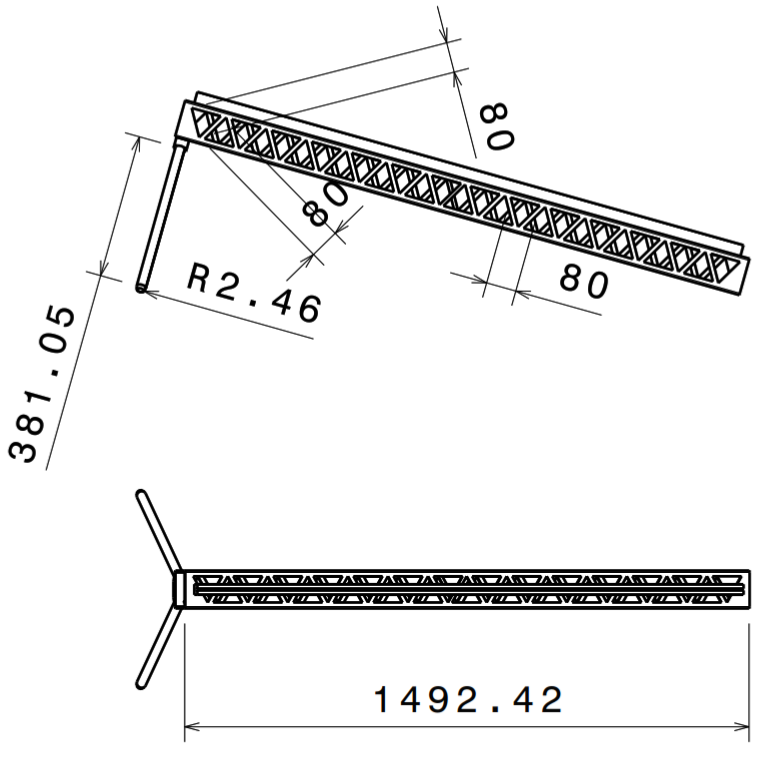
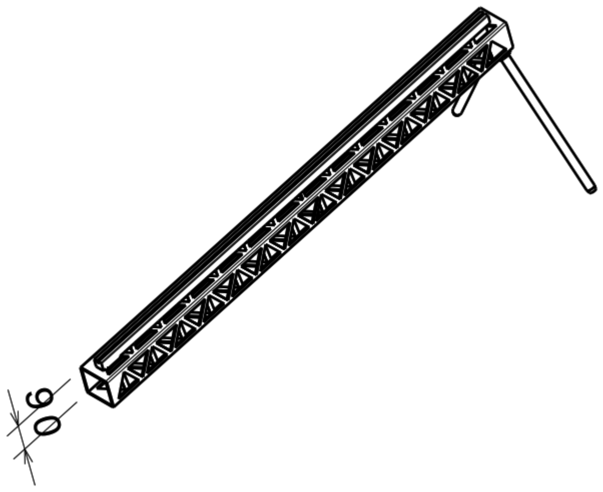
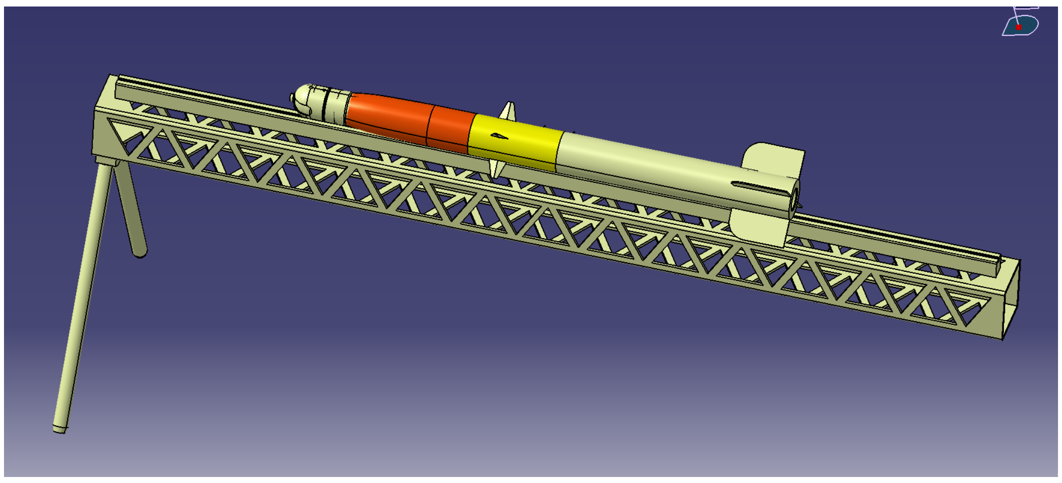
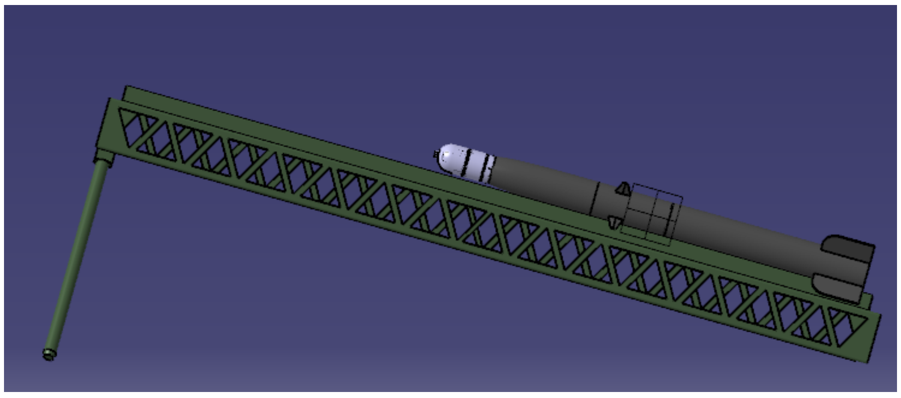
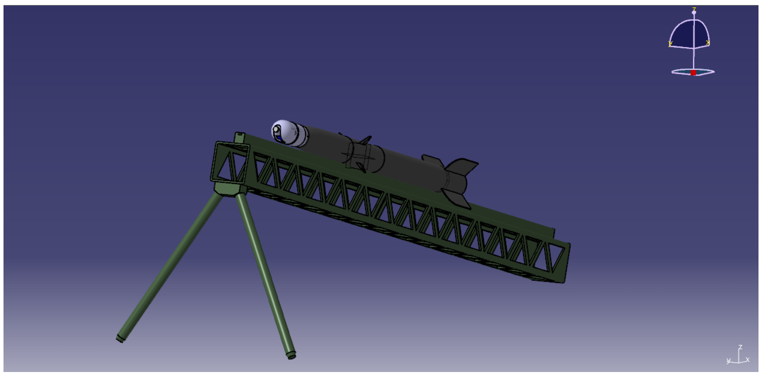


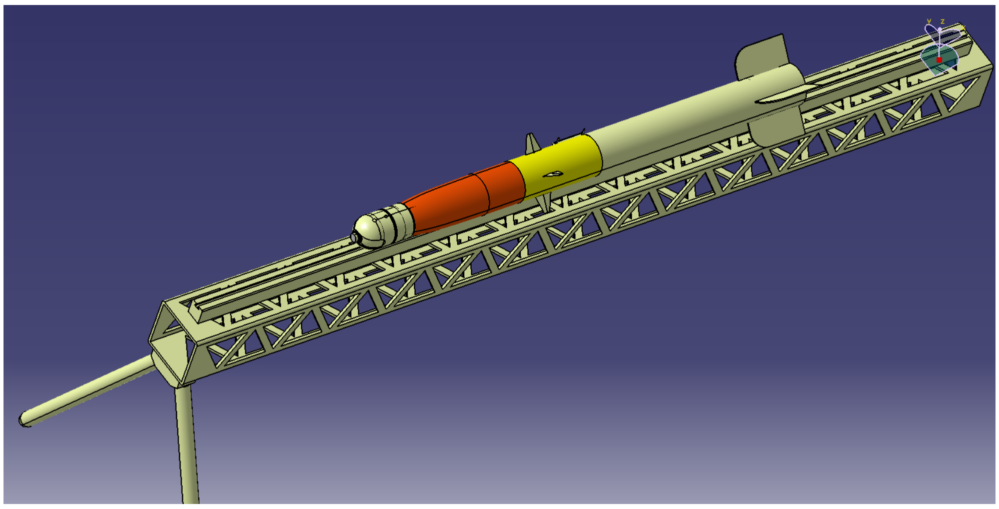
Figure 3.71: Design : Ground Launcher


600









3.3.3 Hand Launcher

The same missile designed can also be hand launched, and thus a hand launcher has been designed.

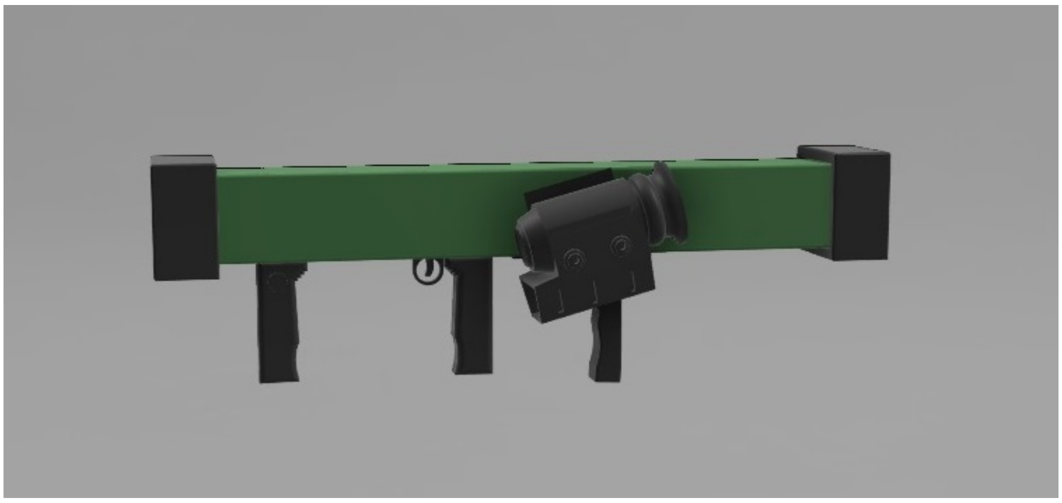
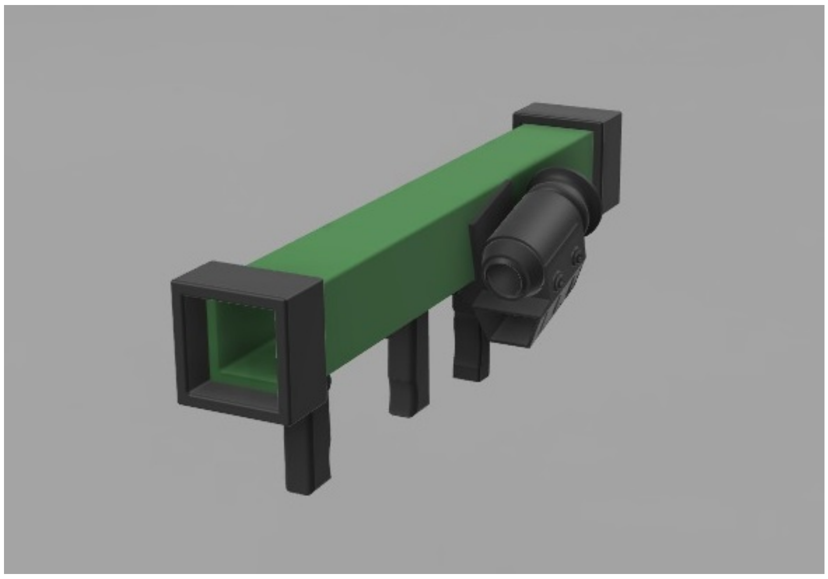
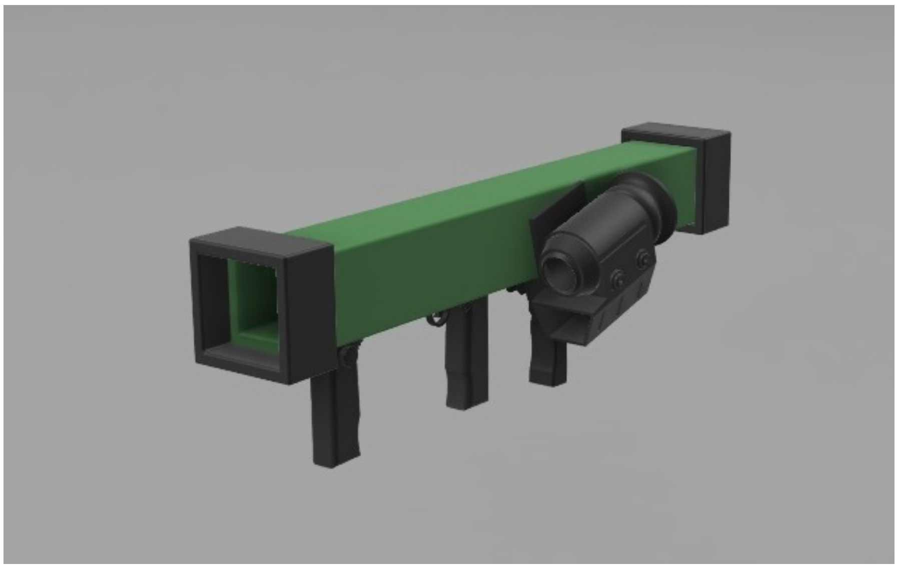


Figure 3.72: Design of Hand Launcher





3.3.4 Design of Weapon Release System A

Weapon Release system A is servo control, has a camera attached to the release system for target lock. This system can be used for Missile Version 1 , 2 , 3. This version launches the missile with the known, calculated guide angle.

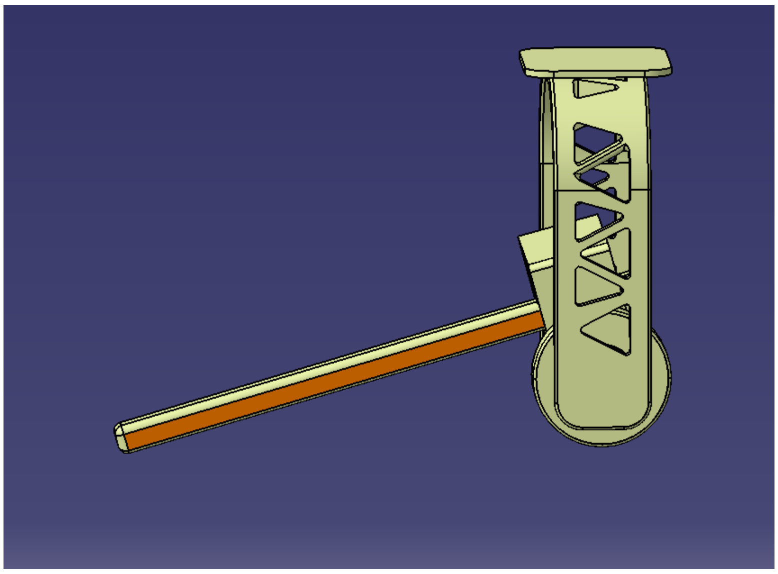
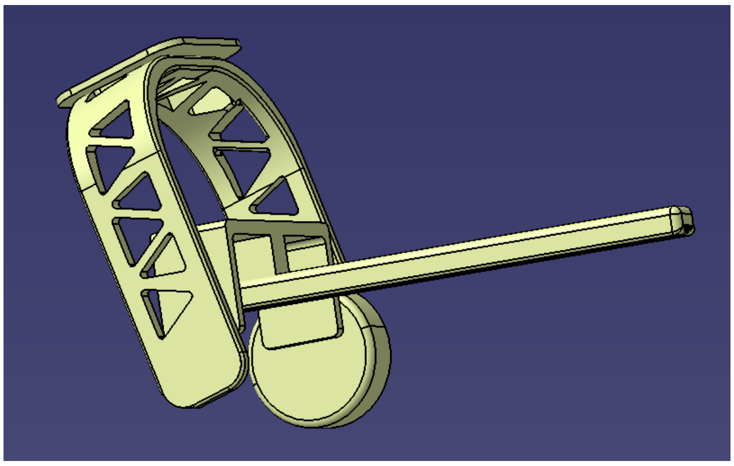
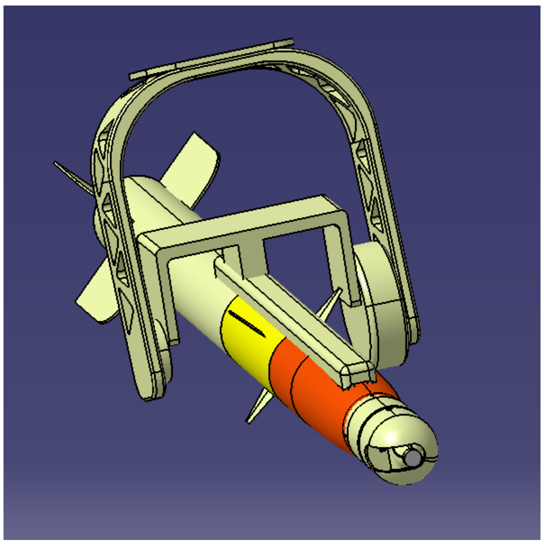
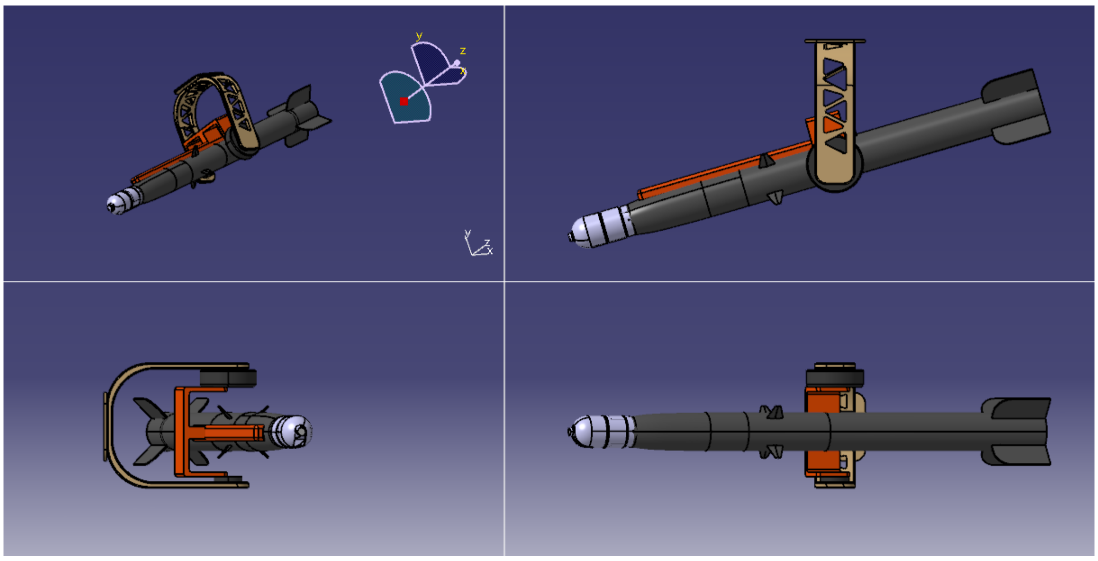
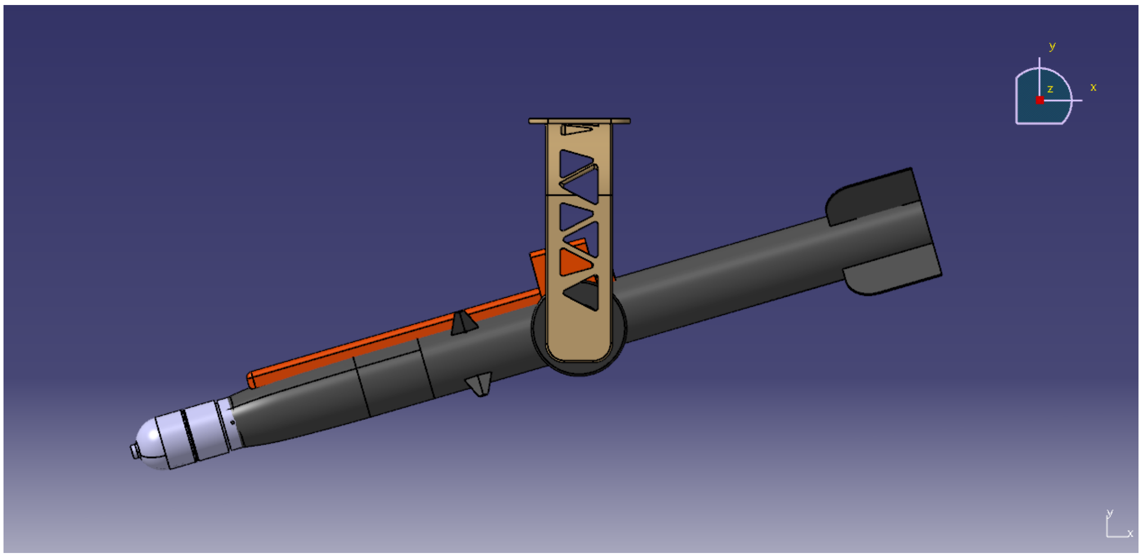


Figure 3.73: Weapon Release System A









3.3.5 Design of Weapon Release System B

Weapon Release system B drops the missile. This system can be used for Missile Version 4 where launch with initial guide angle is not required.

