LEVERAGING LARGE LANGUAGE MODELS FOR ADAPTIVE AND SECURE UAV CONTROL

by

BALAKRISHNAN DHARMALINGAM

A dissertation submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE AND INFORMATICS

2026

Oakland University
Rochester, Michigan

Doctoral Advisory Committee:

Anyi Liu, Ph.D., Chair
Subra Ganesan, Ph.D., Co-Chair
Mohammad Wardat, Ph.D.
Xiaodong Deng, Ph.D.

To my wife, Rekha, and my daughters, Rishma and Rishika.

For your endless love, patience, and unwavering support.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisors, Dr. Anyi Liu and Dr. Subra Ganesan, for their invaluable guidance, mentorship, and continuous support throughout my doctoral studies. Their technical expertise, encouragement, and constructive feedback played a critical role in shaping this research and my development as a scholar.

I am also deeply thankful to my committee members, Dr. Xiaodong Deng and Dr. Mohammad Wardat. Their thoughtful comments, insightful suggestions, and rigorous evaluation significantly elevated the quality and academic merit of this dissertation.

I gratefully acknowledge my colleagues and collaborators—Ibrahim Odat, Sid Patil, Brett Piggott, Raj Mukherjee, and Guohuan Feng—for their technical discussions and hands-on contributions, particularly regarding the synthesis of the cyber-physical datasets and the fine-tuning of the Large Language Models. Their cooperation and dedication were essential to the successful realization of the Aero-LLM framework.

My deepest appreciation goes to my family for their unconditional love, patience, and unwavering support. I am especially grateful to my wife, Rekha Balakrishnan, for her constant encouragement, understanding, and the sacrifices she made throughout this demanding journey. Her belief in me provided immense strength during the most challenging times. I also thank my daughters, Rishma and Rishika, whose love, joy, and inspiration motivated me to persevere and strive for excellence. This achievement would simply not have been possible without them.

Finally, I would like to thank Oakland University and the Department of Computer Science and Engineering for providing the academic environment, computational resources, and opportunities necessary to pursue and complete this research.

Statement on AI Usage: During the preparation of this dissertation, generative AI and writing assistance tools (including Gemini, Grok, Grammarly, QuillBot, and OpenAI models) were utilized strictly to improve language clarity, spelling, grammar, and overall readability. Following the use of these tools, I carefully reviewed, edited, and verified all content. I take full and sole responsibility for the accuracy, originality, and integrity of this dissertation.

Balakrishnan Dharmalingam

ABSTRACT

LEVERAGING LARGE LANGUAGE MODELS FOR ADAPTIVE AND SECURE UAV CONTROL

by

Balakrishnan Dharmalingam

Adviser: Anyi Liu, Ph.D.

Unmanned Aerial Vehicles (UAV) are increasingly integral to mission-critical operations, spanning public safety, decentralized logistics, and autonomous surveillance. However, as UAV operations become highly autonomous within contested environments, their reliance on unencrypted and unauthenticated communication standards exposes a severe cyber-physical attack surface. Kinetic cyber-attacks targeting these vulnerabilities—such as sensor spoofing and Man-in-the-Middle (MITM) intrusions—can yield catastrophic physical consequences. While current research proposes securing UAV operations via deterministic heuristics or conventional deep learning models (e.g., Long Short Term Memory (LSTM), Autoencoders), these methods struggle to adapt to dynamic zero-day threats and operate as opaque "black boxes," lacking the semantic explainability required for safety-critical aviation.

Large Language Models (LLM) demonstrate unprecedented capabilities in contextual reasoning, zero-shot anomaly detection, and semantic explainability. Yet, their deployment in autonomous aviation is severely bottlenecked by the strict Size, Weight, and Power (SWaP) constraints of UAV microcontrollers, necessitating highly optimized, domain-specific training paradigms.

To resolve this dichotomy, this dissertation introduces Aero-LLM, a novel, distributed cyber-physical security framework that harnesses the cognitive reasoning of

LLMs to secure UAV operations. The proposed architecture is comprised of four primary subsystems: (1) A *Data Collector* that synthesizes the Heterogeneous Generative Dataset for UASes (HGDAVE) utilizing Digital Twins (Software-In-The-Loop (SITL)/Hardware-In-The-Loop (HITL)), firmware fuzzing, and AI-generated MITM attacks orchestrated by a generative adversary (Net-GPT); (2) A *Fine-Tuner* that leverages the Zero Redundancy Optimizer (ZeRO), Parameter-Efficient-Fine-Tuning (PEFT), and Reinforment Learing from Human Feedback (RLHF) to compress and align models for the aviation domain; (3) An *Online Controller* deploying specialized models across an Edge-Fog-Cloud distributed architecture for real-time threat mitigation; and (4) An *Offline Processor* that continuously calibrates thresholds to harden the system against emerging attack vectors.

This research contributes a unified cyber-physical data collection pipeline, a resource-efficient generative threat model, and a scalable LLM deployment strategy for embedded flight systems. Experimental evaluations demonstrate the profound efficacy of the proposed architectures. The Net-GPT offensive module achieved an average payload synthesis accuracy of 95.30%, successfully mimicking protocol-compliant network traffic. Defensively, the Aero-LLM anomaly detection framework achieved an accuracy of 92.60%, with a precision of 92.70%, a recall of 92.06%, and an F1-score of 90.55%. While the generative inference introduces a latency overhead of approximately 850ms compared to lightweight neural networks, the framework provides unparalleled semantic explainability and cyber-resilience, establishing a scalable foundation for secure, intelligent autonomous UAV operations.

TABLE OF CONTENTS

TABLE OF CONTENTS—Continued

TABLE OF CONTENTS—Continued

TABLE OF CONTENTS—Continued

LIST OF TABLES

LIST OF TABLES—Continued

LIST OF FIGURES

LIST OF ABBREVIATIONS

C2	Command and Control
CNN	Convolutional Neural Networks
CPS	Cyber-Physical Systems
DoS	Denial of Service
EKF	Extended Kalman Filter
GCS	Ground Control Stations
GPS	Global Positioning Systems
HGDAVE	Heterogeneous Generative Dataset for UASes
HITL	Hardware-In-The-Loop
IMU	Inertial Measurement Unit
LiDAR	Light Detection and Ranging
LLM	Large Language Models
LSTM	Long Short Term Memory
MAVLink	Micro Air Vehicle Communication Link
MEMS	Micro-Electro-Mechanical Systems
MITM	Man-in-the-Middle
MPC	Model Predictive Control
PEFT	Parameter-Efficient-Fine-Tuning
PID	Proportional-Integral-Derivative
PWM	Pulse Width Modulation

QGC	QGroundControl
RC	Radio Control
RLHF	Reinforment Learing from Human Feedback
RNN	Recurrent Neural Network
RTL	Return-to-Launch
RTOS	Real-Time Operating System
SFT	Supervised Fine-Tuning
SITL	Software-In-The-Loop
SWaP	Size, Weight, and Power
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UAV	Unmanned Aerial Vehicles
UDP	User Datagram Protocol
VTOL	Vertical Takeoff and Landing
ZeRO	Zero Redundancy Optimizer

CHAPTER ONE

INTRODUCTION AND BACKGROUND

1.1 Introduction

An Unmanned Aerial Vehicle (UAV), or Unmanned Aircraft System (UAS)—commonly referred to as a drone—is an aircraft operated without a human pilot, crew, or passengers on board, functioning either through remote teleoperation or autonomous control [1]. Driven by rapid evolutionary advancements in both hardware and software, UAVs have become highly advantageous across a multitude of sectors, including military, industrial, and commercial domains [2, 3]. Modern UAVs integrate sophisticated sensor payloads, such as Inertial Measurement Units (IMUs), Global Positioning Systems (GPS), magnetometers, barometers, and visual sensors like Light Detection and Ranging (LiDAR) and optical flow cameras. The continuous fusion of these data streams enables the UAV to accurately estimate its state (position, velocity, and orientation) and navigate complex three-dimensional environments. Consequently, their capacity to operate in a levitated state, maneuver through intricate 3D spatial dimensions, and access hazardous or otherwise inaccessible locations has made them the subject of extensive academic research and industrial application.

