Trajectory Estimation for Dynamic Testing & Military Aircraft Detection Using Deep Learning- YOLOv7 & RT-DETR

Internship Report

Submitted to:

Terminal Ballistics Research Laboratory(TBRL)

Defence Research Development Organization(DRDO)

Under the guidance of

Mr. Subhash Chander(Sc 'F') and Mr. Rohit Khanna(Sc 'C')

RTRS, TBRL

Submitted by:

Charu (Roll No. 12116053)

Department of Mechanical Engineering

National Institute of Technology, Kurukshetra

Jan 2024 – June 2024

Candidate's Declaration

I, Charu, a student pursuing B.Tech. in Mechanical Engineering at the National Institute of Technology, Kurukshetra, hereby affirm that I take full responsibility for the accuracy and integrity of the information, results, and conclusions presented in this project report titled "Trajectory Estimation for Dynamic Testing and Military Aircraft Detection Using Deep Learning-YOLOv7 and RT-DETR." This report is being submitted to TBRL (DRDO), Sector-30, Chandigarh, in fulfillment of the requirements for the internship certificate in the Bachelor of Technology program during my 6^{th} semester.

I acknowledge the contributions of all individuals involved in my project work, both directly and indirectly. Furthermore, I understand that any violation of intellectual property rights or copyrights found in this report will be my sole responsibility.

Charu

Date:

Certificate

This is to certify that the project report titled “Trajectory Estimation for Dynamic Testing and Military Aircraft Detection Using Deep Learning - YOLOv7 and RT-DETR,” submitted by Charu, meets the requirements for the internship in the Bachelor of Technology in Mechanical Engineering program at the Terminal Ballistics Research Laboratory (TBRL), DRDO, Sector-30, Chandigarh.

Charu has demonstrated exceptional dedication, creativity, and technical proficiency in her work. Her project showcases a high level of originality and innovation, reflecting her commitment to excellence. The meticulous research, thorough analysis, and practical application of deep learning techniques in her project are commendable.

Rohit Khanna (Scientist 'C')

RTRS, TBRL

Subhash Chander(Scientist ‘F’)

RTRS TBRL

Acknowledgement

I wish to extend my sincere appreciation to Dr. M Raghavendra Rao, Director , TBRL, Chandigarh, for granting me the opportunity to undertake my internship at TBRL.

I am deeply indebted to Mr. A.C. Sharma (Scientist 'H', Group Director, RTRS), for his unwavering motivation and support throughout this endeavor. My heartfelt gratitude goes to Mr. Sandeep Malik (Scientist 'G', Joint Director, RTRS), my supervisor Mr. Subhash Chander (Scientist 'F') and my mentor, Mr. Rohit Khanna (Scientist 'C'), for their unwavering support, invaluable guidance, and timely assistance throughout my internship. Despite their demanding schedules and departmental commitments, they generously dedicated their time to provide me with direction and knowledge, ensuring my progress on the right trajectory.

Additionally, I am thankful to the staff of the RTRS department at TBRL for their assistance.

Furthermore, I express my profound gratitude to Professor Satnam Singh of the Mechanical Engineering Department at the National Institute of Technology, Kurukshetra, for his significant contributions to the completion of my project titled "Trajectory Estimation for Dynamic Testing and Military Aircraft Detection using Deep Learning - YOLOv7 & RT-DETR."

Lastly, I would like to affirm that this project was completed solely by myself.

Charu

Table of contents

List of figures

List of Graphs

Abstract

This project comprises two critical components aimed at enhancing aerospace and defence capabilities: the development of a user-friendly Graphical User Interface (GUI) using MATLAB for trajectory estimation in dynamic testing and the implementation of state-of-the-art deep learning models, namely YOLOv7 and RT-DETR, for military aircraft detection.

In the initial phase, a MATLAB-based GUI is meticulously crafted to streamline trajectory estimation tasks, offering users intuitive tools to input data, adjust parameters, visualize trajectories, and analyze results comprehensively. This GUI not only simplifies complex tasks but also enhances efficiency and usability in dynamic testing scenarios, thereby advancing research and development activities in aerospace and defence sectors.

The subsequent phase focuses on harnessing advanced deep learning techniques for military aircraft detection. By deploying YOLOv7 and RT-DETR, the project aims to detect, track, and recognize aircraft in real-time aerial surveillance footage. This endeavor contributes significantly to improving situational awareness and defence capabilities, thereby bolstering national security.

Both phases of the project undergo rigorous testing and validation procedures to ensure accuracy, reliability, and performance. The outcomes of this project hold immense potential for defence applications, offering innovative solutions to trajectory estimation challenges and significantly enhancing military aircraft detection capabilities. These advancements not only strengthen defence infrastructure but also pave the way for more secure and resilient national defence systems, aligning with the overarching objectives of defence research and development.

Chapter-1

Defence Research and Development Organisation(DRDO)

1. Introduction:

The Defence Research and Development Organisation (DRDO) is the premier agency under the Department of Defence Research and Development within the Ministry of Defence, Government of India. Headquartered in Delhi, DRDO is responsible for military research and development. Established in 1958 through the merger of the Technical Development Establishment and the Directorate of Technical Development and Production of the Indian Ordnance Factories with the Defence Science Organisation, DRDO has since become a key player in India's defence sector. In 1979, the Defence Research & Development Service (DRDS) was formed, comprising Group 'A' Officers Scientists, operating directly under the administrative control of the Ministry of Defence.

Figure 5.1 Defence Research and Development Organization logo[1]

DRDO is the R&D wing of the Ministry of Defence, Government of India, with a vision to empower India with cutting-edge defence technologies and a mission to achieve self-reliance in critical defence technologies and systems. DRDO aims to equip the armed forces with state-of-the-art weapon systems and equipment in line with the requirements set by the three Services. The organization's pursuit of self-reliance has led to the successful indigenous development and production of strategic systems and platforms such as the Agni and Prithvi series of missiles, the Tejas light combat aircraft, the Pinaka multi-barrel rocket launcher, the Akash air defence system, and a wide range of radars and electronic warfare systems. These advancements have significantly enhanced India's military capabilities, creating effective deterrence and providing crucial leverage.

2. History:

DRDO was established in 1958 through the amalgamation of the already functioning Technical Development Establishment (TDEs) of the Indian Army, the Directorate of Technical Development & Production (DTDP), and the Defence Science Organization (DSO). Initially a small organization with 10 establishments or laboratories, DRDO has expanded significantly over the years, growing in terms of subject disciplines, number of laboratories, achievements, and stature.

In the 1960s, DRDO embarked on its first major project in surface-to-air missiles (SAM) known as Project Indigo. Although Indigo was discontinued without achieving full success, it paved the way for Project Devil and Project Valiant in the 1970s, aimed at developing short-range SAMs and ICBMs. Project Devil eventually led to the development of the Prithvi missile under the

Integrated Guided Missile Development Programme (IGMDP) in the 1980s. The IGMDP, an Indian Ministry of Defence programme running from the early 1980s to 2007, aimed to develop a comprehensive range of missiles, including the Agni, Prithvi, Akash, Trishul, and Nag missiles.

Today, DRDO is a network of over 50 laboratories engaged in developing defence technologies across various disciplines such as aeronautics, armaments, electronics, combat vehicles, engineering systems, instrumentation, missiles, advanced computing and simulation, special materials, naval systems, life sciences, training, information systems, and agriculture. Numerous major projects are underway for the development of missiles, armaments, light combat aircraft, radars, and electronic warfare systems, with significant achievements already made in several of these areas.

3. Organizational Structure:

The Defence Research and Development Organisation (DRDO) operates through a network of laboratories strategically dispersed across the country, each dedicated to specific domains within defence research and development. These laboratories are grouped into clusters based on thematic areas such as aeronautics, armaments, electronics, and life sciences. Under the leadership of eminent scientists and technologists, these clusters play a crucial role in steering the strategic direction of the organization and coordinating research activities. This organizational structure facilitates collaboration and synergy among laboratories, enabling the pooling of expertise and resources to tackle complex defence challenges effectively. Through this collaborative framework, DRDO endeavours to drive innovation, advance technological capabilities, and contribute significantly to India's defence preparedness.

4. Vision:

Empowering the nation with state-of-the-art indigenous defence and security technologies and systems. DRDO is firmly determined to make the nation strong and self-reliant in science and technology, particularly in military technologies.

5. Mission:

- **Design & Development:** Create state-of-the-art sensors, weapon systems, platforms, and allied equipment in defence and security domains, including land, air, sea, space, and cyber.
- **Production & Induction:** Facilitate the production and induction of systems and technologies developed through the Department's R&D ecosystem.
- **Technological Solutions:** Provide technological solutions to the Services to enhance combat effectiveness.
- **Collaboration & Strengthening:** Nurture and strengthen defence R&D capability in Indian industry, S&T institutions, and academia through collaboration.
- **Infrastructure & Development:** Develop infrastructure and test & evaluation facilities, design certification, skill development, and strengthen human resources.

6. Significant Contributions:

- **Strategic Missile Systems:**

DRDO's development of strategic missile systems such as Agni, Prithvi, and BrahMos has significantly bolstered India's deterrence capabilities, enhancing national security and readiness to counter potential threats.

- **Indigenous Fighter Aircraft and Combat Vehicles:**

The successful design and production of indigenous fighter aircraft like Tejas and advanced combat vehicles such as the Arjun Main Battle Tank exemplify DRDO's strides towards self-reliance in defence production, reducing dependency on foreign imports and bolstering India's defence preparedness.

- **Advancements in Electronic Warfare and Communication Technologies:**

DRDO's pioneering advancements in electronic warfare, radar systems, and communication technologies ensure that the Indian armed forces maintain superiority in modern warfare scenarios, enabling effective response to emerging threats and challenges.

- **Breakthroughs in Materials Science:**

DRDO's research in materials science has yielded breakthroughs such as lightweight armour and high-performance alloys, significantly enhancing the protection and survivability of military personnel and equipment in diverse operational environments, contributing to overall defence readiness.

- **Research in Life Sciences:**

DRDO's focus on life sciences has resulted in the development of biomedical devices and protective gear, prioritizing the health and well-being of military personnel. These advancements underscore DRDO's commitment to ensuring the safety and effectiveness of India's armed forces, further strengthening the nation's defence infrastructure and national security.[1]

Chapter-2

Terminal Ballistics Research Laboratory(TBRL)

1. Overview:

The Terminal Ballistics Research Laboratory (TBRL) is a premier Defence Research and Development Organization (DRDO) laboratory located in Chandigarh, India. It plays a crucial role in developing, producing, and evaluating high explosive compositions, warheads, and various armament systems. The laboratory conducts extensive research in ballistics, including blast, lethality, fragmentation studies, and captive flight testing of bombs, missiles, and airborne systems. It also evaluates protective systems like body armor, vehicle armor, and helmets against small arms ammunition.

Figure 2.1 TBRL,DRDO[2]

2. Certifications:

TBRL is certified under the International Quality Management Systems Standard ISO 9001:2000 by the Standardization Testing and Quality Certification Services (STQC), Department of Information Technology (DIT), Government of India. The certification was upgraded to ISO 9001:2008 in 2014.

3. Historical Background:

Established in 1961, TBRL was formed to address the need for indigenous capabilities in designing, developing, and evaluating warheads and weapon systems following World War II. The laboratory became fully operational in 1967 and was officially inaugurated in January 1968 by the then Defence Minister. The main laboratory is in Chandigarh, with a firing range spanning 5000 acres located in Ramgarh, Haryana, 22 km from Chandigarh.

4. Vision:

TBRL envisions achieving self-reliance in developing warhead technologies and providing state-of-the-art diagnostic facilities to assess the terminal effects of armament systems. The laboratory aims to be a leader in technology growth and evaluation capabilities in the field of armaments.

5. Areas of Work:

- **Armor Defeating Projectiles & Immunity Profiles:** Performance studies and development of technologies to defeat armored targets.

- **Blast Damage & Fragmentation Studies:** Analysis of ground shock, blast effects, fragmentation, and lethality of various warheads and ammunition.
- **Safety Templates & Pressure Wave Propagation:** Creation of safety protocols for weapons and studies on underwater detonics and pressure waves.
- **Explosive Technologies:** Development of explosive forming, cladding, and welding techniques, along with high explosive compositions.
- **Impact & Penetration Studies:** Characterization of materials under high strain rates and studies on impact and penetration.
- **High Energy Electrical Pulse Power:** Technology for generating high energy electrical pulses through explosive-driven magnetic flux compression.
- **Captive Flight Testing:** Testing of bombs, missiles, and airborne systems for performance evaluation.
- **Ballistics Evaluation:** Testing of body armor, vehicle armor, and helmets against small arms ammunition.
- **Design & Development:** Development of baffle ranges, warheads, exploders for torpedoes, bund blasting devices, grenades, and non-lethal ammunition.

6. Mission:

- **Technology Development:** Focus on developing warhead-related technologies and products.
- **Diagnostics Facilities:** Provide advanced diagnostic facilities for armament systems testing and evaluation.
- **Infrastructure & Manpower:** Establish a robust technology base in armaments with necessary infrastructure and skilled manpower.

7. Key Technologies:

- **High Energy Electrical Pulse Power:** Generation through explosive-driven magnetic flux compression.
- **Dynamic Shock Compression:** Detonation wave shaping and dynamic shock compression of materials.
- **Shaped Charges & EFP Technologies:** For defeating armored and naval targets.
- **Baffle Ranges:** For small arms firing practice.
- **Small Arms Ammunition:** Development for low-intensity conflict scenarios.
- **Electronic Fuzzes:** Enabling technologies for precision electronic fuses.[2]

Chapter-3

Rail Track Rocket Sled(RTRS)

National Test Facility

1. Introduction:

The Rail Track Rocket Sled (RTRS) facility embodies India's commitment to technological advancement in defence research and development. Established in 1988 under the aegis of the Defence Research and Development Organization (DRDO), RTRS represents a pivotal milestone in India's quest for cutting-edge testing capabilities in the aerospace and defence sectors.

Figure 3.1 Penta Rail, RTRS[3]

RTRS serves as a high-speed testing facility aimed at assessing the performance and dependability of aerospace and defence systems. Employing rocket propulsion mounted on sleds traversing rail tracks, RTRS accelerates test articles to extraordinary velocities. This capability facilitates impact testing of various defence assets including missiles, warheads, aircraft components, and other weapon systems, thereby driving advancements in aerospace and defence technology.

2. Historical Background and Establishment:

The genesis of RTRS can be traced back to India's burgeoning defence research initiatives, culminating in its establishment in 1988 within the Terminal Ballistic Research Lab (TBRL) in Haryana. Motivated by the need for dedicated testing infrastructure to support the development and evaluation of critical defence systems, RTRS was envisioned as a cornerstone of India's defence research and development endeavors.

3. Capabilities:

Equipped with a supersonic penta-rail boasting five rail lines, the RTRS ensures precise alignment and robustness to withstand high loads. Featuring a 4-kilometer track length with multiple lines of wider gauges and continuous welded tracks, it accommodates diverse payloads. Additionally, the RTRS possesses the capability to develop rockets, aerodynamic sleds, and advanced instrumentation systems tailored to specific testing requirements. Trials conducted at the RTRS cover both recovery and non-recovery scenarios, including impact trials for payloads

of varying masses and dimensions. With over 1000 trials to date, achieving velocities from low subsonic to Mach 2 in non-recovery trials, the facility employs thorough data processing through onboard and ground instrumentation modes. Photo-instrumentation plays a critical role in capturing crucial events for comprehensive data analysis. The RTRS finds extensive application in proximity fuse testing, ballistics studies, missile system performance evaluation, and testing of various armament and navigation systems. Its advanced infrastructure and tailored approach position the RTRS as a vital asset in India's defence research and development landscape.

Figure 3.2 Rail track rocket sled[3]

4. Key Specifications:

- **Supersonic Penta-Rail:** RTRS facility features a supersonic penta-rail supporting five rail lines, ensuring robustness and precision in testing operations.
- **Precision Alignment:** The track is meticulously aligned to guarantee accurate testing conditions and withstand high loads, enhancing the reliability and effectiveness of testing.
- **Track Length Extension:** Over the years, the track length has been extended to 4 kilometers, providing ample space for testing operations and accommodating various payloads effectively.
- **Multiple Gauges:** The facility offers multiple lines of wider gauges on the track, enabling the accommodation of diverse payloads and enhancing its versatility in testing capabilities.

5. Vision:

To excel as a global leader in high-speed testing technologies, driving innovation and excellence in aerospace and defence research and development.

6. Mission:

To continually innovate and refine high-speed sled track testing techniques, providing precise and versatile testing capabilities to support the efficient design, testing, and evaluation of aerospace and defence systems.[3]

7. Achievements:

- **Advanced Testing Methodologies:**

The Rail Track Rocket Sled (RTRS) facility has pioneered advanced testing methodologies, ensuring rigorous evaluation of aerospace and defence systems with precision and reliability.

- **Extensive Trial Experience:**

With a track record of conducting over a thousand trials, encompassing both recovery and non-recovery scenarios, the facility demonstrates unparalleled expertise and experience in high-speed testing.

- **Versatile Testing Services:**

RTRS offers a comprehensive range of testing services, including fuse testing, ballistics studies, evaluation of missile systems' performance, and assessment of armament and navigation systems, catering to diverse defence research needs.

- **Impressive Velocities:**

The facility has achieved remarkable velocities, including Mach 2, showcasing its capability to propel test articles at extreme speeds and validate the performance of defence systems under high-speed conditions.

- **Significant Contribution to Defence R&D:**

Through its relentless pursuit of excellence, the RTRS facility has significantly contributed to defence research and development, enhancing India's defence capabilities and furthering national security objectives.

Figure 3.3 Gaganyaan-RTRS Trial[4]

Chapter-4

Project-1

Trajectory Estimation for Dynamic Testing

1. Introduction:

1.1. Overview:

Rail track rocket sled test facility, where high-speed tests are conducted, demands precise trajectory estimation for ensuring safety and performance. In this facility, trajectory estimation is essential for analyzing the behavior of objects, such as projectiles, during high-speed maneuvers. This includes assessing factors like stability, track alignment, and aerodynamic forces to prevent accidents and optimize performance. Software tools like MATLAB, Python, and Simulink play a crucial role in trajectory estimation, enabling accurate modeling, simulation, and analysis of complex systems in high-speed dynamic testing scenarios.

1.2. Objective:

The project aims to develop a user-friendly MATLAB application for estimating trajectories during dynamic testing. It will utilize kinematic equations to ensure accurate trajectory estimations based on user-provided initial conditions. The application will include features for visualizing trajectories and effective input validation for a smooth user experience.

1.3. Why MATLAB?

MATLAB was chosen for this project due to several compelling reasons that align with the goals and requirements of trajectory estimation in dynamic testing:

1. Comprehensive Mathematical Function:

MATLAB offers an extensive library of built-in functions for mathematical computations, making it ideal for implementing the kinematic equations required for trajectory estimation. Functions for numerical integration, differentiation, and solving equations are readily available.

Figure 4.1 MTLAB Logo[5]

2. Advanced Visualization Tools:

MATLAB excels in data visualization, providing robust tools for creating high-quality plots and graphs. This is crucial for dynamically displaying the trajectory of an object, allowing for real-time updates and interactive visualization.

3. User-Friendly Environment:

MATLAB's App Designer provides a graphical environment for creating professional-looking GUIs. It allows for the easy integration of graphical components such as buttons, sliders, and plots, streamlining the development process.

4. Ease of Prototyping and Development:

MATLAB's high-level language and interactive environment facilitate rapid prototyping and development. The ability to test and visualize results immediately helps in quickly iterating and refining the app.

5. Extensive Documentation and Community Support:

MATLAB comes with comprehensive documentation, tutorials, and examples, making it easier to learn and implement new features. Additionally, there is a large and active user community that can provide support and share solutions.

