PLAN-OF-ACTION REPORT

Date: 14th August, 2025
Project Urdhyuth, Phase II, Cohort 07

Intern Name: Aditi Ramakrishnan
Internship Title: Research Intern (Project Urdhyuth)
Organization: NMCAD Lab, Indian Institute of Science (IISc), Bengaluru
Team: Machine Learning Team
Subsystem Chosen: Autonomous Flight Management
Subdomain Chosen: Trajectory Optimization
Mentor: Piyusha Patil (ML Sub Team Lead)

1. Introduction

I am currently undertaking my internship as a **Research Intern** at the **NMCAD Lab, Indian Institute of Science (IISc)**, working within the **Machine Learning team**. My work is focused on the domain of **trajectory optimization** for **autonomous flight management systems** in **defense-oriented electric Vertical Take-Off and Landing (eVTOL) vehicles**.

This project aims to explore and develop advanced algorithms that can enable eVTOLs to autonomously plan and execute optimal flight paths while considering mission objectives, environmental constraints, and defense-specific operational requirements. The internship will involve both **theoretical research** and **practical implementation**, integrating concepts from flight dynamics, control systems, and machine learning to address real-world challenges in defense aviation technology.

2. Objectives

The internship aims to contribute to advanced research in autonomous flight management for defense-oriented eVTOL systems through the following objectives:

1. **Optimal Trajectory Planning:** Formulate and implement algorithms for computing energy-efficient, time-optimal, and mission-compliant flight trajectories under real-world operational constraints.

2. **Advanced Autonomous Flight Management:** Design and test adaptive control strategies capable of maintaining stability, accuracy, and responsiveness in highly dynamic and uncertain environments.

3. **Machine Learning–Driven Enhancements:** Leverage supervised, unsupervised, and reinforcement learning techniques to predict, adapt, and optimize trajectories in real-time, enabling intelligent decision-making.

4. **High-Fidelity Simulation & Verification:** Develop and utilize simulation frameworks to model realistic flight environments, enabling safe and repeatable validation of trajectory and control algorithms.

5. **Defense-Specific Mission Adaptation:** Integrate stealth, rapid response, and mission-specific operational parameters into trajectory and control logic to address unique defense application requirements.

3. Scope of Work

The scope of my internship involves contributing to the design, development, and evaluation of intelligent trajectory optimization and autonomous flight control algorithms tailored for defense-grade electric vertical take-off and landing (eVTOL) platforms. Specifically, the work will encompass:

1. **Dynamic Environment Modeling**
   - Developing high-fidelity simulation environments that incorporate real-world operational constraints such as restricted airspaces, variable weather patterns, GPS-denied conditions, and threat-prone zones.
   - Integration of sensor models (LiDAR, radar, EO/IR cameras) into the simulation for perception-driven trajectory planning.

2. **Machine Learning–Driven Trajectory Optimization**
   - Investigating reinforcement learning and model predictive control (MPC) hybrids for real-time path adaptation under uncertainty.
   - Incorporating aerodynamic, power, and actuator constraints unique to eVTOL flight profiles, especially in rapid ascent/descent and hover-to-cruise transitions.

3. **Threat Avoidance & Mission Adaptability**
   - Designing algorithms for evasive maneuver planning in response to dynamic threats (e.g., missile lock alerts, UAV interception patterns) while maintaining mission objectives.
   - Implementing contingency trajectory replanning for critical events like partial actuator failure, powertrain anomalies, or communication loss.

4. **Energy-Aware Control Strategies**
   - Modeling propulsion energy consumption under diverse maneuvers and altitudes.
   - Coupling trajectory optimization with energy budgeting to ensure mission completion and safe return to base without exceeding battery/energy constraints.

5. **Validation & Evaluation**
   - Testing algorithms in a digital twin environment of the defense eVTOL platform using both synthetic datasets and mission-specific flight logs.
   - Performance evaluation based on mission success rate, threat evasion efficiency, energy utilization, and computational latency.

4. Methodology / Approach

1. **Overview**

   Goal: design, train, and validate a threat-aware, energy- and time-efficient trajectory optimization and flight control stack for a defense eVTOL.