1.1.1 UAV Platform

UAV architectures are primarily categorized into multi-rotors, fixed-wing aircraft, single-rotor helicopters, and Vertical Takeoff and Landing (VTOL) platforms. Multi-rotor configurations offer exceptional maneuverability, precision hovering, and vertical takeoff capabilities, rendering them highly effective for applications such as search and rescue operations and last-mile logistics. Conversely, fixed-wing aircraft provide superior aerodynamic efficiency, facilitating longer operational ranges and higher flight speeds

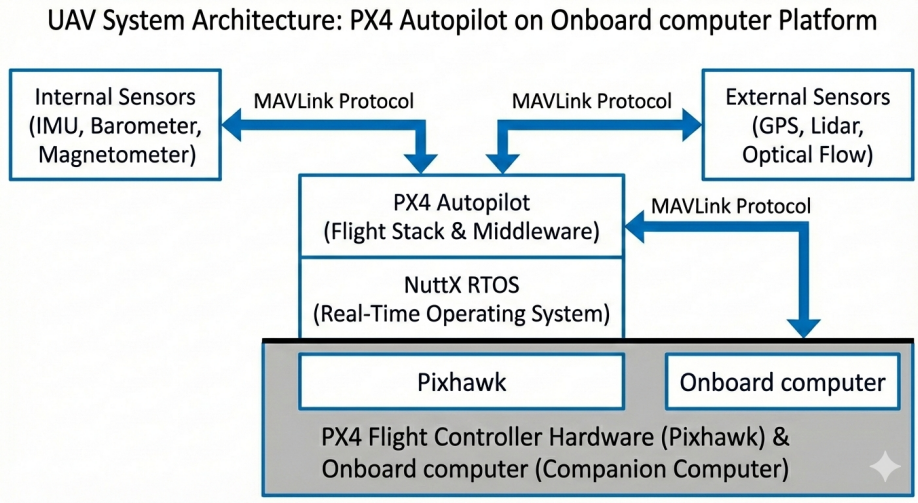


Figure 1.1: UAV Platform

suitable for extensive ground surveying. VTOL hybrid systems bridge these paradigms by combining the hovering proficiency of multi-rotors with the forward-flight efficiency of fixed-wing models. Single-rotor helicopters provide robust payload capacities but require highly complex mechanical linkages, increasing their operational and maintenance difficulty.

The autopilot serves as the central computational hub of a UAV. It comprises a Flight Management System (commonly referred to as the Flight Stack) executing on a Real-Time Operating System (RTOS) embedded within a flight controller hardware module, such as the widely adopted PX4 [4]. PX4 is an open-source flight stack predominantly running on Apache NuttX[5], a highly efficient, POSIX-compliant RTOS tailored for resource-constrained embedded systems. The PX4 flight controller supports a diverse array of vehicle airframes, offering robust flight modes and comprehensive safety protocols. Ultimately, the complete UAV platform ecosystem consists of the PX4 Autopilot, Ground Control Stations (GCS), Pixhawk-standard hardware, and the MAVSDK library [6], which facilitates the integration of companion computers, optical

sensors, and other peripherals via the Micro Air Vehicle Communication Link (MAVLink) communication protocol as illustrated in Figure 1.1.

1.1.2 UAV Applications

UAVs facilitate a broad spectrum of applications across both military and civilian domains. In military contexts, they are extensively utilized for reconnaissance, surveillance, and precision tactical operations. Conversely, in the civilian sector, UAVs are deployed for disaster management, land surveying, agricultural monitoring, structural inspection, and logistics [2, 3, 7]. A categorization of these applications is illustrated in Figure 1.2. The integration of advanced autonomy, sophisticated sensor fusion, artificial intelligence, and swarm technology has significantly expanded their operational utility, rendering them indispensable for numerous Industry 4.0 applications. Autonomous UAVs leverage real-time data processing and control algorithms to navigate dynamically during missions. By utilizing advanced sensors—such as LiDAR, high-resolution optical cameras, and thermal imaging—UAVs execute complex mapping and monitoring tasks. Furthermore, the application of machine learning algorithms enhances object recognition and anomaly detection, capabilities that are particularly critical in precision agriculture and disaster response. The advent of swarm technology further amplifies these capabilities, enabling multiple UAVs to collaborate seamlessly and execute large-scale, distributed tasks with increased efficiency [8].

1.1.2.1 Commercial and Civilian Applications

Driven by continuous technological advancements, the commercial and civilian utility of UAVs has expanded into several key areas:

Figure 1.2: UAV Applications

1. **Aerial Photography and Environmental Monitoring:** UAVs equipped with high-definition cameras are utilized extensively in real estate, media production, and environmental conservation to capture high-fidelity spatial data and imagery.

2. **Precision Agriculture:** Agricultural operations employ UAVs equipped with multi-spectral sensors to monitor crop health, optimize irrigation strategies, and detect pest infestations. Additionally, UAVs outfitted with dispersal mechanisms are used for targeted pesticide application, thereby mitigating pest damage and optimizing crop yields.

3. **Infrastructure Inspection:** UAVs enable the safe and efficient inspection of critical infrastructure, including power lines, bridges, wind turbines, pipelines, and high-rise structures, significantly reducing the risks associated with manual inspections.

4. **Search and Rescue (SAR):** In emergency scenarios, UAVs can rapidly survey vast and difficult-to-navigate geographic areas, providing crucial situational awareness and delivering essential supplies to isolated populations.

5. **Logistics and Package Delivery:** Prominent logistics entities are actively piloting UAV networks designed for the rapid, autonomous delivery of consumer goods, medical supplies, and critical payloads.

6. **Space Exploration:** Given their capacity for autonomous flight in extreme altitudes and environments, advanced UAV designs are increasingly being researched for planetary exploration and extraterrestrial atmospheric studies.

1.2 Logical Architecture of UAV Firmware: PX4 Deep Dive

To effectively understand how anomalies manifest and propagate within a UAV system, it is imperative to dissect the logical architecture of the flight control stack. This research utilizes the PX4 autopilot—an open-source, industry-standard flight control framework—as the foundational reference platform. PX4 is engineered with a modular, layered architecture specifically designed to decouple low-level hardware drivers from high-level flight control logic.

1.2.1 UAV Operation

The operational lifecycle of a UAV begins with mission planning, wherein flight trajectories and objectives are defined and uploaded via a GCS. The GCS transmits these commands to the UAV over wireless communication channels, typically utilizing RF telemetry or Wi-Fi. During execution, the UAV continuously processes these commands and streams real-time state and sensor data back to the GCS for monitoring. Upon mission completion, the UAV executes automated Return-to-Launch (RTL) and landing sequences.

This bidirectional communication is facilitated by the MAVLink [9], a highly efficient, lightweight messaging protocol standard in the drone industry. MAVLink governs the communication between the GCS and the UAV, as well as the internal communication between the onboard flight controller and external peripherals, including GPS modules and optical sensors. The protocol operates primarily on publish-subscribe and point-to-point design patterns; continuous sensor data streams are published to specific message topics for subscribed components, while critical configuration parameters are transmitted point-to-point to ensure reliable delivery.

1.2.2 The μORB Middleware

The architectural backbone of the PX4 flight stack is the Micro Object Request Broker (μORB). This middleware functions as an asynchronous, publish-subscribe messaging bus, enabling diverse software modules—such as hardware drivers, state estimators, and flight controllers—to communicate efficiently without tight programmatic coupling.

- **Publishers:** Components such as sensors and state estimators publish data streams to specific topics, such as `sensor_combined` (yielding raw IMU measurements) or `vehicle_attitude` (providing the estimated spatial orientation).
- **Subscribers:** Downstream control modules subscribe to these topics, utilizing the ingested data to compute precise motor outputs and actuation signals necessary for maintaining stable flight.