6. Integration Capabilities:

MATLAB can easily integrate with other software and hardware, which is beneficial for expanding the app's capabilities in the future. For instance, integrating sensor data or interfacing with hardware devices used in dynamic testing can be achieved seamlessly.

7. Robust Performance:

MATLAB is optimized for performance in numerical computation and large data processing, ensuring that the trajectory calculations are performed efficiently and accurately.

By leveraging MATLAB's strengths, this project ensures accurate and reliable trajectory estimation, combined with a user-friendly interface for interactive and practical use in dynamic testing scenarios.

2. Background and theory:

2.1. Trajectory Estimation:

Trajectory estimation involves predicting the path of an object based on its initial conditions and the forces acting on it. This process commonly utilizes equations of motion derived from Newton's laws of physics.

2.2. Significance:

Accurate trajectory estimation is paramount within specialized environments such as rail track rocket sled (RTRS) experiments. In RTRS experiments, it is essential for predicting an object's

path, which is crucial for ensuring precise design, conducting thorough safety assessments, and evaluating performance in high-speed dynamic testing scenarios.

3. Methodology:

3.1. Tools and software:

- **MATLAB:** An advanced computational environment renowned for its capabilities in numerical computation, data visualization, and programming.
- **App Designer:** A specialized feature within MATLAB tailored for developing sophisticated applications, offering an intuitive drag-and-drop interface for creating professional-grade user interfaces.[6]

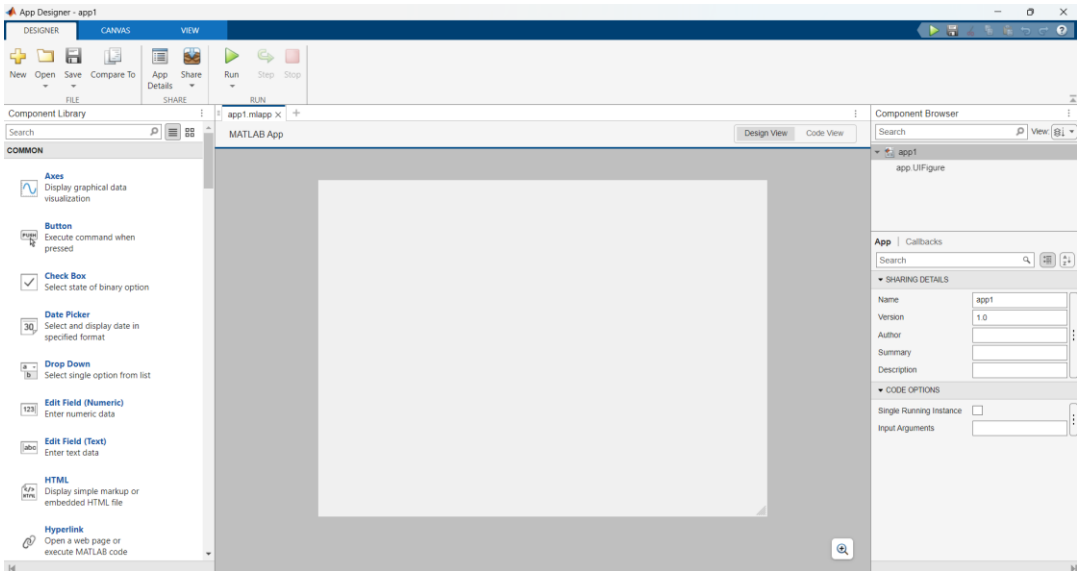


Figure 4.2 MATLAB App designer

3.2. Development Workflow:

- **User Interface Design:** Utilizing the intuitive interface of App Designer to craft a layout comprising essential components such as input fields, labels, buttons, and axes, ensuring a seamless user experience.
- **Callback Function Implementation:** Writing MATLAB code to manage user interactions effectively, process input data, and execute trajectory calculations accurately, thereby ensuring the application's functional integrity.

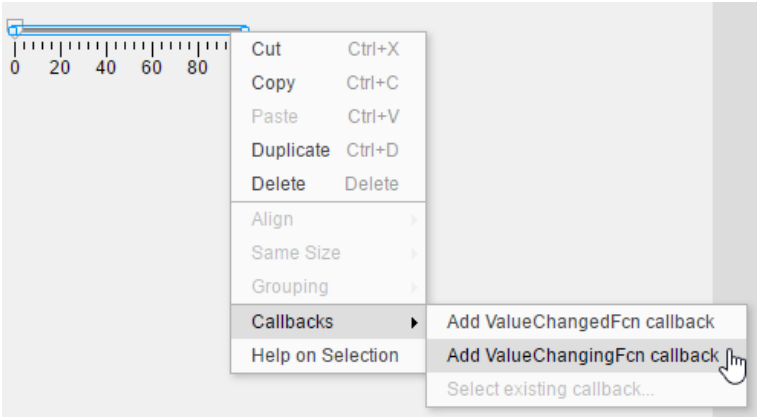


Figure 4.3 Adding callback function[7]

- **Testing and Debugging: Thoroughly evaluating the application's performance across diverse input scenarios to validate its functionality and ensure precise trajectory estimation, thereby guaranteeing reliable and accurate results.**

4. Implementation:

4.1. Theoretical Framework and computational model:

4.1.1. Theoretical Underpinnings:

The fundamental principles of classical mechanics, notably Newtonian dynamics, serve as the bedrock for the trajectory estimation project. These foundational concepts guide our development of mathematical models and computational algorithms, ensuring accurate prediction of object trajectories in rail track rocket sled (RTRS) experiments.

4.1.2. Mathematical Formulations:

In impact testing, where a rocket propels a payload to gain speed before impact, understanding the forces and motion involved is crucial. This requires a careful study of Newtonian mechanics and how different forces interact within the system.

Newton's Second Law: Newton's second law is the cornerstone of our analysis. It states that when a force is applied to an object, it causes the object to accelerate. Mathematically, this is expressed as:

$$F_{net} = ma$$

Here

F_{net} : denotes the resultant force acting on the object

m: signifies its mass

a: represents the ensuing acceleration

Forces in Impact Testing:

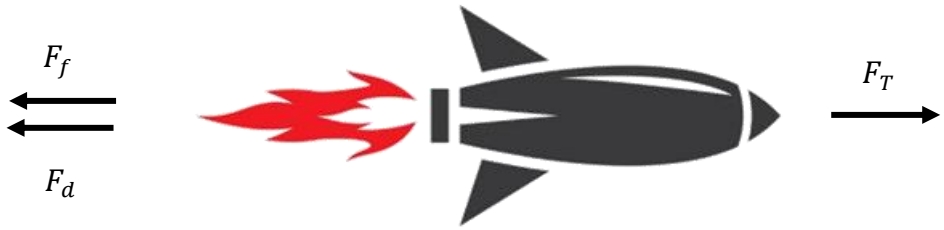


Figure 4.4 Diagrammatic force representation

In impact testing scenarios involving a rocket-driven body, several forces play a significant role:

a) Thrust F_T:

Rocket thrust is the force that propels a rocket forward. It is crucial in impact testing, accelerating the rocket-driven body towards its target. Generated by expelling high-speed exhaust gases from the engine, thrust contributes positively to the net force, pushing the body forward along the positive x-axis. Efficient thrust is essential for achieving desired velocities and trajectories in impact testing.

b) Drag F_d:

As the body moves through the surrounding air, it encounters resistance known as drag. The magnitude of the drag force depends on factors such as fluid density ρ, reference area A, drag coefficient C_d, and velocity v, as governed by the equation:

$$F_d = \frac{1}{2} C_d \rho A v^2$$

c) Friction F_f:

Frictional forces arise from contact between the body and its surroundings, manifesting as surface friction between the body and the ground. The frictional force is determined by the equation:

$$F_f = \mu N$$

where μ is the coefficient of friction, and N is the normal force exerted on the body.

- **Net Force and Acceleration:**

The resultant force F_{net} acting on the rocket-propelled body is the vector sum of the thrust, drag, and frictional forces, expressed as:

$$F_{net} = F_T - F_d - F_f$$

The resulting acceleration a of the body is determined by Newton's second law, which combines the net force and the body's mass.

The total mass m is given by:

$$m = m_c + m_p + m_r$$

where m_c is the mass of the clamps, m_p is the payload's mass and m_r is the mass of the rocket(s) mounted.

- **Acceleration a Equation:**

Acceleration is determined by Newton's second law as:

$$a = \frac{F_{net}}{m}$$

- **Velocity v Equation:**

The velocity of the rocket-propelled body is calculated using the equation of motion:

$$v = u + a(\Delta t)$$

where v is the final velocity, u is the initial velocity, a is the acceleration, and Δt is the elapsed time.

- **Position x Equation:**

To determine the body's position over time, the kinematic equation is utilized:

$$x = u(\Delta t) + \frac{1}{2} a (\Delta t)^2$$

This equation describes the displacement of the body from its initial position due to its initial velocity and the acceleration experienced over time. By integrating the acceleration equation or using alternative kinematic equations, expressions for the body's position at various time intervals during its motion can be derived.

4.2. User Interface Design:

The user interface (UI) design is crucial for creating an intuitive trajectory estimation tool. Using MATLAB's App Designer, the layout includes input fields, buttons, and visualizations for seamless interaction. The design prioritizes usability for easy data input and clear result presentation. The application's intuitive and user-friendly interface enhances efficient analysis of dynamic testing data.

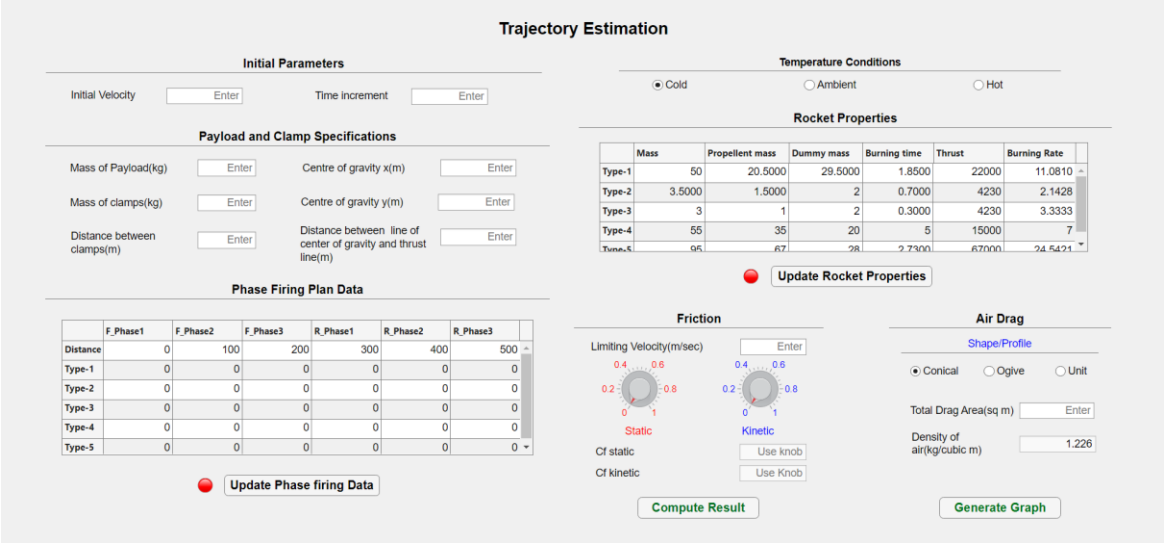


Figure 4.5 GUI for trajectory estimation

4.3. Logic flow in the app:

The app's logic workflow is designed to streamline the trajectory estimation process. It follows these key steps:

1. User Input:

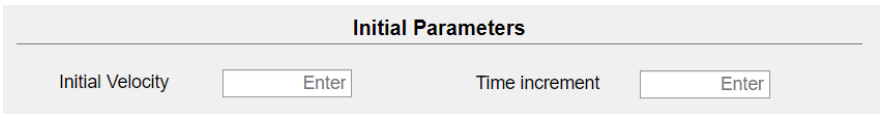


Figure 4.6 Initial Parameters

User input is crucial for accurate trajectory estimation. Users provide initial parameters such as initial velocity and time increment. The initial velocity sets the object's starting speed, while the time increment determines the calculation intervals. These inputs allow the app to simulate and visualize the object's trajectory accurately, ensuring precise analysis in dynamic testing scenarios.

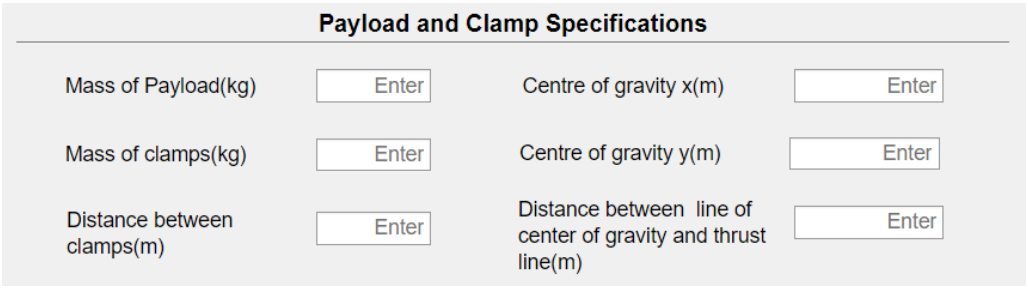
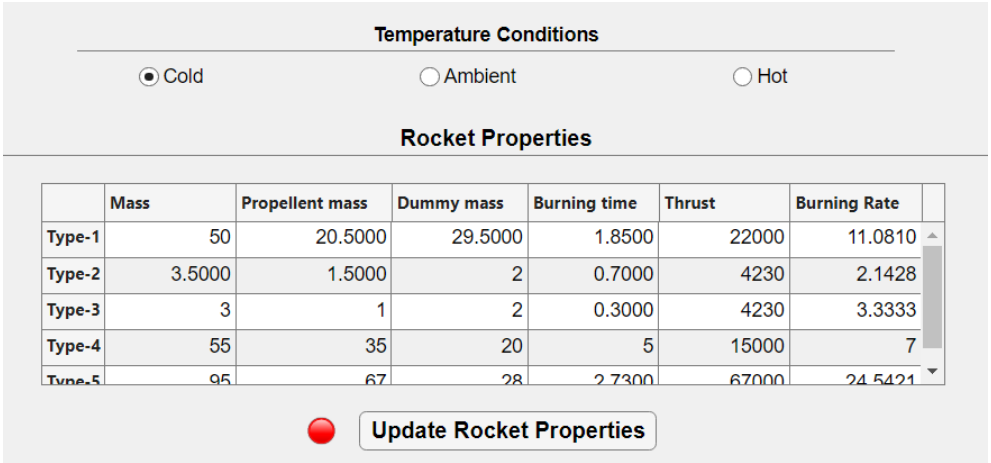


Figure 4.7 Payload and clamp specifications

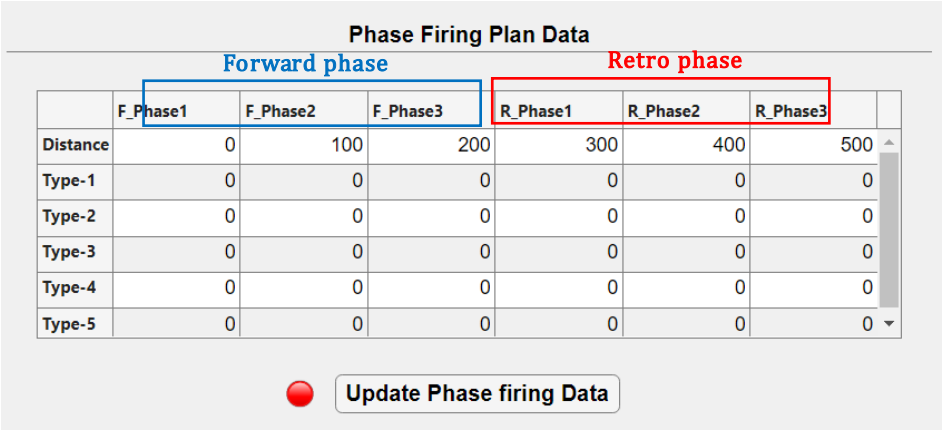
The payload and clamp specifications are vital for accurate trajectory calculations. Users input the mass of the payload and clamp, the coordinates of the centre of gravity (CG), the distance between clamps, and the distance between the CG line and the thrust line. These parameters help determine the object's stability and motion.



	Mass	Propellent mass	Dummy mass	Burning time	Thrust	Burning Rate
Type-1	50	20.5000	29.5000	1.8500	22000	11.0810
Type-2	3.5000	1.5000	2	0.7000	4230	2.1428
Type-3	3	1	2	0.3000	4230	3.3333
Type-4	55	35	20	5	15000	7
Type-5	95	67	28	2.7300	67000	24.5421

Figure 4.8 Rocket Details

Environmental conditions significantly impact the rocket's thrust profile. Users must select the temperature conditions—hot, cold, or ambient—as these affect the rocket's performance. Additionally, users can modify and update rocket properties. When the update button is pushed, an adjacent lamp changes colour from red to green, indicating that the data has been successfully updated. Ensure the lamp is green before computing to confirm that the input data is current and accurate.



	F_Phase1	F_Phase2	F_Phase3	R_Phase1	R_Phase2	R_Phase3
Distance	0	100	200	300	400	500
Type-1	0	0	0	0	0	0
Type-2	0	0	0	0	0	0
Type-3	0	0	0	0	0	0
Type-4	0	0	0	0	0	0
Type-5	0	0	0	0	0	0

Figure 6.9 Phase firing plan

In the phase firing data section, users can adjust the distance at which specific types of rockets are fired and modify the number of rockets used. After making these changes, users must push the update button. The adjacent lamp will change from red to green, indicating that the data has been successfully updated. Ensure the lamp is green before proceeding to confirm that the phase firing data is current and accurate.

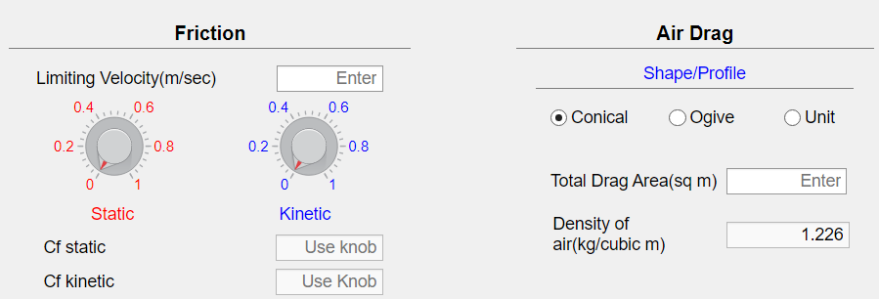


Figure 4.10 Friction and Drag

In the friction section, users must enter the limiting velocity at which the coefficient of friction changes from static to kinetic. Using a knob, users can adjust the static C_s and kinetic C_k friction values.

In the drag section, users select the shape of the object (conical, ogive, or unit, where $C_d = 1$ and enter the frontal area. The air density is set to 1.226 by default.[8]

2. Data Validation:

The app checks the input data for errors or inconsistencies to ensure accurate calculations.

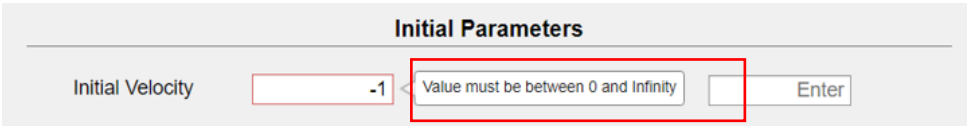


Figure 4.11 Warning (for out of range)

The GUI includes data validation that warns users if they enter invalid values, ensuring all inputs are within the allowed range for accurate calculations.