   Approach: combine physics-based optimal control (for guarantees), machine learning (for adaptation and speed), and high-fidelity simulation (for safety and scale) using ArduPilot SITL.

2. **System architecture**

   a) Perception and environment layer (offline, for planning):

   - Terrain, obstacles, airspace/restricted zones, and weather fields are fused into spatiotemporal "maps" the planner can query.
   - Threat/risk fields are represented as static or time-varying cost layers.

   b) Vehicle layer:

   - 6-DoF dynamics (or high-quality point-mass + turn-rate model for planning) with actuator, energy, and flight-envelope limits.
   - Energy model coupling rotor/prop power to battery state.

   c) Planner + controller layer:

   - Batch/offline global planner: direct optimal control (collocation) to find reference trajectories across the full mission.
   - Real-time local planner: receding-horizon MPC that tracks the reference, re-optimizes under disturbances and threat changes.
   - Learning accelerators: RL policy or supervised surrogate to warm-start or replace expensive solves when deadlines are tight

   d) Integration + execution layer: ArduPilot SITL executes waypoints/trajectories via MAVLink; a bridge script converts planner outputs to flight modes, mission items, or guided setpoints and logs full telemetry.

3. Datasets

A. Vehicle/eVTOL datasets:

- Mass, CG, inertia; rotor geometry/placement; thrust–RPM curves; efficiency maps.
- *Aerodynamics:* lift/drag/moment vs AoA and airspeed (wind-tunnel, CFD, or literature).
- *Limits:* Vmax, climb/descent rates, bank/tilt limits, actuator rate limits.
- *Energy:* battery capacity, discharge curve, internal resistance, power vs thrust.
- *Acquisition:* vendor datasheets or a reference eVTOL configuration; supplement with simplified parameterizations calibrated in SITL.

B. Environment/terrain/airspace

- *Terrain elevation:* DEM/DSM (e.g., SRTM 30 m, Copernicus 30 m).
- *Urban obstacles:* building footprints/heights (e.g., OpenStreetMap + local sources).
- *Airspace:* restricted areas, corridors, takeoff/landing zones (public AIP where permissible; otherwise create synthetic "defense" zones for research).
- *Weather:* gridded wind fields, density, temperature; steady layers for baseline, gust/turbulence models for robustness (e.g., reanalysis like ERA5 for typical profiles, then parametric gusts).
- *Communications/GNSS:* mark simulated GPS-denied or comm-degraded tiles (synthetic).
- *Preprocessing:* resample to a common grid; compute slope/roughness; rasterize no-fly polygons; build wind vector fields per altitude.

C. Threat/risk (research-safe, synthetic)

- Static radar sites with coverage lobes; moving patrols (UAV/UGV) with schedules.
- Detection probability surfaces P(det|position, altitude, time); lethality envelopes.
  Electronic-warfare zones (GPS/comm degraded).
- Sources: synthetic generation using physics-based detection models or simplified heuristics; will not use sensitive/real threat intel.

D. Mission/operations

- Start/goal coordinates, mandatory waypoints, time windows, payload mass.
- *Mission rules:* maximum exposure time, minimum standoff ranges, abort/RTB conditions.

E. Simulation/training logs

- *SITL telemetry:* position/velocity/attitude, control outputs, battery, wind, mode changes.
- *Planner traces:* states, controls, constraint multipliers, objective breakdowns.
- *Perturbation catalogs:* randomized seeds for wind, sensor noise, actuator faults.

F. Derived planning layers

- Energy cost map (Wh per meter per altitude cell).
- Risk map (expected detection/time-in-threat).
- Feasibility mask (forbidden cells by terrain, airspace, or kinematics).
- Time-indexed risk (threat patrols) for time-expanded planning.

4. Data engineering

- Standardize coordinates to ENU/NED local frames for planning; maintain WGS84 for SITL.
- Build a tiling scheme (e.g., 50–200 m grid) with multi-altitude layers; cache on disk.
- Interpolation services: bilinear for terrain/obstacles, spatial-temporal for wind/threats.
- Validation scripts: unit tests that check bounds, continuity, and CRS consistency.