1.2.3 The Flight Stack Hierarchy

The software architecture of the PX4 flight stack is organized into a rigid, deterministic hierarchy of interdependent modules. This layered approach ensures that

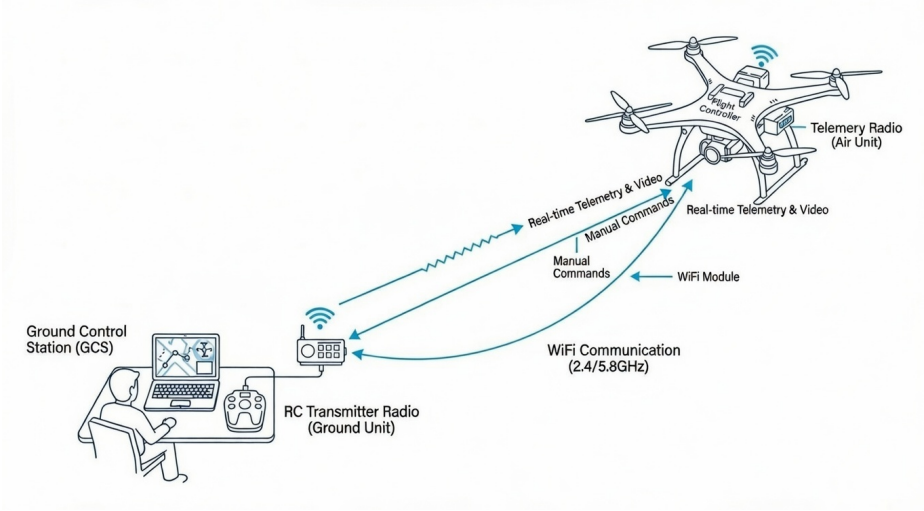


Figure 1.3: PX4 Autopilot UAV System and GCS

low-level hardware interactions are cleanly abstracted from the high-level decision-making and autonomous navigation processes. The primary components of this hierarchy include:

1. **Drivers:** Low-level software abstractions interfacing directly with physical sensors (e.g., gyroscopes, accelerometers, magnetometers, and barometers) utilizing standard hardware communication protocols such as SPI, I2C, and UART.

2. **Estimators (EKF2):** The state estimation module, predominantly leveraging an Extended Kalman Filter (EKF), fuses asynchronous and noisy raw sensor data to probabilistically estimate the vehicle's state. This encompasses its spatial orientation (represented via quaternions, q), velocity vector (v), and global position (p).

3. **Commander:** A central finite state machine responsible for managing the vehicle's operational state. It handles safety-critical pre-flight checks, arming and disarming

sequences, and transitions between discrete flight modes (e.g., from manual *Stabilized* flight, to *Position Hold*, to autonomous *Mission* execution).

4. **Navigator:** The mission execution engine that translates pre-planned autonomous missions or dynamic waypoints into continuous, executable setpoints (desired position and velocity vectors) for the underlying control loops.

5. **Controllers:** A cascaded, multi-loop control architecture (typically utilizing Proportional-Integral-Derivative (PID) or PID variants) that computes the necessary error corrections. It converts the kinematic setpoints generated by the Navigator into low-level motor mixing and precise actuator commands required to stabilize the airframe.

1.2.4 Fundamental UAV Operational Sequence

The flight operational sequence commences with the "arming" of the UAV, a requisite state transition that energizes the onboard motors and actuators, bringing the system out of its safe standby mode. This procedure is typically executed via a GCS interface, such as QGroundControl (QGC), or a companion computer. Upon successful initialization and completion of pre-flight checks by the Commander module, a takeoff command is transmitted, prompting the propulsion system to generate sufficient thrust to overcome gravity.

During the active flight phase, the operator or autonomous agent governs the aircraft's trajectory and orientation by issuing specific flight control inputs—specifically defining the roll, pitch, and yaw axes—through either the GCS, a manual Radio Control (RC) transmitter, or the Navigator module. The operation concludes with a landing command, which executes a controlled, sensor-guided descent and the subsequent disarming of the propulsion system to ensure ground safety.

1.3 Flight Control and Sensor Fusion

The stabilization and navigation of a multicopter represent a classic problem in controlling underactuated, highly coupled non-linear systems. To effectively manage this complexity, the flight control architecture employs a cascaded, multi-loop structure. In this paradigm, outer control loops manage slower translational dynamics (e.g., global position and velocity), while inner loops regulate the highly responsive rotational dynamics (e.g., spatial attitude and angular body rates).

1.3.1 The Control Pipeline

The hierarchical control pipeline processes navigational goals sequentially, translating high-level spatial objectives into low-level mechanical actuation. The standard control flow cascades through the following functional domains:

$$\text{Position Error} \rightarrow \text{Velocity Setpoint} \rightarrow \text{Attitude Setpoint} \rightarrow \text{Rate Setpoint} \rightarrow \text{Actuator Output} \tag{1.1}$$

Mathematically, let p_d denote the desired spatial position and p_e denote the estimated current position. The outer position controller computes a positional error vector, which the velocity controller utilizes to generate a desired velocity setpoint v_d via a Proportional (P) gain configuration:

$$v_d = K_p(p_d - p_e) \tag{1.2}$$

This desired velocity is translated into an attitude setpoint, which cascades down to the inner rate controller. The rate controller employs a PID algorithm to minimize the rotational error $e(t)$, defined as the difference between the desired angular rate ω_{des} and the measured angular rate ω_{meas} (i.e., $e(t) = \omega_{des} - \omega_{meas}$). The control law is formulated

as:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \tag{1.3}$$

where $u(t)$ represents the desired torque command vector. This computed control effort is subsequently fed into a control allocation matrix (commonly referred to as the motor mixer), which maps the virtual aerodynamic torques and thrust demands into specific actuation signals for individual motor hardware.

1.3.2 Autonomous Mission Execution and Trajectory Generation

An autonomous mission comprises a deterministic sequence of `MAVLink` micro-commands uploaded from the Ground Control Station to the UAV's onboard memory. These missions are fundamentally structured around waypoints, which are defined utilizing global geographic coordinates (Latitude, Longitude, and Altitude in the WGS84 coordinate system[10].

Rather than navigating abruptly from point to point, the flight stack's Navigator module performs real-time spatial interpolation between these discrete waypoints. By computing a dynamically feasible path that respects the vehicle's kinematic constraints, the Navigator generates a continuous, smooth trajectory setpoint. The underlying position and velocity controllers then actively track this generated trajectory to ensure precise and efficient mission execution.

1.4 State of the Art in Anomaly Detection

Modern UAVs are complex Cyber-Physical Systems (CPS) that rely heavily on continuous sensor data ingestion and wireless communication protocols, such as MAVLink, to maintain stable flight. However, this heavy reliance on external signals and distributed architecture introduces significant attack surfaces. UAVs are increasingly vulnerable to sophisticated cyber-attacks, including False Data Injection (FDI), GPS

spoofing, sensor blinding, and malicious command interception. Detecting these anomalies—particularly those induced by stealthy cyber-attacks that mimic natural environmental disturbances—remains a critical challenge. Current approaches to anomaly detection in UAVs generally fall into three distinct architectural categories.

1.4.1 Deterministic Limit Checking and Rule-Based Failsafes

The most fundamental form of anomaly detection involves deterministic limit checking, which is currently the standard mechanism implemented in commercial flight controllers like PX4 (commonly referred to as "failsafes"). This approach relies on pre-defined operational boundaries for critical flight parameters. For instance, if the telemetry data indicates that the `battery_voltage` has dropped below a hardcoded threshold (e.g., 10.5V), the flight controller deterministically triggers a safety protocol, such as a RTL or emergency landing sequence.