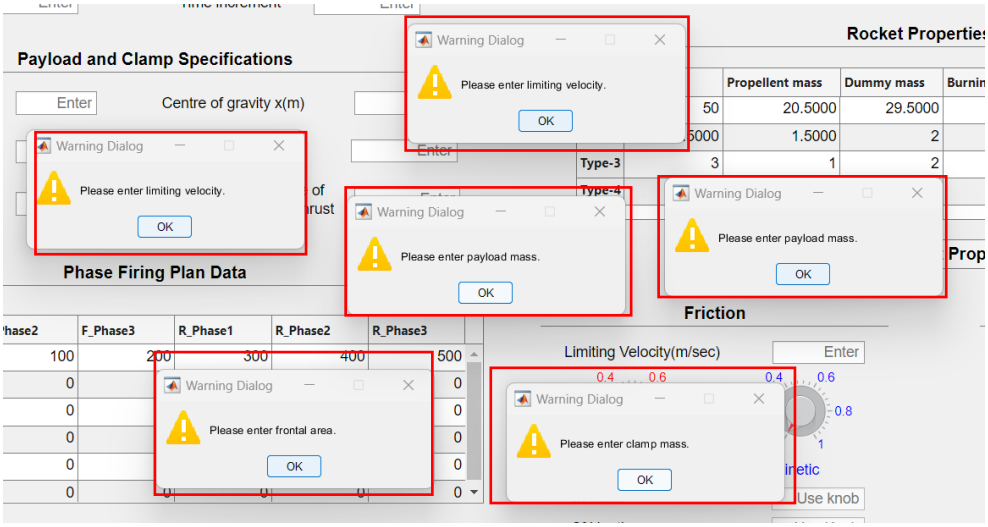


Figure 4.12 Warning (if empty)

The GUI also includes a warning dialog box that alerts users if any necessary fields are left empty. This ensures that all required information is provided before proceeding with the trajectory estimation process.

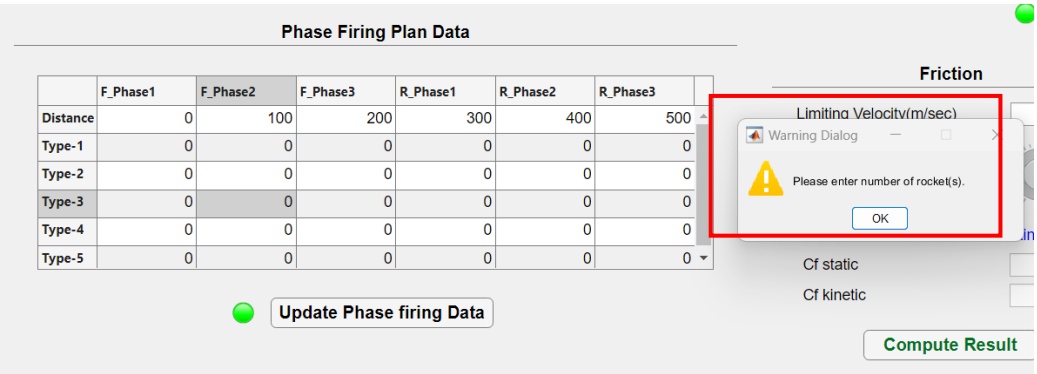


Figure 4.13 Warning (zero rockets)

If the user enters zero as the number of rockets, a warning message will appear, prompting the user to enter a valid value. This ensures that the calculation can proceed accurately with the necessary parameters.

3. Calculation Execution:

Upon validation, the app uses the provided inputs to perform trajectory calculations based on Newtonian mechanics and relevant equations of motion.

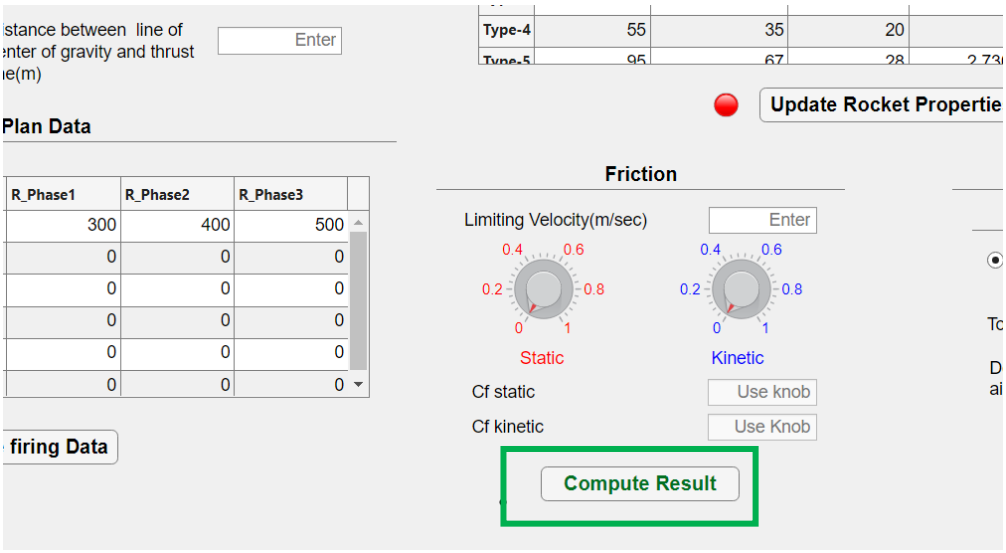


Figure 4.14 Compute Result Button

Upon clicking the "Compute Results" button, the calculation is executed, and an output window opens using the concept of a multi-window app in App Designer. This separate window displays the trajectory estimation results, allowing users to analyze and interpret the data conveniently.

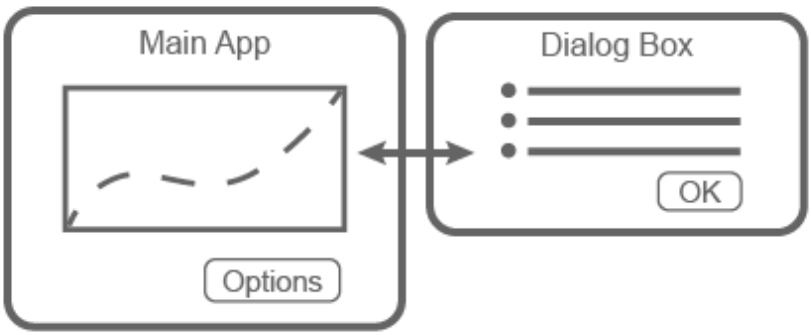


Figure 4.15 Multiwindow Concept[9]

4. Visualization:

The results of the calculations, including trajectory paths and key metrics, are graphically displayed on the axes within the app.

The application includes robust data visualization capabilities for enhanced analysis and interpretation of the trajectory estimation results. Users can generate graphs by pressing the "Generate Graph" button. These visual representations facilitate a comprehensive understanding of the dynamic behavior of the payload during its flight, enabling effective evaluation of performance metrics and identification of any anomalies in the trajectory.

The visual data aids in making informed decisions by providing clear and intuitive insights into the complex motion characteristics of the rocket-driven payload.

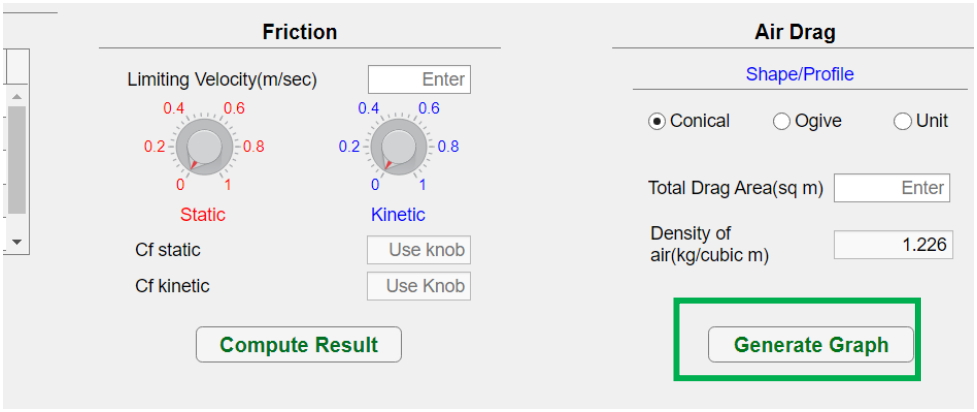


Figure 4.16 Generate graph button

The "Generate Graph" button plots several key graphs for visualization:

1. Velocity vs. Distance
2. Velocity vs. Time
3. Thrust vs. Time

Each graph provides valuable insights into the dynamics of the object's motion during the trajectory estimation process, enhancing the user's understanding of the data.

5. Output Presentation:

The connected 'Output' app displays all relevant results for comprehensive outcomes presentation. Users can conveniently view trajectory estimation results, including velocity, acceleration, thrust, and other pertinent data, facilitating a thorough analysis of the object's motion dynamics.

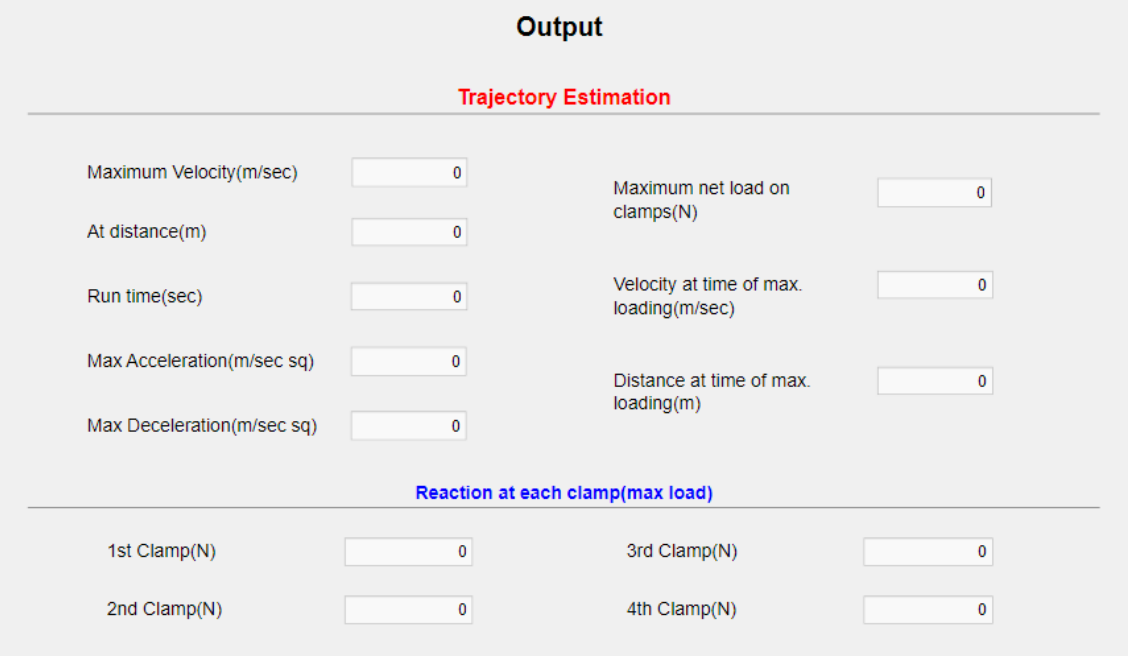


Figure 4.17 Output App

6. User Interaction:

The 'Output' app offers an interactive interface where users can modify input parameters and initiate new calculations, allowing for the exploration of various scenarios and their impacts on the trajectory.

Users can dynamically alter initial conditions, payload specifications, friction and drag properties, and rocket firing data to examine how these changes influence the object's trajectory and key performance metrics such as maximum velocity, maximum acceleration, and overall runtime. This iterative process enables a thorough evaluation and optimization of the trajectory under different testing conditions.

The workflow is designed to be intuitive, guiding users through a logical sequence of adjustments and re-calculations, thereby ensuring an efficient and seamless experience for trajectory estimation in dynamic testing environments.

5. Results:

5.1. Sample input and outputs:

The results display showcases sample input parameters alongside their corresponding output from the trajectory estimation process.
This allows users to compare their input data with the calculated results, providing insight into the accuracy and reliability of the estimation method.

5.1.1. Single-phase:

This results presentation illustrates sample input parameters and their corresponding outputs from the trajectory estimation process, focusing on a single phase of rocket firing simulation.