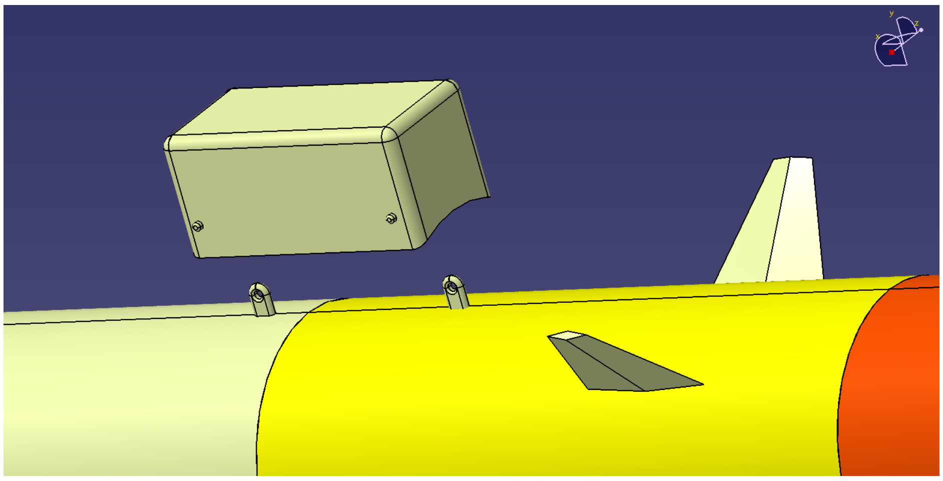
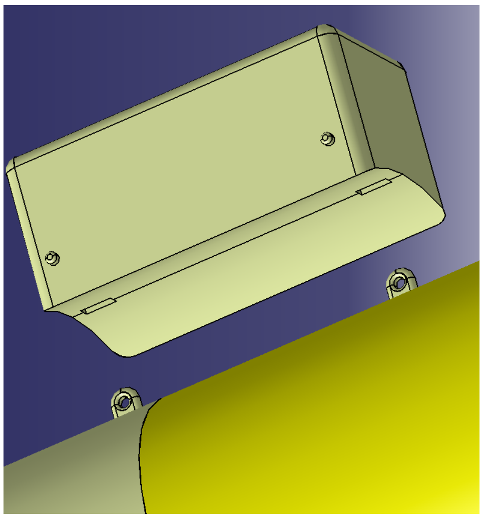
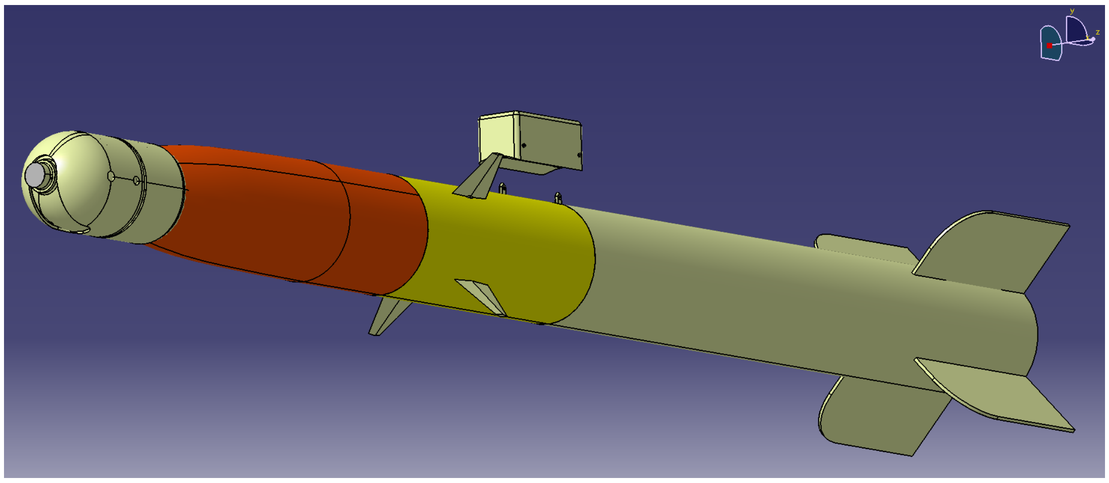
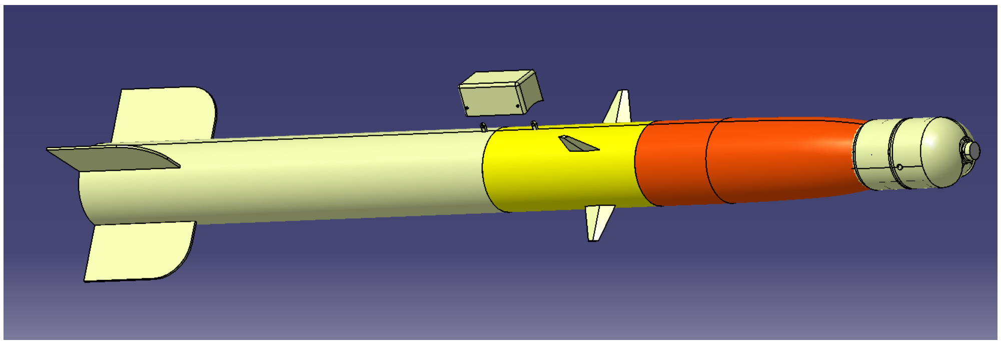


Figure 3.74: Weapon Release System B







CHAPTER 4

TRAJECTORY SIMULATION & PREDICTION METHODOLOGY

4.1 STABILITY ANALYSIS

The stability is the tendency of the rocket to return to its equilibrium position after it has been disturbed. The disturbance may be generated by the thrust action or atmospheric phenomena. The atmospheric disturbance can be wind gusts, wind gradients, or turbulent air.

The stability analysis begins with the defining the equilibrium. If a rocket is to remain in steady uniform flight, the resultant force as well as the resultant moment about the centre of gravity (CG) must be equal to zero. A rocket satisfying this requirement is said to be in equilibrium. On the other hand, if the forces and moment do not sum to zero, the rocket will be subjected to transnational and rotational acceleration.

The subject of stability is divided into static and dynamic stability. Static stability is the initial tendency of the vehicle to return to its equilibrium state after a disturbance. In dynamic stability we are concerned with the time history of the motion of the vehicle after it is disturbed from its equilibrium point.

4.2 STATIC STABILITY

Static stability is the initial tendency of the vehicle to return to its equilibrium state after a disturbance. The pitching, yawing and rolling moment coefficients are denoted as C_m, C_n, C_l. For this to happen, three conditions have to be satisfied :

1. Longitudinal static stability condition $C_m\alpha < 0$, this condition has to be satisfied by the horizontal tail.

2. Directional static Stability Condition $C_n\beta > 0$, this condition has to be satisfied by the vertical tail

3. Lateral static stability condition $C_l\beta < 0$, this condition has to be satisfied by vertical tail.

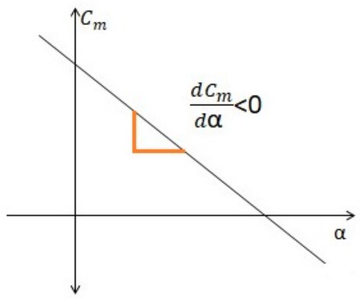


Figure 4.1: Longitudinal static Stability Condition

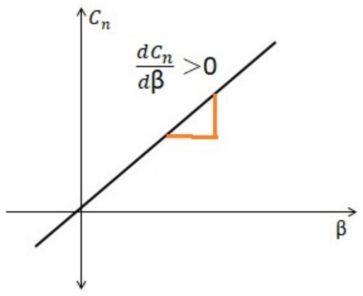


Figure 4.2: Directional static Stability Condition

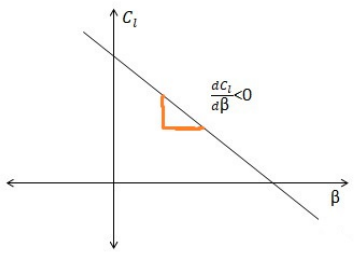


Figure 4.3: Lateral static stability condition

4.3 DYNAMIC STABILITY

If the rocket were disturbed from its equilibrium conditions. Note that the vehicle can be statically stable but dynamically unstable. Static stability, therefore does not guarantee dynamic stability. However, for the vehicle to be dynamically stable it must be statically stable.

- **Conservation of Linear Momentum Equations**

$$F_x - mg\sin\theta = m\left[\dot{u} + qw - rv\right] \tag{4.1}$$

$$F_y + mg\cos\theta\sin\phi = m\left[\dot{v} + ru - pw\right] \tag{4.2}$$

$$F_z + mg\cos\theta\cos\phi = m\left[\dot{w} + pv - qu\right] \tag{4.3}$$

  Where F_x, F_y, F_z and u,v,w are the components of the force and velocity along the x,y and z axes respectively. The force components are composed of contributions due to the aerodynamic, propulsive and gravitational forces acting on the rocket. L,M,N are the components of the moment along the x,y and z axis respectively. m is mass of the rocket and g is the gravitational force. I_{xx}, I_{yy} and I_{zz} are the mass moment of inertia of the rocket about the x,y,z axis respectively. The term with the mixed indexes are called the product of inertial.

- **Conservation of Angular Momentum equations**

$$L = I_{xx}\dot{p} - I_{xz}\dot{r} + qr\left[I_{zz} - I_{yy}\right] - I_{xz}pq \tag{4.4}$$

$$M = I_{yy}\dot{q} + rq\left[I_{xx} - I_{zz}\right] + I_{xz}\left[p^2 - r^2\right] \tag{4.5}$$

$$N = -I_{xz}\dot{p} + I_{zz}\dot{r} + pq\left[I_{yy} - I_{xx}\right] + I_{xz}qr \tag{4.6}$$

- **Body Angular Velocities in terms of Euler Angles and Euler Rates (Navigation Equation** The relationship between the angular velocities in the body frame p,q and r are roll, pitch and yaw rates. The Euler rates ($\dot{\phi}$, $\dot{\psi}$, $\dot{\theta}$) are

$$p = \dot{\phi} - \dot{\psi}\sin\theta \tag{4.7}$$

$$q = \dot{\theta}\cos\phi + \dot{\psi}\cos\theta\sin\phi \tag{4.8}$$

$$r = \dot{\psi}\cos\theta\cos\phi - \dot{\theta}\sin\phi \tag{4.9}$$

Small disturbance theory is used in linearising the above equations. In applying the small disturbance theory we assume that the motion of rocket consists of small deviation about a steady flight condition. The term with the indexes '0' represents the reference value i,e steady state values. The linearised small-disturbance longitudinal and lateral rigid body equation of motions are as follows [66]

- **Longitudinal Equations**

$$F_x - mg\sin\theta = m\left[\dot{u} + qw - rv\right] \tag{4.10}$$

$$F_z + mg\cos\theta\cos\phi = m\left[\dot{w} + pv - qu\right] \tag{4.11}$$

$$M = I_{yy}\dot{q} + rq\left[I_{xx} - I_{zz}\right] + I_{xz}\left[p^2 - r^2\right] \tag{4.12}$$

- **Lateral Equations**

$$F_y + mg\cos\theta\sin\phi = m\left[\dot{v} + ru - pw\right] \tag{4.13}$$

$$L = I_{xx}\dot{p} - I_{xz}\dot{r} + qr\left[I_{zz} - I_{yy}\right] - I_{xz}pq \tag{4.14}$$

$$N = -I_{xz}\dot{p} + I_{zz}\dot{r} + pq\left[I_{yy} - I_{xx}\right] + I_{xz}qr \tag{4.15}$$

4.3.1 Stability Derivatives

Stability Derivatives	
Lateral Directional	Longitudional Stability
$Y_p = \frac{C_{y\beta t}\bar{q}Sr_d}{mu_0}$	$X_u - \frac{\left[C_{Du} + 2C_{Do}\right]\bar{Q}_0S}{mu_0}$
$Y_r = \frac{\bar{q}SC_{y\beta}l_v}{mu_0}$	$X_\alpha = \frac{\bar{Q}_0S\left[-C_{D\alpha} + C_{Lss}\right]}{m}$
$Y_\beta = \frac{\bar{Q}_0SC_{y\beta}}{m}$	$X_q = 0$
$L_p = -\frac{C_{L\alpha}\bar{Q}_0Cb^2}{12U_0I_{xx}}$	$Z_u = -\frac{\bar{Q}_0S\left[C_{Lu} + 2C_{Lss}\right]}{mu_0}$
$L_r = -\frac{\bar{Q}_0Sr_dC_{y\beta}l_v}{U_0I_{xx}}$	$Z_q = -\frac{\bar{Q}_0Sl_tC_{L\alpha}}{mu_0}$
$L_\beta = \frac{\bar{Q}_0Sr_dC_{y\beta}}{I_{xx}}$	$Z_\alpha = -\frac{\bar{Q}_0S\left[C_{L\alpha} + C_{Dss}\right]}{m}$
$N_p = \frac{\bar{Q}_0SC_{y\beta}r_dl_v}{U_0Izz}$	$Z_q = -\frac{\bar{Q}_0S\bar{c}C_{mu}}{I_{yy}u_0}$
$N_r = \frac{\bar{Q}_0Sl_v^2C_{y\beta}}{U_0I_{zz}}$	$M_\alpha = \frac{\bar{Q}_0S\bar{c}C_{m\alpha}}{I_{yy}}$
$N_\beta = -\frac{\bar{Q}_0Sl_vC_{y\beta}}{Izz}$	$M_q = -\frac{C_{L\alpha t}l_t^2\bar{Q}_0S}{u_0I_{yy}}$

Table 4.1: Stability Derivatives

4.3.2 Aerodynamic Characteristics of Missile

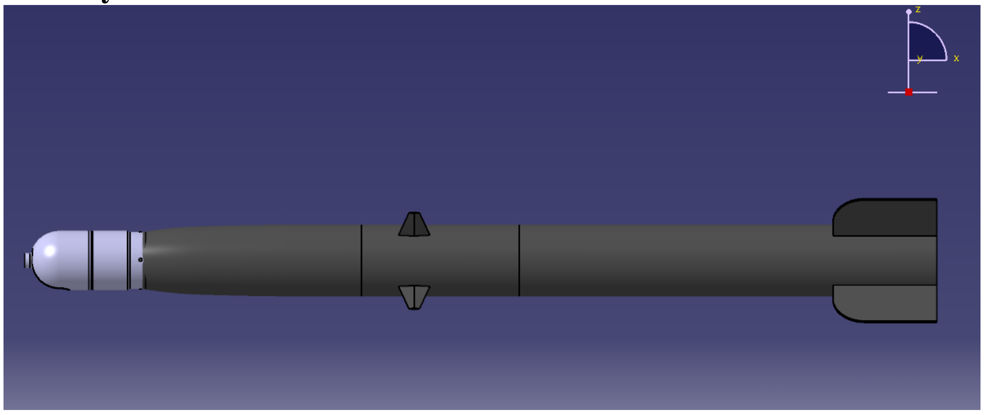
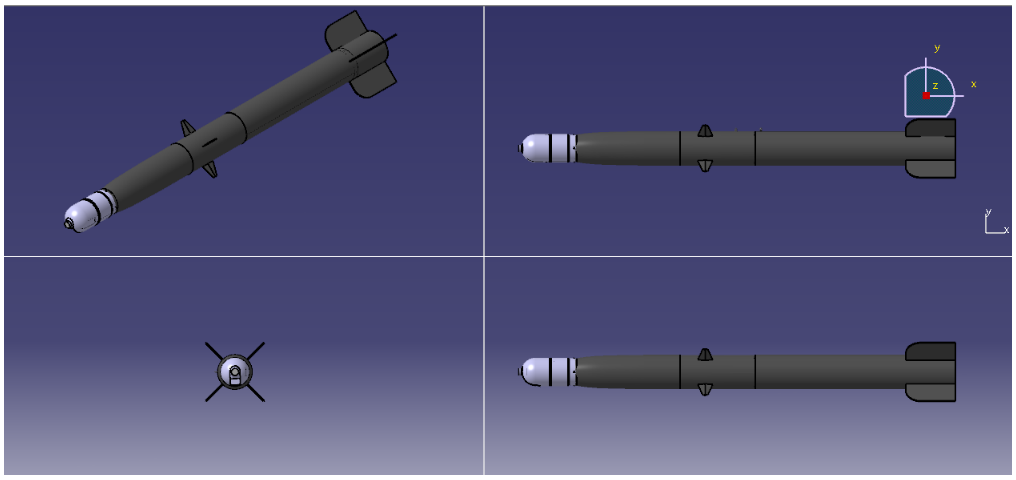


Figure 4.4: Missile Design



Computational Simulation

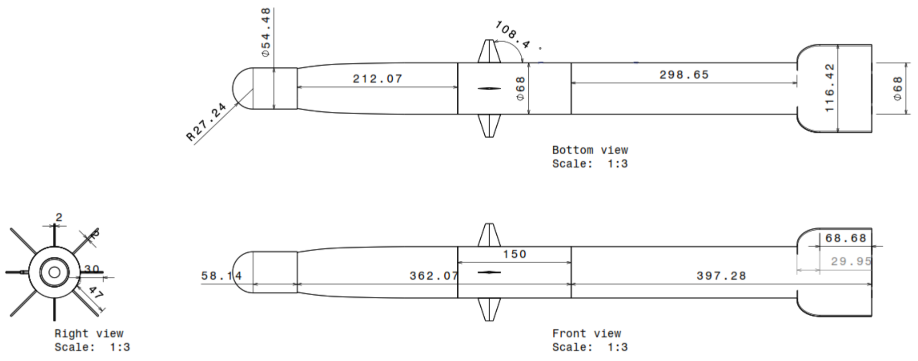
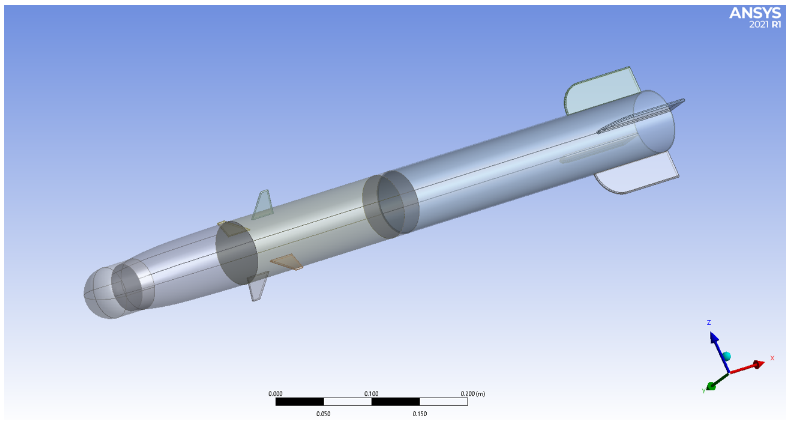
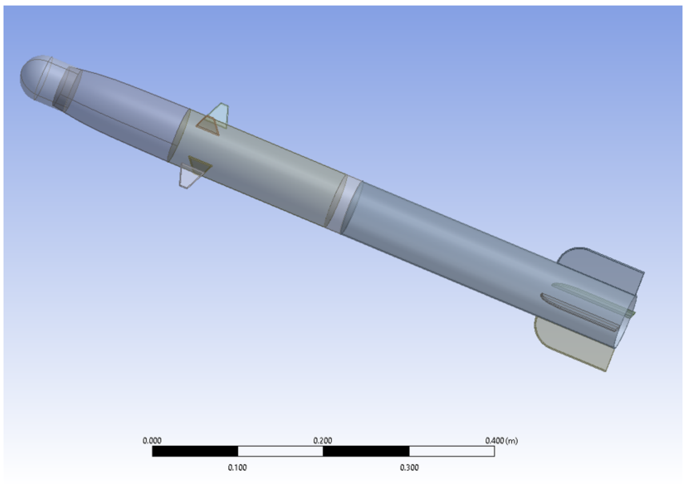
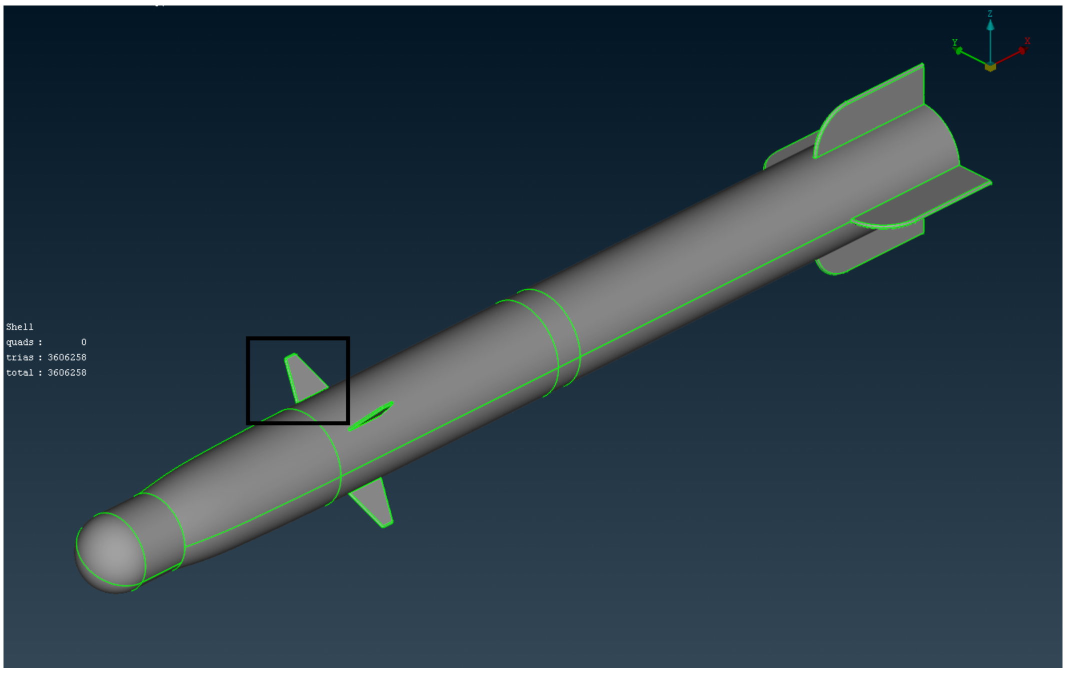