The primary advantage of rule-based limit checking is its extremely low computational latency and absolute interpretability. However, its critical limitation lies in its rigidity. Deterministic rules cannot detect subtle, progressive system degradations or context-dependent anomalies. For example, high accelerometer variance (vibration) might be perfectly normal during turbulent, high-wind hovering, but indicative of a mechanical failure or a structural cyber-attack during calm forward flight. Consequently, limit checking is highly susceptible to evasion by intelligent cyber-attacks that keep manipulated parameters just within the acceptable threshold bounds.

1.4.2 Analytical Model-Based Methods

To address the limitations of static rules, model-based methods leverage mathematical representations of the UAV's physical dynamics and kinematics. These architectures utilize a state observer—most commonly an EKF—to predict the vehicle's

expected subsequent state based on current control inputs and historical data. Anomaly detection is performed by evaluating the "residual," which is the quantifiable divergence between the model's predicted state and the actual physical observations obtained from the sensors.

Mathematically, the residual vector r_t at time t is defined as:

$$r_t = y_t - \hat{y}_t \tag{1.4}$$

where y_t represents the actual measured sensor observation and $\hat{y}_t$ is the predicted state generated by the analytical model. If the magnitude of the residual r_t exceeds a dynamically calculated statistical threshold (e.g., a 3σ deviation), an anomaly is flagged. While highly effective for detecting physical hardware faults, model-based methods struggle significantly with unmodeled environmental dynamics and are often inadequate for detecting cyber-attacks that simultaneously spoof both the control inputs and the sensor feedback, thereby artificially minimizing the residual.

1.4.3 Data-Driven and Deep Learning Paradigms

Given the complexity of modeling highly non-linear aerodynamics and complex cyber-threats, recent literature has pivoted toward data-driven Machine Learning (ML) and Deep Learning (DL) methodologies. These approaches learn the nominal flight envelope directly from vast datasets of historical telemetry rather than relying on explicit physics equations. Prominent architectures include:

- **Autoencoders (AE) [11]:** Neural networks trained to compress and subsequently reconstruct normal flight telemetry. During inference, if the network encounters anomalous data (such as an injected cyber-attack signature), it fails to reconstruct the input accurately. A high reconstruction error directly implies the presence of an anomaly.

- **Long Short-Term Memory (LSTM) Networks [12]:** A specialized class of Recurrent Neural Networks (RNNs) [13] highly adept at capturing temporal dependencies in time-series data. LSTMs are utilized to forecast future sensor values based on sequential historical context, flagging deviations from the forecasted trajectory.

- **One-Class Support Vector Machines (OC-SVM):** A semi-supervised learning algorithm that maps nominal multidimensional flight data into a high-dimensional feature space, establishing a strict hyper-boundary around the normal operational envelope to isolate outliers.

While traditional deep learning models demonstrate high accuracy in identifying statistical outliers, they inherently lack semantic understanding and deep contextual reasoning. They often generate high false-positive rates during aggressive, yet normal, flight maneuvers. Furthermore, standard ML architectures struggle to generalize against novel, zero-day cyber-attacks that differ substantially from their training distribution. This critical gap necessitates the exploration of LLM, which possess unprecedented zero-shot reasoning capabilities, vast contextual memory, and the ability to correlate complex, multi-modal semantic relationships to detect sophisticated cyber-physical intrusions.

1.5 Taxonomy of Cyber-Physical Attacks on UAVs

To establish a comprehensive threat landscape for the proposed anomaly detection framework, it is imperative to categorize the diverse attack vectors through which a UAV can be compromised. Unlike traditional, purely digital Information Technology (IT) systems, UAVs operate as CPS and are therefore susceptible to both network-layer intrusions and physical-layer manipulations. The following taxonomy categorizes these threats based on the specific layer of the UAV's logical architecture they target.

1.5.1 Perception Layer Attacks (Sensor Spoofing)

Perception layer attacks specifically target the UAV's exteroceptive and proprioceptive sensory systems. By directly corrupting the raw observational inputs (y_t) prior to their ingestion by the state estimator (e.g., the Extended Kalman Filter), attackers can manipulate the vehicle's perceived reality, bypassing traditional logical failsafes.

- **GPS/GNSS Spoofing:** GPS/GNSS spoofing involves the transmission of counterfeit satellite signals possessing higher signal-to-noise ratios than legitimate GPS/GNSS broadcasts. Utilizing Software Defined Radios (SDR), adversaries can exploit the "Capture Effect," forcing the UAV's receiver to lock onto the malicious signal. Consequently, the attacker can hijack the drone's navigational trajectory while the flight controller's state estimation falsely converges on a stationary hover. This is illustrated in Figure 1.4. Detecting such sophisticated spoofing necessitates multi-modal cross-validation; for instance, a LLM can semantically correlate conflicting data streams, identifying that a high GPS-derived velocity coupled with zero proper acceleration from the IMU directly violates fundamental kinematic principles.

- **Optical Flow and LiDAR Spoofing:** Ranging and optical sensors are highly susceptible to saturation or structured light attacks via high-intensity strobes or targeted lasers. By injecting false proximity measurements, adversaries can trigger unwarranted collision avoidance protocols. This forces the UAV to execute aggressive, uncommanded evasive maneuvers or sudden altitude drops, potentially resulting in catastrophic physical crashes.

- **IMU and Magnetometer Interference:** The onboard magnetometer, which is critical for accurate heading estimation, is highly vulnerable to localized magnetic

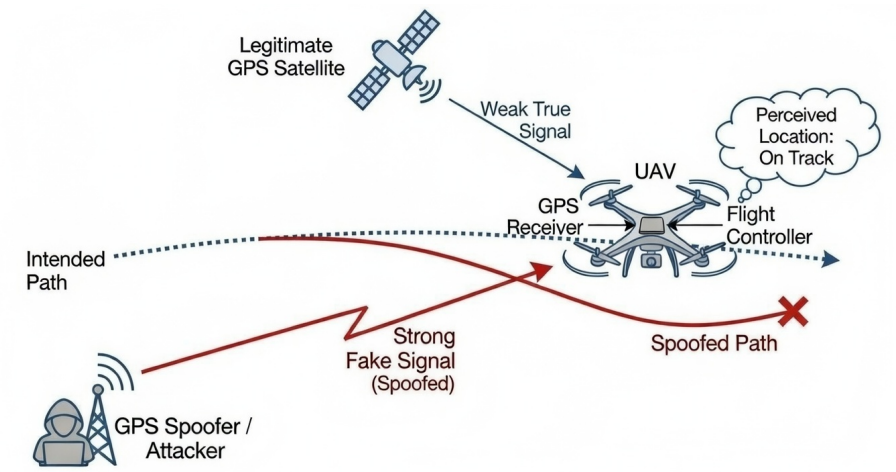


Figure 1.4: Architecture of a GPS Spoofing Attack on UAV Navigation.

interference. The deliberate introduction of anomalous magnetic fields skews the directional vector, confusing the flight controller's yaw control. This conflict between the magnetometer and the gyroscopes often induces an unstable, diverging spiral trajectory—colloquially recognized in the aerospace domain as the "toilet bowl effect"—ultimately leading to a complete loss of spatial orientation.

1.5.2 Network Layer Attacks (Communication Links)

Network layer attacks compromise the Command and Control (C2) communication links, which typically operate over standard Radio Frequency (RF) telemetry bands (e.g., 915 MHz or 433 MHz) or wireless local area networks (e.g., 2.4 GHz or 5.8 GHz Wi-Fi). These attacks aim to sever, degrade, or manipulate the data exchange between the GCS and the UAV.

- **Radio Frequency (RF) Jamming (Denial of Service):** RF Jamming acts as a localized Denial of Service (DoS) attack by saturating the operational frequency spectrum with high-power, broadband noise. While this typically severs the C2 link

and triggers deterministic failsafe mechanisms—such as an automated RTL—sophisticated adversaries can deploy intelligent, intermittent jamming. By repeatedly triggering and releasing these failsafes, attackers can exhaust the UAV's energy reserves or effectively enforce geographic area denial.