5. Mathematical formulation

- *States x(t):* 3D position, velocity, attitude (or heading), battery SoC.
- *Controls u(t):* rotor thrust allocations or simplified speed/turn/climb commands.
- *Dynamics:* “next state = physics(x,u) + wind + noise”; obey actuator and envelope limits.
- *Objective J:* weighted sum of mission time, energy used, and cumulative risk exposure, plus smoothness terms.

- *Constraints:* start/finish conditions, obstacle/airspace avoidance, threat exposure caps, SoC ≥ reserve, and vehicle limits.

6. **Optimization strategy**

A. Global reference planning (offline)

- Direct collocation/pseudospectral (e.g., with CasADi/Pyomo) on a time grid.
- Warm-start from graph search (A*/ARA*) on a coarse 3D+time lattice using cost layers.
- Produce a dynamically feasible, threat-aware reference trajectory with time schedule.

B. Real-time local planning and control

- Nonlinear MPC tracks the reference; horizon 2–8 s, replanning at 5–20 Hz (sim).
- Constraints in MPC: control rates, bank/tilt, SoC budget, no-fly buffers.
- Disturbance estimation (EKF/UKF) feeds wind bias to MPC.

C. Learning accelerators (optional)

- Supervised surrogate: train a neural net to predict good warm-starts (states → controls) from the offline optimal solutions; used to cut solver time.
- RL policy (e.g., PPO/SAC) in SITL-like gym: reward = −time − energy − risk − constraint violations; deploy as fallback when solver fails or as a proposal generator.

7. **Training procedure:**

Step 1: Generate datasets

- Sample missions across terrain/wind/threat seeds; solve offline optimal control to create a corpus of (state, environment features) → (optimal control/trajectory) pairs.
- Log SITL flights using those trajectories under varied disturbances to get "imperfect execution" data.

Step 2: Supervised learning

- Inputs: local map patches (terrain slope, risk, wind vectors), current state, goal vector.
- Targets: optimal control increments or next-waypoint deltas from the offline solver.
- Train/val split by map regions and seeds; early stop by validation loss and constraint satisfaction on held-out maps.

Step 3: RL fine-tuning

- Environment: ArduPilot SITL or a high-fidelity proxy with identical dynamics limits.
- Curriculum: start with calm wind, sparse threats; progressively add gusts, moving patrols, GPS degradation.
- Safety shaping: hard penalties for constraint violations; terminate episodes on safety breaches.

Step 4: Controller tuning

- MPC cost weights tuned via Bayesian optimization on SITL runs (minimize tracking error, energy, violations).
- Fallback autopilot: PID/LQR gains tuned to ensure safe behavior on solver timeouts.

8. Simulation, integration, and evaluation

a. SITL loop

Planner (Python) → setpoints/mission items via MAVLink/DroneKit → ArduPilot SITL executes → logs to .bin/.tlog → analysis scripts compute metrics.

b. Key metrics

- Mission: success rate, total time, distance, SoC remaining.
- Safety: obstacle/airspace incursions (zero tolerated), min separation, threat exposure seconds and cumulative risk.
- Control quality: RMS tracking error, max attitude/tilt, control rate usage.
- Efficiency: Wh/km, peak power, thermal margins.
- Computation: mean and tail latency (p95/p99), success of warm-starts.

c. ***Robustness tests***
- Monte Carlo over wind fields, sensor noise, mass/payload changes, actuator faults (saturated rotor, stuck tilt), GPS dropouts.
- Domain shift: new maps/unseen cities; unseen threat layouts.

d. ***Ablations***
- Offline only vs MPC only vs MPC+surrogate vs MPC+RL.
- With/without risk layer; with/without wind.

9. Tooling and stack
- *Planning/learning*: Python, CasADi/Pyomo, NumPy, PyTorch/JAX.
- *Autopilot/sim:* ArduPilot SITL, MAVProxy, DroneKit/MAVSDK.
- *Data:* GDAL/rasterio for DEM/DSM; shapely/geopandas for airspace/obstacles.
- *Filters:* filterpy or custom EKF/UKF; cvxpy for convex subproblems.
- *Experiment ops:* Hydra/Weights & Biases (if allowed) for configs and logs; unit tests with pytest; Docker for reproducibility.