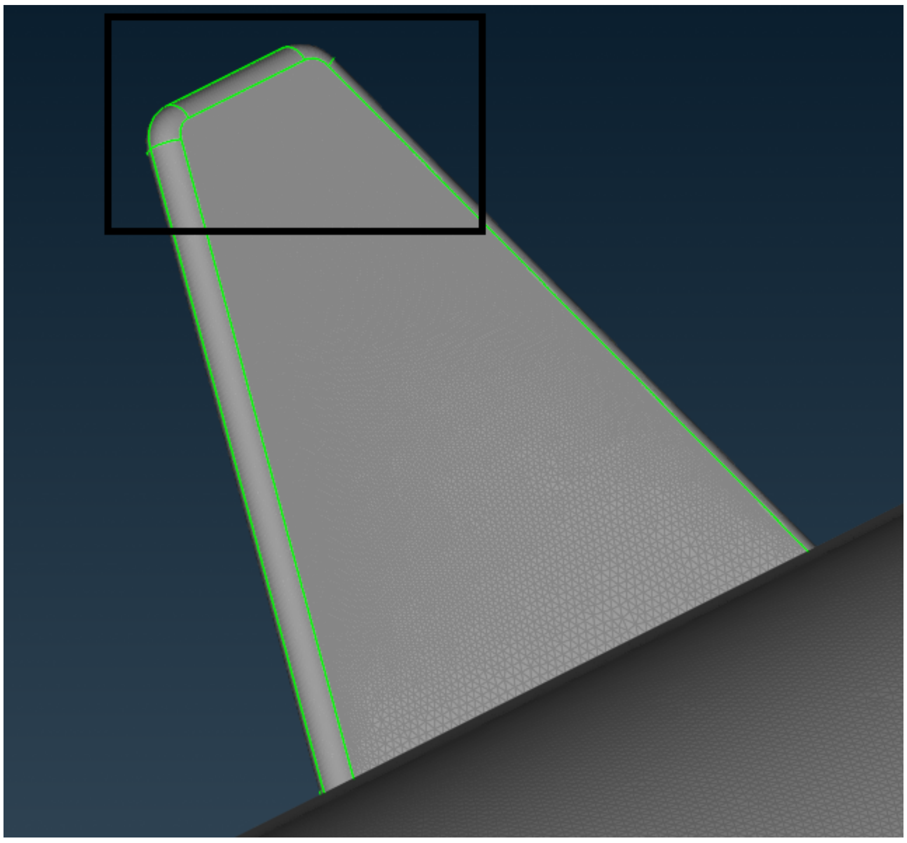
Figure 4.5: Missile CAD

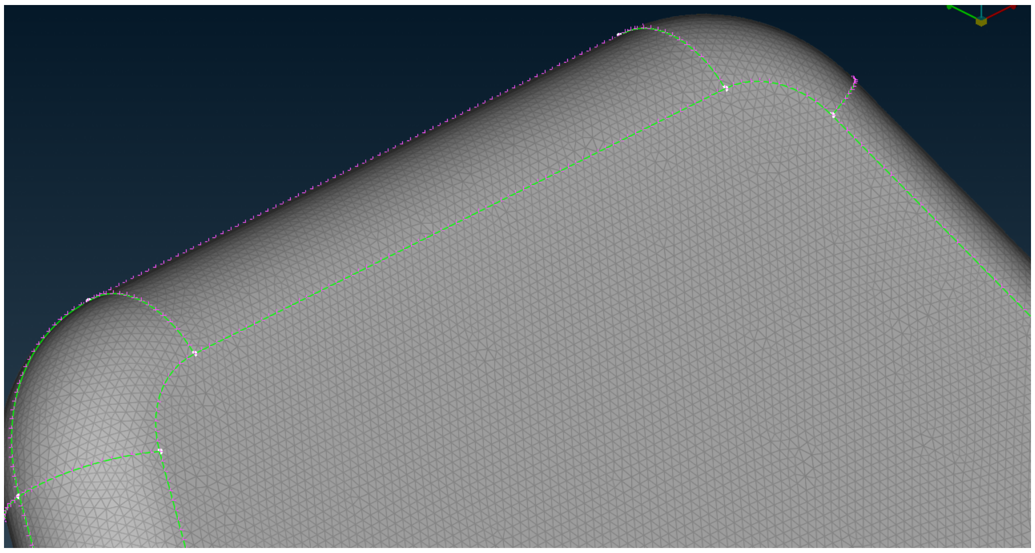


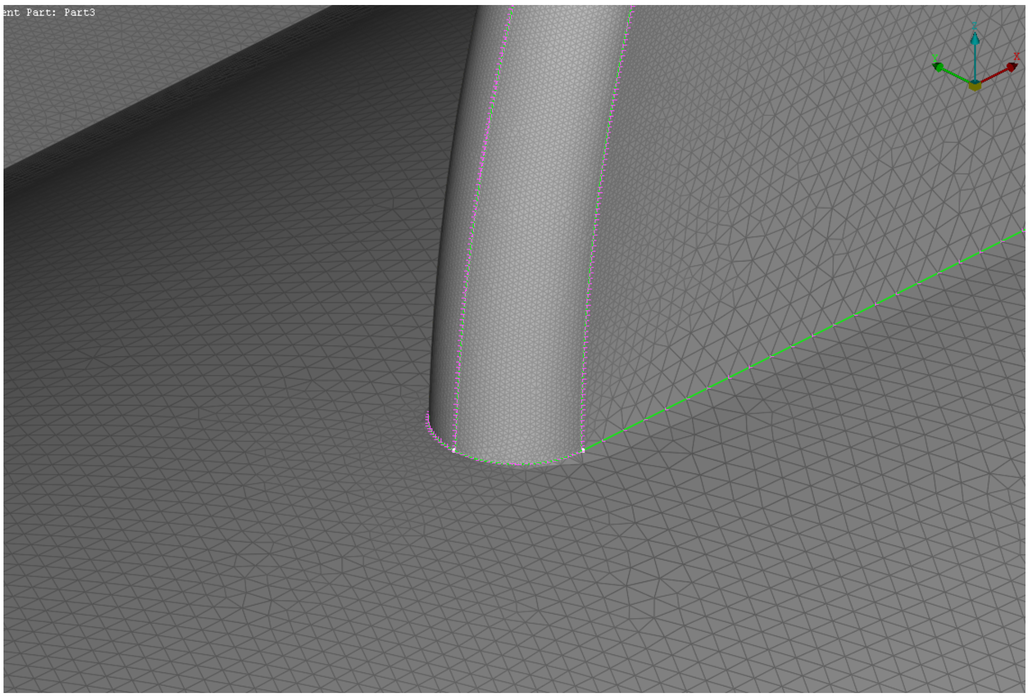




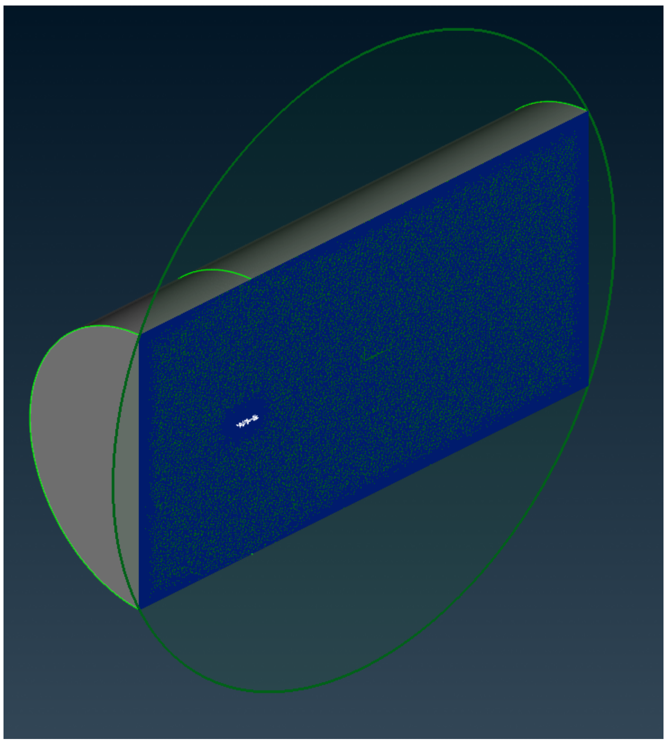
```
Shell
quads :        0
trias : 3606258
total : 3606258
```

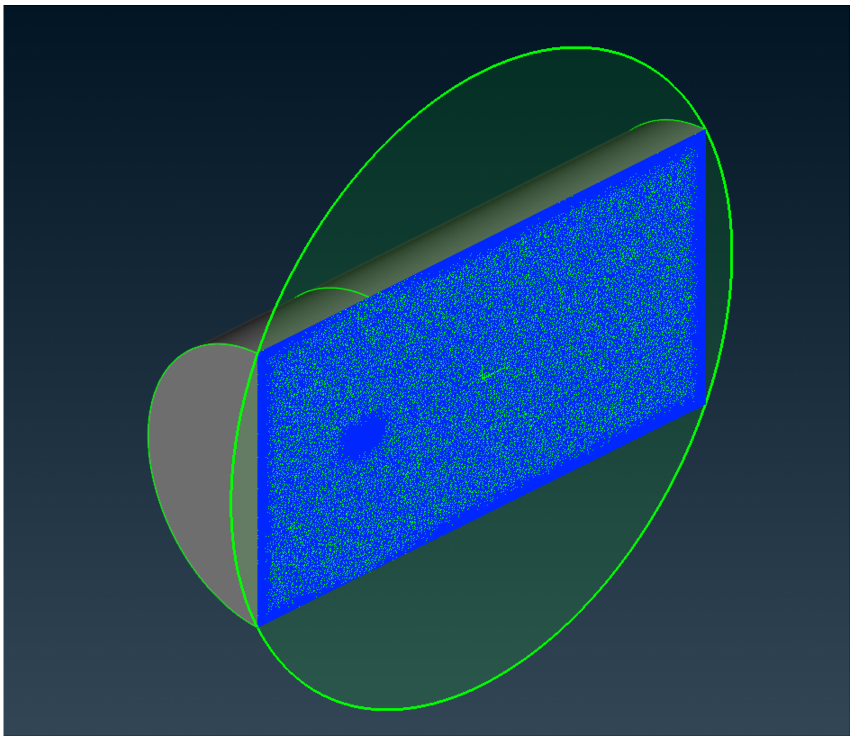


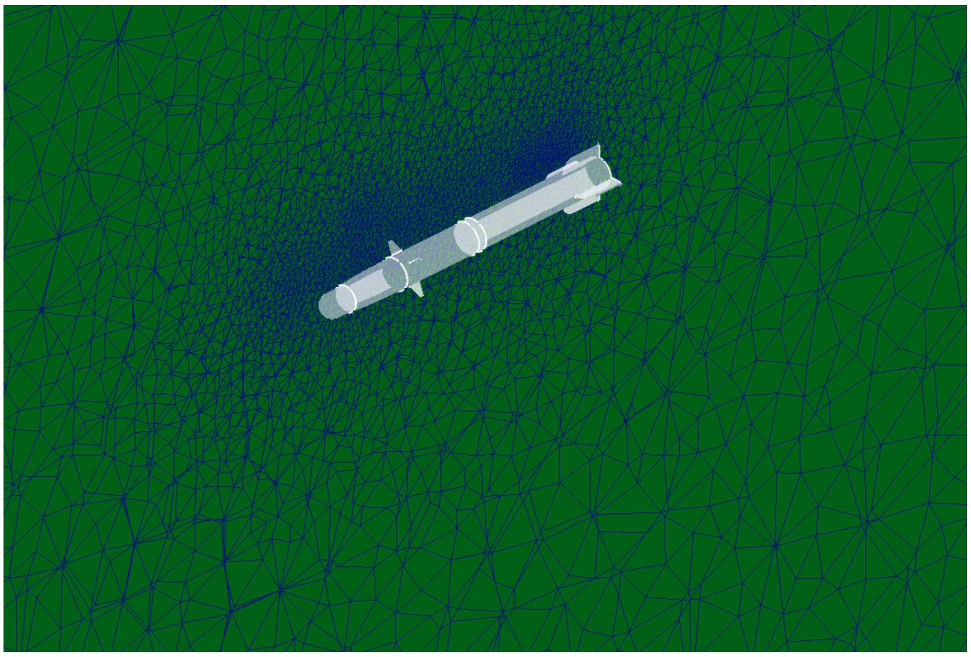


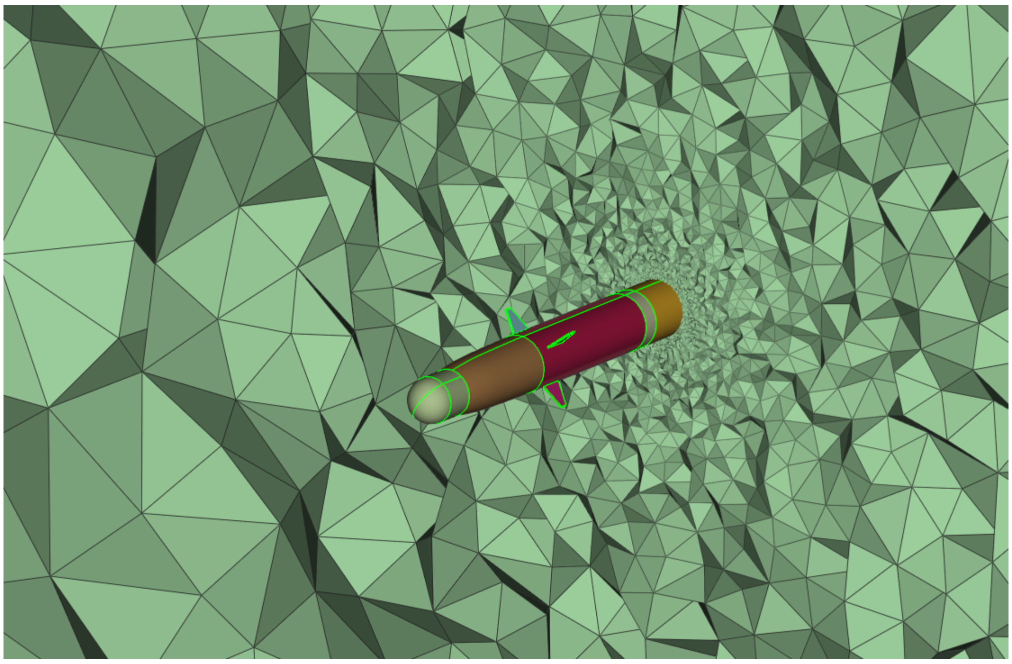


Surface mesh on missile body with 3606258 trias elements, 1034611 nodes.









Volume mesh on missile body with 65862454 tetras elements, 19079505 nodes.

The commercial CFD software Fluent 2021R1 by ANSYS Inc. was used for the simulations. in order to obtain the flow solution, Fluent is used to numerically solve the 3-D, viscous compressible, steady-state RANS equations. And the two equations SST $K - \omega$ turbulance model is chosen to closure the RANS equations. For Solving these equations, pressure-based solver provided by FLUENT is selected. The coupled pressure-velocity coupling approach is employed to solve for all variables, and as a result it has higher accuracy. Simulation has been run for Mach number ranging from 0.3 to 1.6, for angle of attacks 0 to 20 degrees.

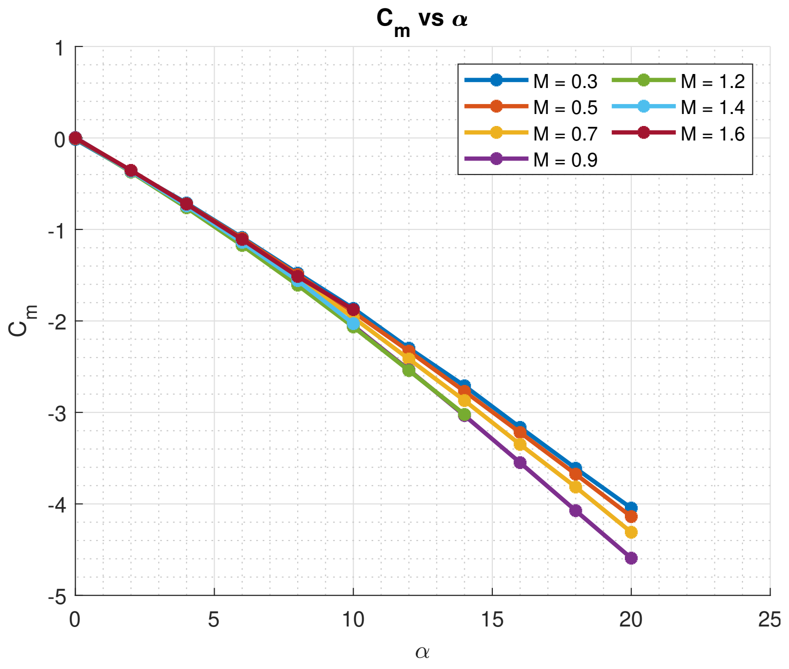


Figure 4.6: C_m vs α

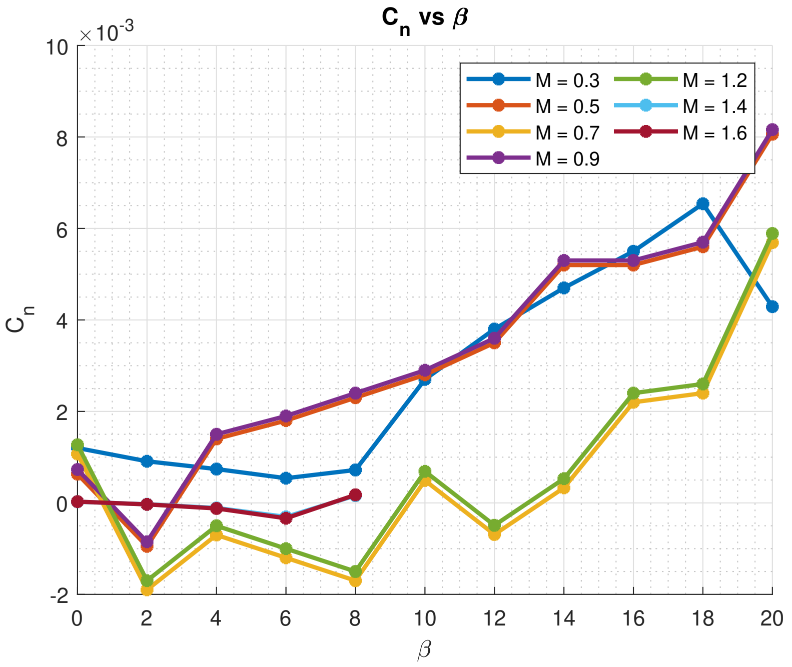


Figure 4.7: C_n vs β

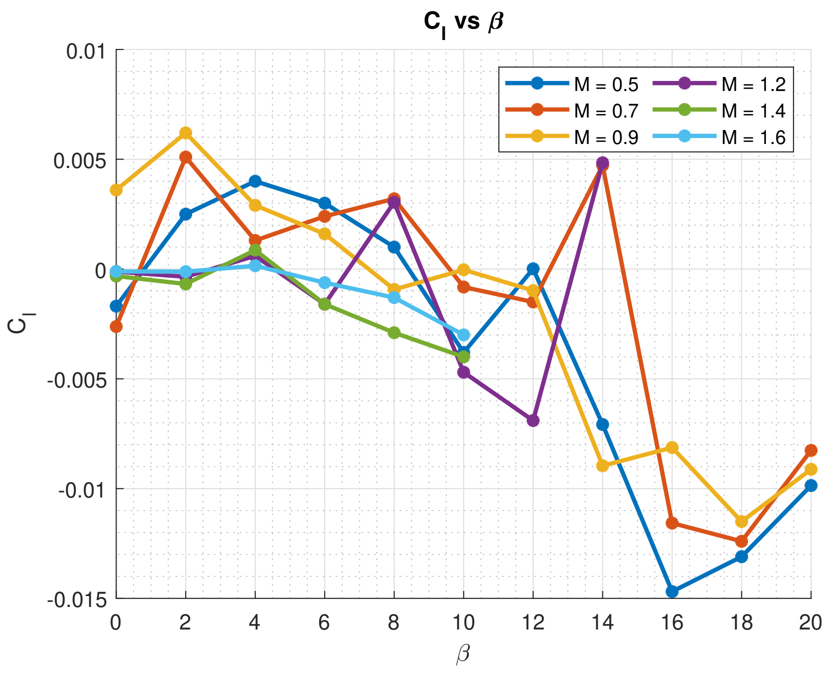


Figure 4.8: C_l vs β

* Longitudinal static stability condition $C_m\alpha < 0$ is satisfied in figure 4.6