5.1.1.1. Input Parameters:

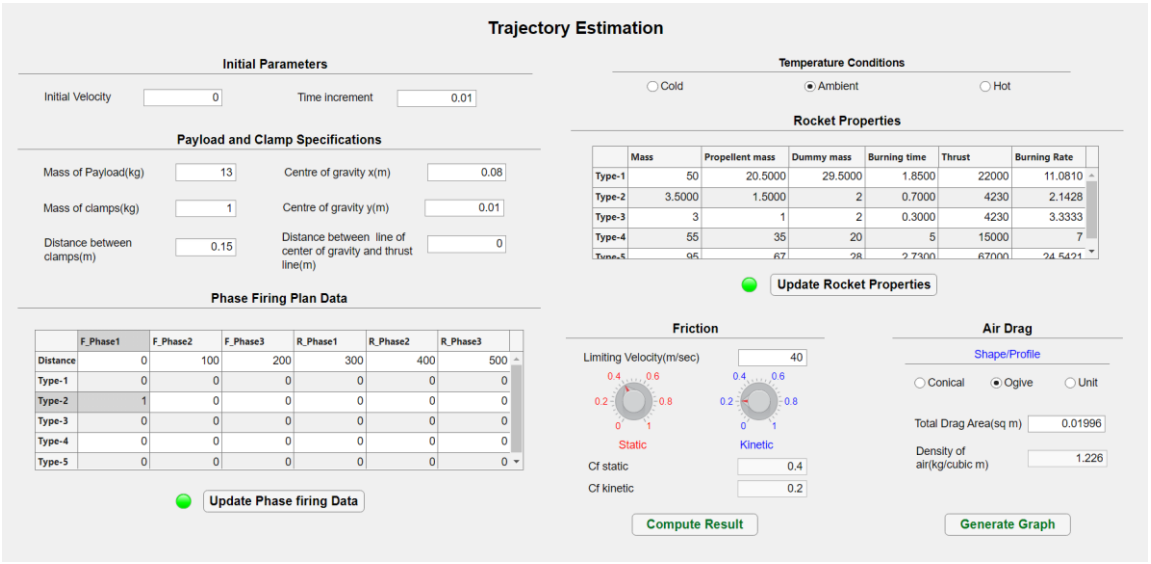


Figure 4.18 Sample input values

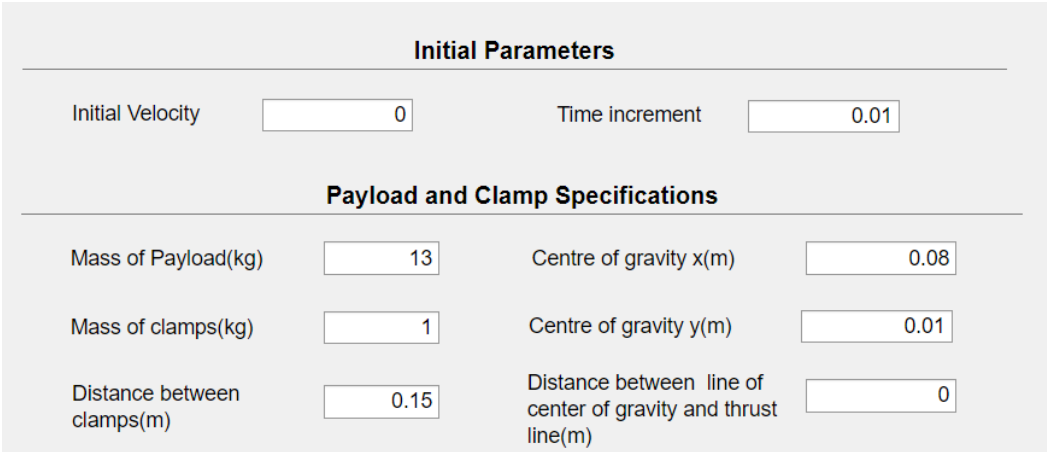


Figure 4.19 Zoom-in initial parameters & Payload and clamp specifications

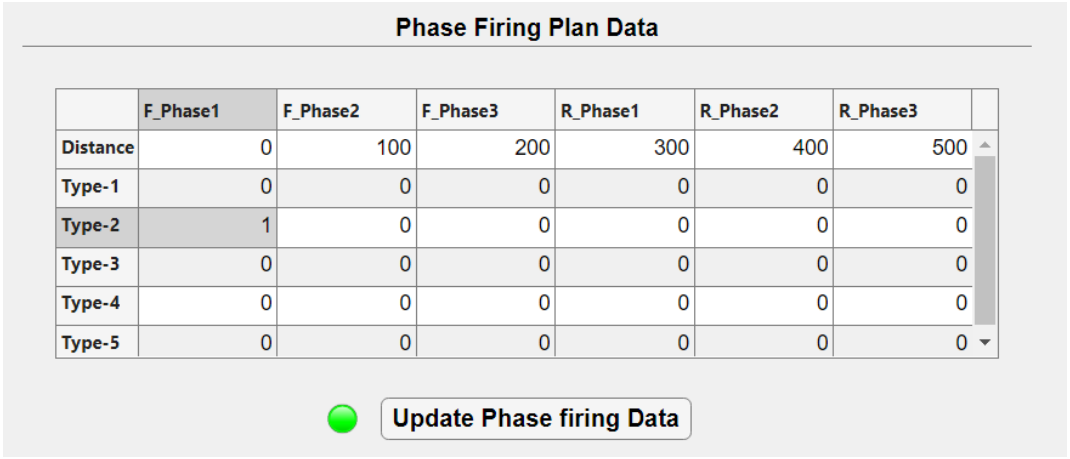


Figure 4.20 Zoom-in phase firing data

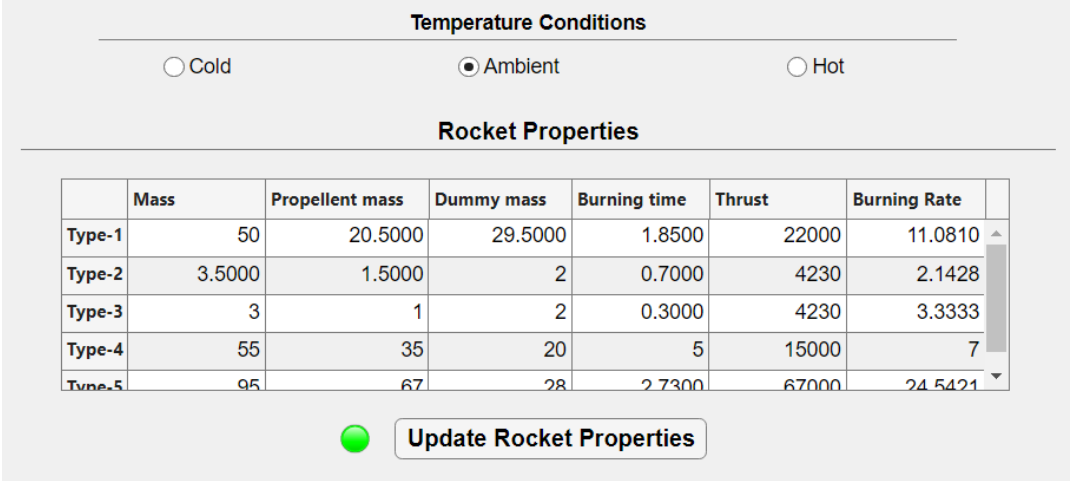


Figure 4.21 Zoom-in rocket details

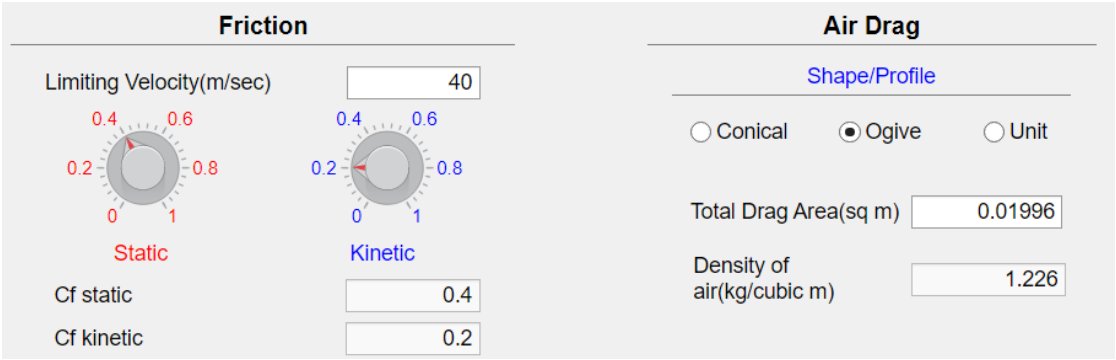


Figure 4.22 Zoom-in Friction and Drag

5.1.1.2. Output:

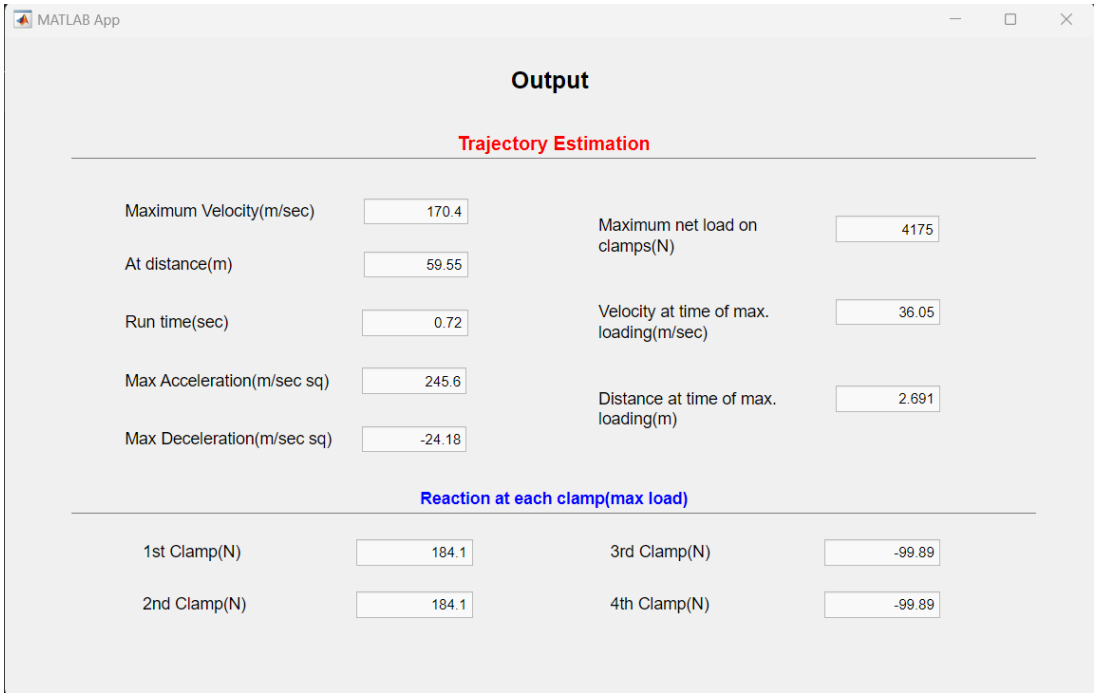


Figure 4.23 Output-results

5.1.1.3. Graphs:

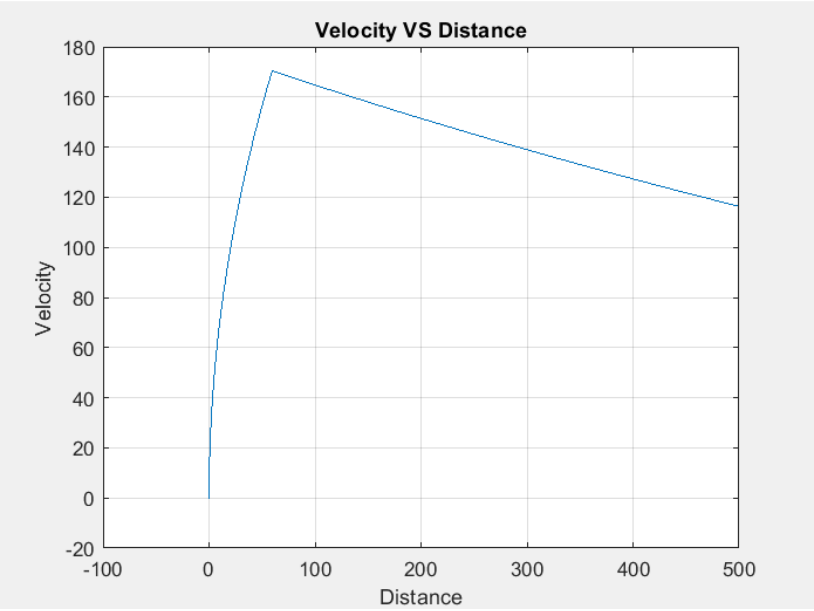


Figure 4.24 Velocity vs Distance plot

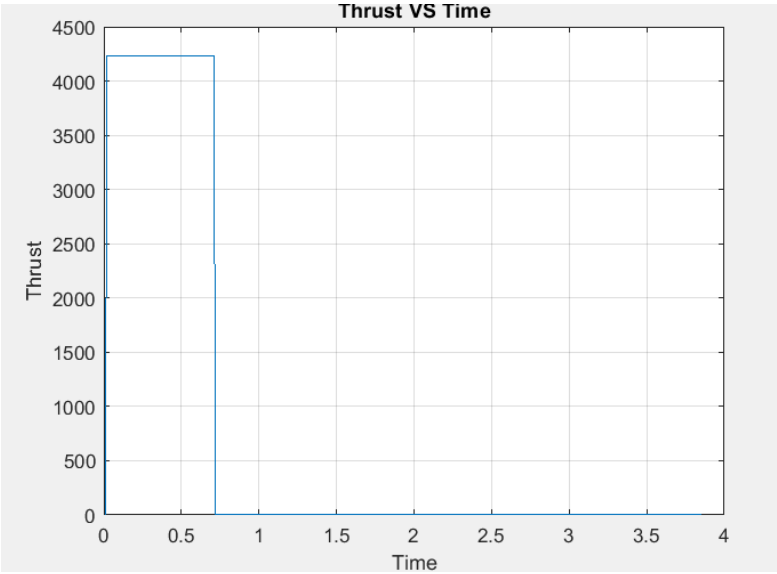


Figure 4.25 Thrust vs Time plot

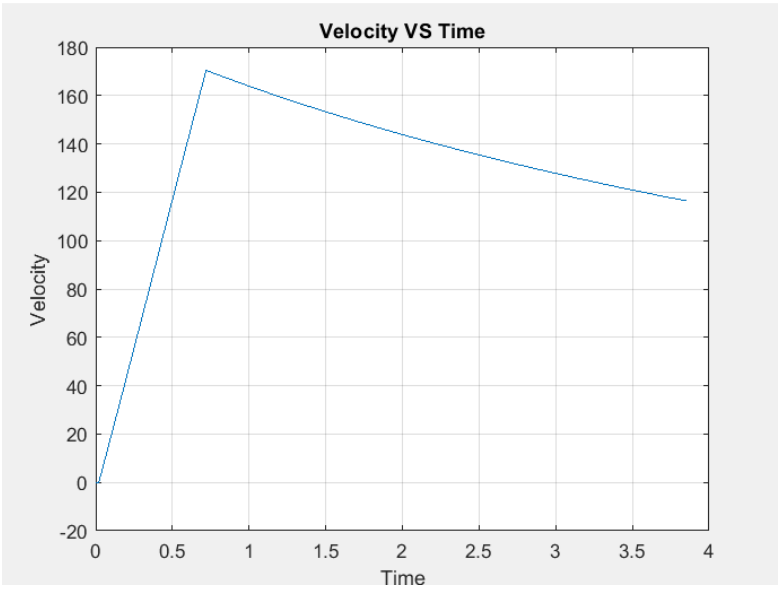


Figure 4.26 Velocity vs Time plot

5.1.2. Multi-phase:

5.1.2.1. Example-A:

In this section, a multi-phase scenario is explored, utilizing a rocket with a constant thrust profile. The methodology employed remains consistent with the previous case, with the sole modifications occurring in the phase firing data, while all other parameters are held constant. This methodology enables a comparative analysis, shedding light on the effects of different phase firing configurations on trajectory estimation outcomes. By maintaining consistency in the methodology and only altering phase firing data, the impact of these specific parameters on the trajectory can be isolated, providing valuable insights into their influence on the overall trajectory dynamics.

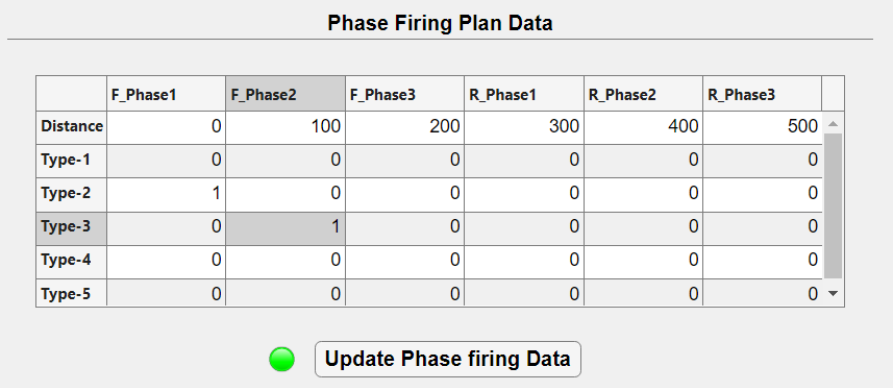


Figure 4.27 Modified Phase firing data

Output:

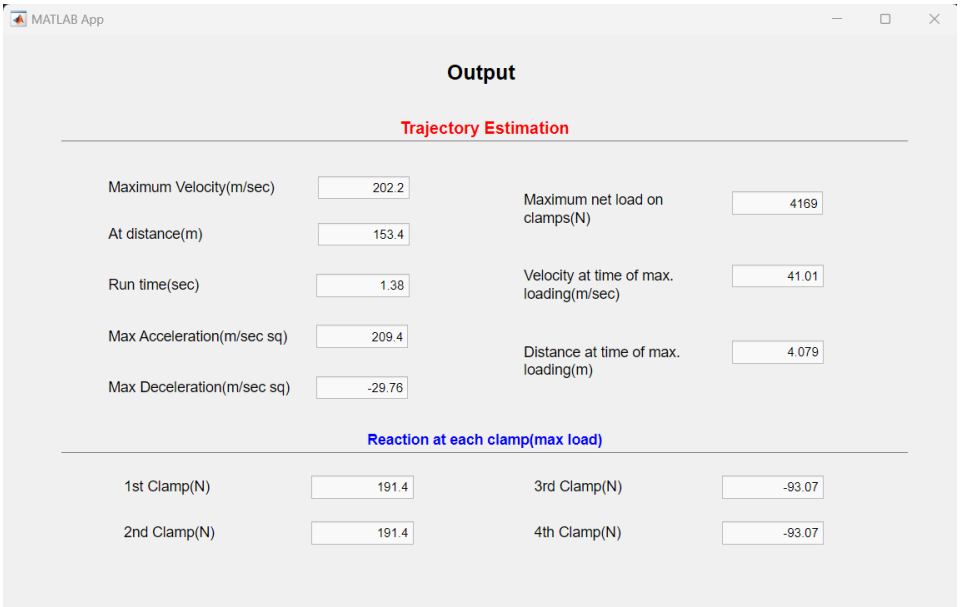


Figure 4.28 Output for eg-a

Graphs:

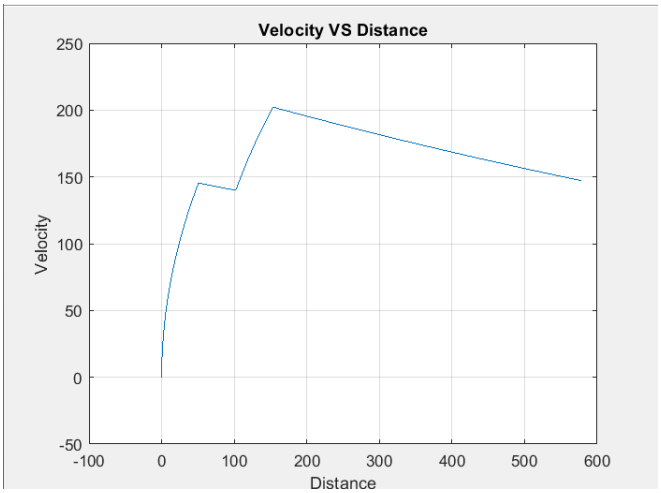


Figure 4.29 Velocity vs Distance for eg-a

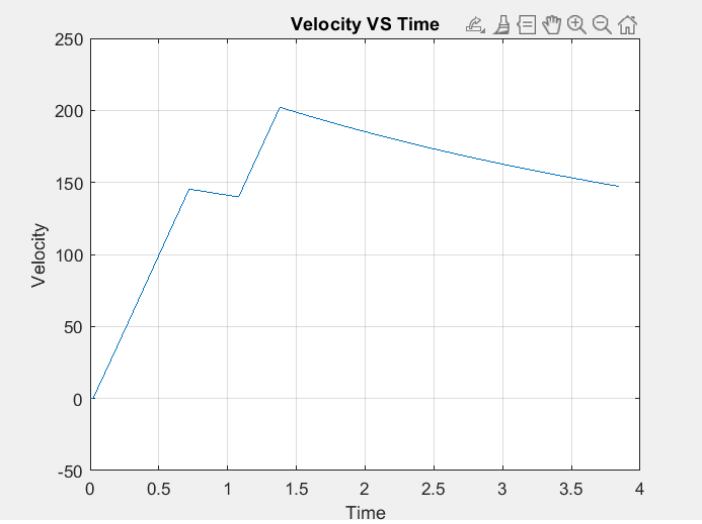


Figure 4.30 Velocity vs Time for eg-a

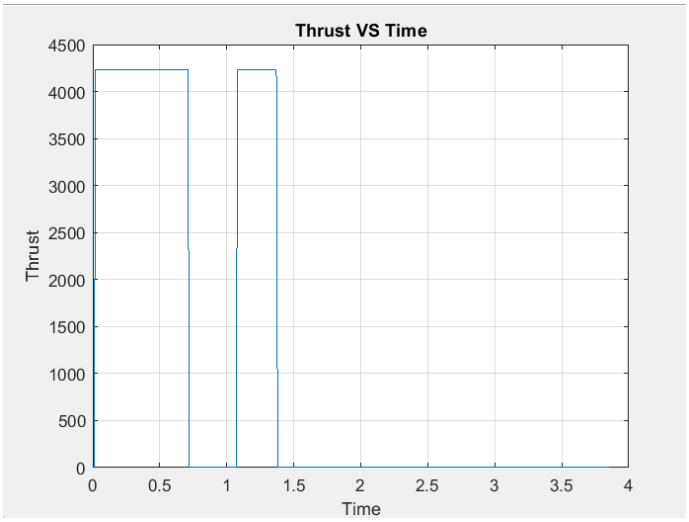


Figure 4.31 Thrust vs Time for eg-a

5.1.2.2. Example B:

In Example B, a significant number of parameters were modified to simulate a specific scenario, encompassing adjustments to phase firing data and other essential parameters. This meticulous approach facilitates a comprehensive analysis of the trajectory estimation outcomes, providing insights into the effects of varying parameters on the trajectory.

Notably, the use of a rocket with variable thrust over time is a crucial aspect to consider. This characteristic plays a pivotal role in the trajectory estimation process, as the evolving thrust profile directly impacts the motion dynamics of the object throughout its trajectory. Understanding and accounting for these variations are essential for accurate and reliable trajectory predictions.

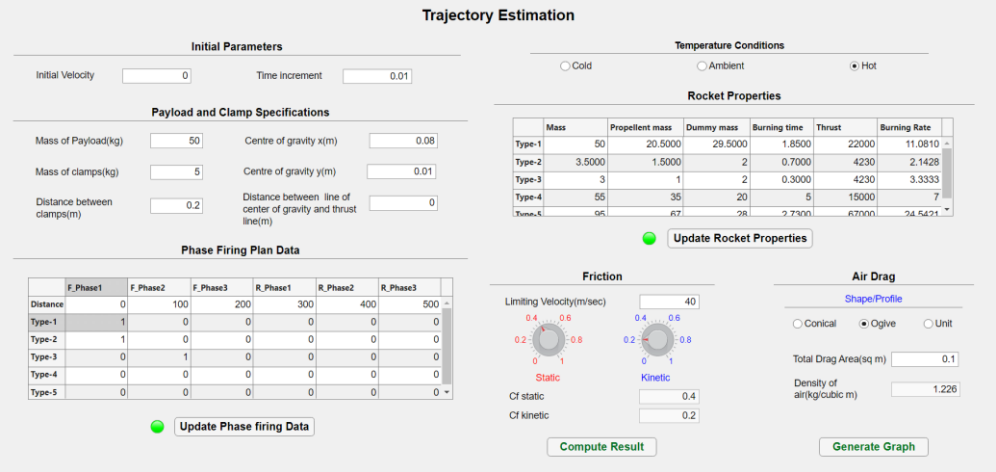


Figure 4.32 eg-b input parameters

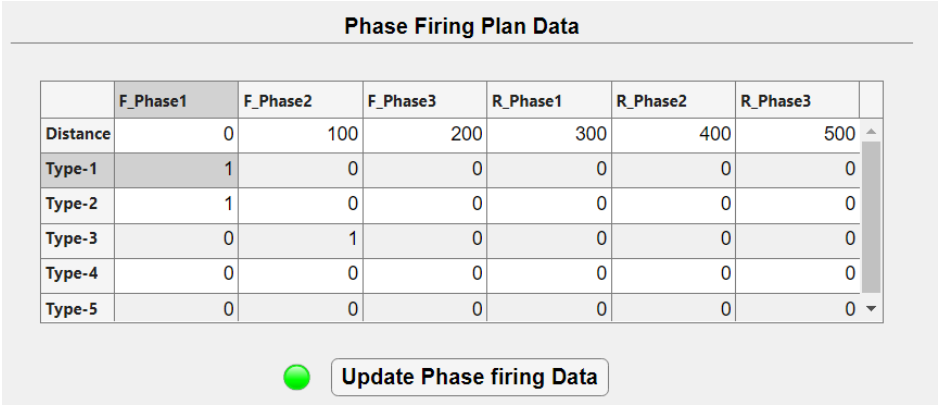


Figure 4.33 Zoom-in phase firing data

Output:

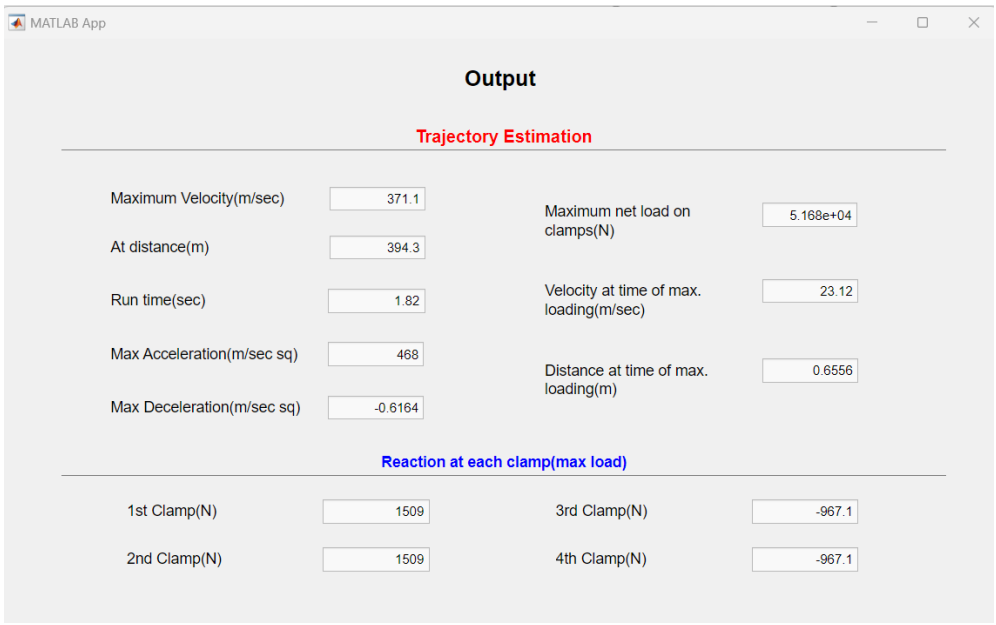


Figure 4.34 Output for eg-

Graphs:

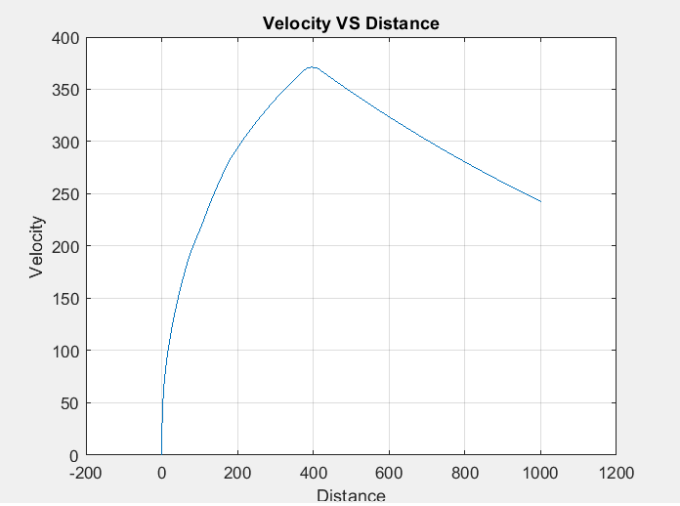


Figure 4.35 Velocity vs Distance for eg-b

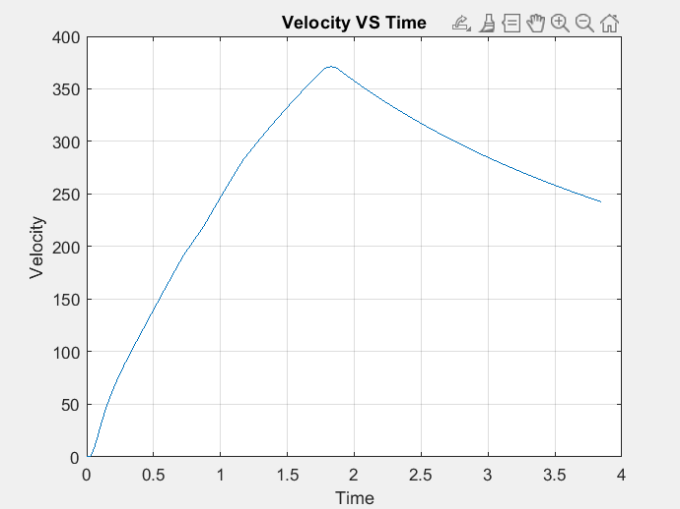


Figure 4.36 Velocity vs Time for eg-b

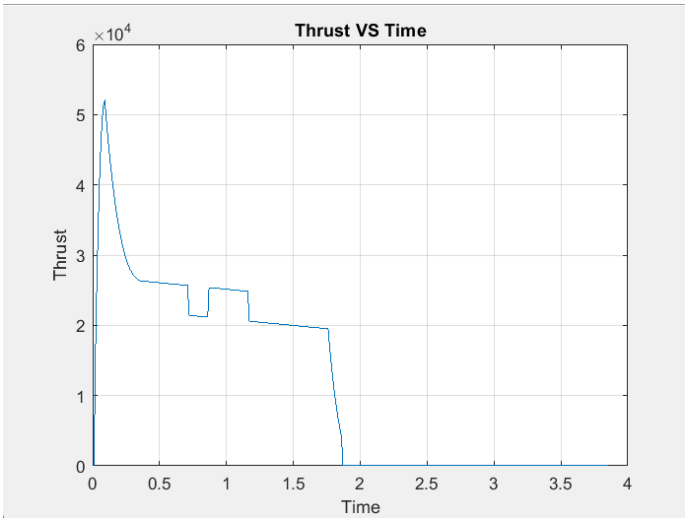


Figure 4.37 Thrust vs Time for eg-b

5.2. Accuracy and validation:

Practical data-driven tuning enables the refinement of the trajectory estimation model, ensuring that it aligns closely with real-world observations. By comparing simulated results with empirical data obtained from experiments or field tests, any discrepancies or inaccuracies can be identified and addressed. This iterative process of validation and refinement enhances the reliability and credibility of the trajectory estimation system, ultimately leading to more accurate predictions and assessments in dynamic testing scenarios.

6. Discussion:

6.1. Challenges and solutions:

Addressing challenges and providing solutions is essential in any project. Let's highlight some common challenges encountered in trajectory estimation and potential solutions:

6.1.1. Challenges:

1. **Uncertainty in Environmental Factors:** Variations in wind speed, air density, and other environmental conditions can affect the accuracy of trajectory estimation.
2. **Complex Motion Dynamics:** Objects undergoing dynamic testing may exhibit complex motion patterns, making it challenging to accurately model their trajectories.
3. **Data Variability:** Real-world data may exhibit variability due to measurement errors or external factors, leading to inaccuracies in trajectory estimation.

6.1.2. Solutions:

1. **Environmental Sensing and Calibration:** Implementing environmental sensors and calibration techniques can help account for variations in environmental factors, improving the accuracy of trajectory estimation.
2. **Advanced Modeling Techniques:** Employing advanced mathematical models and simulation algorithms can better capture the complex motion dynamics of objects, leading to more accurate trajectory predictions.
3. **Data Fusion and Filtering:** Integrating data from multiple sources and applying filtering techniques such as Kalman filtering can mitigate the effects of data variability, enhancing the reliability of trajectory estimation.

By addressing these challenges with appropriate solutions, accuracy and reliability of trajectory estimation in dynamic testing scenarios can be improved.

6.2. Improvements and future work:

Improvements and future work can focus on two key areas: data management and user customization.

1. **Data Saving Capability:** Implementing a feature to save input parameters, simulation results, and user preferences allows for easy retrieval and comparison of past experiments. This enhances the usability and efficiency of the trajectory estimation tool.

2. **Customizable Thrust Profile Editing:** Introducing functionality that enables users to edit the thrust profile of a rocket according to their specifications or experimental requirements adds flexibility and customization to the tool. This empowers users to simulate a wider range of scenarios and better tailor the trajectory estimation process to their specific needs.

By incorporating these improvements, the trajectory estimation tool becomes more versatile, user-friendly, and capable of meeting the evolving demands of dynamic testing applications.

7. Conclusion:

In summary, the trajectory estimation tool presented here offers a reliable means of predicting object motion during dynamic testing scenarios. Through advanced modeling and user-friendly features, it supports precise design, safety, and performance evaluations across aerospace, automotive testing, and robotics. Looking ahead, refining data management and user customization will further elevate the tool's utility and accessibility, meeting the evolving demands of dynamic testing applications.

Chapter-5

Project-2

Military Aircraft Detection using Deep learning- YOLOv7 and RT-DETR

1. Introduction:

1.1. Overview:

The primary objective of the project is to develop a robust deep learning system capable of automatically detecting and classifying military aircraft in aerial imagery. By leveraging state-of-the-art models like YOLOv7 and RT-DETR, the system aims to improve the accuracy and efficiency of military surveillance tasks. Additionally, the project aims to explore potential applications of the developed model in other defence domains, such as the detection and classification of various warheads, if suitable datasets are available. Ultimately, the goal is to enhance situational awareness and decision-making capabilities in military scenarios, thereby contributing to the advancement of defence operations.[10]

1.2. Objective:

1. Develop a deep learning system to detect and classify military aircraft in images.
2. Utilize advanced models like YOLOv7 and RT-DETR for improved accuracy.
3. Enhance efficiency in military surveillance tasks.
4. Contribute to better situational awareness and decision-making in defence operations.

2. Background:

Automated detection and classification of military aircraft are vital for ensuring timely threat detection, strategic planning, and effective mission execution in defence operations. By harnessing the power of deep learning techniques, this project aims to enhance situational awareness and decision-making capabilities in military scenarios.

The utilization of advanced algorithms allows for the rapid and accurate identification of aircraft types, aiding defence personnel in assessing potential threats and formulating appropriate responses.

Additionally, with suitable datasets, this model could potentially be applied at the RTRS (Rail Track Rocket Sled), DRDO test facility for the detection and classification of various warheads, thereby extending its utility and impact across diverse defence applications.

3. Methodology:

The deep learning pipeline involves several key steps:

3.1. Data Preprocessing:

Aerial imagery of military aircraft undergo preprocessing to ensure compatibility with deep learning models. This includes resizing images, normalizing pixel values, and augmenting the dataset to enhance its quality and diversity. These steps optimize the dataset for improved model training and accuracy in military aircraft recognition tasks.

3.2. Model Training:

The YOLOv7 and RT-DETR models are trained using annotated datasets of military aircraft images. During training, the models learn to recognize and classify various aircraft types by minimizing a predefined loss function, refining their ability to accurately detect and classify objects within the images.

3.3. Model Optimization:

Following the initial training phase, the YOLOv7 and RT-DETR models undergo fine-tuning and hyperparameter tuning processes to refine their performance. This iterative optimization involves adjusting various parameters to maximize accuracy and minimize inference time, ensuring the models achieve optimal performance levels for their specific tasks and datasets.

4. Data collection:

For gathering data, Kaggle datasets are used. These datasets are chosen for their quality and relevance to the project.[11]

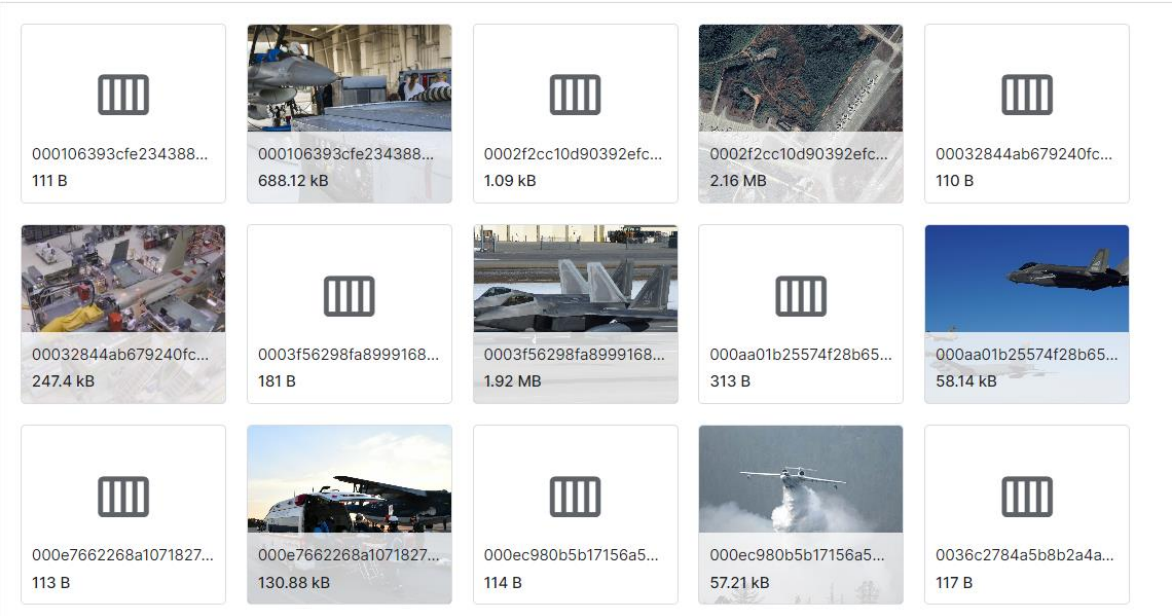


Figure 5.1 Dataset Directory

```
In [6]: from google.colab import files
        files.upload()

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving kaggle.json to kaggle.json
Out[6]: {'kaggle.json': b'{"username":"charurathee","key":"04c3cf646f58ad25a997a23ee66c3da0"}'}

In [7]: !ls -lha kaggle.json

-rw-r--r-- 1 root root 67 May 21 21:02 kaggle.json

In [8]: !pip install -q kaggle

In [9]: # The Kaggle API client expects this file to be in ~/.kaggle,
        # so move it there.
        !mkdir -p ~/.kaggle
        !cp kaggle.json ~/.kaggle/

        # This permissions change avoids a warning on Kaggle tool startup.
        !chmod 600 ~/.kaggle/kaggle.json
```

Figure 5.2 Setting up Kaggle Configuration[12]

The Kaggle API configuration was set up in the Google Colab environment to facilitate seamless access to Kaggle datasets. Initially, the Kaggle API key file was uploaded, and then the Kaggle package was installed.

Subsequently, the key file was moved to the appropriate directory (~/.kaggle/) to align with the Kaggle API client's expectations. Lastly, appropriate permissions were set for the key file to ensure secure access. This configuration allows for streamlined retrieval and utilization of Kaggle datasets for the project.

```
In [12]: !kaggle datasets download -d a2015003713/militaryaircraftdetectiondataset

Dataset URL: https://www.kaggle.com/datasets/a2015003713/militaryaircraftdetectiondataset
License(s): unknown
Downloading militaryaircraftdetectiondataset.zip to /content
100% 12.6G/12.6G [07:20<00:00, 37.2MB/s]
100% 12.6G/12.6G [07:20<00:00, 30.7MB/s]

In [13]: # Unzip the dataset
         !unzip militaryaircraftdetectiondataset.zip -d /content/militaryaircraftdetectiondataset

Streaming output truncated to the last 5000 lines.
  inflating: /content/militaryaircraftdetectiondataset/dataset/d3bb7e234d752428218e9b1e31314f22.csv
  inflating: /content/militaryaircraftdetectiondataset/dataset/d3bb7e234d752428218e9b1e31314f22.jpg
  inflating: /content/militaryaircraftdetectiondataset/dataset/d3bba2d98cc91eed0be188c02eef652b.csv
  inflating: /content/militaryaircraftdetectiondataset/dataset/d3bba2d98cc91eed0be188c02eef652b.jpg
  inflating: /content/militaryaircraftdetectiondataset/dataset/d3bd64c61df9f77e872122ab9e065e65.csv
  inflating: /content/militaryaircraftdetectiondataset/dataset/d3bd64c61df9f77e872122ab9e065e65.jpg
  inflating: /content/militaryaircraftdetectiondataset/dataset/d3bd6ae57951c6c23dbb07663d019ec9.csv
  inflating: /content/militaryaircraftdetectiondataset/dataset/d3bd6ae57951c6c23dbb07663d019ec9.jpg
```

Figure 5.3 Downloading and Unzipping Kaggle Dataset

The "Military Aircraft Detection Dataset" was procured from Kaggle, boasting a substantial size of 12.6 GB. Upon successful download, the dataset underwent unzipping, with all files extracted to the specified directory located at /content/militaryaircraftdetectiondataset. This meticulous process was essential to prepare the dataset for comprehensive analysis and utilization within the project. By organizing the dataset in the designated directory, it became readily accessible for data preprocessing, model training, and validation processes. The dataset's size underscores the richness and depth of information contained within, providing ample resources for robust model development and optimization.

```
In [15]: # Path to the unzipped dataset
dataset_path = '/content/militaryaircraftdetectiondataset/dataset'

# Collect CSV and JPG paths
csv_paths = glob.glob(os.path.join(dataset_path, '*.csv'))
jpg_paths = glob.glob(os.path.join(dataset_path, '*.jpg'))
csv_paths.sort()
jpg_paths.sort()
print('number of images:', len(csv_paths))

# Create necessary directories in the Colab environment
os.makedirs('/content/ultralytics/data/train/images', exist_ok=True)
os.makedirs('/content/ultralytics/data/train/labels', exist_ok=True)

os.makedirs('/content/ultralytics/data/valid/images', exist_ok=True)
os.makedirs('/content/ultralytics/data/valid/labels', exist_ok=True)

os.makedirs('/content/ultralytics/data/test/images', exist_ok=True)
os.makedirs('/content/ultralytics/data/test/labels', exist_ok=True)

number of images: 14088
```

Figure 5.4 setting directory structure

```
                f.write(output_string)
        except:
            print(txt_file_path)
            raise

print('test:', len(glob.glob('/content/ultralytics/data/test/labels/*.txt')))
print('valid:', len(glob.glob('/content/ultralytics/data/valid/labels/*.txt')))
print('train:', len(glob.glob('/content/ultralytics/data/train/labels/*.txt')))

14088it [00:42, 332.21it/s]
test: 0
valid: 2635
train: 0
```

Figure 5.5 Preparing Dataset Annotations and File Structure

After establishing the dataset directory structure, the annotations and file preparation tasks commence. This involves organizing images and creating label files according to provided annotations. Such meticulous organization ensures alignment with the chosen model architecture and streamlines data management for efficient model training.

```
print(yaml_content)

train: /content/ultralytics/data/train
val: /content/ultralytics/data/valid
test: /content/ultralytics/data/test
nc: 49
names: ["A10", "A400M", "AG600", "AV8B", "B1", "B2", "B52", "Be200", "C130", "C2", "C17", "C5", "E2", "E7", "EF2000", "F11
7", "F14", "F15", "F16", "F18", "F22", "F35", "F4", "JAS39", "MQ9", "Mig31", "Mirage2000", "P3", "RQ4", "Rafale", "SR71",
"Su34", "Su57", "Tu160", "Tu95", "Tornado", "U2", "US2", "V22", "XB70", "YF23", "Vulcan", "J20", "KC135", "J10", "Su25", "S
u24", "H6", "JF17"]
```

Figure 5.6 YAML configuration file

After setting up the dataset directories, a YAML configuration file is generated. This file specifies the paths to the training, validation, and test datasets, the number of classes, and their names. This configuration ensures the model can correctly identify the dataset locations and class information, facilitating efficient training and evaluation.

5. Implementation:

The following libraries are utilized in the implementation:

- **Numpy:** Used for numerical computations and array manipulation.
- **Pandas:** Employed for data manipulation and analysis, particularly in handling tabular data.
- **matplotlib.pyplot:** Utilized for data visualization, including plotting charts, graphs, and histograms.
- **glob:** Enables file pattern matching, allowing for easy retrieval of file paths matching specified patterns.
- **shutil:** Used for high-level file operations such as copying, moving, and deleting files and directories.
- **cv2:** OpenCV library, essential for image processing tasks such as reading, writing, and manipulating images.
- **os:** Provides a portable way of using operating system-dependent functionality, used for various file and directory operations.
- **tqdm:** Used to create progress bars for iterative processes, enhancing the user experience during tasks that take a significant amount of time.
- **Ultralytics:** A package providing utilities for object detection tasks, such as YOLOv5 implementation and evaluation.
- **Torch:** The PyTorch library, used for building and training deep learning models, including neural networks for tasks such as object detection and classification.