10. Validation and verification
- Unit tests for every dynamics/constraint function.
- Scenario testbench with seeded randomness; fixed baselines (straight-line, Dubins, energy-only).
- Cross-validator that rejects models failing any safety constraint on held-out maps.
- *Human-in-the-loop review:* visualize paths over terrain/threat maps; flight replays in SITL/GCS.

11. Ethical, legal, and safety stance
- All threat data are synthetic or generic physics-based; no sensitive or real-world targeting data.
- Results are for research; no operational evasion tactics.
- Emphasis on safety constraints and transparency of limitations.

12. Learning and reference plan

a. **Optimal Control**
- *Textbooks:* Bryson & Ho, *Applied Optimal Control*; Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*; Kirk, *Optimal Control Theory: An Introduction*.

- *Numerical methods:* CasADi documentation; GPOPS-II collocation method references; direct transcription techniques from Betts and Rao papers.
- *Research:* Recent UAV path planning papers using optimal control and trajectory optimization in IEEE Transactions on Aerospace and Electronic Systems.

b. Model Predictive Control (MPC)

- *Textbooks:* Borrelli, Bemporad & Morari, *Predictive Control for Linear and Hybrid Systems*; Mayne et al., *Model Predictive Control: Theory and Design*.
- *Online resources:* ACADO Toolkit tutorials; MATLAB MPC Toolbox documentation; UAV-specific MPC examples from PX4 and ArduPilot forums.
- *Research:* Nonlinear MPC for UAV guidance in *Control Engineering Practice* and *IFAC Journal of Systems and Control*.

c. Reinforcement Learning (RL) & Imitation Learning

- *Foundations:* Sutton & Barto, *Reinforcement Learning: An Introduction*; Levine et al., "Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems."
- *Practical guides:* OpenAI's *Spinning Up* tutorials; Ray RLlib documentation; Stable Baselines3 examples for UAV control.
- *Advanced:* Model-based RL survey papers (Moerland et al., *Machine Learning* journal); imitation learning methods for autonomous systems (e.g., DAgger, GAIL).

d. UAV / Autopilot Systems

- *Primary sources:* ArduPilot SITL & PX4 SITL documentation; MAVLink protocol specification; DroneKit-Python examples.
- *Control & Estimation:* PX4 EKF2 and EKF3 notes; Beard & McLain, *Small Unmanned Aircraft: Theory and Practice*.
- *Research:* Multi-rotor dynamics and control papers from the *Journal of Field Robotics* and *AIAA Guidance, Navigation, and Control Conference*.

e. GIS & Terrain Awareness

- *Libraries:* GDAL, rasterio, pyproj for coordinate systems; DEM (Digital Elevation Model) handling with USGS SRTM datasets.

- *Guides:* GDAL Cookbook; QGIS documentation for terrain preprocessing; coordinate transformation best practices.
- *Research:* UAV terrain-following strategies using DEMs and onboard sensing (IEEE Robotics and Automation Letters).

f. Cross-cutting References

- *Flight dynamics:* Etkin, *Dynamics of Atmospheric Flight*; Stevens & Lewis, *Aircraft Control and Simulation*.
- *Numerical optimization:* Nocedal & Wright, *Numerical Optimization* for algorithmic foundations.
- *Software engineering:* ROS2 best practices; GitHub repositories for UAV RL/MPC frameworks.

13. Risks and mitigations

- Solver instability/latency → warm-starts, simplified dynamics, surrogate.
- Data gaps (e.g., building heights) → conservative buffers, synthetic heights.
- Sim-to-real gap → parameter identification in SITL, domain randomization.
- Overfitting to maps → strict geographic hold-outs and unseen threat layouts.

5. Deliverables

A. Dataset Repository

- A well-structured and annotated collection of flight trajectory datasets aggregated from simulation platforms (ArduPilot SITL, PX4 SITL, JSBSim) and publicly available UAV/VTOL datasets.
- Metadata documentation describing the source, scenario type (urban, defense, evasive maneuvers, etc.), and preprocessing steps for each dataset.