* Directional static Stability Condition $C_n\beta > 0$, this condition is satisfied by the vertical tail, visually from figure 4.7

* Lateral static stability condition $C_l\beta < 0$, this condition is satisfied by vertical tail, from figure 4.8

4.4 TRAJECTORY SIMULATION IN MATLAB

The equations of motions that describe the 6-DOF model is mentioned below which is used to do derive the trajectory 2 D simulation for Version 1, Version 2.

$$\begin{aligned}
F_x - mg\sin\theta &= m[\dot{u} + qw - rv] \\
F_y + mg\cos\theta\sin\phi &= m[\dot{v} + ru - pw] \\
F_z + mg\cos\theta\cos\phi &= m[\dot{w} + pv - qu] \\
L &= I_{xx}\dot{p} - I_{xz}\dot{r} + qr\left[I_{zz} - I_{yy}\right] - I_{xz}pq \\
M &= I_{yy}\dot{q} + rq\left[I_{xx} - I_{zz}\right] + I_{xz}\left[p^2 - r^2\right] \\
N &= -I_{xz}\dot{p} + I_{zz}\dot{r} + pq\left[I_{yy} - I_{xx}\right] + I_{xz}qr
\end{aligned} \tag{4.16}$$

$$m\frac{dV}{dt} = T - D - mg\sin\gamma \tag{4.17}$$

$$\frac{dV}{dt} = \frac{(T - D - mg\sin\gamma)}{m} \tag{4.18}$$

$$m\frac{dV_t}{dt} = L - mg\cos\gamma \tag{4.19}$$

$$m\frac{d}{dt}(\gamma\omega) = L - mg\cos\gamma \tag{4.20}$$

$$\frac{d(\gamma\omega)}{dt} = \frac{d\gamma}{dt}\omega + \frac{d\omega}{dt}\gamma \tag{4.21}$$

$\frac{d\omega}{dt}$ tends to zero

$$mV\frac{d\gamma}{dt} = L - mg\cos\gamma \tag{4.22}$$

$$\frac{d\gamma}{dt} = \frac{L - mg\cos\gamma}{mv} \tag{4.23}$$

$$\frac{dx}{dt} = V \cos \gamma \tag{4.24}$$

$$\frac{dy}{dt} = V \sin \gamma \tag{4.25}$$

ODE 45

MATLAB's standard solver for ordinary differential equations (ODEs) is the function ode45. This function implements a Runge-Kutta method with a variable time step for efficient computation. ode45 is designed to handle the following general problem

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(t, \mathbf{x}), \quad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.26}$$

where t is the independent variable, $\mathbf{x}$ is a vector of dependent variables to be found and $\mathbf{f}(t, \mathbf{x})$ is a function of t and $\mathbf{x}$. The mathematical problem is specified when the vector of functions on the right-hand side of Eq. (4.26), $\mathbf{f}(t, \mathbf{x})$, is set and the initial conditions, $\mathbf{x} = \mathbf{x}_0$ at time t_0 are given.

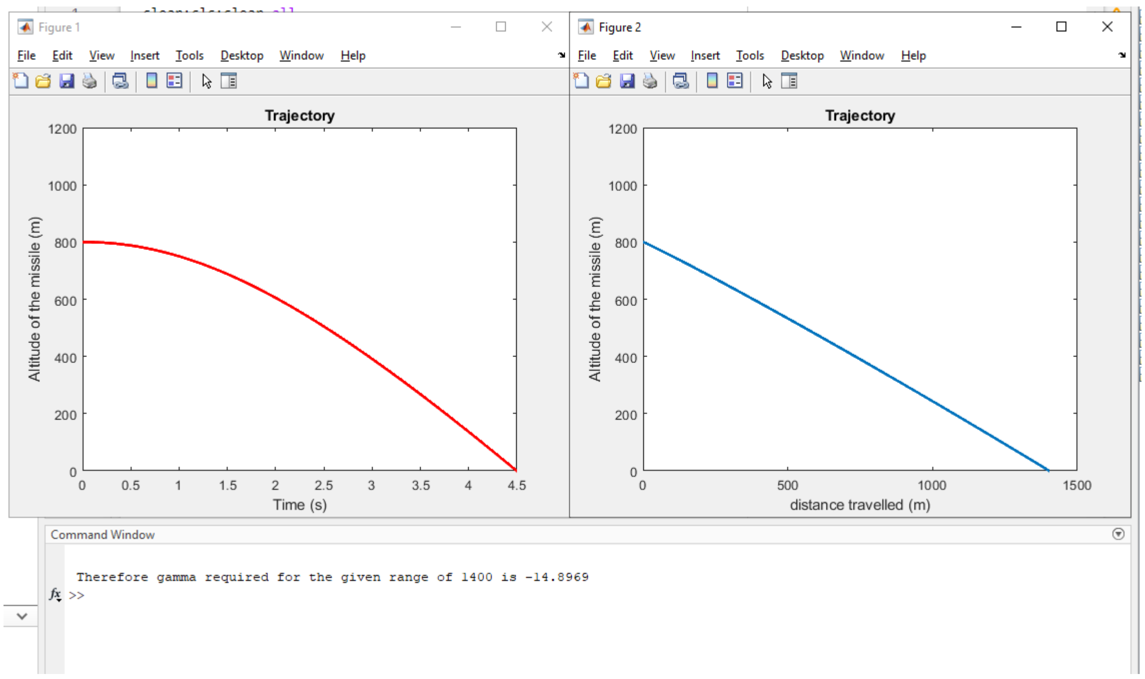
4.4.1 Version 1

Using, ODE45 for equations (4.18), (4.23), (4.24), (4.25), given the initial conditions, It is possible to find all the possible x, y, u, v, γ datasets in a matrix., Now for a possible range of γ from $-\pi/2$ to $+\pi/2$, we can find optimum value of $\gamma_{required}$ inserting the required final position (x_f, y_f) in ODE45 run. This $\gamma_{required}$ is the value that is needed depression angle, to launch missile to reach the required range, and to hit the target.

The code is given in Appendix B.3

To the given Matlab code, when given range value, it returns the angle of depression required for the desired trajectory. Also returns the trajectory graphs.

The figure 4.9 is a trajectory obtained when the input given to matlab code is,, that we launch from 800 meter altitude, for the target range of 1400 meter. Then, Matlab returns the output as " Therefore gamma required for the given range of 1400 is -14.8969 "



```
Therefore gamma required for the given range of 1400 is -14.8969
>>
```

Figure 4.9: Version 1.0 : Trajectory

4.4.2 Version 2

Expected Trajectory is shown in figure 4.10

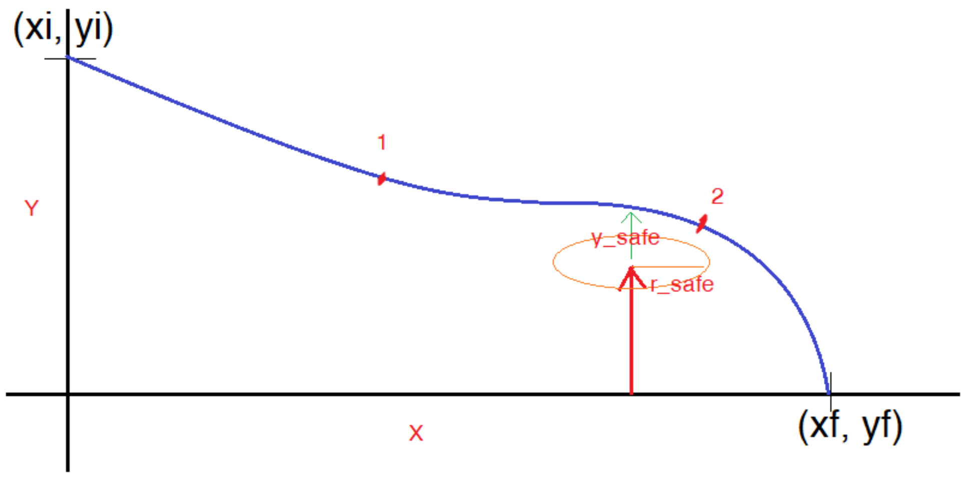


Figure 4.10: Caption

1. Find the optimal gamma to reach the safe point 1
2. Find the C_l of canards required to apply for 0.5 seconds to turn the missile, such that the gamma at end of turn equals zero is found, and applied
3. Find the C_l of canards required to apply for 0.5 seconds to change the gamma to a value, at the end of 0.5 seconds, which will lead the rest un-powered projectile path to intersect at target location

The Matlab code for the respective logic is attached in appendix B.4.

Checking matlab code

Input

1. Obstacle (650,350)
2. Radius of Obstacle = 100 meter
3. X0, Y0 = (950,0)
4. Target (X,Y) = (830,0)

Output from code

1. Initial gamma = -16 degree

2. Intermediate gamma = 0 degree

3. Final gamma = -32 degree

4. 0.5 seconds, we had Cl of 0.025 for 1st Turn. For First turn 1,3 canard deflection, = 14 degree

5. 0.5 seconds, we had Cl of 0.035 for 2nd turn. For Second turn 1,3 canard defection, = -23 degree

6. The figure 4.11 is a trajectory obtained

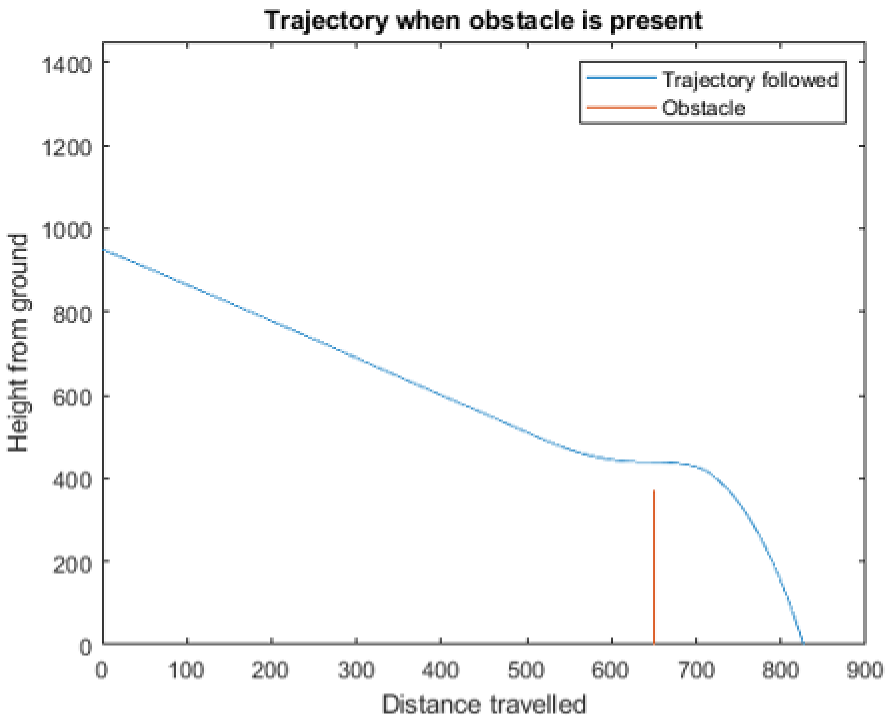


Figure 4.11: Version 2.0 : Trajectory

4.5 TRAJECTORY PREDICTION USING ARTIFICIAL INTELLIGENCE

The traditional method used for trajectory predictions use models and numerical integration, which requires a lot of computing resources. Even though missile guidance [68]is a well-studied focus, conventional algorithms[69] underperform when the target is rapidly changing direction simultaneously exerting tremendous acceleration. [67]

4.5.1 Artificial Intelligence

Artificial Intelligence (AI) is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence. It combines computer science and robust datasets, to enable problem-solving.[70]

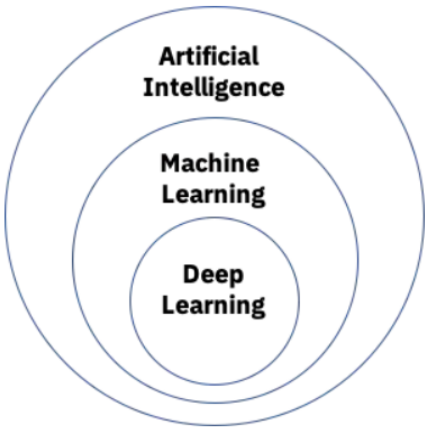


Figure 4.12: Design of Conventional Deep Neural Networks

It also encompasses sub-fields of machine learning and deep learning, which are frequently mentioned in conjunction with artificial intelligence. These disciplines are comprised of AI algorithms which seek to create expert systems which make predictions or classifications based on input data.

4.5.2 Machine Learning

Machine Learning (ML) is the field of AI that gives computers the capability to learn from past experiences without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the [71]computer that makes it more similar to humans. Various ML algorithms are utilized in this work to compare and contrast the trajectory predictions with DL algorithms.[72]

4.5.3 Deep Learning

DL is a subset of AI and ML that drives many AI applications and services that improve automation, performing analytical and physical tasks without human intervention. DL distinguishes itself from classical ML by the type of data that it works with and the methods in which it learns.

DL is an important element of data science, which includes statistics and predictive modeling. It is extremely beneficial for tasks that includes collecting, analyzing and interpreting large amounts of data as DL makes this process faster [73]and easier. At its simplest, DL can be thought of as a way to automate predictive analytics. While traditional ML algorithms are linear, DL algorithms are stacked in a hierarchy of increasing complexity and abstraction.

It eliminates some of data pre-processing that is typically involved with ML. Then, through the processes of gradient descent and forward propogation or backpropagation, the DL algorithm adjusts [74]and fits itself for accuracy, allowing it to make predictions about a trejectory with increased precision.

4.5.4 Deep Neural Network

Deep Neural Networks (DNN) consist of multiple layers of interconnected nodes, each building upon the[75] previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called visible[77] layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.[76]

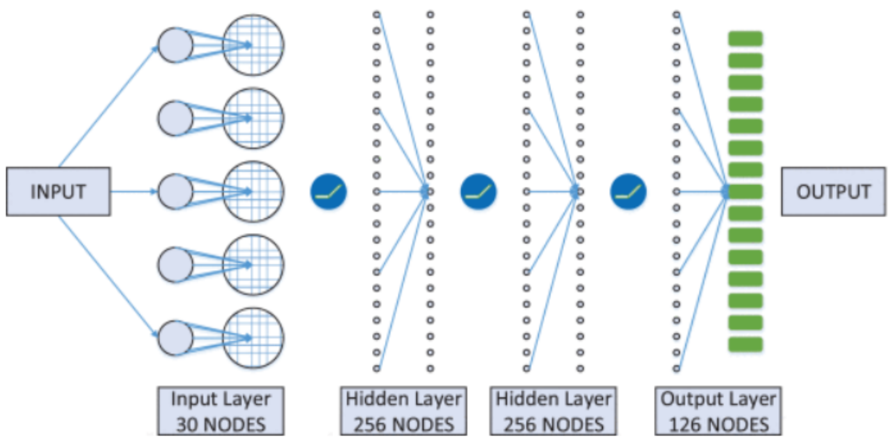


Figure 4.13: Design of Conventional Deep Neural Networks

Another process called backpropagation uses algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model. Together, forward[78] propagation and backpropagation allow a neural network to make predictions and correct for any errors accordingly. Over time, the algorithm becomes gradually more accurate.[79]

4.5.5 Model Input & Output

Since the missile's flight trajectory is continuous, the network's input should include the present[80] point as well as numerous points prior to it, allowing the DNN to learn the changing law of the flight trajectory. To make continuous fitting of the DNN easier, the output of the DNN should likewise be a collection of continuous points derived backward from the current point. The network's input is set to the position[81] and velocity of the current time point and the four points preceding it. The network's output is the position and speed of the next time point, as well as the next 25 points.

4.5.6 Dataset Description

To work with ML or DL, we need a huge amount of data, because, without the data, one cannot train AI models. Collecting and preparing the dataset is one of the most crucial parts while creating an ML/AI prediction. A dataset is a collection of data in which data is arranged in some order. A tabular dataset can be understood as a database table or matrix, where each column corresponds to a particular variable, and each row corresponds to the fields of the dataset. The most supported file type for a tabular dataset is "Comma Separated File," or CSV. The dataset comprises various fields related to missile trajectories and aerodynamics. The features considered are time t, altitude h, propulsive thrust T, time step t, flight path angle γ, time derivative T, velocity V in u, v, and w, acceleration A in x, y, and z direction. pitch, roll, yarn, roll rate, pitch rate and yarn rate respectively.

4.5.7 Other Algorithms Adopted for Comaparison

Random Forest Regressor

The random forest model[84] is a type of added substance model that makes forecasts by combining choices from many base models. Each base classifier in this case is a decision tree. Model ensembling refers to the broad process of combining many models to improve [82]predictive performance. All of the basic models in arbitrary sequence are produced independently using a different sub-test of the information. We used the sklearn package to import Rando nForest Regressor and computed the provided scores for comparison.[83]

XGB Regressor

XGB regression is an abbreviation for Extreme Gradient Boosting, also known as XGBoost. XGboost is an ensemble model approach to tackle problems with classification or regression forecasting. Ensembles are built using decision tree models. Each tress are added in the ensemble model to rectify misclassification rate produced[85] by previous models. To fit models, any arbitrarily discrete error rate and the optimization approach are utilised. XGBoost dominates structural datasets when it comes to regression and classification predictive modelling.[86]

Linear regression

The most elementary regression process is linear regression, which demands us to determine the predictive y [87]coordinate from the input x. A linear regression model assumes a straight-line between both the average value of the output and the [88]value of the input variable. Parameters and are almost always undetermined and should be deduced from data.[89]

Lasso Regression

The abbreviation Least Absolute Shrinkage and Selection Operator (LASSO) stands for Least Absolute Shrinkage and Selection Operator. Regularisation is a form of Lasso regression.[90] It is favoured over regression techniques for more accurate forecasting.[91] This model capitalises on shrinking. This type of regression is suitable for models with a significant level of multivariate regression or when you want to automate certain portions of model evaluation, such as parameter elimination.[92] Lasso Regression employs the L1 regularisation approach. It is used when there are a huge number of features since it automatically selects features.[93]

Logistic Regression

Logistic regression (LR) is a statistical technique that, like linear regression, [94]finds an expression which forecasts an occurrence for a categorical variable depending across one or even more output responses. To forecast group[95] membership, LR uses the logarithmic odds ratio instead of probabilities, and the resultant model is built using an incremental expectation - maximization approach.[96] It has always been the case that the application of the approach (and how well it works, for example, the classification error) takes precedence above statistical assumptions.[97]

4.6 PERFORMANCE METRICS

The performance was analysed on the basis of various factors such as:

Mean Absolute Error

Mean Absolute Error is the amount of error in your measurements. It is the difference between the measured value and “true” value. The Goal is to minimize the MAE.

Mean Squared Error

Mean Squared Error (MSE) measures the average of the squares of the errors that is, the average of the squared difference between the estimated values and the actual value. MSE is a risk function, pointing to the value of expectation of the squared error loss. The Goal is to minimize the MSE.

R2-Score

It is (total variance explained by model) / total variance. So if it is 100, the two variables are perfectly, i.e., with no variance at all. A low value would show a minimum correlation, meaning a regression model that is not valid, but not in all cases. Lower the R2 score the more accurate the model will be and that is not valid in all cases.

4.6.1 Result Analysis

Methods	MAE	MSE	R2 Score
Random Forest	0.64986	0.6160	0.7355
SVM	0.7648	0.6558	0.3771
XGB	0.2162	0.5148	0.1096
DNN	0.9974	0.1450	0.3912
Linear Regression	0.4952	0.3586	0.3942
Lasso Regression	0.6416	0.4458	0.2021
Logistic Regression	0.5002	0.9874	0.3771

Table 4.2: Performance analysis of intelligent models

From the above table, it is evident that DNN outperforms all the state-of-art algorithms by achieving lowest error rate. Thus the predicted trajectory will the be the best one for the missile to reach the target.