- **MITM Attacks:** In a MITM attack, the adversary surreptitiously intercepts the bidirectional telemetry stream. Because the attacker sits logically between the GCS and the UAV, they can relay benign status updates back to the GCS—maintaining the illusion of normal operation for the human operator—while simultaneously injecting malicious navigational setpoints or uncommanded actuation signals directly into the UAV's flight controller as shown in Figure 1.5.

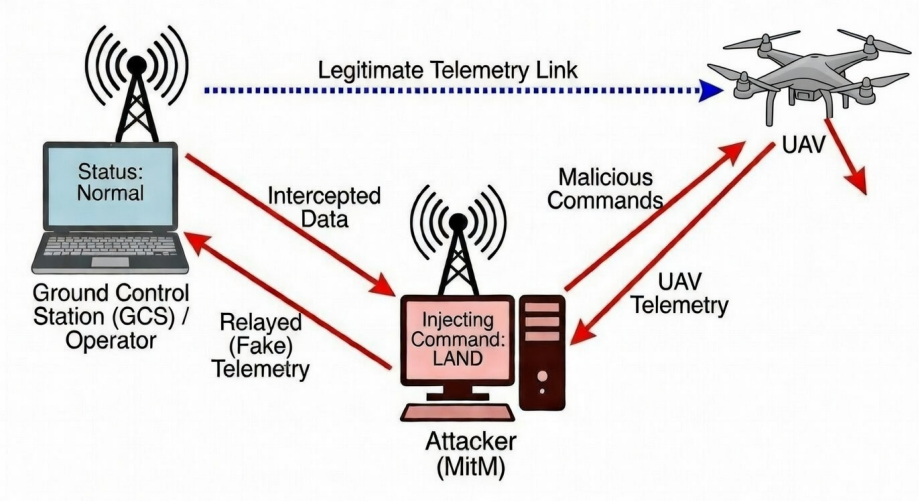


Figure 1.5: MITM Attack on UAV Telemetry Link.

1.5.3 Protocol Layer Attacks (MAVLink)

As the PX4 flight stack relies heavily on the MAVLink protocol for both internal and external communication, protocol-specific vulnerabilities represent a critical attack surface. MAVLink, particularly in its legacy or unencrypted configurations, often lacks robust cryptographic signing by default, rendering it highly susceptible to exploitation.

- **Command Injection:** Command Injection involves the fabrication and transmission of valid MAVLink packets (e.g., `MAV_CMD_NAV_LAND` or `MAV_CMD_DO_MOTOR_TEST`) into the telemetry stream. Without rigorous message authentication protocols in place, the flight controller processes these forged commands as legitimate GCS directives, completely bypassing operator intent and assuming rogue behavioral states.
- **Heartbeat Storming:** Heartbeat Storming exploits the fundamental publish-subscribe architecture of the MAVLink protocol. By flooding the communication interface with a high volume of redundant `HEARTBEAT` messages, an attacker can induce severe resource exhaustion. This artificially overloads the onboard microcontroller, leading to high processing latency, buffer overflows, and ultimately the total starvation of the high-frequency inner flight control loops.

1.5.4 Physical and Side-Channel Attacks

Beyond traditional cyber-intrusions, UAVs are susceptible to sophisticated physical and side-channel attacks targeting their micro-mechanical hardware.

- **Acoustic Gyroscope Resonance:** By directing focused, high-amplitude acoustic waves at the specific resonant frequency of the onboard Micro-Electro-Mechanical Systems (MEMS) gyroscope, attackers can induce severe structural vibrations. This resonance translates into wildly inaccurate angular rate measurements (y_t). As the

inner rate controller attempts to compensate for these phantom rotational forces, it generates asymmetric motor outputs, typically resulting in an uncontrollable loss of vehicle attitude and catastrophic structural failure.

1.5.5 Summary of Attack Vectors and Observable Anomalies

Table 1.1 summarizes the primary cyber-physical attack vectors discussed in this section, mapping each threat to its targeted hardware component and the corresponding multi-modal anomaly signature. This mapping serves as the foundational threat model for the proposed LLM anomaly detection framework, which relies on these observable kinematic and systemic contradictions to identify malicious intrusions.

1.6 Limitations of Current Anomaly Detection Methodologies

Despite significant advancements in data-driven and deep learning techniques, current anomaly detection solutions face critical theoretical and practical limitations that hinder the deployment of fully autonomous, self-healing UAVs:

1.6.1 Lack of Interpretability (The "Black Box" Paradigm)

Traditional deep learning models, such as LSTMs and Convolutional Neural Networkss (CNNs), excel at pattern recognition but suffer from severe opacity. They typically provide a strictly binary or probabilistic classification output (e.g., Normal vs. Anomalous) but lack semantic explainability. These architectures are fundamentally incapable of articulating the underlying causal mechanisms of an anomaly—leaving the human operator or the autonomous recovery system unable to distinguish whether the abnormal behavior stems from physical propeller icing, a degraded sensor bias, or external wind shear. This lack of root-cause analysis significantly delays necessary mitigation responses.

Category	Specific Attack	Target Component	Observable Anomaly Signature
Perception	GPS Spoofing	GNSS Receiver	Kinematic divergence between GPS-derived velocity and IMU acceleration integration (physics violation).
Physical	Acoustic Resonance	MEMS Gyroscope	Anomalous high-frequency resonance isolated to specific gyroscope axes; conflicting attitude estimates.
Network	RF Jamming	Telemetry Radio	Abrupt degradation of Received Signal Strength Indicator (RSSI) and communication link quality.
Protocol	Command Injection	MAVLink Parser	Uncommanded flight mode transitions or unauthorized state changes lacking operator origin.
Physical	Power Source Compromise	Power Module	Non-linear voltage degradation strictly inconsistent with current throttle and thrust demands.

Table 1.1: Taxonomy of Cyber-Physical Attacks and Corresponding Anomaly Signatures.

1.6.2 Vulnerability to Zero-Day and Out-of-Distribution Anomalies

Supervised anomaly detection models implicitly assume that future data will be drawn from the same distribution as the training data. Consequently, they require rigorously labeled datasets for every anticipated failure mode. However, creating an exhaustively labeled dataset that encompasses the infinite manifold of potential cyber-physical attacks and environmental failures is fundamentally intractable. While unsupervised models (such as Autoencoders) attempt to mitigate this by modeling only

nominal flight behaviors and flagging novelty, they frequently fail to generalize. Unsupervised systems consistently struggle to mathematically distinguish between a truly dangerous, Out-of-Distribution (OOD) cyber-attack and a safe, albeit aggressive, unmodeled evasion maneuver.

1.6.3 Absence of Semantic and Contextual Awareness

Perhaps the most significant limitation of conventional detection algorithms is their lack of contextual reasoning. Traditional detectors treat sensor data as abstract, isolated time-series numbers. They suffer from a severe semantic disconnect; they process telemetry without any high-level awareness of the vehicle's current mission phase. For example, a sudden, rapid decrease in altitude constitutes a critical anomaly if the UAV is operating in a stable "Loiter" or "Hold" mode; however, that identical kinematic data is perfectly nominal if the flight controller has transitioned into an automated "Land" sequence. Current mathematical models lack the cognitive architecture required to integrate this high-level operational context with low-level sensor streams dynamically.

This critical gap—the inability of traditional architectures to perform contextual, linguistic reasoning over multi-modal numerical data—serves as the primary motivation for investigating the integration of Large Language Models (LLMs) into the UAV anomaly detection pipeline.

1.7 Large Language Models in UAVs

Despite the ubiquitous deployment of UAVs across diverse operational domains, their autonomous capabilities remain constrained by several critical challenges as explained in Section 1.6:

- **Adaptive Control:** Traditional adaptive control mechanisms are often insufficient to maintain optimal flight stability and mission performance amidst highly dynamic

environmental conditions, such as severe weather perturbations or the introduction of unmapped dynamic obstacles.