Figure 5.7 Library import

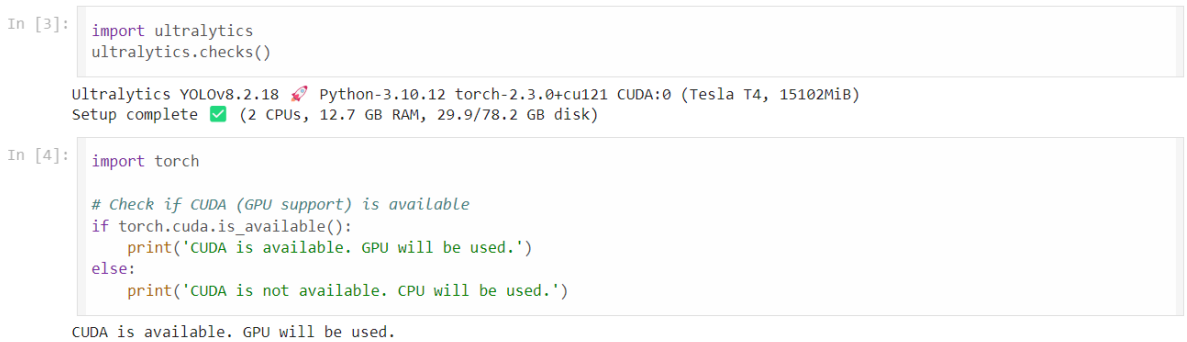


Figure 5.8 Environment Checks

Detailed Steps for Developing and Integrating a Deep Learning Model into the System:

5.1. Model Selection:

When selecting a model, factors such as performance metrics, computational efficiency, and task suitability are carefully considered. The focus is on balancing accuracy and speed to ensure optimal performance given the project's requirements. By evaluating these aspects thoroughly, a model that best meets the needs is chosen, delivering reliable results efficiently and effectively. This approach underscores the commitment to leveraging cutting-edge technology to achieve project objectives with maximum efficiency and effectiveness.

The specific models chosen for this project:

1. YOLOv7 (You Only Look Once version 7):

- YOLOv7 is a state-of-the-art object detection model known for its real-time performance and high accuracy.
- It operates by dividing the input image into a grid and predicting bounding boxes and class probabilities for each grid cell.
- YOLOv7 is chosen for its balance between accuracy and speed, making it suitable for military aircraft detection tasks requiring real-time inference capabilities.[13]

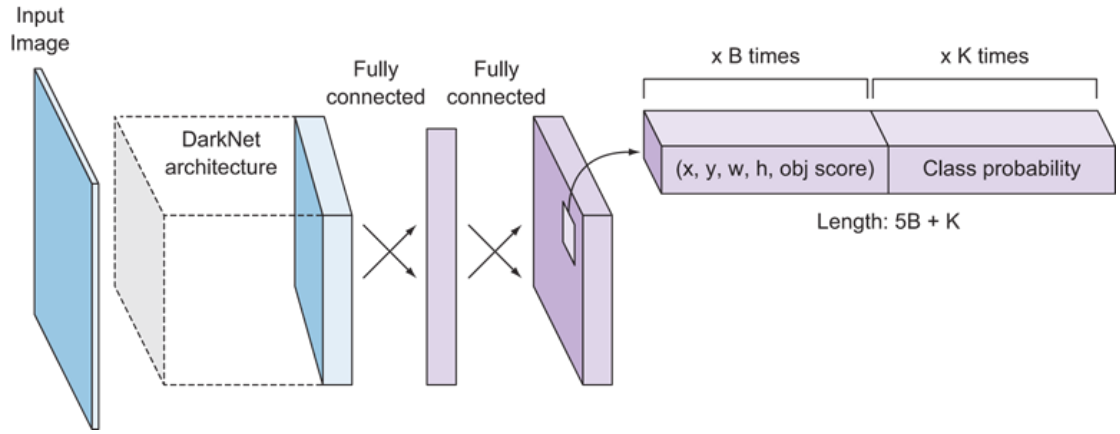


Figure 5.9 Yolo Model Architecture[14]

2. RT-DETR (Real-Time Detection Transformer):

- RT-DETR is a transformer-based object detection model that leverages self-attention mechanisms to capture global context information.
- Unlike traditional convolutional neural networks (CNNs), RT-DETR directly predicts object bounding boxes and class labels without the need for anchor boxes or region proposal networks.
- RT-DETR is selected for its ability to handle complex scenes and varying object scales efficiently, making it suitable for military aircraft detection in diverse environment.[16]

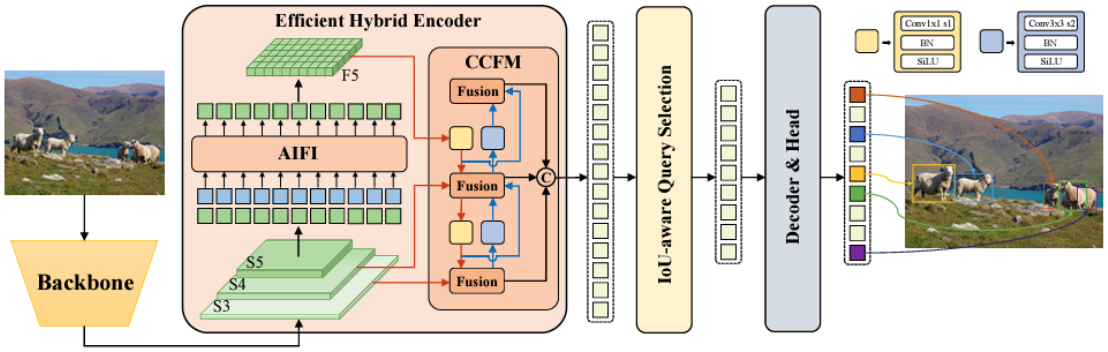


Figure 5.10 RT-DETR model Architecture[15]

The models are selected based on their proven performance, efficiency, and suitability for military aircraft detection tasks. Rigorous evaluations ensure accurate identification and categorization of aircraft types in dynamic environments. Their efficiency in terms of computational resources and inference speed aligns with real-time operational needs. By leveraging these models, the project aims to achieve robust results meeting military-grade standards. Selection prioritizes models with a track record of success in similar applications, ensuring accurate and timely insights for defense operations.

5.2. Model Training:

5.2.1. Using YOLOv7

```
In [ ]: !git clone https://github.com/WongKinYiu/yolov7
        %pip install -r yolov7/requirements.txt
        !pip uninstall -y wandb
        %pip install yt-dlp
        %pip install ffmpeg

Cloning into 'yolov7'...
```

Figure 5.11 Setting Up YOLOv7 Environment

```
In [ ]: with open('./yolov7/detect.py') as f:
            data_lines = f.read()
        data_lines = data_lines.replace('line_thickness=1', 'line_thickness=4')
        with open('./yolov7/detect.py', mode='w') as f:
            f.write(data_lines)
```

Figure 5.12 Modifying YOLOv7 Detection Script

To set up the YOLOv7 environment, the repository was cloned, and necessary dependencies were installed. Additionally, the `detect.py` script was modified to adjust the line thickness for better visual clarity in detection outputs.

```
In [ ]: hyp = {
            'lr0': 0.0002,  # initial learning rate (SGD=1E-2, Adam=1E-3)
            'lrf': 0.001,  # final OneCycleLR learning rate (lr0 * lrf)
            'momentum': 0.937,  # SGD momentum/Adam beta1
            'weight_decay': 0.0005,  # optimizer weight decay 5e-4
            'warmup_epochs': 2.0,  # warmup epochs (fractions ok)
            'warmup_momentum': 0.8,  # warmup initial momentum
            'warmup_bias_lr': 0.0001,  # warmup initial bias lr
            'box': 0.05,  # box loss gain
            'cls': 0.3,  # cls loss gain
            'cls_pw': 1.0,  # cls BCELoss positive_weight
            'obj': 0.7,  # obj loss gain (scale with pixels)
            'obj_pw': 1.0,  # obj BCELoss positive_weight
            'iou_t': 0.20,  # IoU training threshold
            'anchor_t': 4.0,  # anchor-multiple threshold
            # anchors: 3  # anchors per output layer (0 to ignore)
            'fl_gamma': 0.0,  # focal loss gamma (efficientDet default gamma=1.5)
            'hsv_h': 0.015,  # image HSV-Hue augmentation (fraction)
            'hsv_s': 0.7,  # image HSV-Saturation augmentation (fraction)
            'hsv_v': 0.4,  # image HSV-Value augmentation (fraction)
            'degrees': 0.5,  # image rotation (+/- deg)
            'translate': 0.25,  # image translation (+/- fraction)
            'scale': 0.45,  # image scale (+/- gain)
            'shear': 0.5,  # image shear (+/- deg)
            'perspective': 0.0,  # image perspective (+/- fraction), range 0-0.001
            'flipud': 0.15,  # image flip up-down (probability)
            'fliplr': 0.5,  # image flip left-right (probability)
            'mosaic': 0.8,  # image mosaic (probability)
            'mixup': 0.35,  # image mixup (probability)
            'copy_paste': 0,  # image copy paste (probability)
            'paste_in': 0,  # image copy paste (probability), use 0 for faster training
            'loss_ota': 1, # use ComputeLossOTA, use 0 for faster training
        }
        with open('./yolov7/data/hyp.yaml', 'w') as f:
            yaml.dump(hyp, f)
```

Figure 5.13 Configuring YOLOv7 Hyperparameters

The model training configuration for YOLOv7 involves setting various hyperparameters and specifying the environment for effective training. The hyperparameters (`hyp`) are defined as follows:

- **Learning Rate:** The initial and final learning rates, crucial for adjusting the model's weight updates during training.
- **Momentum:** Used to accelerate gradients vectors in the right directions, leading to faster converging.
- **Weight Decay:** Regularizes the model to prevent overfitting by penalizing large weights.
- **Warmup Parameters:** Gradually increases the learning rate during the initial epochs to stabilize training.
- **Loss Gains:** Adjusts the contribution of each component (box, class, object) to the total loss.
- **IoU Threshold:** Intersection over Union threshold for object detection.
- **Anchor Threshold:** Determines the matching threshold for anchors.
- **Data Augmentation Parameters:** Applies various transformations to the training images to enhance model generalization.
- **Focal Loss Gamma:** Used to address class imbalance by focusing more on hard-to-classify examples.
- **Loss OTA: Option to use the ComputeLossOTA method for calculating the loss.**

```
In [ ]: # YOLOv-W6
        os.makedirs('./yolov7/checkpoints', exist_ok=True)
        shutil.copy('D:\\work\\aircraft_detection_yolo7\\Data\\yolov7-w6-best.pt', './yolov7/checkpoints/yolov7-w6.pt')

Out[ ]: './yolov7/checkpoints/yolov7-w6.pt'

In [ ]: import os
        os.environ["PYTORCH_CUDA_ALLOC_CONF"] = "expandable_segments:True"

In [ ]: !python ./yolov7/test.py \
        --batch-size 4 \
        --img-size 1280 \
        --name yolov7_mad_12 \
        --data ./yolov7/data/mad.yaml \
        --weights ./yolov7/checkpoints/yolov7-w6.pt \
        --v5-metric

Namespace(weights=['./yolov7/checkpoints/yolov7-w6.pt'], data='./yolov7/data/mad.yaml', batch_size=4, img_size=1280, conf_thres=
ngle_cls=False, augment=False, verbose=False, save_txt=False, save_hybrid=False, save_conf=False, save_json=False, project='runs
_trace=False, v5_metric=True)
YOLOR  v0.1-128-ga207844 torch 2.2.2 CUDA:0 (NVIDIA GeForce RTX 3050 Laptop GPU, 4095.5MB)
```

Figure 5.14 YOLOv7-W6 Model Evaluation

The training script sets up directories, pre-trained weights, and optimizes memory allocation. YOLOv7 training commences with specified parameters, ensuring effective model training.

5.2.2. Using RT-DETR

```
In [18]: if MODE == 'train':
             from ultralytics import RTDETR
             import os

             # Set the path to the pretrained model
             pretrained_model_path = '/content/rtdetr-l.pt'
             # Set the path to the data YAML file
             data_yaml_path = '/content/ultralytics/data/mad.yaml'

             # Initialize the RTDETR model
             model = RTDETR(model='rtdetr-l.pt')

             # Train the model
             result = model.train(
                 name='RT-DETR_001',
                 epochs=1,
                 imgsz=640,  # 1280 on colab A100
                 batch=4,  # 8 on colab A100
                 # optimizer='auto',

                 # augmentation
                 close_mosaic=0,
                 hsv_h=0.02,
                 hsv_s=0.75,
                 hsv_v=0.45,
                 degrees=0.1,
                 translate=0.25,
                 scale=0.55,
                 shear=0.1,
                 perspective=0.0,
                 flipud=0.1,
                 fliplr=0.5,
                 mosaic=1.0,
                 mixup=0.5,

                 pretrained=pretrained_model_path,
                 data=data_yaml_path,
             )
```

Figure 5.15 Training RT-DETR Model

- **Initialization:**

Pretrained model path (`pretrained_model_path`) and data YAML file path (`data_yaml_path`) are specified to set up the training environment.

- **Model Setup:**

The RT-DETR model is initialized using the `RTDETR` class from the `ultralytics` library, passing the pretrained model path as an argument. This sets up the model for training.

- **Training:**

The model is trained using the `train` method, specifying parameters such as the model's name (`RT-DETR_001`), number of epochs (1), image size (640 pixels), batch size (4), and augmentation settings. Augmentation techniques such as mosaic, mixup, flipping, translation, scaling, rotation, and perspective are applied to enrich the diversity of the training dataset, improving the model's robustness and generalization capability.

```
In [20]: # validate trained weights
         !yolo val \
         model='/content/rtdetr-l-best_49cls.pt' \
         data='/content/ultralytics/data/mad.yaml' \
         augment \
         batch=8 \
         imgsz=640
```

Figure 5.16 Validating Trained Weights

The trained weights are validated using the YOLO validation command. The model weights, specified by the path to the trained weights file (`rtdetr-l-best_49cls.pt`), along with the data configuration file (`mad.yaml`), are utilized for validation. Augmentation techniques are applied during validation to ensure robustness. This process assesses the model's performance and ensures its effectiveness in detecting and classifying military aircraft in aerial imagery.

6. Evaluation:

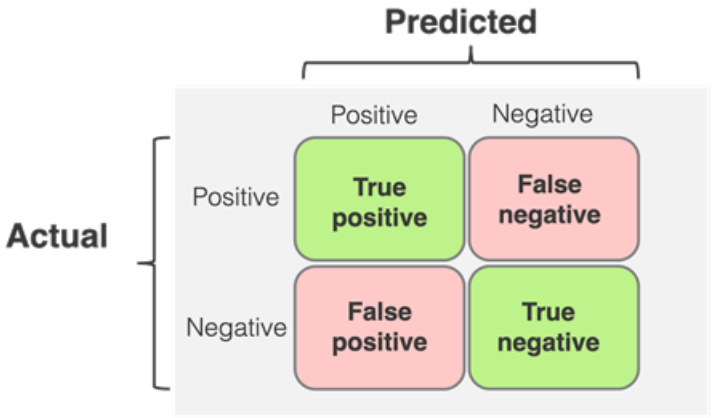


Figure 5.17 Confusion Matrix[17]

A confusion matrix summarizes a classification model's performance by comparing its predictions against actual outcomes. It provides key metrics like accuracy, precision, recall, and F1 score, offering insights into the model's effectiveness across different classes.

Model evaluation involves several key metrics to understand performance comprehensively:

Accuracy	Predictions/ Classifications	$\frac{\text{Correct}}{\text{Correct + Incorrect}}$
Precision	Predictions/ Classifications	$\frac{\text{True Positive}}{\text{True Positive + False Positive}}$
Recall	Predictions/ Classifications	$\frac{\text{True Positive}}{\text{True Positive + False Negative}}$
F1	Predictions/ Classifications	$\frac{\text{2 * True Positive}}{\text{True Positive + 0.5 (False Positive + False Negative)}}$

Figure 5.18 Key Formulae[18]

- **Precision:**

This measures the proportion of true positive predictions out of all positive predictions made by the model. High precision means the model has a low rate of false positives, which is crucial when the cost of a false positive is high.

- **Recall:**

This measures the proportion of true positive predictions out of all actual positive cases. High recall indicates the model effectively identifies most of the actual positives, which is important when the cost of a false negative is high.

- **F1 Score:**

This is the harmonic mean of precision and recall, providing a single metric that balances both. It's particularly useful when you need to consider both false positives and false negatives and when you have an imbalanced dataset.

- **Accuracy:**

This measures the overall proportion of correct predictions (both true positives and true negatives) out of all predictions made. While easy to interpret, accuracy can be misleading in imbalanced datasets, where the model might predict the majority class well but fail on minority classes.

- **Mean Average Precision (mAP):**

This calculates the average precision for each class and then averages these values. It offers a detailed evaluation, especially valuable in multi-class classification scenarios, highlighting the model's performance across all classes rather than just an overall score.

Collectively, these metrics provide a comprehensive view of the model's strengths and weaknesses, guiding further improvements and optimizations to ensure the model meets performance requirements.[18]

6.1. YOLOv7 Results:

6.2.1. Graphs:

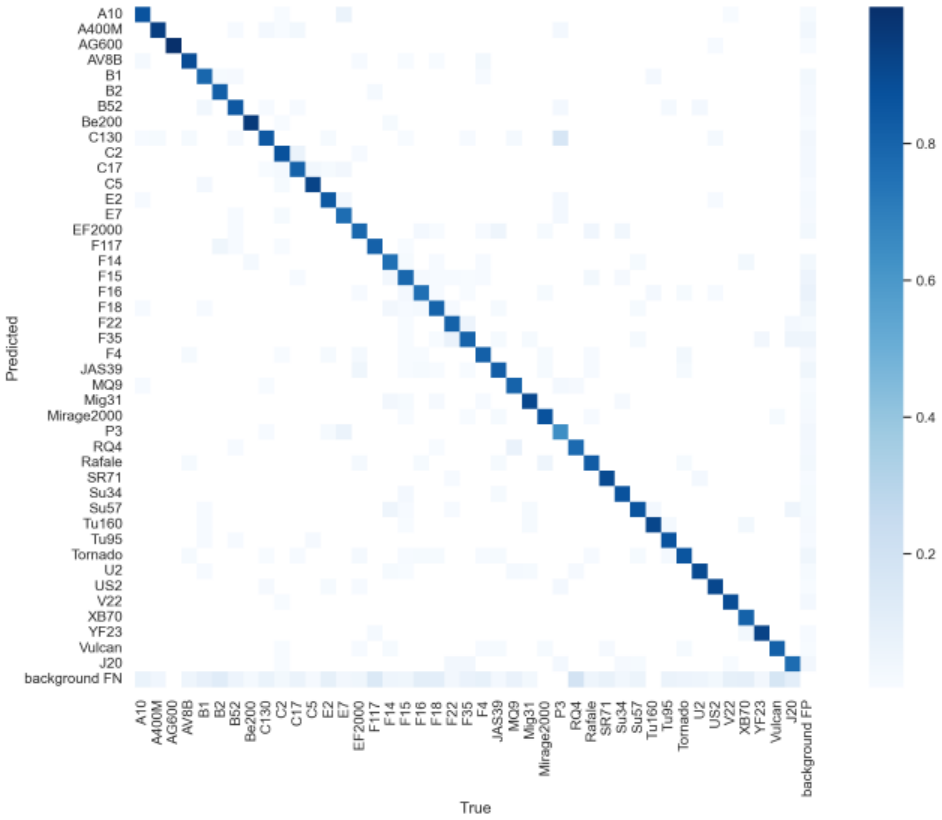


Figure 5.19 Confusion matrix-YOLOv7

The evaluation of the model reveals several key insights into its performance. One prominent observation is the phenomenon of diagonal dominance within the confusion matrix. This dominance indicates that the model has made correct predictions for the majority of instances, as most of the true positive classifications align with the diagonal of the matrix.

Furthermore, the low values observed off the diagonal signify a minimal number of false positives and false negatives. This suggests that the model has achieved high precision and recall across all classes, demonstrating its ability to accurately classify instances from different categories.

Overall, the balanced performance depicted by the confusion matrix illustrates the model's robustness and generalization capability. Its capacity to distinguish between classes effectively enhances its reliability and suitability for practical deployment in real-world scenarios.