CHAPTER 5

CONCLUSION

1. Solid Rocket motor V2.0 for the range of 2000 m with $\Delta\theta < 45$ deg with thrust of 1000 N for 2.5 s burnout is designed.

2. Warhead payload profile is designed, for 2500 gram and 2100 gram, and named WP V 2.0, and WP V 3.0 respectively

3. Implemented : Optimising structural weight without compromising on performance : Carbon composite material selected for Missile frame, to provide same strength as that of aluminum alloy, with 50 percent less weight than aluminum alloy

4. Successfully passed hydrotest for the material selected to validate the maximum pressure Rocket Motor case can handle, and it is found the material withstands 50 bar

5. Nozzle throat Rocket motor igniter is selected

6. Matlab code developed for, Trust calculation using the Pressure data obtained in firing of propellant using BEM

7. 2 Canard mechanism designed and fabricated

8. Custom 2 axis gimbal for missile has been designed with lot of iterations, for the missile.

9. Stability analysis of Missile is calculated and found missile is statically stable.

10. Design of Ground Launcher, Hand Launcher, Weapon Release System A and Weapon Release System B is complete

11. FPGA Architecture achitecture is studied and found that FPGA methodology is the best for tracking development of missile version 4.0

12. Studied and finalised Sensors, Single Board Computers (SBC), Carrier circuit, and python code for csrt tracking with pan-tilt servo configuration is implemented

13. Trajectory Simulation is done using matlab for version 1, version 2

14. Trajectory Prediction using Machine Learning and Deep Learning is implemented.

5.1 FUTURE SCOPE

1. Fabrication, Assembling and Testing can be conducted for the designed models. Key variables at each stage can be validated in real field

2. Further development can be based on real field validation, The data obtained can be used for further simulations

3. Coding can be developed using FPGA architecture for Tracking

APPENDIX A

CEA ANALYSIS

A.1 SRP V 2.0 CEA ANALYSIS REPORT

```
*******************************************************************************

         NASA-GLENN CHEMICAL EQUILIBRIUM PROGRAM CEA2, MAY 21, 2004
                   BY  BONNIE MCBRIDE AND SANFORD GORDON
      REFS: NASA RP-1311, PART I, 1994 AND NASA RP-1311, PART II, 1996

*******************************************************************************



problem    o/f=2.125,
    rocket  equilibrium  frozen  nfz=1  tcest,k=3800
  p,bar=40,
  sup,ae/at=4.23,
react
  oxid=NH4CLO4(I) wt=68  t,k=300
  fuel=AL(cr) wt=18  t,k=300
  fuel=HTPB  wt=14  t,k=300
    h,kj/mol=-51.8  C 4 H 6
end

OPTIONS: TP=F  HP=F  SP=F  TV=F  UV=F  SV=F  DETN=F  SHOCK=F  REFL=F  INCD=F
RKT=T  FROZ=T  EQL=T  IONS=F  SIUNIT=T  DEBUGF=F  SHKDBG=F  DETDBG=F  TRNSPT=F

TRACE= 0.00E+00  S/R= 0.000000E+00  H/R= 0.000000E+00  U/R= 0.000000E+00

Pc,BAR =    40.000000

Pc/P =

SUBSONIC AREA RATIOS =

SUPERSONIC AREA RATIOS =     4.2300

NFZ=  1  Mdot/Ac= 0.000000E+00  Ac/At= 0.000000E+00

   REACTANT          WT.FRAC   (ENERGY/R),K   TEMP,K  DENSITY
       EXPLODED FORMULA
O: NH4CLO4(I)       1.000000  -0.355439E+05   300.00  0.0000
         N  1.00000  H  4.00000  CL 1.00000  O  4.00000
F: AL(cr)           0.562500   0.538838E+01   300.00  0.0000
         AL 1.00000
F: HTPB             0.437500  -0.623007E+04   300.00  0.0000
         C  4.00000  H  6.00000

 SPECIES BEING CONSIDERED IN THIS SYSTEM
(CONDENSED PHASE MAY HAVE NAME LISTED SEVERAL TIMES)
```

```
LAST thermo.inp UPDATE:    9/09/04

g12/97  *AL             tpis96  ALC             tpis96  ALC2
tpis96  ALCL            tpis96  ALCL2           tpis96  ALCL3
tpis96  ALH             tpis96  ALHCL           tpis96  ALHCL2
tpis96  ALH2            tpis96  ALH2CL          tpis96  ALH3
tpis96  ALN             tpis96  *ALO            tpis96  ALOCL
tpis96  ALOCL2          tpis96  ALOH            tpis96  ALOHCL
tpis96  ALOHCL2         tpis96  ALO2            tpis96  AL(OH)2
tpis96  AL(OH)2CL       tpis96  AL(OH)3         tpis96  AL2
tpis96  AL2C2           tpis96  AL2CL6          tpis96  AL2O
tpis96  AL2O2           tpis96  AL2O3           g 7/97  *C
g 8/99  CCL             g 8/99  CCL2            n12/93  CCL3
tpis91  CCL4            tpis79  *CH             g 9/99  CHCL
n12/93  CHCL2           g 7/99  CHCL3           g 4/02  CH2
g12/99  CH2CL           tpis91  CH2CL2          g 4/02  CH3
tpis91  CH3CL           g11/00  CH2OH           g 7/00  CH3O
g 8/99  CH4             g 7/00  CH3OH           srd 01  CH3OOH
g 8/99  *CN             g12/99  CNN             tpis79  *CO
tpis91  COCL            tpis91  COCL2           tpis91  COHCL
g 9/99  *CO2            tpis91  COOH            tpis91  *C2
tpis91  C2CL            g 5/02  C2CL2           tpis91  C2CL3
g 5/02  C2CL4           g 5/02  C2CL6           g 6/01  C2H
g 5/02  C2HCL           g 5/02  C2HCL3          g 1/91  C2H2,acetylene
g 5/01  C2H2,vinylidene tpis91  C2H2CL2         g 4/02  CH2CO,ketene
g 3/02  O(CH)2O         srd 01  HO(CO)2OH       g 7/01  C2H3,vinyl
g 5/02  C2H3CL          srd 01  CH2CL-COOH      g 9/00  CH3CN
g 6/96  CH3CO,acetyl    g 1/00  C2H4            g 8/88  C2H4O,ethylen-o
g 8/88  CH3CHO,ethanal  g 6/00  CH3COOH         srd 01  OHCH2COOH
g 7/00  C2H5            g 7/00  C2H6            g 8/88  CH3N2CH3
g 8/88  C2H5OH          g 7/00  CH3OCH3         srd 01  CH3O2CH3
g 7/00  CCN             tpis91  CNC             srd 01  OCCN
tpis79  C2N2            g 8/00  C2O             tpis79  *C3
n 4/98  C3H3,1-propynl  n 4/98  C3H3,2-propynl  g 2/00  C3H4,allene
g 1/00  C3H4,propyne    g 5/90  C3H4,cyclo-     g 3/01  C3H5,allyl
g 2/00  C3H6,propylene  g 1/00  C3H6,cyclo-     g 6/01  C3H6O,propylox
g 6/97  C3H6O,acetone   g 1/02  C3H6O,propanal  g 7/01  C3H7,n-propyl
g 9/85  C3H7,i-propyl   g 2/00  C3H8            g 2/00  C3H8O,1propanol
g 2/00  C3H8O,2propanol srd 01  CNCOCN          g 7/88  C3O2
g tpis  *C4             g 7/01  C4H2,butadiyne  g 8/00  C4H4,1,3-cyclo-
n10/92  C4H6,butadiene  n10/93  C4H6,1butyne    n10/93  C4H6,2butyne
g 8/00  C4H6,cyclo-     n 4/88  C4H8,1-butene   n 4/88  C4H8,cis2-buten
n 4/88  C4H8,tr2-butene n 4/88  C4H8,isobutene  g 8/00  C4H8,cyclo-
g10/00  (CH3COOH)2      n10/84  C4H9,n-butyl    n10/84  C4H9,i-butyl
g 1/93  C4H9,s-butyl    g 1/93  C4H9,t-butyl    g12/00  C4H10,n-butane
g 8/00  C4H10,isobutane g 6/01  C4N2            g 8/00  *C5
g 5/90  C5H6,1,3cyclo-  g 1/93  C5H8,cyclo-     n 4/87  C5H10,1-pentene
g 2/01  C5H10,cyclo-    n10/84  C5H11,pentyl    g 1/93  C5H11,t-pentyl
n10/85  C5H12,n-pentane n10/85  C5H12,i-pentane n10/85  CH3C(CH3)2CH3
g 2/93  C6H2            g11/00  C6H5,phenyl     g 8/00  C6H5O,phenoxy
g 8/00  C6H6            g 8/00  C6H5OH,phenol   g 1/93  C6H10,cyclo-
n 4/87  C6H12,1-hexene  g 6/90  C6H12,cyclo-    n10/83  C6H13,n-hexyl
g 6/01  C6H14,n-hexane  g 7/01  C7H7,benzyl     g 1/93  C7H8
g12/00  C7H8O,cresol-mx n 4/87  C7H14,1-heptene n10/83  C7H15,n-heptyl
n10/85  C7H16,n-heptane n10/85  C7H16,2-methylh n 4/89  C8H8,styrene
n10/86  C8H10,ethylbenz n 4/87  C8H16,1-octene  n10/83  C8H17,n-octyl
n 4/85  C8H18,n-octane  n 4/85  C8H18,isooctane n10/83  C9H19,n-nonyl
g 3/01  C10H8,naphthale n10/83  C10H21,n-decyl  g 8/00  C12H9,o-bipheny
```

```
 g 8/00  C12H10,biphenyl  g 7/97  *CL              g 6/95  CLCN
 tpis89  CLO              g 7/93  CLO2             tpis89  CL2
 tpis89  CL2O             g 6/97  *H               tpis96  HALO
 tpis96  HALO2            g 6/01  HCN              g 1/01  HCO
 tpis89  HCCN             g 6/01  HCCO             tpis89  HCL
 g 6/01  HNC              g 7/00  HNCO             g10/01  HNO
 tpis89  HNO2             g 5/99  HNO3             g 1/01  HOCL
 g 4/02  HO2              tpis78  *H2              g 5/01  HCHO,formaldehy
 g 6/01  HCOOH            g 8/89  H2O              g 6/99  H2O2
 g 6/01  (HCOOH)2         g 5/97  *N               g 6/01  NCO
 g 4/99  *NH              g 3/01  NH2              tpis89  NH3
 tpis89  NH2OH            tpis89  *NO              g 4/99  NOCL
 g 4/99  NO2              g 4/99  NO2CL            j12/64  NO3
 tpis78  *N2              g 6/01  NCN              g 5/99  N2H2
 tpis89  NH2NO2           g 4/99  N2H4             g 4/99  N2O
 g 4/99  N2O3             tpis89  N2O4             g 4/99  N2O5
 tpis89  N3               g 4/99  N3H              g 5/97  *O
 g 4/02  *OH              tpis89  *O2              g 8/01  O3
 coda89  AL(cr)           coda89  AL(L)            tpis96  ALCL3(cr)
 tpis96  ALCL3(L)         tpis96  ALH3(a)          tpis96  ALN(cr)
 tpis96  ALN(cr)          tpis96  ALN(L)           tpis96  ALN(L)
 tpis96  AL(OH)3(a)       tpis96  AL2O3(a)         tpis96  AL2O3(a)
 tpis96  AL2O3(a)         tpis96  AL2O3(L)         tpis96  AL4C3(cr)
 tpis96  AL4C3(cr)        n 4/83  C(gr)            n 4/83  C(gr)
 n 4/83  C(gr)            g11/99  H2O(cr)          g 8/01  H2O(L)
 g 8/01  H2O(L)           j 9/65  NH4CL(II)        j 9/65  NH4CL(III)
 j 9/65  NH4CL(III)

O/F =   2.125000

                       EFFECTIVE FUEL     EFFECTIVE OXIDANT        MIXTURE
ENTHALPY                  h(2)/R              h(1)/R               h0/R
(KG-MOL)(K)/KG        -0.50278394E+02     -0.30252904E+03     -0.22180883E+03

KG-FORM.WT./KG             bi(2)               bi(1)               b0i
 *N                    0.00000000E+00      0.85114308E-02      0.57877729E-02
 *H                    0.48529833E-01      0.34045723E-01      0.38680638E-01
 *CL                   0.00000000E+00      0.85114308E-02      0.57877729E-02
 *O                    0.00000000E+00      0.34045723E-01      0.23151092E-01
 *AL                   0.20847588E-01      0.00000000E+00      0.66712283E-02
 *C                    0.32353222E-01      0.00000000E+00      0.10353031E-01

POINT ITN      T             N           H            CL           O
                   AL           C
  1   22    2568.230     -13.324      -8.923      -22.064      -20.511
                -13.398      -10.874
ADD  AL2O3(L)
  1    6    3284.231     -13.785      -9.257      -20.497      -19.009
                -16.449      -12.142
Pinf/Pt = 1.731179
  2    4    3098.873     -13.947      -9.413      -20.825      -19.529
                -16.947      -12.184
Pinf/Pt = 1.733822
  2    2    3098.362     -13.947      -9.413      -20.826      -19.530
                -16.948      -12.184
  3    5    2228.111     -14.644     -10.115      -22.834      -22.915
                -21.423      -11.864
PHASE CHANGE, REPLACE  AL2O3(L)       WITH  AL2O3(a)
```

```
  3    3    2397.704     -14.774     -10.242     -22.622     -22.069
                -20.106     -12.549
ADD  AL2O3(L)
  3    3    2327.000     -14.721     -10.189     -22.704     -22.407
                -20.676     -12.272
  3    3    2327.000     -14.684     -10.152     -22.666     -22.407
                -20.676     -12.197
  3    2    2327.000     -14.683     -10.152     -22.666     -22.407
                -20.676     -12.197





              THEORETICAL ROCKET PERFORMANCE ASSUMING EQUILIBRIUM

           COMPOSITION DURING EXPANSION FROM INFINITE AREA COMBUSTOR

Pin =   580.2 PSIA
CASE =

            REACTANT                    WT FRACTION      ENERGY      TEMP
                                         (SEE NOTE)     KJ/KG-MOL      K
OXIDANT     NH4CLO4(I)                   1.0000000   -295529.716   300.000
FUEL        AL(cr)                       0.5625000        44.802   300.000
FUEL        HTPB                         0.4375000    -51800.000   300.000

O/F=    2.12500  %FUEL= 32.000000  R,EQ.RATIO= 1.921799  PHI,EQ.RATIO= 2.659239

                CHAMBER   THROAT     EXIT
Pinf/P            1.0000   1.7338   21.811
P, BAR            40.000   23.070   1.8339
T, K             3284.23  3098.36  2327.00
RHO, KG/CU M    4.0063 0 2.4678 0 2.6622-1
H, KJ/KG        -1844.23 -2376.08 -4398.41
U, KJ/KG        -2842.66 -3310.95 -5087.28
G, KJ/KG        -33642.5 -32374.7 -26928.6
S, KJ/(KG)(K)     9.6821   9.6821   9.6821

M, (1/n)          27.350   27.556   28.086
MW, MOL WT        25.268   25.365   25.685
(dLV/dLP)t      -1.01518 -1.01146 -1.00179
(dLV/dLT)p        1.2853   1.2247   0.0000
Cp, KJ/(KG)(K)    3.7669   3.4131   0.0000
GAMMAs            1.1340   1.1378   0.9982
SON VEL,M/SEC     1064.0   1031.4    829.2
MACH NUMBER        0.000    1.000    2.726

PERFORMANCE PARAMETERS

Ae/At                      1.0000   4.2300
CSTAR, M/SEC               1571.6   1571.6
CF                         0.6562   1.4381
Ivac, M/SEC                1937.8   2565.0
Isp, M/SEC                 1031.4   2260.2


MOLE FRACTIONS
```

```

 *AL               0.00016  0.00007  0.00000
 ALCL              0.00823  0.00545  0.00016
 ALCL2             0.00061  0.00039  0.00001
 ALCL3             0.00022  0.00019  0.00003
 ALH               0.00004  0.00002  0.00000
 ALHCL             0.00003  0.00001  0.00000
 ALHCL2            0.00006  0.00004  0.00000
 *ALO              0.00014  0.00006  0.00000
 ALOCL             0.00035  0.00021  0.00000
 ALOH              0.00456  0.00259  0.00005
 ALOHCL            0.00059  0.00032  0.00001
 ALOHCL2           0.00076  0.00055  0.00006
 AL(OH)2           0.00012  0.00005  0.00000
 AL(OH)2CL         0.00017  0.00010  0.00001
 AL(OH)3           0.00003  0.00002  0.00000
 AL2O              0.00009  0.00003  0.00000
 AL2O2             0.00002  0.00001  0.00000
 *CO               0.25407  0.25535  0.25781
 *CO2              0.00748  0.00722  0.00810
 *CL               0.00778  0.00646  0.00138
 CL2               0.00001  0.00001  0.00000
 *H                0.03062  0.02457  0.00470
 HALO2             0.00001  0.00000  0.00000
 HCN               0.00001  0.00001  0.00000
 HCO               0.00002  0.00001  0.00000
 HCL               0.12553  0.13172  0.14685
 *H2               0.32881  0.33699  0.35704
 H2O               0.07681  0.07243  0.06379
 NH2               0.00001  0.00000  0.00000
 NH3               0.00001  0.00001  0.00000
 *NO               0.00017  0.00009  0.00000
 *N2               0.07302  0.07334  0.07433
 *O                0.00016  0.00008  0.00000
 *OH               0.00312  0.00204  0.00016
 *O2               0.00001  0.00001  0.00000
 AL2O3(a)          0.00000  0.00000  0.03739
 AL2O3(L)          0.07613  0.07953  0.04812

  * THERMODYNAMIC PROPERTIES FITTED TO 20000.K

    PRODUCTS WHICH WERE CONSIDERED BUT WHOSE MOLE FRACTIONS
    WERE LESS THAN 5.000000E-06 FOR ALL ASSIGNED CONDITIONS

 ALC              ALC2             ALH2             ALH2CL           ALH3
 ALN              ALOCL2           ALO2             AL2              AL2C2
 AL2CL6           AL2O3            *C               CCL              CCL2
 CCL3             CCL4             *CH              CHCL             CHCL2
 CHCL3            CH2              CH2CL            CH2CL2           CH3
 CH3CL            CH2OH            CH3O             CH4              CH3OH
 CH3OOH           *CN              CNN              COCL             COCL2
 COHCL            COOH             *C2              C2CL             C2CL2
 C2CL3            C2CL4            C2CL6            C2H              C2HCL
 C2HCL3           C2H2,acetylene   C2H2,vinylidene  C2H2CL2          CH2CO,ketene
 O(CH)2O          HO(CO)2OH        C2H3,vinyl       C2H3CL           CH2CL-COOH
 CH3CN            CH3CO,acetyl     C2H4             C2H4O,ethylen-o  CH3CHO,ethanal
 CH3COOH          OHCH2COOH        C2H5             C2H6             CH3N2CH3
 C2H5OH           CH3OCH3          CH3O2CH3         CCN              CNC
 OCCN              C2N2              C2O               *C3               C3H3,1-propynl
 C3H3,2-propynl    C3H4,allene       C3H4,propyne      C3H4,cyclo-       C3H5,allyl
 C3H6,propylene    C3H6,cyclo-       C3H6O,propylox    C3H6O,acetone     C3H6O,propanal
 C3H7,n-propyl     C3H7,i-propyl     C3H8              C3H8O,1propanol   C3H8O,2propanol
 CNCOCN            C3O2              *C4               C4H2,butadiyne    C4H4,1,3-cyclo-
 C4H6,butadiene    C4H6,1butyne      C4H6,2butyne      C4H6,cyclo-       C4H8,1-butene
 C4H8,cis2-buten   C4H8,tr2-butene   C4H8,isobutene    C4H8,cyclo-       (CH3COOH)2
 C4H9,n-butyl      C4H9,i-butyl      C4H9,s-butyl      C4H9,t-butyl      C4H10,n-butane
 C4H10,isobutane   C4N2              *C5               C5H6,1,3cyclo-    C5H8,cyclo-
 C5H10,1-pentene   C5H10,cyclo-      C5H11,pentyl      C5H11,t-pentyl    C5H12,n-pentane
 C5H12,i-pentane   CH3C(CH3)2CH3     C6H2              C6H5,phenyl       C6H5O,phenoxy
 C6H6              C6H5OH,phenol     C6H10,cyclo-      C6H12,1-hexene    C6H12,cyclo-
 C6H13,n-hexyl     C6H14,n-hexane    C7H7,benzyl       C7H8              C7H8O,cresol-mx
 C7H14,1-heptene   C7H15,n-heptyl    C7H16,n-heptane   C7H16,2-methylh   C8H8,styrene
 C8H10,ethylbenz   C8H16,1-octene    C8H17,n-octyl     C8H18,n-octane    C8H18,isooctane
 C9H19,n-nonyl     C10H8,naphthale   C10H21,n-decyl    C12H9,o-bipheny   C12H10,biphenyl
 CLCN              CLO               CLO2              CL2O              HALO
 HCCN              HCCO              HNC               HNCO              HNO
 HNO2              HNO3              HOCL              HO2               HCHO,formaldehy
 HCOOH             H2O2              (HCOOH)2          *N                NCO
 *NH               NH2OH             NOCL              NO2               NO2CL
 NO3               NCN               N2H2              NH2NO2            N2H4
 N2O               N2O3              N2O4              N2O5              N3
 N3H               O3                AL(cr)            AL(L)             ALCL3(cr)
 ALCL3(L)          ALH3(a)           ALN(cr)           ALN(L)            AL(OH)3(a)
 AL4C3(cr)         C(gr)             H2O(cr)           H2O(L)            NH4CL(II)
 NH4CL(III)

 NOTE. WEIGHT FRACTION OF FUEL IN TOTAL FUELS AND OF OXIDANT IN TOTAL OXIDANTS





           THEORETICAL ROCKET PERFORMANCE ASSUMING FROZEN COMPOSITION

 Pin =   580.2 PSIA
 CASE =

             REACTANT                    WT FRACTION      ENERGY      TEMP
                                          (SEE NOTE)     KJ/KG-MOL      K
 OXIDANT     NH4CLO4(I)                   1.0000000   -295529.716    300.000
 FUEL        AL(cr)                       0.5625000        44.802    300.000
 FUEL        HTPB                         0.4375000    -51800.000    300.000

 O/F=    2.12500  %FUEL= 32.000000  R,EQ.RATIO= 1.921799  PHI,EQ.RATIO= 2.659239

                 CHAMBER   THROAT
 Pinf/P            1.0000   1.7656
 P, BAR            40.000   22.655
 T, K             3284.23  3000.20
 RHO, KG/CU M    4.0063 0 2.4839 0
 H, KJ/KG        -1844.23 -2386.97
 U, KJ/KG        -2842.66 -3299.05
 G, KJ/KG        -33642.5 -31435.2
 S, KJ/(KG)(K)     9.6821   9.6821

 M, (1/n)          27.350   27.350
```

```
MW, MOL WT         25.268   25.268
Cp, KJ/(KG)(K)     1.9182   1.9032
GAMMAs             1.1883   1.1901
SON VEL,M/SEC      1089.3   1041.9
MACH NUMBER         0.000    1.000

PERFORMANCE PARAMETERS

Ae/At                      1.00000
CSTAR, M/SEC                1545.7
CF                          0.6740
Ivac, M/SEC                 1917.3
Isp, M/SEC                  1041.9

MOLE FRACTIONS

*AL             0.00016   ALCL            0.00823   ALCL2           0.00061
ALCL3           0.00022   ALH             0.00004   ALHCL           0.00003
ALHCL2          0.00006   *ALO            0.00014   ALOCL           0.00035
ALOH            0.00456   ALOHCL          0.00059   ALOHCL2         0.00076
AL(OH)2         0.00012   AL(OH)2CL       0.00017   AL(OH)3         0.00003
AL2O            0.00009   AL2O2           0.00002   *CO             0.25407
*CO2            0.00748   *CL             0.00778   CL2             0.00001
*H              0.03062   HALO2           0.00001   HCN             0.00001
HCO             0.00002   HCL             0.12553   *H2             0.32881
H2O             0.07681   NH2             0.00001   NH3             0.00001
*NO             0.00017   *N2             0.07302   *O              0.00016
*OH             0.00312   *O2             0.00001   AL2O3(L)        0.07613

  * THERMODYNAMIC PROPERTIES FITTED TO 20000.K

   PRODUCTS WHICH WERE CONSIDERED BUT WHOSE MOLE FRACTIONS
   WERE LESS THAN 5.000000E-06 FOR ALL ASSIGNED CONDITIONS

ALC             ALC2            ALH2            ALH2CL          ALH3
ALN             ALOCL2          ALO2            AL2             AL2C2
AL2CL6          AL2O3           *C              CCL             CCL2
CCL3            CCL4            *CH             CHCL            CHCL2
CHCL3           CH2             CH2CL           CH2CL2          CH3
CH3CL           CH2OH           CH3O            CH4             CH3OH
CH3OOH          *CN             CNN             COCL            COCL2
COHCL           COOH            *C2             C2CL            C2CL2
C2CL3           C2CL4           C2CL6           C2H             C2HCL
C2HCL3          C2H2,acetylene  C2H2,vinylidene C2H2CL2         CH2CO,ketene
O(CH)2O         HO(CO)2OH       C2H3,vinyl      C2H3CL          CH2CL-COOH
CH3CN           CH3CO,acetyl    C2H4            C2H4O,ethylen-o CH3CHO,ethanal
CH3COOH         OHCH2COOH       C2H5            C2H6            CH3N2CH3
C2H5OH          CH3OCH3         CH3O2CH3        CCN             CNC
OCCN            C2N2            C2O             *C3             C3H3,1-propynl
C3H3,2-propynl  C3H4,allene     C3H4,propyne    C3H4,cyclo-     C3H5,allyl
C3H6,propylene  C3H6,cyclo-     C3H6O,propylox  C3H6O,acetone   C3H6O,propanal
C3H7,n-propyl   C3H7,i-propyl   C3H8            C3H8O,1propanol C3H8O,2propanol
CNCOCN          C3O2            *C4             C4H2,butadiyne  C4H4,1,3-cyclo-
C4H6,butadiene  C4H6,1butyne    C4H6,2butyne    C4H6,cyclo-     C4H8,1-butene
C4H8,cis2-buten C4H8,tr2-butene C4H8,isobutene  C4H8,cyclo-     (CH3COOH)2
C4H9,n-butyl    C4H9,i-butyl    C4H9,s-butyl    C4H9,t-butyl    C4H10,n-butane
C4H10,isobutane C4N2            *C5             C5H6,1,3cyclo-  C5H8,cyclo-
C5H10,1-pentene C5H10,cyclo-    C5H11,pentyl    C5H11,t-pentyl  C5H12,n-pentane
```

```
C5H12,i-pentane CH3C(CH3)2CH3   C6H2            C6H5,phenyl     C6H5O,phenoxy
C6H6            C6H5OH,phenol   C6H10,cyclo-    C6H12,1-hexene  C6H12,cyclo-
C6H13,n-hexyl   C6H14,n-hexane  C7H7,benzyl     C7H8            C7H8O,cresol-mx
C7H14,1-heptene C7H15,n-heptyl  C7H16,n-heptane C7H16,2-methylh C8H8,styrene
C8H10,ethylbenz C8H16,1-octene  C8H17,n-octyl   C8H18,n-octane  C8H18,isooctane
C9H19,n-nonyl   C10H8,naphthale C10H21,n-decyl  C12H9,o-bipheny C12H10,biphenyl
CLCN            CLO             CLO2            CL2O            HALO
HCCN            HCCO            HNC             HNCO            HNO
HNO2            HNO3            HOCL            HO2             HCHO,formaldehy
HCOOH           H2O2            (HCOOH)2        *N              NCO
*NH             NH2OH           NOCL            NO2             NO2CL
NO3             NCN             N2H2            NH2NO2          N2H4
N2O             N2O3            N2O4            N2O5            N3
N3H             O3              AL(cr)          AL(L)           ALCL3(cr)
ALCL3(L)        ALH3(a)         ALN(cr)         ALN(L)          AL(OH)3(a)
AL4C3(cr)       C(gr)           H2O(cr)         H2O(L)          NH4CL(II)
NH4CL(III)

NOTE. WEIGHT FRACTION OF FUEL IN TOTAL FUELS AND OF OXIDANT IN TOTAL OXIDANTS

WARNING!!  CALCULATIONS WERE STOPPED BECAUSE NEXT POINT IS MORE
THAN 50 K BELOW THE TEMPERATURE RANGE OF A CONDENSED SPECIES (ROCKET)
```