- **Cyber-security:** Commercial UAV firmware frequently lacks robust intrinsic cybersecurity protocols, rendering these platforms highly susceptible to sophisticated network intrusions, localized RF jamming, and sensor spoofing attacks that can severely compromise navigational integrity.
- **Resource Constraints:** Onboard embedded systems are fundamentally limited by strict SWaP constraints. This inherent deficit in computational bandwidth restricts the real-time processing, inference, and analysis of high-dimensional sensor telemetry at the tactical edge.

Historically, UAV flight stabilization and navigation have relied heavily on classical control methodologies, such as PID controllers, which maintain stability by linearly adjusting actuator outputs based on immediate sensor feedback. More advanced paradigms, such as Model Predictive Control (MPC) [14], optimize control actions over a defined prediction horizon [15]. While these tactics are highly effective in structured or semi-structured environments, their efficacy degrades significantly in complex, unpredictable scenarios characterized by adverse weather, fluctuating payload dynamics, and evolving mission requirements. Furthermore, calibrating these controllers demands extensive system identification, massive volumes of empirical flight data, and rigid mathematical modeling, which often fails to generalize smoothly across different UAV airframes or dynamic mission profiles [16, 17].

To address these systemic deficiencies, Large Language Models (LLMs) offer a transformative, paradigm-shifting solution. LLMs introduce unprecedented capabilities in zero-shot in-context learning, deep semantic reasoning, and adaptive decision-making, allowing them to infer concealed patterns and adapt to novel cyber-physical tasks with

minimal task-specific fine-tuning [18]. Modern foundational LLMs—including GPT-5 [19], DeepSeek-R1 [20], Grok 4 [21], Genie 3 [22], and Llama-4 [23]—are constructed on sophisticated, highly scalable transformer architectures. These encompass encoder-only models (e.g., BERT [24]) optimized for dense state embeddings, decoder-only models (e.g., GPT [25]) designed for autoregressive generative tasks, and versatile encoder-decoder frameworks (e.g., T5 [26]) capable of translating complex multi-modal inputs into structured operational outputs.

The integration of LLMs into UAV ecosystems represents a critical evolution from rigid, rule-based automation to true cognitive autonomy. This architecture bridges high-level semantic decision-making with low-level traditional sensor-actuator control loops. By ingesting and synthesizing a heterogeneous array of data streams—ranging from raw kinematic sensor inputs and MAVLink telemetry to natural language operator commands and geofencing regulatory constraints—LLMs can generate highly contextualized, actionable insights [19, 23]. For instance, an onboard LLM agent can rapidly detect subtle cyber-physical anomalies by correlating real-time state deviations against its vast internalized knowledge base, or autonomously re-route flight trajectories in response to sudden weather degradation. This facilitates the deployment of advanced applications such as context-aware path optimization, zero-day threat detection, and resilient mission planning [17]. Ultimately, LLMs enhance the robustness of UAVs by dynamically revising behavioral rules and recursively learning from environmental feedback [27], thereby establishing a self-healing, fully adaptive flight control framework.

1.7.1 LLM Architecture

Fundamentally, a LLM operates as an autoregressive probabilistic framework built upon the Transformer architecture. These models are pre-trained on massive, highly diverse corpora of textual data to predict the conditional probability of a subsequent token,

t_i, given an extended sequence of preceding tokens $(t_1, t_2, \ldots, t_{i-1})$. This predictive distribution is mathematically formalized as:

$$P(t_i \mid t_1, \ldots, t_{i-1}) = \text{Softmax}(Wh_i + b) \tag{1.5}$$

where h_i represents the contextualized hidden state vector generated by the Transformer layers, W denotes the learned weight matrix mapping to the vocabulary space, and b is the bias term. While originally engineered for Natural Language Processing (NLP) tasks, the underlying mechanism—the Self-Attention mathematical operator—is inherently domain-agnostic. Self-Attention allows the neural network to dynamically weigh the importance of different segments within an input sequence regardless of their positional distance, thereby enabling the highly effective capture of long-range dependencies and complex multivariate correlations.

1.7.2 Applicability to Anomaly Detection and Control

A foundational premise of this dissertation is that a UAV flight mission can be conceptually and mathematically modeled as a continuous semantic narrative. Within this paradigm:

- **Syntax** corresponds to the rigid laws of physics, kinematic constraints, and flight dynamics (e.g., gravitational acceleration, aerodynamic drag, and inertial momentum).
- **Semantics** represents the high-level operational intent and the current state machine phase of the vehicle (e.g., executing a precision loiter for structural inspection versus executing an aggressive evasion maneuver).

Traditional data-driven anomaly detectors successfully learn the syntactic correlations of the flight envelope but remain completely agnostic to the underlying

semantics. LLMs, conversely, offer a unique and disruptive advantage: zero-shot contextual reasoning. By embedding raw sensor telemetry into a structured prompt alongside discrete system logs and state machine flags, an LLM can transcend basic threshold flagging to perform diagnostic reasoning. For example, an LLM agent can deduce: *"The Z-axis acceleration variance is highly anomalous, yet the GPS-derived velocity remains near zero; this contradiction suggests localized mechanical vibration—such as a propeller imbalance—rather than external wind turbulence."*

A primary challenge in deploying LLMs for cyber-physical systems is the inherent modality gap: foundational LLMs natively process discrete text tokens, whereas UAV sensors generate continuous, high-frequency numerical time-series data. To bridge this modality gap, this research leverages specialized temporal architectures, specifically Time-LLM [28] and TimesNet [29]. Time-LLM acts as a reprogramming framework that transforms continuous time-series patches into text-aligned prototypes, allowing the frozen LLM backbone to reason over temporal data, while TimesNet provides robust 2D-variation modeling to capture complex intra-period and inter-period flight dynamics.

1.7.3 UAV Datasets and Empirical Limitations

The successful integration of LLM-based anomaly detection frameworks necessitates extensive fine-tuning and validation using vast quantities of multi-modal flight telemetry. While publicly available repositories exist, they often lack the breadth required for comprehensive cyber-security research. For instance, the widely referenced UA Dataset [30] is constrained to a limited threat taxonomy, predominantly featuring GPS spoofing and localized MAVLink jamming. Furthermore, the anomalies within such datasets are often generated via simplistic SITL simulator hooks or basic ping flooding, which fail to accurately encapsulate the complex thermodynamic and aerodynamic realities of advanced hardware-level intrusions across diverse airframes.

To address this empirical gap and capture a comprehensive taxonomy of cyber-physical intrusions (encompassing perception, network, protocol, and physical side-channel attacks), existing datasets are fundamentally insufficient. Consequently, this research proposes a novel, high-fidelity data collection pipeline (detailed extensively in Section 3.2.2). This custom architecture is specifically designed to harvest synchronized, multi-modal telemetry across a diverse spectrum of benign operational regimes and sophisticated, zero-day attack scenarios.

1.8 Research Problem

While traditional UAV control architectures, including PID and MPC systems, demonstrate high efficacy within structured environments, they exhibit fundamental limitations in scalability and adaptability. These deterministic controllers are inherently constrained when operating in highly dynamic or unpredictable scenarios, such as severe weather perturbations, rapidly evolving mission parameters, and sophisticated cyber-physical threats. Furthermore, the optimization of these controllers necessitates extensive empirical flight data, rigid mathematical modeling, and intensive manual parameter tuning, rendering them difficult to generalize across diverse UAV platforms and dynamic operational envelopes.

LLMs present a disruptive paradigm shift, enabling true cognitive autonomy through their innate capabilities in zero-shot contextual reasoning, adaptive decision-making, and multi-modal data synthesis. Unlike traditional controllers, LLMs can dynamically ingest multi-dimensional sensor telemetry, synthesize environmental context, and recalibrate flight parameters or mission tactics in real time. Moreover, their advanced semantic pattern recognition capabilities offer unprecedented potential for identifying complex cyber threats and mitigating anomalies, thereby significantly enhancing UAV resilience and operational security.