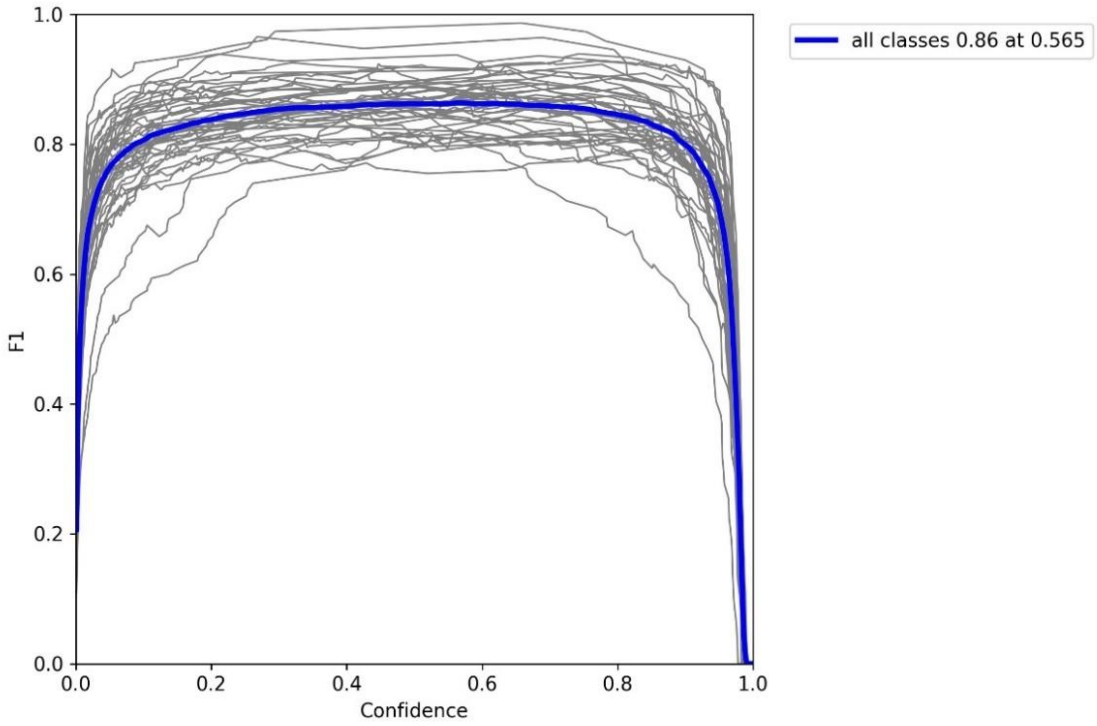


Figure 5.20 F1 vs confidence curve

Observation:

At a confidence threshold of 0.565, all classes achieve a high F1 score of 0.86.

Implications:

- **Balanced Precision and Recall:** The high F1 score indicates a good balance between precision and recall, ensuring accurate identification of true positives while minimizing false positives and false negatives.
- **Effective Threshold:** The confidence threshold of 0.565 strikes an optimal balance, neither overly strict nor lenient, maximizing model performance.
- **Consistent Class Performance:** The uniform high F1 scores across all classes suggest the model generalizes well and performs consistently across different categories, enhancing its reliability.
- **Deployment Readiness:** The strong performance at this threshold renders the model suitable for deployment in real-world applications where accurate predictions are vital for decision-making.
- **Model Robustness:** The high F1 scores across all classes indicate the model's robustness and its ability to handle various data scenarios effectively, ensuring dependable performance in diverse settings.

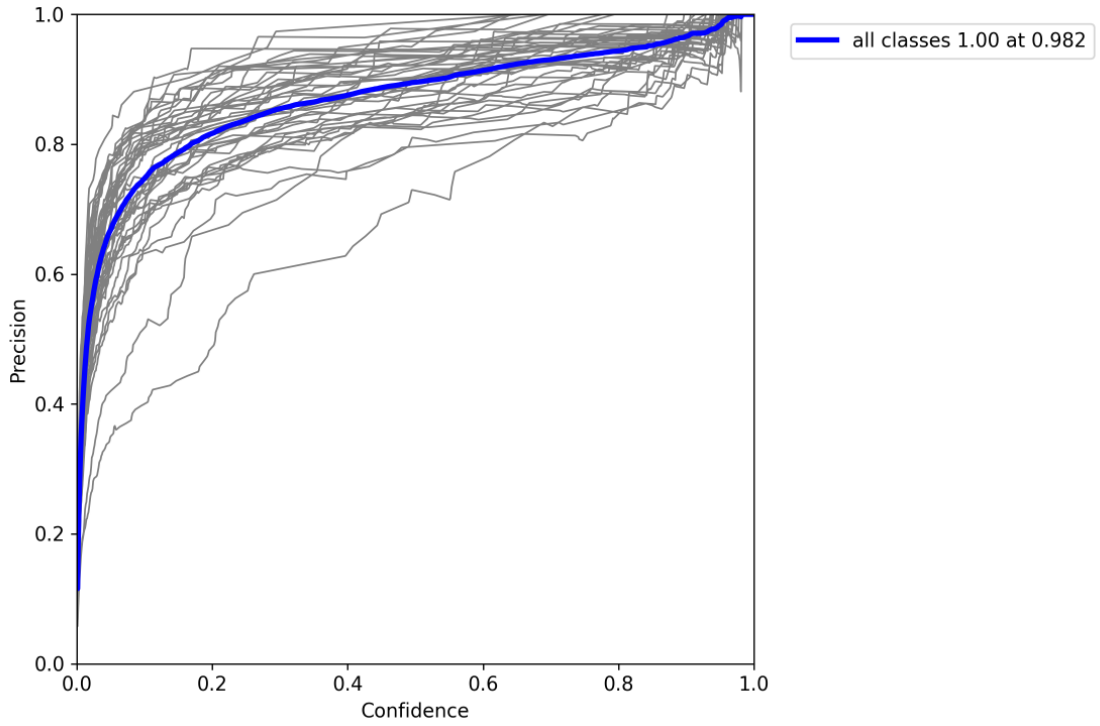


Figure 5.21 Precision vs Confidence curve

Observation:

All classes achieve perfect precision (precision score of 1) at a confidence threshold of 0.982 on the Precision vs. Confidence curve.

Implications:

- **Flawless Precision:** The model's predictions at this threshold are entirely accurate with no false positives, ensuring reliable results.
- **High Confidence Standard:** The model demonstrates exceptional confidence in its predictions at the 0.982 threshold, indicating robust performance.
- **Reduced False Alarms:** Perfect precision minimizes the occurrence of false alarms, crucial for applications where precision is paramount, enhancing trust in the model's predictions.
- **Potential Trade-off:** While achieving high precision, there might be a compromise on recall due to the stringent threshold, necessitating a balance between precision and recall based on the application's requirements.
- **Applicability in Critical Situations:** This precision threshold is particularly suited for applications where the cost of false positives is high, such as medical diagnoses or security systems, ensuring accurate and reliable outcomes in critical scenarios.

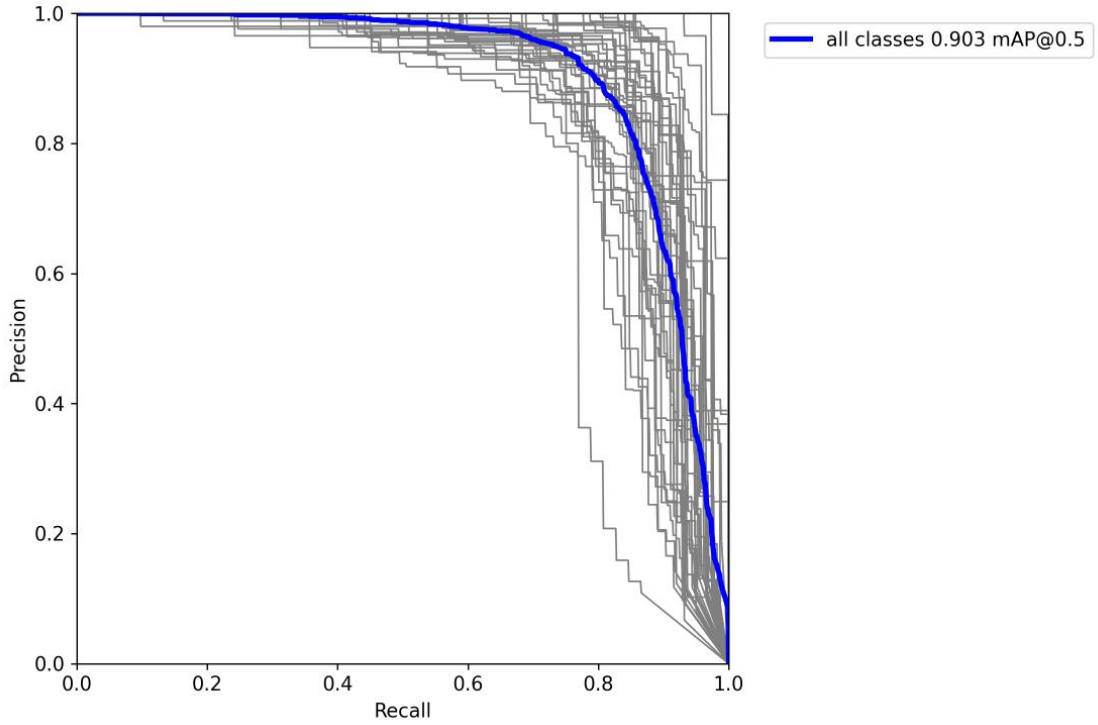


Figure 5.22 Precision vs Recall Curve

Observation:

Consistently high precision of 0.903 across all classes at mAP@0.5 indicates reliable detection accuracy across different categories, highlighting the model's robust performance.

Implications:

- **Reliable Detection Accuracy:** The model consistently maintains a high precision score of 0.903, showcasing its ability to accurately detect objects across various categories.
- **Robust Performance:** The consistent precision across different classes suggests that the model generalizes well and performs reliably in diverse scenarios.
- **Confidence in Predictions:** The high precision score instills confidence in the model's predictions, making it suitable for real-world applications where accuracy is critical.
- **Effective Object Detection:** Achieving a high mAP@0.5 indicates that the model effectively detects objects with a high level of precision, enhancing its utility in tasks such as surveillance, autonomous driving, and object recognition.
- **Performance Benchmark:** The model's performance, as reflected by its consistent precision at mAP@0.5, serves as a benchmark for evaluating and comparing other object detection systems.

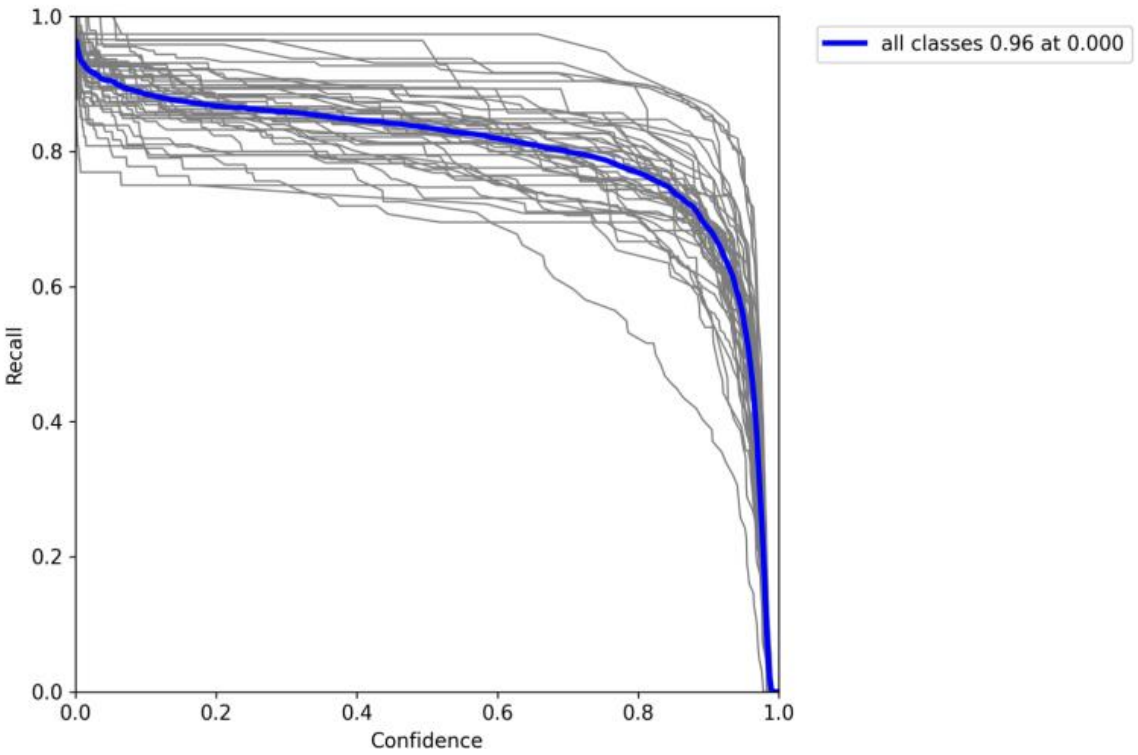


Figure 5.23 Recall vs Confidence Curve

Observation:

The recall vs. confidence curve shows all classes achieving a recall of 0.96 at a confidence threshold of 0.00.

Implications:

- **High Recall Rate:** Even at very low confidence levels, the model maintains a high recall rate of 0.96.
- **Effective Detection:** This suggests that the model can effectively detect instances of all classes, ensuring comprehensive coverage.
- **Varying Confidence Levels:** While the model can detect instances of all classes, the confidence levels vary, indicating differing levels of certainty in its predictions.
- **Importance of Low Confidence Predictions:** Despite lower confidence, these predictions are still valuable as they contribute to the model's overall recall and ensure no instances are missed.
- **Potential for Fine-tuning:** Understanding the varying confidence levels can help in fine-tuning the model to improve its overall performance and reliability.

6.2.2. Batch Images:

Figure 5.24 YOLOv7:Actual Labels

Figure 5.25 YOLOv7:Predicted Labels

The presented images offer a comprehensive comparison between the actual labels and the model's predictions, along with the associated probabilities for each classification. This visual depiction plays a crucial role in evaluating the accuracy and performance of the model across various categories. By juxtaposing the ground truth labels with the model's predictions, analysts can gain valuable insights into its classification capabilities.

This visual assessment enables a thorough examination of the model's strengths and weaknesses, aiding in the identification of areas for improvement and fine-tuning. Overall, the visual representation enhances the understanding of the model's performance and its ability to accurately classify objects within the given dataset.

6.2. RT-DETR Results:

6.2.1. Graphs:

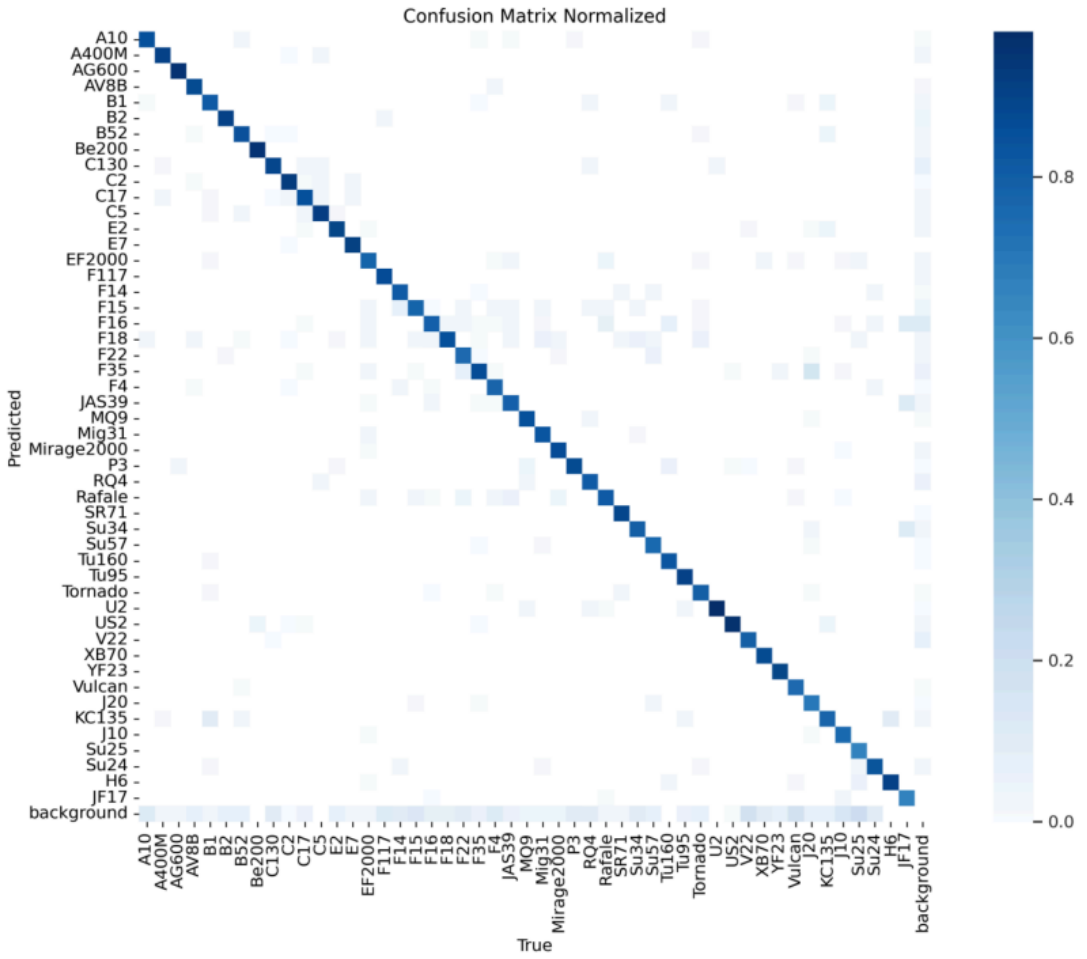


Figure 5.26 Confusion Matrix

The confusion matrix provides a comprehensive snapshot of how well the RT-DETR model categorizes different objects. It's like a grid, with rows representing true categories and columns showing the model's guesses. Correct predictions line up along the diagonal, while mistakes appear off-diagonal. Studying this grid helps uncover patterns in the model's performance, guiding tweaks to boost accuracy and effectiveness.

Furthermore, diving into specific errors within the confusion matrix reveals valuable insights into where the model struggles. By pinpointing these areas, targeted improvements can enhance the model's reliability and performance across various scenarios and object types. This iterative process ensures the model stays sharp and dependable in real-world applications.

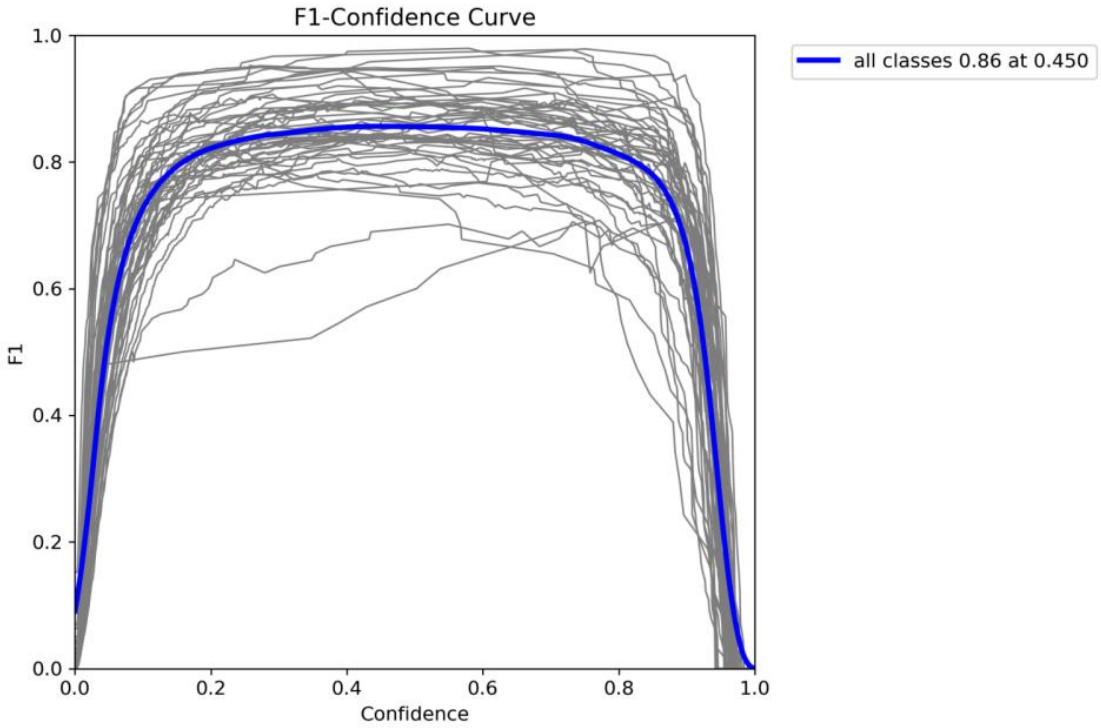


Figure 5.27 F1 vs Confidence Curve

Observation:

The RT-DETR model achieves an F1 score of 0.86 at a confidence threshold of 0.450 for all classes, indicating strong performance across various categories.