B. Trajectory Optimization Model

- A trained machine learning model capable of generating optimized flight paths for defense eVTOL under multiple constraints (fuel efficiency, time-to-target, obstacle avoidance, stealth profile).
- Model documentation including architecture, training parameters, and evaluation metrics.

C. Simulation Demonstrations

- Fully functional simulation environments showing the model in action, both in controlled and randomized defense mission scenarios.
- Recorded video runs and performance summaries comparing baseline autopilot performance vs. optimized model performance.

D. Codebase & Tools

- Modular, well-commented Python code integrating dataset preprocessing, model training, evaluation, and SITL simulation pipelines.
- Scripts for dataset augmentation, parameter tuning, and trajectory visualization.

E. Technical Report

- A consolidated report detailing background research, methodology, system architecture, datasets used, training process, results, and limitations.
- Includes a future work section identifying opportunities for real-world integration and scaling.

F. Knowledge Artifacts

- Cheat-sheet documentation of simulation setup for ArduPilot, PX4, and JSBSim so future interns can replicate experiments.
- A curated list of academic papers, datasets, and code repositories that informed model design and training.

6. Timeline (one-month)

Week 1(Aug 14 – Aug 20): Orientation & Infrastructure Setup

- ***Objectives***:
  - Understand project scope in the context of the six-month goal.
  - Finalize simulation platforms (ArduPilot, PX4, Microsoft AirSim, Gazebo, RotorS).
  - Install required simulation and ML environments (ROS, Python, TensorFlow/PyTorch, OpenCV).
  - Identify relevant UAV datasets from public repositories (IEEE Dataport, Kaggle, DroneDeploy, Roboflow).
- ***Deliverable***:
  - Installed and validated simulation environments.
  - Curated list of dataset sources with metadata (type, format, usage rights).

Week 2 (Aug 21 – Aug 27): Dataset Collection & Preprocessing

- ***Objectives***:
  - Generate synthetic datasets using ArduPilot, PX4, and AirSim scenarios.
  - Download and organize real-world UAV datasets from identified repositories.
  - Perform dataset preprocessing (resizing, normalization, annotation formatting to COCO/PASCAL VOC).
  - Begin preliminary augmentation pipeline (weather conditions, object occlusion, lighting variations).
- ***Deliverable***:
  - Consolidated dataset repository (raw + processed + augmented data).
  - Documentation on dataset characteristics (classes, resolution, formats).

Week 3 (Aug 28 – Sept 3): Baseline Model Development

- ***Objectives***:
  - Implement baseline object detection models (YOLOv8, Faster R-CNN, SSD) for UAV image/video analysis.
  - Train baseline models on a small subset of the dataset to establish initial performance benchmarks.
  - Record performance metrics (mAP, precision, recall) for future comparison.
- ***Deliverable***:
  - Trained baseline model weights.

- Initial benchmarking report

<u>Week 4 (Sept 4 – Sept 14): Evaluation & Documentation</u>

- ***Objectives:***
    - Evaluate baseline model performance across both synthetic and real datasets.
    - Document strengths, weaknesses, and gaps in data coverage.
    - Prepare interim report including:
        - Summary of tools used.
        - Dataset acquisition and preparation steps.
        - Baseline results and key findings.
        - Recommendations for next phase (multi-modal data integration, domain adaptation).
- ***Deliverables:***
    - Interim project report (formatted for IISc review).
    - Presentation slides with visuals from simulations and model outputs.

7. Resources Required

- High-performance laptop/workstation with multi-core CPU, dedicated GPU, and ≥16GB RAM for simulation, data processing, and ML training
- Access to UAV simulation platforms: ArduPilot SITL, PX4 SITL, Gazebo, and compatible mission-planning tools (Mission Planner, QGroundControl)
- Open-source UAV datasets: ArduPilot flight logs, PX4 log archives, NASA UAV data repositories, and academic research datasets
- Relevant Python, ROS, and MAVLink libraries for data handling and integration
- Collaboration with Pranjal Talapatra (Avionics Team) for avionics domain expertise, dataset interpretation, and simulation validation