Despite their theoretical applicability across various domains, the practical integration of LLMs into resource-constrained UAV systems—as outlined in Section 1.6—poses significant architectural, computational, and security challenges that remain systematically unaddressed in current literature. Therefore, this research investigates the efficacy and feasibility of integrating foundational LLMs directly into the UAV flight control and monitoring stack. By designing a distributed, LLM-driven UAV framework, this dissertation seeks to overcome the limitations of classical deterministic architectures and establish a highly scalable, intelligent, and cyber-resilient autonomous flight system.

1.9 Research Objectives

To bridge the critical gap between traditional UAV control limitations and the cognitive capabilities offered by LLMs, this dissertation pursues the following specific research objectives:

- **Objective 1:** Synthesize a comprehensive, multi-modal training dataset leveraging diverse simulation architectures, including HITL and SITL environments. This will incorporate high-fidelity benign flight telemetry alongside malicious data generated via dynamic penetration testing, protocol fuzzing, and adversarial exploitation.
- **Objective 2:** Investigate the efficacy of utilizing LLMs as generative adversarial agents capable of autonomously executing MITM attacks and synthesizing sophisticated, context-aware spoofed telemetry for robust dataset generation and vulnerability assessment.
- **Objective 3:** Develop and optimize task-specific temporal LLMs by applying advanced tuning methodologies—including PEFT, Supervised Fine-Tuning (SFT), and RLHF—to execute critical, real-time UAV tasks such as multi-modal anomaly detection and autonomous path optimization.

- **Objective 4:** Systematically evaluate foundational LLM architectures for their applicability to temporal forecasting and anomaly classification, establishing selection criteria based on parameter efficiency, predictive accuracy, and strict onboard SWaP constraints.

- **Objective 5:** Benchmark the proposed distributed LLM framework against established baseline control algorithms, rigorously evaluating performance across metrics including Precision, Recall, F1-Score, inference latency, and overall systemic cyber-resilience.

1.10 Contributions

The primary intellectual and technical contributions of this dissertation are manifold:

- **Synthesis of the HGDAVE Dataset:** Construction of the Heterogeneous Generative Dataset for Unmanned Autonomous Systems, providing a novel, high-fidelity, and multi-modal benchmark dataset specifically tailored for Connected and Autonomous Vehicles (CAVs) and UAVs [31].

- **Adversarial LLM Agent Design:** Design and implementation of an LLM-Empowered Man-in-the-Middle framework capable of semantically parsing UAV network protocols and executing intelligent, context-aware cyber-attacks.

- **Task-Specific LLM Optimization:** Methodological advancements in fine-tuning foundational models utilizing SFT and RLHF to specifically address the unique temporal and kinetic requirements of UAV anomaly detection and autonomous decision-making [32].

- **Development of Aero-LLM:** The architectural formulation and deployment of a novel, distributed LLM framework (Aero-LLM) engineered to balance high-level cognitive processing with the stringent real-time computational constraints of tactical UAV missions [32].
- **Advanced Cyber-Threat Mitigation:** Empirical demonstration of LLM-based anomaly detection frameworks significantly outperforming traditional architectures in identifying and mitigating complex cyber-intrusions, including localized spoofing, protocol manipulation, and network jamming.
- **Comprehensive Benchmarking:** Rigorous empirical evaluation of the Aero-LLM framework within both simulated physics environments and real-world hardware deployments, establishing new performance benchmarks for accuracy, inference latency, and cyber-resilience.

1.11 Organization of the Dissertation

This chapter establishes the foundational context, motivation, and objectives for the proposed research. The remainder of this dissertation delves into the technical design, architectural implementation, and rigorous empirical evaluation of the proposed framework [32], and is organized as follows: Chapter 2 presents a comprehensive Literature Review of UAV control systems, anomaly detection methodologies, and Large Language Models. Chapter 3 details the core Methodology, outlining the dataset synthesis, architectural design of Aero-LLM, and the specific fine-tuning procedures employed. Chapter 4 presents the Experimental Results and comparative performance benchmarking. Finally, Chapter 5 provides the Conclusion, summarizing the research findings, addressing current limitations, and outlining pathways for future work.

CHAPTER TWO

LITERATURE REVIEW

2.1 Knowledge Domains and Taxonomy

Instead of a chronological survey, the existing literature is systematically taxonomized into five core thematic domains. This structured categorization facilitates a rigorous comparative analysis of methodological paradigms and precisely highlights the intersecting gaps in current research. The foundational knowledge domains are defined as follows:

1. **Flight Control and Trajectory Planning:** This domain encompasses a spectrum of navigation methodologies, ranging from classical deterministic feedback loops (e.g., PID, MPC) to advanced learning-based control schemes and cooperative multi-UAV path planning [33, 14, 34, **?**]. The primary focus of these works is the kinematic, aerodynamic, and spatial command of the autonomous vehicle.

2. **Time-Series Forecasting and Anomaly Detection:** This category examines predictive modeling for sequential telemetry data, spanning foundational statistical baselines, state-space models, and sophisticated deep learning architectures optimized for temporal forecasting [29, 28]. The emphasis is on learning nominal sensor distributions to identify critical kinematic deviations and systemic anomalies.

3. **Large Language Models for Robotics and Autonomy:** Literature in this emerging domain explores the application of foundational language models for high-level robotic reasoning, autonomous task planning, and the semantic interpretation of time-stamped kinematic data [35, 28].

4. **Distributed Edge-Fog-Cloud Architectures:** This domain investigates the optimal spatial distribution of computational workloads. It addresses the critical latency and bandwidth trade-offs associated with executing resource-intensive AI inference directly on resource-constrained edge hardware (the UAV) versus offloading tasks to proximate fog nodes or centralized cloud servers [36].

5. **Cybersecurity and Cyber-Physical Resilience:** The final theme catalogues the diverse threat landscape—including adversarial attacks on sensors, communication radios, and onboard machine learning modules—while surveying contemporary algorithmic countermeasures and intrusion detection systems [37, 32].

A high-level synthesis of these representative domains is provided in Table 2.1. A critical analysis of this taxonomy reveals a pervasive structural limitation within the current state-of-the-art: very few proposed architectures successfully intersect more than one of these isolated domains.

2.2 Flight Control and Planning: Advances and Limitations

2.2.1 Deterministic and Robust Control Architectures

PID controllers and Linear-Quadratic Regulators (LQR) remain the ubiquitous foundational architectures in commercial and academic UAV flight control. These deterministic methodologies provide rigorous mathematical provability, ease of implementation, and clear pathways to aerospace certification [33]. Advanced paradigms, such as MPC, extend these capabilities by formulating trajectory tracking as an active optimization problem. MPC models future plant states over a receding horizon to compute optimal control inputs while strictly enforcing kinematic and environmental constraints [14]. However, the receding horizon optimization fundamental to MPC imposes a severe computational burden, frequently exceeding the SWaP constraints of embedded UAV

microcontrollers. Consequently, engineers are forced to tune MPC parameters conservatively or implement robust margins, which inevitably blunts dynamic performance. Furthermore, a critical vulnerability in these classical schemes is their reliance on the assumption of pristine communication links and nominal sensor data, largely ignoring the potential for adversarial data injection or malicious exploitation.

2.2.2 Adaptive and Learning-Based Controllers

Deterministic controllers inevitably exhibit performance degradation when subjected to non-stationary aerodynamic disturbances or highly non-linear flight regimes. To address this, the literature has increasingly pivoted toward adaptive and intelligent control mechanisms. Gain-scheduling techniques dynamically adjust control parameters based on the current flight envelope, while fuzzy logic controllers blend heuristic rules to navigate non-linearities. More recently, Reinforcement Learning (RL) has been heavily leveraged to autonomously discover complex control policies through high-volume simulation [39]. Each of these approaches introduces distinct trade-offs. Most notably, learning-based models frequently succumb to the "sim-to-real" reality gap; policies trained within idealized physics engines exhibit catastrophic degradation when deployed on physical hardware due to unmodeled aerodynamic perturbations, stochastic sensor noise, and actuator saturation [55]. While safety layers, such as Control Barrier Functions (CBFs), can provide mathematical guarantees for bounding unsafe behaviors, they introduce further computational overhead. A notable gap in current research remains the empirical validation of these resource-intensive, adaptive algorithms on the heavily constrained, low-power embedded processors utilized in tactical UAV platforms.