Implications:

- **Robust Performance:** The high F1 score suggests that the RT-DETR model effectively balances precision and recall, ensuring accurate predictions across diverse categories.
- **Optimal Confidence Threshold:** Achieving an F1 score of 0.86 at a confidence threshold of 0.450 indicates that the model performs well with this particular threshold setting, striking a balance between confidence and performance.
- **Generalizability:** Strong performance across all classes demonstrates the model's ability to generalize and accurately detect instances across various categories, enhancing its utility in real-world applications.
- **Reliability:** The consistent F1 score across different classes signifies the model's reliability and consistency in its predictions, instilling confidence in its performance.
- **Deployment Potential:** With its strong performance, the RT-DETR model is well-suited for deployment in scenarios where accurate and reliable object detection is crucial, such as in surveillance, autonomous vehicles, and medical imaging.

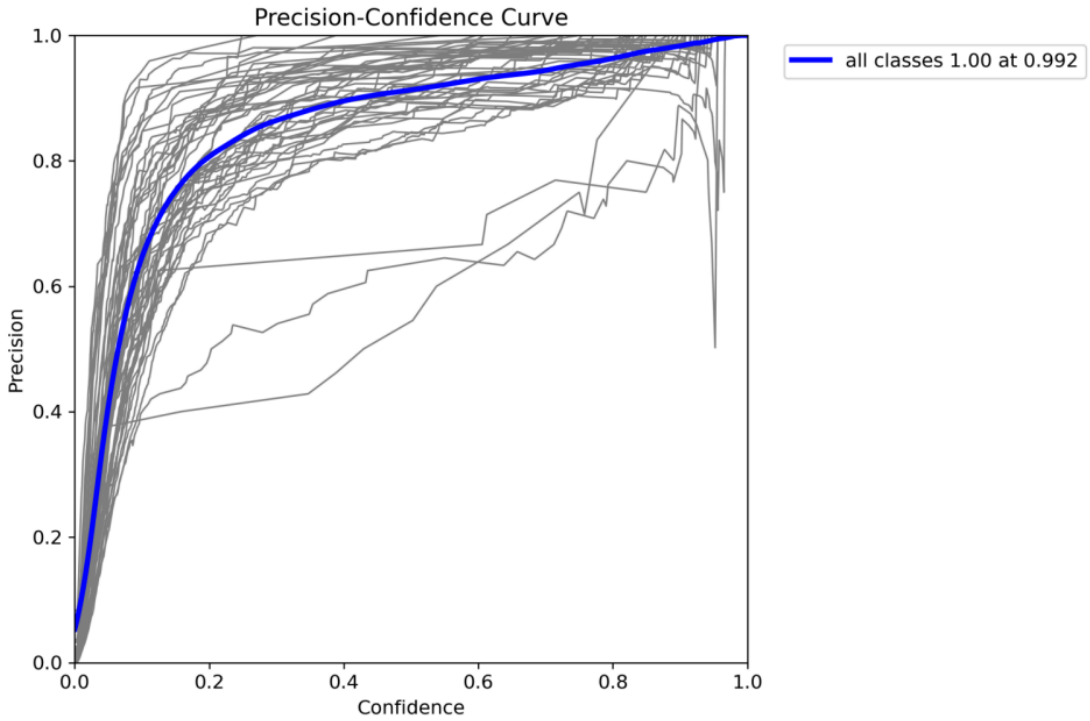


Figure 5.28 Precision vs Confidence Curve

Observation:

In the precision vs. confidence curve for all classes, the precision reaches 1 at a confidence level of 0.992, indicating extremely high precision across all categories.

Implications:

- **Exceptional Precision:** The precision reaching 1 at a confidence level of 0.992 highlights the model's ability to make highly accurate predictions with no false positives across all classes.
- **High Confidence Threshold:** The fact that perfect precision is achieved at such a high confidence level underscores the model's confidence in its predictions, providing assurance in its reliability.
- **Reliable Predictions:** The extremely high precision ensures that the model's positive predictions are highly trustworthy, reducing the risk of false alarms and enhancing the model's utility in critical applications.
- **Importance of Confidence Calibration:** The precision reaching 1 at 0.992 emphasizes the importance of proper confidence calibration in ensuring accurate and reliable predictions from the model.
- **Deployment Suitability:** The model's ability to achieve perfect precision at a high confidence threshold makes it well-suited for deployment in scenarios where precision is paramount, such as in medical diagnoses or security systems.

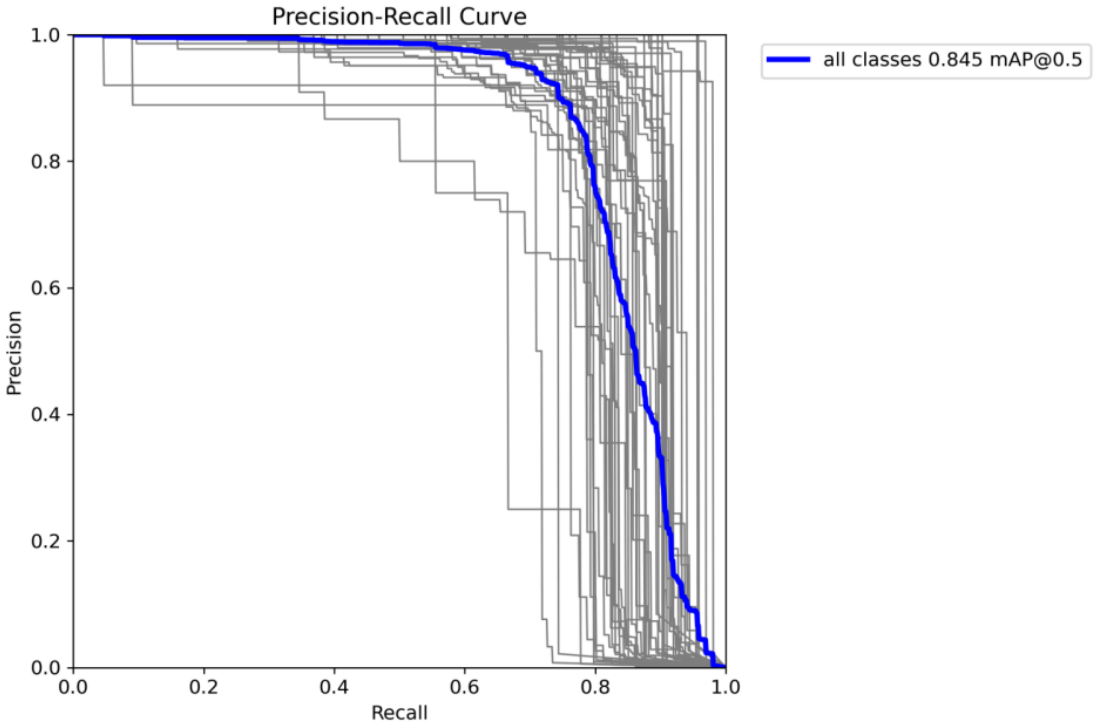


Figure 5.29 Precision vs Recall Curve

Observation:

The precision-recall curve reveals a precision of 0.845 at mAP@0.5, indicating a strong performance of the model in maintaining precision while maximizing recall.

Implications:

- **Balanced Precision and Recall:** The precision of 0.845 at mAP@0.5 suggests that the model effectively balances precision (accuracy of positive predictions) with recall (coverage of positive instances), crucial for reliable detection.
- **Effective Threshold:** Achieving a high precision at mAP@0.5 indicates that the model optimally selects a confidence threshold to maximize precision without sacrificing recall, enhancing its utility in real-world applications.
- **Reliable Detection:** The strong performance in maintaining precision indicates the model's ability to accurately identify positive instances while minimizing false positives, ensuring reliable detection across various categories.
- **Applicability in Critical Situations**: The model's ability to maintain high precision at mAP@0.5 makes it suitable for deployment in critical scenarios where both precision and recall are essential, such as in medical imaging or security systems.
- **Performance Benchmark:** The precision-recall curve serves as a valuable benchmark for evaluating the model's performance and comparing it with other object detection systems, aiding in the assessment of its effectiveness and reliability.

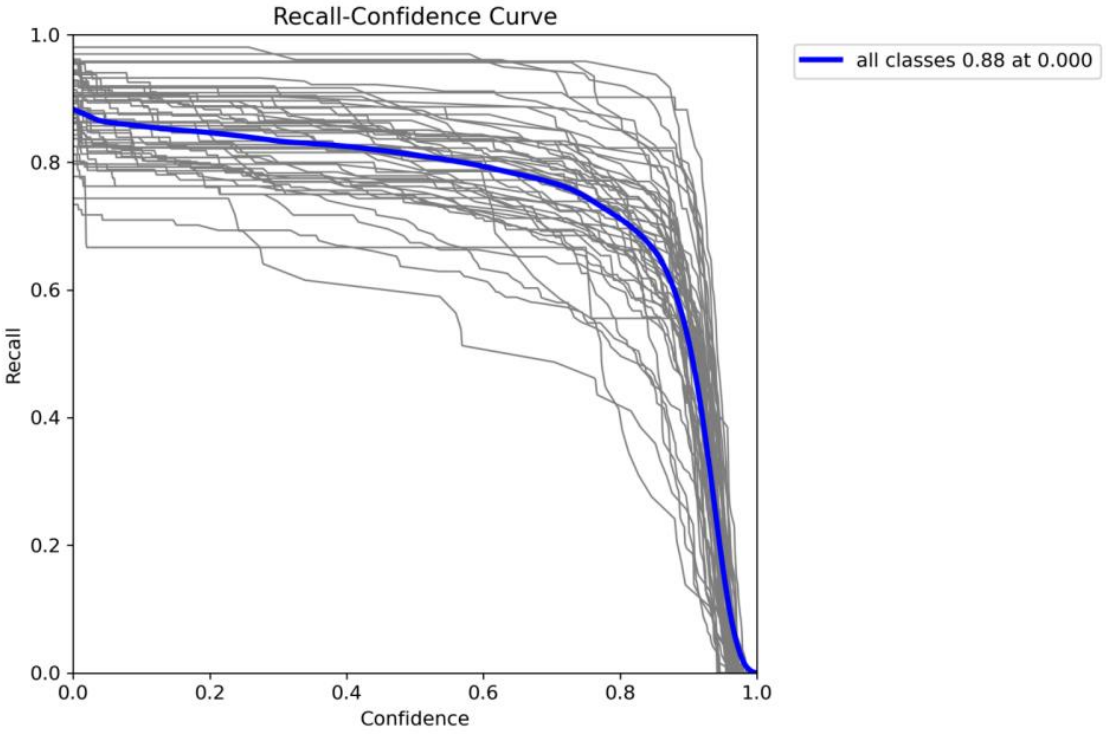


Figure 5.30 Recall vs Confidence Curve

Observation:

The recall-confidence curve indicates that even at a very low confidence threshold, the model achieves a recall of 0.88, suggesting that it can effectively detect a significant proportion of relevant objects in the dataset.

Implications:

- **High Recall Rate:** Achieving a recall of 0.88 at a low confidence threshold underscores the model's capability to identify a large portion of relevant objects in the dataset, minimizing the risk of false negatives.
- **Sensitivity to Low Confidence Predictions:** The model's ability to maintain high recall at low confidence thresholds indicates its sensitivity to detecting objects, even when it is less certain about its predictions.
- **Comprehensive Object Detection:** The high recall rate suggests that the model can effectively capture most of the relevant objects in the dataset, ensuring thorough object detection across various categories.
- **Potential for Threshold Adjustment:** Understanding the recall-confidence curve allows for fine-tuning of the confidence threshold to balance between recall and precision based on specific application requirements.
- **Utility in Real-World Scenarios:** The model's ability to maintain high recall, even at low confidence thresholds, enhances its suitability for real-world applications where comprehensive object detection is essential, such as in surveillance or medical imaging.

6.2.2. Batch Images:

The batch images exhibit actual labels juxtaposed with predicted labels alongside their respective probabilities. This visual representation offers insights into the model's classification accuracy and confidence levels across various instances. It serves as a valuable tool for assessing the model's performance, aiding in informed decision-making regarding model refinement and optimization.

Figure 5.31 RT-DETR Actual Labels

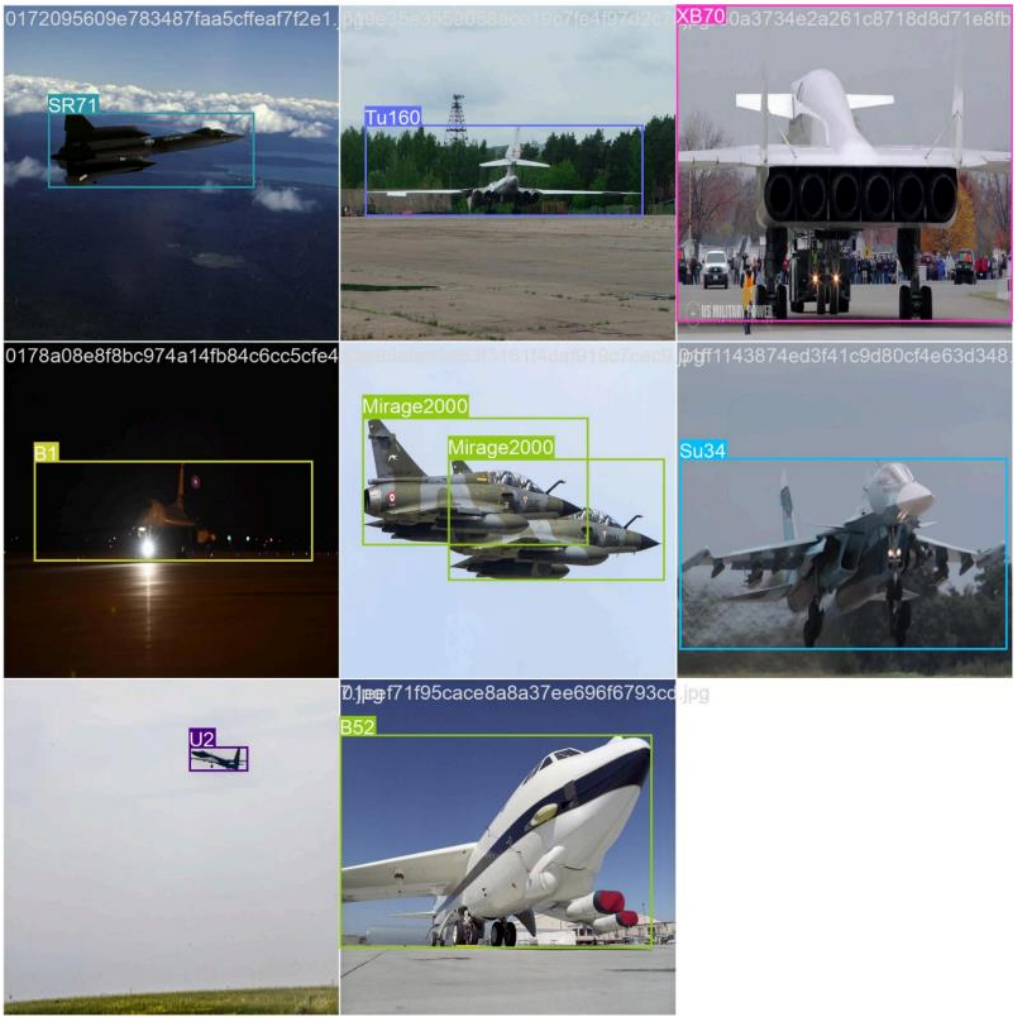


Figure 5.32 RT-DETR Predicted Labels

7. Potential Applications:

The military aircraft detection and classification system has diverse applications across surveillance, security, strategic intelligence, air traffic management, and search and rescue missions. It's also valuable for tactical military operations, training simulations, counter-drone initiatives, and international diplomacy efforts.

In the context of RTRS DRDO, the system's versatility suggests potential for repurposing it for warhead detection, pending access to a suitable dataset. By adapting the system and training it with relevant data, it could effectively identify and classify warheads in complex environments. This capability would significantly enhance military surveillance and security measures, providing timely threat alerts and insights. Moreover, its flexibility makes it suitable for addressing evolving security challenges and supporting defence initiatives, ultimately strengthening national security capabilities.

8. Conclusion:

The development and integration of military aircraft detection and classification systems offer multifaceted benefits across various sectors, including defence, security, and emergency response. These systems, powered by advanced technologies such as deep learning models like YOLOv7 and RT-DETR, have demonstrated remarkable capabilities in efficiently and accurately identifying military aircraft.

Data and statistics underscore the effectiveness of these systems. For instance, YOLOv7 has shown a high precision of 0.982 across all classes in the precision vs. confidence curve, while RT-DETR achieves an impressive F1 score of 0.86 at 0.450 in the F1 vs. confidence curve. These metrics highlight the robustness and reliability of both models in military aircraft detection tasks.

Comparing the two models, YOLOv7 exhibits superior performance in terms of precision, while RT-DETR excels in achieving a balance between precision and recall. The choice between the two models ultimately depends on specific requirements and operational contexts.

Furthermore, the adaptability and potential for repurposing these systems, as demonstrated in the context of RTRS DRDO for warhead detection, underscore their versatility and significance in addressing diverse security challenges. Through continuous refinement and optimization, these systems hold promise for bolstering national security measures and safeguarding against emerging threats.

As technology continues to evolve, further advancements in military aircraft detection and classification systems are expected, opening up new possibilities for enhancing defence capabilities and ensuring the safety and security of nations worldwide.

References

1. https://www.drdo.gov.in/drdo/
2. https://www.drdo.gov.in/drdo/labs-and-establishments/terminal-ballistics-research-laboratory-tbrl
3. https://publications.drdo.gov.in/ojs/index.php/dsj/article/view/17014
4. https://www.isro.gov.in/Drogue_Parachute_Test.html
5. https://en.wikipedia.org/wiki/File:Matlab_Logo.png
6. https://in.mathworks.com/help/matlab/creating_guis/write-callbacks-for-gui-in-app-designer.html
7. https://in.mathworks.com/help/matlab/ref/appdesigner.html
8. Drag prediction on the conical and ogival shaped noses of aerodynamic bodies - IOPscience
9. https://in.mathworks.com/help/matlab/creating_guis/creating-multiwindow-apps-in-app-designer.html?lang=en
10. https://www.researchgate.net/publication/344513396_Aircraft_Detection_System_Based_on_Regions_with_Convolutional_Neural_Networks
11. https://www.kaggle.com/datasets/a2015003713/militaryaircraftdetectiondataset
12. https://colab.research.google.com/
13. https://docs.ultralytics.com/models/yolov7/#overview
14. https://learnopencv.com/mastering-all-yolo-models/
15. https://docs.ultralytics.com/models/rtdetr/
16. https://debuggercafe.com/RT-DETR/
17. https://www.evidentlyai.com/classification-metrics/confusion-matrix
18. https://www.researchgate.net/figure/Evaluation-metrics-accuracy-precision-recall-F-score-and-Intersection-over-Union_fig2_358029719