APPENDIX B

MATLAB CODE

B.1 METHOD OF CHARACTERISTICS

```
clc; close all; clear all;

p_1 = 4000000  ;
T_1 = 3284;
FT = 0;
m_dot = 0.4914;
ALT = 1000;
g = 1.2;
R = 329;

T = 15.04 - 0.00649*ALT;
p_o = 1000*(101.29*((T+273.1)/288.08)^5.256);

PR = p_o/p_1;
PR2 = (p_o/p_1)^((g-1)/g);
TT = (2*g*R*T_1)/(g-1);
p_t = ((2/(g+1))^(g/(g-1)))*2.068;
v_t = sqrt((2*g*R*T_1)/(g+1));
v_e = sqrt(TT*(1-PR2));

if m_dot==0
    m_dot=FT/v_e;
elseif FT==0
    FT = m_dot/v_e;
else
    fprintf('You can either set desired thrust
        OR mass flow rate')
end

T_e = T_1*(p_o/p_1)^((g-1)/g);
a_e = sqrt(g*R*T_e);

Me = v_e/a_e;


TR = 7.8; %throat radius (mm)
RTOD = 180/pi;
DTOR = pi/180;
P = [];

A = sqrt((g+1)/(g-1));
B = (g-1)/(g+1);
v_PM = @(x) A*atan(sqrt(B*(x^2-1))) - atan(
   sqrt(x^2-1));

T_max = 0.5*v_PM(Me)*RTOD;
DT = (90-T_max) - fix(90-T_max);
T(1) = DT*DTOR;
n = T_max*2;

for m = 2:n+1
    T(m) = (DT + (m-1))*DTOR;
    %Mach from T(i) using T(i) = v_PM (FALSE
       POSITION)
    x_int = [1 1.01*Me];
    func = @(x) T(m) - v_PM(x);
    M(m) = fzero(func,x_int);
    P(m) = 0 + TR*tan(T(m)); %X-AXIS POINTS
    %RRSLOPES
    RR(m) = -TR/P(m);
    %LR slopes
    LR(m) = tan(T(m)+asin(1/M(m)));
    SL(m) = -RR(m);
end

P(1) = [];
l = length(P);

for j = 1:l
    P1 = [0 TR];
    P2 = [P(j) 0];
    plot(P2,P1,'k')
    hold on
    xlabel('CENTERLINE (mm)')
    ylabel('RADIUS (mm)')
end
hold on;
LR(1) = []; RR(1) = [];
SL(1) = [];
F = RR(m-1);

for c = 1:length(P)-1
    x(c) = (TR+SL(c)*P(c))/(SL(c)-F);
    y(c) = F*x(c)+TR;
    X_P = [P(c) x(c)];
    Y_P = [0 y(c)];
    plot(X_P,Y_P,'b');
end
hold on



TM = T_max*DTOR;
```

```
xw(1) = (TR+SL(1)*P(1))/(SL(1)-tan(TM));
yw(1) = tan(TM)*xw(1)+TR;
X_P2 = [P(1) xw];
Y_P2 = [P(2) yw];
plot(X_P2,Y_P2,'g');

DTW = tan(TM)/(length(P)-1);
s(1) = tan(TM);
b(1) = TR;

    for k = 2:length(P)-1
        s(k) = tan(TM)-(k-1)*DTW; %slope
        b(k) = yw(k-1)-s(k)*xw(k-1); %y-int
        xw(k) = (b(k)+SL(k)*P(k))/(SL(k)-s(k))
           ;
        yw(k) = s(k)*xw(k)+b(k);
        X_P3 = [x(k) xw(k)];
        Y_P3 = [y(k) yw(k)];
        plot(X_P3,Y_P3,'r');
    end
    hold on

    LAST POINT
    xf = (b(length(b))+SL(length(SL))*P(length
       (P)))/SL(length(SL));
    yf = b(length(b));
    X_F = [P(length(P)) xf];
    Y_F = [0 yf];
    plot(X_F,Y_F,'r');

    xw = [0 xw];
    yw = [TR yw];
    RTHROAT = TR;
    REXIT = yw(length(yw));

    AR = (RTHROAT/REXIT)^2

    xlswrite('PARAMS.xlsx',transpose(xw),'PTS'
       ,'A1:A62');
    xlswrite('PARAMS.xlsx',transpose(yw),'PTS'
       ,'B1:B62');
```

B.2 THRUST CALCULATION AT EACH TIME-STEP

```
A = 0.001168999;
Astar = 0.000261586;
pb = 101325;
Te = 1445;
gamma = 1.2;
R = 330.9;

t = load('Time.mat','VarName1');
t = struct2cell(t);
t = t{1};
t = transpose(t);
P = load('Pressure.mat','Pb');
P = struct2cell(P);
P = P{1};
P = transpose(P);
n = length(P);
%P = P.*100000;

T = findthrust(P,n,pb,Te,gamma,A,Astar,R);

```

```
figure(1)
plot(t,P)
xlabel('Time')
ylabel('Pressure')
title('Pressure vs time')

figure(2)
plot(t,T)
xlabel('Time')
ylabel('Thrust')
title('Thrust vs time')

function Thrust = findthrust(P,n,pb,Te,gamma,A
    ,Astar,R)
% Calculations
a = A/Astar;
fprintf('\nThe value of A/Astar is %g\n',a)
g = gamma;
c1 = (g+1)/(2*(g-1));
c2 = (g-1)/2;
c3 = (g+1)/2;
m = 0.01:0.0000001:1;
val1 = ((1./m).*(((1 + c2.*m.*m)./c3).^c1)) -
    a;
[~,ind] = min(abs(val1));
M1 = m(1,ind);   % subsonic mach
fprintf('\n The value of Subsonic Mach number
    for given A/Astar value is %g\n',M1)
m = 1:0.0000001:4;
val2 = ((1./m).*(((1 + c2.*m.*m)./c3).^c1)) -
    a;
[~,ind] = min(abs(val2));
M2 = m(1,ind);   %super sonic mach
```

```
fprintf('\n The value of Supersonic Mach
   number for given A/Astar value is %g\n',M2)

prat1 = (1 + c2*M1*M1)^(g/(1-g));
p01 = pb/prat1;
fprintf('\nP01 value is %g\n',p01)

prat2 = (1 + c2*M2*M2)^(g/(1-g));
p02 = pb/prat2;
fprintf('\nP02 value is %g\n',p02);

Md = (((g-1)*M2^2 + 2)/(2*g*M2^2 - (g-1)))
   ^0.5;
pratd = (1 + c2*Md*Md)^(g/(1-g));

p21 = ((((g+1)*M2^2)/(((g-1)*M2^2)+2))^(g/(g
   -1)))*(((g+1)/((2*g*M2^2)-(g-1)))^(1/(g-1))
   );

pbi = pratd*p21;

p0i = pb/pbi;
fprintf('\nP0i value is %g\n',p0i);

Mi = ((1-((pb/p0i)^((1-g)/g)))*(2/(1-g)))
   ^00.5;
fprintf('\n Value of Mi is %g\n',Mi)
M = zeros(1,n); % Exit Mach number
mdot = zeros(1,n); % Mass flow rate
PE = zeros(1,n); % Exit Pressure

for i = 1:1:n
    pc = P(1,i);
    if pc <= p01
        M(1,i) = ((1-((pb/pc)^((1-g)/g)))
            *(2/(1-g)))^00.5;
        mdot(1,i) = A*pc*(g/R)^0.5 * M(1,i)*(1
            + c2* M(1,i)*M(1,i))^(-c1);
        PE(1,i) = pb;
    end
    if pc > p01 && pc <= p0i
        pbc = pb/pc;
        Asrat = findAs(pbc,a,g);
        As = Asrat*Astar;
        m = 1:0.0000001:4;
        val2 = ((1./m).*(((1 + c2.*m.*m)./c3)
            .^c1)) - Asrat;
        [~,ind] = min(abs(val2));
        Mup = m(1,ind);   %super sonic mach
        Mdo = ((1 + c2*Mup*Mup)/(g*Mup*Mup -
            c2))^0.5;
        A2star = Mdo.*As.*(1./((1./Mdo).*(((1
            + c2.*Mdo.*Mdo)./c3).^c1)));
        m = 0.01:0.0000001:1;
        val1 = ((1./m).*(((1 + c2.*m.*m)./c3)
            .^c1)) - (A./A2star);
        [~,ind] = min(abs(val1));
        M(1,i) = m(1,ind);   % subsonic mach
        mdot(1,i) = A*pc*(g/R)^0.5 * M(1,i)*(1
            + c2* M(1,i)*M(1,i))^(-c1);
        Pt1t0 = ((((g+1)*Mup^2)/(((g-1)*Mup^2)
            +2))^(g/(g-1)))*(((g+1)/(2*g*Mup^2
            - (g-1)))^(1/(g-1)));
        Pt1 = Pt1t0*pc;
        PE(1,i) = Pt1* (1 + c2*M(1,i)^2)^(-g/(
            g-1));
    end
    if pc > p0i
        M(1,i) = M2;
        mdot(1,i) = A*pc*(g/R)^0.5 * M(1,i)*(1
            + c2* M(1,i)*M(1,i))^(-c1);
        PE(1,i) = pc* (1 + c2*M(1,i)^2)^(-g/(g
           -1));
    end
end

Thrust = -mdot.*M.*((g*R*Te)^0.5).*0.1 - (PE -
    pb).*A;
end
```

B.2.1 Normal Shock Area function

```
#save this file in same directory as findAs.m

function Asrat = findAs(pbc,a,g)

    Ae_At        = a;

       % Ratio of specific heats []
    setPbP0_NSN = pbc;



    gm1    = g-1;
    gp1    = g+1;
    gogm1  = g/gm1;
    gm1o2  = gm1/2;
    gogp1  = g/gp1;


    Me_Sub     = ISENTROPIC_FLOW(Ae_At,'Asub',g,'M');
    Pe_P0_Sub  = (1 + gm1o2*Me_Sub^2)^(-gogm1);

    Me_Sup     = ISENTROPIC_FLOW(Ae_At,'Asup',g,'M');
    Pb_P0_Sup  = (1 + gm1o2*Me_Sup^2)^(-gogm1);

    P2_P1      = 1 + ((2*gogp1)*(Me_Sup^2-1));
    Pe_P0_NSE  = (P2_P1)*(Pb_P0_Sup);
    Me_NSE     = NORMAL_SHOCK(Me_Sup,'M1',g,'M2');


    if (setPbP0_NSN > Pe_P0_NSE && setPbP0_NSN < Pe_P0_Sub)


        errTol   = 1e-6;
        Arat_Lo  = 1.00001;
        Arat_Hi  = Ae_At;
        err      = inf;


        while (err > errTol)


            Arat_Mid = 0.5*(Arat_Hi + Arat_Lo);


            Pb_P0_Lo   = NS_NOZZLE(Arat_Lo,Ae_At,g);
            Pb_P0_Mid  = NS_NOZZLE(Arat_Mid,Ae_At,g);
            Pb_P0_Hi   = NS_NOZZLE(Arat_Hi,Ae_At,g);


            errLo   = setPbP0_NSN - Pb_P0_Lo;
            errMid  = setPbP0_NSN - Pb_P0_Mid;
            errHi   = setPbP0_NSN - Pb_P0_Hi;


            if (errLo*errMid < 0)
                Arat_Hi = Arat_Mid;
            elseif (errHi*errMid < 0)
                Arat_Lo = Arat_Mid;
            end


            err = abs(errMid-errLo);
        end


        ANS_A1s_Iter = Arat_Mid;

        term1 = -(1/gm1);
```

```
          term2 = 1/(gm1^2);
          term3 = 2/gm1;
          term4 = (2/gp1)^(gp1/gm1);
          term5 = (setPbP0_NSN*Ae_At)^-2;
          Me    = sqrt(term1 + sqrt(term2 + term3*term4*term5));


          P0e_Pe = (1 + (gm1o2*Me^2))^(g/gm1);
          P02_P01 = P0e_Pe*setPbP0_NSN;


          problem.objective = @(M) exp(-gogm1*log(((2 + gm1*M^2)/(
              gp1*M^2))*...
                                                    (1+(2*g/gp1)*(M^2-1))) +
                                                       ...
                                                    log((1+(2*g/gp1)*(M^2-1)
                                                       ))) - P02_P01;
          problem.x0          = [1+1e-5 50];
          problem.solver      = 'fzero';
          problem.options     = optimset(@fzero);
          M1                  = fzero(problem);


          num = 1 + gm1o2*M1^2;
          den = (g*M1^2) - gm1o2;
          M2  = sqrt(num/den);


          ANS_A1s_Dir = ISENTROPIC_FLOW(M1,'M',g,'AAs');
      else
          fprintf('Set pressure ratio does not give NS in nozzle!\
              n\n');
          ANS_A1s_Iter = 1;
          ANS_A1s_Dir  = 1;
      end
      Asrat = ANS_A1s_Iter;
  end
```

B.3 TRAJECTORY : VERSION 1

```
clear;clc;clear all
m = 5;
```

```
g = 9.81;
cd0 = 0.01; k = 0; cl = 0; rho = 1.225; S =
   0.6;

T1 = load('Thrusttime.mat');
T1 = struct2cell(T1);
T1 = T1{1};
T1 = transpose(T1);

t = linspace(0,10,10000);

V0 = 1; gamma0 = -73.33*pi/180; x0 = 0; y0 =
   2000;
[t1,y1] = ode45(@(t,y)flig(t,y,m,g,cd0,cl,k,S,
   rho,T1),t,[V0;gamma0;x0;y0]);
z1 =linspace(8,57,10000);

figure(1)
tiledlayout(2,1)
nexttile
plot(y1(:,3),y1(:,4),'linewidth',2)
ylim([0 (y0+400)])
xlabel('distance travelled (m)')
ylabel('Altitude of the missile (m)')
title('Trajectory')

nexttile
plot(t1,y1(:,4),'r','linewidth',2)
ylim([0 (y0+400)])
xlabel('Time (s)')
ylabel('Altitude of the missile (m)')
title('Trajectory')

```

```
gamma = 0:-0.01:-pi/2.1;
ran = zeros(1,length(gamma));


for i = 1:length(gamma)
     gamma0 = gamma(1,i);
    [t1,y1] = ode45(@(t,y)flig(t,y,m,g,cd0,cl,k
       ,S,rho,T),t,[V0;gamma0;x0;y0]);
    ran(1,i) = findrange(y1);
end

reqrange = 1500;
[~,b] = min(abs(ran - reqrange));
reqgamma = gamma(1,b);
fprintf('\n Therefore gamma required for the
   given range of %g is %g \n',reqrange,
   reqgamma*180/pi)

[t1,y1] = ode45(@(t,y)flig(t,y,m,g,cd0,cl,k,S,
   rho,T),t,[V0;reqgamma;x0;y0]);

figure(3)
plot(y1(:,3),y1(:,4),'linewidth',2)
ylim([0 (y0+400)])
xlabel('distance travelled (m)')
ylabel('Altitude of the missile (m)')
title('Trajectory')


function r = findrange(y)
```

```
[~,a] = min(abs(y(:,4)));
r = y(a,3);
end

function dy = flig(t,y,m,g,cd0,cl,k,S,rho,T1)
T = interp1(T1(:,1),T1(:,2),t);
D = 0.5*rho*S*y(1)*y(1)*(cd0 + k*cl*cl);
dy(1,1) = (T - D - m*g*cos(y(2)))/m;
dy(2,1) = (-m*g*cos(y(2)))/(m*y(1));
dy(3,1) = y(1)*cos(y(2));
dy(4,1) = y(1)*sin(y(2));
end
```

B.4 TRAJECTORY : VERSION 2

```
clear; clear all; clc;
m = 5;
g = 9.81;
cd0 = 0.01; k = 0; cl = 0; rho = 1.225; S =
    0.6;

T1 = load('Thrusttime.mat');
T1 = struct2cell(T1);
T1 = T1{1};
T1 = transpose(T1);

t = linspace(0,10,10000);
V0 = 100; x0 = 0; y0 = 950; xob = 650; yob =
    350; rob = 100;
rsafe = 50; ysafe = 20; xdes = xob + 600; d0 =
     3;
```

```
gamma = 0:-0.01:-pi/2.1;
y1val = zeros(1,length(gamma));

for i = 1:length(gamma)
    gamma0 = gamma(1,i);
   [t1,y1] = ode45(@(t,y)flig(t,y,m,g,cd0,cl,k
     ,S,rho,T),t,[V0;gamma0;x0;y0]);
   y1val(1,i) = interp1(y1(:,3),y1(:,4),xob-(
     rob+rsafe));
end

[~,ind] = min(abs(y1val - (yob+8*ysafe)));
gamma0 = gamma(1,ind);

[t01,y01] = ode45(@(t,y)flig(t,y,m,g,cd0,cl,k,
   S,rho,T),t,[V0;gamma0;x0;y0]);
[~,ind] = min(abs(y01(:,3)-(xob-(rob+rsafe))))
   ;
tp1 = t01(ind,1); yp1 = y01(ind,4); xp1 = y01(
   ind,3); gamma1 = y01(ind,2);
V1 = y01(ind,1);

I = 0.36;
t = linspace(tp1,tp1+0.5,10000);

clc = 0.00001:10000:1;
yval2 = zeros(1,length(clc));

for i = 1:length(clc)
    clcan = clc(1,i);
    t = linspace(0,10,10000);
    [t2,y2] = ode45(@(t,y)turn(t,y,m,g,cd0,cl,
        k,S,rho,T,clcan,xob,rob,rsafe,d0,I),t,[
        V1;gamma1;xp1;yp1;0]);
    A = find(y2(:,2)>0);
    B = A(1,1);
    yval2(1,i) = t2(B,1);
end

[~,k0] = min(abs(yval2-0.5));
clcan = clc(1,k0);

t = linspace(tp1,tp1+0.5,10000);
[t02,y02] = ode45(@(t,y)turn(t,y,m,g,cd0,cl,k,
   S,rho,T,clcan,xob,rob,rsafe,d0,I),t,[V1;
   gamma1;xp1;yp1;0]);

[~,ind1] = min(abs(y02(:,2)));
yp2 = y02(ind1,4); xp2 = y02(ind1,3); gamma2 =
    y02(ind1,2);
V2 = y02(ind1,2);
tp2 = t02(ind1,1);

t = linspace(tp2,tp2+0.15,10000);
[t03,y03] = ode45(@(t,y)turn(t,y,m,g,cd0,cl,k,
   S,rho,T,-clcan*4,xob,rob,rsafe,d0,I),t,[V1;
   gamma2;xp2;yp2;0]);
tp3 = t03(end,1);

t = linspace(tp3,tp3+10,10000);
[t04,y04] = ode45(@(t,y)flig(t,y,m,g,cd0,cl,k,
   S,rho,0),t,[y03(end,1)/20;y03(end,2);y03(
   end,3);y03(end,4)]);

figure(1)
```

```
P1 = [y01(1:ind,3) y01(1:ind,4)];
P2 = [y02(1:ind1,3) y02(1:ind1,4)];
P3 = [y03(:,3) y03(:,4)];
P4 = [y04(:,3) y04(:,4)];
P = [P1;P2;P3;P4];

Vf1 = [y01(1:ind,1) y01(1:ind,2)];
Vf2 = [y02(1:ind1,1) y02(1:ind1,2)];
Vf3 = [y03(:,1) y03(:,2)];
Vf4 = [y04(:,1) y04(:,2)];
Vf = [Vf1;Vf2;Vf3;Vf4];

time = [t01(1:ind,1);t02(1:ind1,1);t03(:,1);
   t04(:,1)];

plot(P(:,1),P(:,2),[xob,xob],[0,yob+25])
xlabel('Distance travelled')
ylabel('Height from ground')
title('Trajectory when obstacle is present')
legend('Trajectory followed','Obstacle')
hold on
ylim([0 y0+500])

figure(2)
plot(time(:,1),P(:,2))
xlabel('Time')
ylabel('Height from ground')
title('Height vs time')
function dy = flig(t,y,m,g,cd0,cl,k,S,rho,T1)
D = 0.5*rho*S*y(1)*y(1)*(cd0 + k*cl*cl);
if t > 4
    T = interp1(T1(:,1),T1(:,2),t);
end
```

```
dy(1,1) = (T - D - m*g*sin(y(2)))/m;
dy(2,1) = (-m*g*cos(y(2)))/(m*y(1));
dy(3,1) = y(1)*cos(y(2));
dy(4,1) = y(1)*sin(y(2));
end

function dy = turn(t,y,m,g,cd0,cl,k,S,rho,T1,
   clcan,xob,rob,rsafe,d0,I)
D = 0.5*rho*S*y(1)*y(1)*(cd0 + k*cl*cl);
L = 0.5*rho*S*y(1)*y(1)*clcan;
if t > 4
    T = interp1(T1(:,1),T1(:,2),t);
end
dy(1,1) = (T - D - m*g*sin(y(2)))/m;
dy(2,1) = ((-m*g*cos(y(2)))/(m*y(1)) + y(5));
dy(3,1) = y(1)*cos(y(2));
dy(4,1) = y(1)*sin(y(2));
dy(5,1) = 2*L*d0/I;
end
```

APPENDIX C

TRACKING CODE

C.1 COLOR DETECTION

```
import cv2
import numpy as np

img = cv2.imread('yellow_object.JPG', 1)

img = cv2.resize(img, (0,0), fx=0.2, fy=0.2)
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

lower_range = np.array([24, 100, 100], dtype=np.uint8)
upper_range = np.array([44, 255, 255], dtype=np.uint8)
mask = cv2.inRange(hsv, lower_range, upper_range)
cv2.imshow('mask',mask)
cv2.imshow('image', img)

while(1):
  k = cv2.waitKey(0)
  if(k == 27):
    break

cv2.destroyAllWindows()
```

C.2 BGR TO HSV

```
import sys
import numpy as np
import cv2

blue = sys.argv[1]
green = sys.argv[2]
red = sys.argv[3]

color = np.uint8([[[blue, green, red]]])
hsv_color = cv2.cvtColor(color, cv2.COLOR_BGR2HSV)
```

```python

hue = hsv_color[0][0][0]

print("Lower bound is :"),
print("[" + str(hue-10) + ", 100, 100]\n")

print("Upper bound is :"),
print("[" + str(hue + 10) + ", 255, 255]")
```

C.3 OBJECT TRACKING IN RECORDED VIDEO

```python
from imutils.video import VideoStream
from imutils.video import FPS
import argparse
import imutils
import time
import cv2

ap = argparse.ArgumentParser()
ap.add_argument("-v", "--video", type=str,
        help="path to input video file")
ap.add_argument("-t", "--tracker", type=str, default="kcf
   ",
        help="OpenCV object tracker type")
args = vars(ap.parse_args())

(major, minor) = cv2.__version__.split(".")[:2]
if int(major) == 3 and int(minor) < 3:
        tracker = cv2.Tracker_create(args["tracker"].
           upper())
else:
        OPENCV_OBJECT_TRACKERS = {
                "csrt": cv2.TrackerCSRT_create,
                "kcf": cv2.TrackerKCF_create,
                "boosting": cv2.TrackerBoosting_create,
                "mil": cv2.TrackerMIL_create,
                "tld": cv2.TrackerTLD_create,
                "medianflow": cv2.
                   TrackerMedianFlow_create,
                "mosse": cv2.TrackerMOSSE_create
        }
        tracker = OPENCV_OBJECT_TRACKERS[args["tracker"]]()
initBB = None

if not args.get("video", False):
        print("[INFO] starting video stream...")
        vs = VideoStream(src=0).start()
        time.sleep(1.0)
else:
        vs = cv2.VideoCapture(args["video"])
fps = None

while True:
        frame = vs.read()
        frame = frame[1] if args.get("video", False) else frame

        if frame is None:
                break
        frame = imutils.resize(frame, width=500)
        (H, W) = frame.shape[:2]


        if initBB is not None:

                (success, box) = tracker.update(frame)

                if success:
                        (x, y, w, h) = [int(v) for v in box]
                        cv2.rectangle(frame, (x, y), (x + w, y + h),
                                (0, 255, 0), 2)

                fps.update()
                fps.stop()

                info = [
                        ("Tracker", args["tracker"]),
                        ("Success", "Yes" if success else "No"),
                    ("FPS", "{:.2f}".format(fps.fps()
                       )),
                ]

                for (i, (k, v)) in enumerate(info):
                        text = "{}: {}".format(k, v)
                        cv2.putText(frame, text, (10, H -
                            ((i * 20) + 20)),
                                cv2.FONT_HERSHEY_SIMPLEX,
                                   0.6, (0, 0, 255), 2)
        cv2.imshow("Frame", frame)
        key = cv2.waitKey(1) & 0xFF

        if key == ord("s"):
                initBB = cv2.selectROI("Frame", frame,
                   fromCenter=False,
                        showCrosshair=True)
                tracker.init(frame, initBB)
                fps = FPS().start()

        elif key == ord("q"):
                break
if not args.get("video", False):
        vs.stop()
else:
        vs.release()
cv2.destroyAllWindows()
```

C.4 OBJECT TRACKING IN LIVE VIDEO

```
from collections import deque
import numpy as np
import argparse
import imutils
import cv2

ap = argparse.ArgumentParser()
ap.add_argument("-v", "--video",
        help="path to the (optional) video file")
ap.add_argument("-b", "--buffer", type=int, default=64,
        help="max buffer size")
```

```
args = vars(ap.parse_args())

colorLower = (24, 100, 100)
colorUpper = (44, 255, 255)
pts = deque(maxlen=args["buffer"])

if not args.get("video", False):
        camera = cv2.VideoCapture(0)
else:
        camera = cv2.VideoCapture(args["video"])

while True:
        (grabbed, frame) = camera.read()

        if args.get("video") and not grabbed:
                break

        frame = imutils.resize(frame, width=600)
        frame = imutils.rotate(frame, angle=180)
        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

        mask = cv2.inRange(hsv, colorLower, colorUpper)
        mask = cv2.erode(mask, None, iterations=2)
        mask = cv2.dilate(mask, None, iterations=2)

        cnts = cv2.findContours(mask.copy(), cv2.
           RETR_EXTERNAL,
                cv2.CHAIN_APPROX_SIMPLE)[-2]
        center = None

        if len(cnts) > 0:
                c = max(cnts, key=cv2.contourArea)
                ((x, y), radius) = cv2.minEnclosingCircle
                   (c)
                M = cv2.moments(c)
                center = (int(M["m10"] / M["m00"]), int(M
                   ["m01"] / M["m00"]))

                if radius > 10:
                        cv2.circle(frame, (int(x), int(y)
                           ), int(radius),
                                        (0, 255, 255), 2)
                                cv2.circle(frame, center, 5, (0,
                                    0, 255), -1)

                pts.appendleft(center)

                for i in range(1, len(pts)):
                        if pts[i - 1] is None or pts[i] is None:
                                continue

                        thickness = int(np.sqrt(args["buffer"] /
                            float(i + 1)) * 2.5)
                        cv2.line(frame, pts[i - 1], pts[i], (0,
                            0, 255), thickness)

                cv2.imshow("Frame", frame)
                key = cv2.waitKey(1) & 0xFF
                # if the 'q' key is pressed, stop the loop
                if key == ord("q"):
                        break

camera.release()
cv2.destroyAllWindows()
```

C.5 GET OBJECT COORDINATE IN LIVE VIDEO

```
from __future__ import print_function
from imutils.video import VideoStream
import imutils
import time
import cv2
import os
import RPi.GPIO as GPIO
redLed = 21
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(redLed, GPIO.OUT)
def mapObjectPosition (x, y):
    print ("[INFO] Object Center coordenates at X0 = {0}
        and Y0 =  {1}".format(x, y))
vs = VideoStream(0).start()
```

```
time.sleep(2.0)
colorLower = (24, 100, 100)
colorUpper = (44, 255, 255)
GPIO.output(redLed, GPIO.LOW)
ledOn = False
while True:
        frame = vs.read()
        frame = imutils.resize(frame, width=500)
        frame = imutils.rotate(frame, angle=180)
        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

        mask = cv2.inRange(hsv, colorLower, colorUpper)
        mask = cv2.erode(mask, None, iterations=2)
        mask = cv2.dilate(mask, None, iterations=2)

        cnts = cv2.findContours(mask.copy(), cv2.
           RETR_EXTERNAL,
                cv2.CHAIN_APPROX_SIMPLE)
        cnts = cnts[0] if imutils.is_cv2() else cnts[1]
        center = None

        if len(cnts) > 0:
                c = max(cnts, key=cv2.contourArea)
                ((x, y), radius) = cv2.minEnclosingCircle
                   (c)
                M = cv2.moments(c)
                center = (int(M["m10"] / M["m00"]), int(M
                   ["m01"] / M["m00"]))

                if radius > 10:
                        cv2.circle(frame, (int(x), int(y)
                           ), int(radius),
                                (0, 255, 255), 2)
                        cv2.circle(frame, center, 5, (0,
                           0, 255), -1)

                        mapObjectPosition(int(x), int(y))

                        if not ledOn:
                                GPIO.output(redLed, GPIO.
                                   HIGH)
```

```
                                        ledOn = True
            elif ledOn:
                    GPIO.output(redLed, GPIO.LOW)
                    ledOn = False
            cv2.imshow("Frame", frame)

            key = cv2.waitKey(1) & 0xFF
            if key == 27:
                break
GPIO.cleanup()
cv2.destroyAllWindows()
vs.stop()
```

C.6 LIVE OBJECT TRACK IN 2 AXIS GIMBAL

```
from __future__ import print_function
from imutils.video import VideoStream
import argparse
import imutils
import time
import cv2
import os
import RPi.GPIO as GPIO
panServo = 27
tiltServo = 17
redLed = 21
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(redLed, GPIO.OUT)

def positionServo (servo, angle):
    os.system("python angleServoCtrl.py " + str(servo) +
       " " + str(angle))
    print("[INFO] Positioning servo at GPIO {0} to {1}
       degrees\n".format(servo, angle))

def mapServoPosition (x, y):
    global panAngle
    global tiltAngle
    if (x < 220):
        panAngle += 10
        if panAngle > 140:
            panAngle = 140
        positionServo (panServo, panAngle)

    if (x > 280):
        panAngle -= 10
        if panAngle < 40:
            panAngle = 40
        positionServo (panServo, panAngle)

    if (y < 160):
        tiltAngle += 10
        if tiltAngle > 140:
            tiltAngle = 140
        positionServo (tiltServo, tiltAngle)

    if (y > 210):
        tiltAngle -= 10
        if tiltAngle < 40:
            tiltAngle = 40
        positionServo (tiltServo, tiltAngle)

print("[INFO] waiting for camera to warmup...")
vs = VideoStream(0).start()
time.sleep(2.0)
colorLower = (24, 100, 100)
colorUpper = (44, 255, 255)

GPIO.output(redLed, GPIO.LOW)
ledOn = False

global panAngle
panAngle = 90
global tiltAngle
tiltAngle =90

positionServo (panServo, panAngle)
positionServo (tiltServo, tiltAngle)

while True:
        frame = vs.read()
        frame = imutils.resize(frame, width=500)
        frame = imutils.rotate(frame, angle=180)
        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

        mask = cv2.inRange(hsv, colorLower, colorUpper)
        mask = cv2.erode(mask, None, iterations=2)
        mask = cv2.dilate(mask, None, iterations=2)

        cnts = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL,
                cv2.CHAIN_APPROX_SIMPLE)
        cnts = cnts[0] if imutils.is_cv2() else cnts[1]
        center = None

        if len(cnts) > 0:
                c = max(cnts, key=cv2.contourArea)
                ((x, y), radius) = cv2.minEnclosingCircle(c)
                M = cv2.moments(c)
                center = (int(M["m10"] / M["m00"]), int(M["m01"] / M["m00"]))
                if radius > 10:
                        cv2.circle(frame, (int(x), int(y)), int(radius),
                                (0, 255, 255), 2)
                        cv2.circle(frame, center, 5, (0, 0, 255), -1)

                        mapServoPosition(int(x), int(y))

                        if not ledOn:
                                GPIO.output(redLed, GPIO.HIGH)
                                ledOn = True

        elif ledOn:
                GPIO.output(redLed, GPIO.LOW)
                ledOn = False
        cv2.imshow("Frame", frame)
        key = cv2.waitKey(1) & 0xFF
        if key == 27:
            break
positionServo (panServo, 90)
positionServo (tiltServo, 90)
GPIO.cleanup()
cv2.destroyAllWindows()
vs.stop()
```

APPENDIX D

PYTHON CODE FOR ML & DL TRAJECTORY PREDICTION ANALYSIS

D.1 IMPORT LIBRARY FILE AND LOAD DATASET

```
%matplotlib inline
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,
   plot_confusion_matrix
from imblearn.over_sampling import SMOTE
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import
   GradientBoostingClassifier
from sklearn.utils import class_weight
from keras import models
from keras import layers
from keras import regularizers
from keras import optimizers
import tensorflow as tf
import random as rn

os.environ['PYTHONHASHSEED'] = '0'
np.random.seed(1)
rn.seed(2)
tf.random.set_seed(3)

df = pd.read_csv('/content/drive/MyDrive/dataset.csv
   ')
```

D.2 DISPLAY THE FIRST 5 ROWS

```
df.head()
df.info()
```

D.3 SMOTE TECHNIQUE FOR CLASS BALANCING

```
X_train_oversampled, y_train_oversampled = SMOTE().
    fit_resample(X_train, y_train)

print("Oversampled train X: ", X_train_oversampled.
    shape)
print("Oversampled train y: ", y_train_oversampled.
    shape)

pd.DataFrame(y_train_oversampled).value_counts()
# Create x, where x the 'scores' column's values as
    floats
x = df[['Velocity']].values.astype(float)
x
```

D.4 NORMALIZATION

```
# Create a minimum and maximum processor object
min_max_scaler = preprocessing.MinMaxScaler()

# Create an object to transform the data to fit
    minmax processor
x_scaled = min_max_scaler.fit_transform(x)

# Run the normalizer on the dataframe
#df_normalized = pd.DataFrame(x_scaled)
x_scaled
```

D.5 FILLING NULL VALUES TEST TRAIN SPLIT

```
df['Velocity'] = x_scaled
df
df.head()
df = df.fillna(df.mean())
from sklearn.model_selection import train_test_split
```

```
y = df['x','y']
x = df.drop('x','y', axis = 1)
x_train, x_test, y_train, y_test = train_test_split(
    x, y, test_size = 0.3, random_state = 42)

print(x_train)
print(x_test)
print(y_train)
print(y_test)
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

D.6 RANDOM FOREST REGRESSOR

```
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(max_depth=2,
    random_state=0, n_estimators=100)
regressor.fit(x_train, y_train)
b_pred = regressor.predict(x_test)
from sklearn.metrics import mean_squared_error as
    mse
from sklearn.metrics import mean_absolute_error as
    mae
from sklearn.metrics import r2_score
print('MSE␣=', mse(y_pred, y_test))
print('MAE␣=', mae(y_pred, y_test))
print('R2␣Score␣=', r2_score(y_pred, y_test))
```

D.7 ARTIFICIAL NEURAL NETWORK

```
from keras.callbacks import ModelCheckpoint
from keras.models import Sequential
from keras.layers import Dense, Activation, Flatten
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error
from matplotlib import pyplot as plt
import seaborn as sb
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import warnings
from keras.callbacks import History
warnings.filterwarnings('ignore')
warnings.filterwarnings('ignore', category=
    DeprecationWarning)
NN_model = Sequential()

# The Input Layer :
NN_model.add(Dense(128, kernel_initializer='normal',
    input_dim = a_train.shape[1], activation='relu'))

# The Hidden Layers :
NN_model.add(Dense(256, kernel_initializer='normal',
    activation='relu'))
NN_model.add(Dense(256, kernel_initializer='normal',
    activation='relu'))
NN_model.add(Dense(256, kernel_initializer='normal',
    activation='relu'))

# The Output Layer :
NN_model.add(Dense(1, kernel_initializer='normal',
    activation='linear'))

# Compile the network :
NN_model.compile(loss='mean_absolute_error',
    optimizer='adam', metrics=['mean_absolute_error'])
NN_model.summary()
from keras.callbacks import History
history = History()
History=NN_model.fit(a_train, b_train, epochs=50,
    batch_size=500, validation_split = 0.2, callbacks
    =[history])
```

```
print(history.history.keys())
plt.plot(History.history['mean_absolute_error'])
plt.ylabel('mean_absolute_error')
plt.xlabel('epoch')

plt.plot(History.history['mean_absolute_error'])
plt.plot(History.history['val_mean_absolute_error'])
plt.title('mean_absolute_error')
plt.ylabel('mean_absolute_error')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper␣left')
plt.show()
 # summarize history for loss
plt.plot(History.history['loss'])
plt.plot(History.history['val_loss'])
plt.title('model␣loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper␣left')
plt.show()
```

D.8 XGB REGRESSOR

```
from xgboost import XGBRegressor
from sklearn.metrics import mean_absolute_error
XGBModel = XGBRegressor()
XGBModel.fit(a_train,b_train , verbose=False)
accuracy = XGBModel.score(a_test,b_test)
print('accuracy␣=',accuracy*100)

# Get the mean absolute error on the validation data
    :
XGBpredictions = XGBModel.predict(a_test)
MAE = mean_absolute_error(b_test , XGBpredictions)
print('MAE␣=␣',MAE)
```

D.9 LINEAR REGRESSION

```
from sklearn.linear_model import LinearRegression
linear_regression = LinearRegression()
linear_regression.fit(X_train,y_train)
accuracy = linear_regression.score(X_test,y_test)
print(accuracy*100,'%')
```

D.10 LASSO REGRESSION

```
from sklearn.linear_model import Lasso
lasso = Lasso(alpha=0.3,max_iter=1)
lasso.fit(X_train,y_train)
accuracy = lasso.score(X_test,y_test)
print(accuracy*100,'%')
```

D.11 CORRELATION HEATMAP

```
import matplotlib.pyplot as plt
import seaborn as sb

C_mat = dataset.corr()
fig = plt.figure(figsize = (15,15))

sb.heatmap(C_mat, vmax = .8, square = True)
plt.show()
```

REFERENCES

[1] Darpa Defense Advanced Research Projects Agency 1958-2018, 60 years edition `https://www.darpa.mil/attachments/DARAPA60_publication-no-ads.pdf`

[2] Squadron Leader Piyush Raj, Integration Of Guided Weapon On Indigenously Developed UAV, 2021 [MTech Thesis]. IIT Madras.

[3] Sqn Ldr Raviraj S R, Design Of Rocket For Indigenously Developed UAV, 2020 [MTech Thesis]. IIT Madras

[4] Niharika, Developing A Guidance System For A Low-Cost Missile, 2021, [DDP]. IIT Madras

[5] Zaikang, Qi, and Lin Defu. Design of Guidance and Control Systems for Tactical Missiles. CRC Press, 2019.

[6] Yanushevsky, R. T. (2018). Modern missile guidance. CRC Press.

[7] Siouris, George M. Missile guidance and control systems. Springer Science Business Media, 2004.

[8] Zarchan, Paul. Tactical and strategic missile guidance. American Institute of Aeronautics and Astronautics, Inc., 2012.

[9] Zarchan, Paul, ed. "Tactical and Strategic Missile Guidance, -SET." (2019).

[10] White, Brian A., and Antonios Tsourdos. "Modern missile guidance design: An overview." IFAC Proceedings Volumes 34.15 (2001): 431-436.

[11] Rogers, Steve. "Missile guidance comparison." In AIAA Guidance, Navigation, and Control Conference and Exhibit, p. 4882. 2004.

[12] Adler, Fred P. "Missile guidance by three-dimensional proportional navigation." Journal of Applied Physics 27.5 (1956): 500-507.

[13] Li, Z., Xia, Y., Su, C. Y., Deng, J., Fu, J., He, W. (2014). Missile guidance law based on robust model predictive control using neural-network optimization. IEEE transactions on neural networks and learning systems, 26(8), 1803-1809.

[14] Wang, Xianghua, and Jinzhi Wang. "Partial integrated missile guidance and control with finite time convergence." Journal of Guidance, Control, and Dynamics 36,

no. 5 (2013): 1399-1409.

[15] Imado, F., Kuroda, T. and Tahk, M.J., 1998, August. A new missile guidance algorithm against a maneuvering target. In Guidance, navigation, and control conference and exhibit (p. 4114).

[16] Zarchan, P. (1979). Complete statistical analysis of nonlinear missile guidance systems-SLAM. Journal of guidance and control, 2(1), 71-78.

[17] Rodriguez, José F., James P. Thomas, and John E. Renaud. "Maximizing the strength of fused-deposition ABS plastic parts." 1999 International Solid Freeform Fabrication Symposium. 1999.

[18] Thostenson, Erik T., and Tsu-Wei Chou. "Processing-structure-multi-functional property relationship in carbon nanotube/epoxy composites." Carbon 44, no. 14 (2006): 3022-3029.

[19] Fiorina, M., A. Seman, Bruno Castanié, K. M. Ali, C. Schwob, and L. Mezeix. "Spring-in prediction for carbon/epoxy aerospace composite structure." Composite Structures 168 (2017): 739-745.

[20] Zhou, Yuanxin, Farhana Pervin, Lance Lewis, and Shaik Jeelani. "Fabrication and characterization of carbon/epoxy composites mixed with multi-walled carbon nanotubes." Materials Science and Engineering: A 475, no. 1-2 (2008): 157-165.

[21] Mcmanus, H. L., Springer, G. S. (1992). High temperature thermomechanical behavior of carbon-phenolic and carbon-carbon composites, i. analysis. Journal of Composite Materials, 26(2), 206-229.

[22] Torres-Herrador, Francisco, Alessandro Turchi, Kevin M. Van Geem, Julien Blondeau, and Thierry E. Magin. "Determination of heat capacity of carbon composites with application to carbon/phenolic ablators up to high temperatures." Aerospace Science and Technology 108 (2021): 106375.

[23] Kuo, H.H., Lin, J.C. and Ju, C.P., 2005. Effect of carbonization rate on the properties of a PAN/phenolic-based carbon/carbon composite. Carbon, 43(2), pp.229-239.

[24] Asim, Mohd, Naheed Saba, Mohammad Jawaid, Mohammad Nasir, Mohammed Pervaiz, and Othman Y. Alothman. "A review on phenolic resin and its composites." Current Analytical Chemistry 14, no. 3 (2018): 185-197.

[25] Davenas, A. ed., 2012. Solid rocket propulsion technology. Newnes.

[26] Kamran, A., Guozhu, L. (2012). An integrated approach for optimization of solid rocket motor. Aerospace Science and Technology, 17(1), 50-64.

[27] Mishra, A. K., Jadhav, S., Akshay, M. (2022). Theoretical Aspects on Design and Performance Characteristics of solid rocket motor. International Journal of All Research Education and Scientific Methods, 10(2), 894-898.
[28] Wong, E. "Solid rocket nozzle design summary." In 4th Propulsion Joint Specialist Conference, p. 655. 1968.
[29] Jianding, Huang, and Ye Dingyou. "The development of space solid rocket motors in China." Acta astronautica 40.2-8 (1997): 607-612.
[30] Tao, Yang, Xun-bo Wu, Xiao-qian Chen, Yang Tao, Xun-bo Wu, and Xiao-qian Chen. "Design of a micro solid rocket motor." In 33rd Joint Propulsion Conference and Exhibit, p. 2864. 1997.
[31] Mukunda, H. S., Jain, V. K., Paul, P. J. (1979). A review of hybrid rockets: Present status and future potential. Proceedings of the Indian Academy of Sciences Section C: Engineering Sciences, 2(2), 215-242.
[32] Mukunda, H. S. Understanding aerospace chemical propulsion. IK International Publishing House Pvt. Limited, 2017.
[33] Jain, S.R., 2002. Solid propellant binders.
[34] Dombe, G., Jain, M., Singh, P. P., Radhakrishnan, K. K., Bhattacharya, B. (2008). Pressure casting of composite propellant.
[35] Ribereau, D., P. Le Breton, and S. Ballereau. "Casting process effect on composite solid propellant burning rate." In 37th Joint Propulsion Conference and Exhibit, p. 3946. 2001.
[36] Steinberger, R. and Drechsel, P.D., 1969. Manufacture of cast double-base propellant.
[37] Ramesh, Kurva, Shekhar N. Jawalkar, Swati Sachdeva, and Bikash Bhattacharya. "Development of a composite propellant formulation with a high performance index using a pressure casting technique." Central European Journal of Energetic Materials 9, no. 1 (2012): 49-58.
[38] Cho, In Hyun, and Seung Wook Baek. "Numerical analysis of ignition transient in an axisymmetric solid rocket motor equipped with rear ignition system." Combustion science and technology 152, no. 1 (2000): 81-98.
[39] Manuprasad E S, D. C. S. R., Development of Aluminized and Non-Aluminized composite solid propellant with Low pressure index, volume 86. International Conference on Mechanical and Aerospace Engineering, 2016.

[40] Apinhapt, P., and N. Pittayaprasertku. "Experimental investigation on pyrotechnic igniter for solid rocket motor." In The 5th International Conference on Chemical Engineering and Applications (IPCBEE). 2014.
[41] Willemse, B., Angelone, M., Drevet, O., Serraglia, F., Vita, G. (2007). An Overview of the Development of the VEGA Solid Rocket Motor Igniters. In 43rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference Exhibit (p. 5813).
[42] Kulkarni, A. K., Kumar, M., Kuo, K. K. (1982). Review of solid-propellant ignition studies. AIAA journal, 20(2), 243-244.
[43] Salama, Ahmed, and Hatem M. Belal. "Experimental Investigations of a Pyrotechnic Igniter." In AIAA SCITECH 2022 Forum, p. 1480. 2022.
[44] Ulas A, Risha GA, Kuo KK. An Investigation of the Performance of a Boron/Potassium-Nitrate Based Pyrotechnic Igniter. Propellants, Explosives, Pyrotechnics: An International Journal Dealing with Scientific and Technological Aspects of Energetic Materials. 2006 Aug;31(4):311-7.
[45] Rossi, C., Larangot, B., Lagrange, D., Chaalane, A. (2005). Final characterizations of MEMS-based pyrotechnical microthrusters. Sensors and Actuators A: Physical, 121(2), 508-514.
[46] ADAMS, DAVID M. "Igniter performance in solid-propellant rocket motors." Journal of Spacecraft and Rockets 4, no. 8 (1967): 1024-1029.
[47] Yilmaz, Alper, Omar Javed, and Mubarak Shah. "Object tracking: A survey." Acm computing surveys (CSUR) 38.4 (2006): 13-es.
[48] Challa, S., Morelande, M. R., Mušicki, D., Evans, R. J. (2011). Fundamentals of object tracking. Cambridge University Press.
[49] Ullah, K., Ahmed, I., Ahmad, M., Khan, I. (2019, March). Comparison of person tracking algorithms using overhead view implemented in OpenCV. In 2019 9th annual information technology, electromechanical engineering and microelectronics conference (IEMECON) (pp. 284-289). IEEE.
[50] Singh, Suryansh Pratap. "Comparing various tracking algorithms in opencv." Turkish Journal of Computer and Mathematics Education (TURCOMAT) 12, no. 6 (2021): 5193-5198.
[51] Mittal, Akshat, Suryansh Singh, and Manas Gupta. Comparing Various Tracking Algorithms in OpenCV. No. 5422. EasyChair, 2021.
[52] Li, S. A., Weng, C. W., Chen, Y. H., Lo, C. H., Yang, M. H., Lin, Y. C., ... Wong,

C. C. (2012, November). Servo motor controller design for robotic manipulator. In 2012 International Symposium on Intelligent Signal Processing and Communications Systems (pp. 254-257). IEEE.

[53] Micro Maestro 6-Channel USB Servo Controller `https://www.pololu.com/product/1350`

[54] NVIDIA JETSON TX2 `https://developer.nvidia.com/embedded/jetson-tx2`

[55] J120 carrier board `https://auvidea.eu/j120/`

[56] PIXHAWK MODULE `https://ardupilot.org/copter/docs/common-pixhawk-overview.html`

[57] Pocketbeagle SBC from Beaglebone `https://beagleboard.org/pocket`

[58] e-CAM132-TX2 `https://www.e-consystems.com/13mp-autofocus-nvidia-jetson-tx2-camera-board.asp`

[59] Monmasson, Eric, and Marcian N. Cirstea. "FPGA design methodology for industrial control systems—A review." IEEE transactions on industrial electronics 54, no. 4 (2007): 1824-1842.

[60] Birem, Merwan, and François Berry. "DreamCam: A modular FPGA-based smart camera architecture." Journal of Systems Architecture 60.6 (2014): 519-527.

[61] Shi, Y., Tsui, T. (2007, November). An FPGA-based smart camera for gesture recognition in HCI applications. In Asian conference on Computer vision (pp. 718-727). Springer, Berlin, Heidelberg.

[62] Dias, F., Berry, F., Sérot, J., Marmoiton, F. (2007, September). Hardware, design and implementation issues on a FPGA-based smart camera. In 2007 First ACM/IEEE International Conference on Distributed Smart Cameras (pp. 20-26). IEEE.

[63] Muscoloni, Alessandro, and Stefano Mattoccia. "Real-time tracking with an embedded 3D camera with FPGA processing." In 2014 international conference on 3d imaging (IC3D), pp. 1-7. IEEE, 2014.

[64] Köhler, Tim, F. Röchter, Jens Peter Lindemann, and Ralf Möller. "Bio-inspired motion detection in an FPGA-based smart camera module." Bioinspiration biomimetics 4, no. 1 (2009): 015008.

[65] Zhou, Weiguo, Yunhui Liu, Congyi Lyu, Weihua Zhou, Jianqing Peng, Ruijia Yang, and Haiyang Shang. "Real-time implementation of panoramic mosaic camera based on FPGA." In 2016 IEEE International Conference on Real-time Computing and

Robotics (RCAR), pp. 204-209. IEEE, 2016.

[66] Nelson, R. C. Flight Stability and Control Automatic Control.

[67] Benterki, Abdelmoudjib, Moussa Boukhnifer, Vincent Judalet, and Choubeila Maaoui. "Artificial intelligence for vehicle behavior anticipation: Hybrid approach based on maneuver classification and trajectory prediction." IEEE Access 8 (2020): 56992-57002.

[68] Payeur, Pierre, Hoang Le-Huy, and Clement M. Gosselin. "Trajectory prediction for moving objects using artificial neural networks." IEEE Transactions on Industrial Electronics 42, no. 2 (1995): 147-158.

[69] Murray, B. and Perera, L.P., 2020. A dual linear autoencoder approach for vessel trajectory prediction using historical AIS data. Ocean Engineering, 209, p.107478.

[70] Winston, P. H. (1992). Artificial intelligence. Addison-Wesley Longman Publishing Co., Inc.

[71] Mitchell, T. M., Mitchell, T. M. (1997). Machine learning (Vol. 1, No. 9). New York: McGraw-hill.

[72] Mahesh, B. (2020). Machine learning algorithms-a review. International Journal of Science and Research (IJSR).[Internet], 9, 381-386.

[73] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521, no. 7553 (2015): 436-444.

[74] Yan, L. C., Yoshua, B., Geoffrey, H. (2015). Deep learning. nature, 521(7553), 436-444.

[75] Samek, Wojciech, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. "Evaluating the visualization of what a deep neural network has learned." IEEE transactions on neural networks and learning systems 28, no. 11 (2016): 2660-2673.

[76] Canziani, A., Paszke, A., Culurciello, E. (2016). An analysis of deep neural network models for practical applications. arXiv preprint arXiv:1605.07678.

[77] Bau, D., Zhu, J. Y., Strobelt, H., Lapedriza, A., Zhou, B., Torralba, A. (2020). Understanding the role of individual units in a deep neural network. Proceedings of the National Academy of Sciences, 117(48), 30071-30078.

[78] Schiffmann, W., Joost, M., Werner, R. (1993, April). Comparison of optimized backpropagation algorithms. In ESANN (Vol. 93, pp. 97-104).

[79] Abbass, H. A. (2003). Speeding up backpropagation using multiobjective evolu-

tionary algorithms. Neural Computation, 15(11), 2705-2726.

[80] Zhang, J., Zheng, Y., Qi, D., Li, R., Yi, X. (2016, October). DNN-based prediction model for spatio-temporal data. In Proceedings of the 24th ACM SIGSPATIAL international conference on advances in geographic information systems (pp. 1-4).

[81] Orekondy, T., Schiele, B., Fritz, M. (2019). Prediction poisoning: Towards defenses against dnn model stealing attacks. arXiv preprint arXiv:1906.10908.

[82] Nakashima, Hayato, Ismail Arai, and Kazutoshi Fujikawa. "Passenger counter based on random forest regressor using drive recorder and sensors in buses." In 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 561-566. IEEE, 2019.

[83] Kim, S., Jeong, M., Ko, B. C. (2021). Self-supervised keypoint detection based on multi-layer random forest regressor. IEEE Access, 9, 40850-40859.

[84] Ding, J., Bar-Joseph, Z. (2017). MethRaFo: MeDIP-seq methylation estimate using a Random Forest Regressor. Bioinformatics, 33(21), 3477-3479.

[85] Jiang, D., Lin, W., Raghavan, N. (2021). A Gaussian mixture model clustering ensemble regressor for semiconductor manufacturing final test yield prediction. IEEE Access, 9, 22253-22263.

[86] Barrionuevo, Germán Omar, Sergio Ríos, Stewart W. Williams, and Jorge Andrés Ramos-Grez. "Comparative evaluation of machine learning regressors for the layer geometry prediction in wire arc additive manufacturing." In 2021 IEEE 12th International Conference on Mechanical and Intelligent Manufacturing Technologies (ICMIMT), pp. 186-190. IEEE, 2021.

[87] Weisberg, Sanford. Applied linear regression. Vol. 528. John Wiley Sons, 2005.

[88] Su, Xiaogang, Xin Yan, and Chih-Ling Tsai. "Linear regression." Wiley Interdisciplinary Reviews: Computational Statistics 4.3 (2012): 275-294.

[89] Yan, X., Su, X. (2009). Linear regression analysis: theory and computing. world scientific.

[90] Ranstam, J., and J. A. Cook. "LASSO regression." Journal of British Surgery 105.10 (2018): 1348-1348.

[91] Meier, Lukas, Sara Van De Geer, and Peter Bühlmann. "The group lasso for logistic regression." Journal of the Royal Statistical Society: Series B (Statistical Methodology) 70, no. 1 (2008): 53-71.

[92] Reid, Stephen, Robert Tibshirani, and Jerome Friedman. "A study of error variance

estimation in lasso regression." Statistica Sinica (2016): 35-67.

[93] Li, Fan, Yiming Yang, and Eric Xing. "From lasso regression to feature vector machine." Advances in neural information processing systems 18 (2005).

[94] Wright, Raymond E. "Logistic regression." (1995).

[95] Menard, Scott. Applied logistic regression analysis. No. 106. Sage, 2002.

[96] Kleinbaum, D. G., Dietz, K., Gail, M., Klein, M., Klein, M. (2002). Logistic regression (p. 536). New York: Springer-Verlag.

[97] LaValley, Michael P. "Logistic regression." Circulation 117.18 (2008): 2395-2399.