Critical Assessment A pervasive limitation in contemporary control literature is the isolated evaluation paradigm. Algorithmic advancements are overwhelmingly assessed in

sterile, closed-loop configurations. Evaluations rarely encompass the holistic operational environment, largely ignoring the impact of stochastic communication channels, real-time energy consumption bounds, or systemic resilience against adversarial perturbations (e.g., GPS/sensor spoofing). System certification and security considerations are frequently treated as reactive afterthoughts rather than foundational design constraints.

2.3 Time-Series Forecasting and Temporal Anomaly Detection

2.3.1 Statistical and Hybrid Baselines

The interpretation and predictive modeling of UAV telemetry constitute a foundational challenge in autonomous systems. Classical linear methodologies, including Autoregressive Integrated Moving Average (ARIMA) models, Kalman filters, and switching state-space models, provide high interpretability and strong performance within tightly constrained, linear operational envelopes. However, the kinematic reality of UAV flight is inherently non-stationary; high-frequency state transitions, such as aggressive evasive maneuvers or abrupt aerodynamic perturbations (e.g., wind shear), rapidly violate the linear assumptions of these classical models, leading to significant predictive divergence.

2.3.2 Deep Temporal Architectures

To address these non-linearities, recent literature has increasingly adopted deep learning architectures for temporal modeling. Transformer-based architectures, notably Temporal Fusion Transformers (TFT) and Informer, have demonstrated exceptional efficacy in capturing long-range temporal dependencies through self-attention mechanisms. Furthermore, advanced frameworks such as PatchTST and TimesNet leverage patching mechanisms and 2D-variation modeling to efficiently process local semantic segments, significantly reducing computational complexity [29, 28]. While these

models achieve state-of-the-art performance on standardized benchmark datasets (e.g., electrical grid demand or macro-meteorological forecasting), these domains exhibit vastly slower temporal dynamics compared to the high-frequency control loops of UAVs. Crucially, there is a pronounced scarcity of empirical validation demonstrating the robustness of these deep temporal models against adversarial signal perturbations, stochastic packet loss, or active cyber-intrusions.

2.3.3 Anomaly Detection in Telemetry

The identification of cyber-physical anomalies represents a parallel, highly coupled challenge to temporal forecasting. Unsupervised deep learning paradigms, specifically Autoencoders (AE), Variational Autoencoders (VAE), and Generative Adversarial Networks (GAN), are frequently employed to model the nominal flight manifold, classifying deviations by quantifying reconstruction error. Architectures such as OmniAnomaly and USAD exemplify this approach, demonstrating high sensitivity to statistical outliers. However, these detectors suffer from severe out-of-distribution (OOD) fragility; they are hypersensitive to benign operational shifts and highly vulnerable to intelligent adversarial manipulation. Consequently, the literature exhibits a critical dearth of studies quantifying false-positive rates under realistic environmental stressors, such as turbulent wind gusts, localized sensor drift, or intermittent telemetry degradation.

Critical Assessment Although the broader temporal forecasting literature is extensive, its direct translation to UAV operational environments remains highly theoretical. A vast majority of proposed architectures systematically ignore the severe constraints of tactical edge deployment, including high-frequency sampling requirements, strict onboard energy budgets, and the complex multi-variate correlations inherent to 6-Degree-of-Freedom

(6-DoF) sensor arrays. Most importantly, the evaluation of these models under active, sophisticated adversarial scenarios remains an unsolved, largely neglected research gap.

2.4 LLMs for Robotics and Autonomy

2.4.1 Semantic Task Planning and Tool-Augmented Autonomy

Large Language Models (LLMs) have demonstrated profound capabilities in zero-shot instruction following and deep contextual reasoning. Within the robotics domain, architectures such as SayCan and the Robotic Transformer (RT) series illustrate that LLMs, when augmented with specific tool-use Application Programming Interfaces (APIs), can systematically map high-level natural language mission descriptions into executable sequences of low-level control actions [44, 45]. For UAV operations, this paradigm introduces the capability to autonomously synthesize heterogeneous, unstructured data streams—such as Notices to Air Missions (NOTAMs), real-time meteorological reports, and internal telemetry—to execute complex dynamic decision-making (e.g., autonomously calculating a mission abort trajectory). Concurrently, emerging research paradigms, such as Time-LLM, propose reprogramming foundational language models to process numerical time-series data by patching and projecting continuous telemetry into a discrete token space. This allows the system to leverage pre-trained attention weights for temporal forecasting and anomaly detection without altering the underlying model architecture [28].

2.4.2 Parameter-Efficient Fine-Tuning and Edge Deployment

Deploying massive foundational models on embedded edge platforms necessitates aggressive architectural optimization. PEFT methodologies, notably Low-Rank Adaptation (LoRA) [50], facilitate the domain-specific specialization of large networks by isolating and training only a minimal subset of low-rank weight matrices, thereby

preserving the vast majority of pre-trained parameters. To further mitigate onboard memory footprints, quantization techniques map high-precision floating-point weights to lower bit-width representations, an approach fully realized in unified frameworks like QLoRA [51]. Additionally, memory optimization protocols, such as the ZeRO optimizer [49], distribute state tensors to make localized fine-tuning computationally tractable. Despite these architectural optimizations, the literature lacks comprehensive frameworks for resolving the severe latency mismatch between autoregressive LLM inference and the high-frequency deterministic control loops of autonomous UAVs. Furthermore, critical security vulnerabilities specific to generative AI—such as adversarial prompt injection and the malicious exploitation of tool-use APIs—are only beginning to be systematically addressed in the literature.

Critical Assessment The intersection of LLMs and autonomous robotics is nascent, with the majority of current literature constrained to theoretical proofs-of-concept operating within simulated environments. Rigorous empirical investigations concerning inference latency, onboard energy consumption (SWaP constraints), and systemic resilience under active cyber-physical attacks remain critical, unaddressed gaps in the literature.

2.5 Edge–Fog–Cloud Systems for UAV AI

2.5.1 Distributed Offloading and Hardware Acceleration

To reconcile the dichotomy between resource-intensive artificial intelligence workloads and the strict real-time latency constraints of autonomous flight, contemporary architectures increasingly employ distributed computational offloading. This paradigm—fundamentally rooted in the evolution of cloudlets and fog computing—distributes inference tasks dynamically across the onboard UAV processor

(the edge), proximate ground control stations (fog nodes), and centralized cloud infrastructure [36]. Furthermore, to optimize localized execution on constrained edge devices, researchers leverage specialized software stacks and hardware-accelerated inference engines, such as TensorRT and the ONNX Runtime. These frameworks maximize throughput and minimize latency via computational graph optimization and precision quantization. Concurrently, dynamic task schedulers are utilized to preemptively prioritize safety-critical control loops over heavy analytical workloads. To mitigate the impact of fluctuating wireless bandwidth during offloading, techniques such as speculative decoding and advanced telemetry compression are frequently deployed. Despite these architectural advancements, the prevailing literature predominantly evaluates distributed systems through the narrow metrics of throughput and energetic efficiency, largely neglecting cyber-physical safety and systemic security.

Critical Assessment Although systems-level research has achieved significant advancements in optimizing inference performance and resource allocation, these architectures are rarely evaluated through the lens of adversarial resilience. Existing literature seldom stress-tests distributed frameworks under conditions of severe communication intermittency, active Radio Frequency (RF) jamming, or malicious traffic injection. Consequently, the critical interplay between dynamic task offloading, network degradation, and deterministic flight control guarantees remains a significantly under-explored domain.

2.6 UAV Cybersecurity and Resilience

2.6.1 The Cyber-Physical Threat Landscape

Autonomous UAVs are inherently deployed within contested, zero-trust environments: they continuously broadcast their spatial coordinates and fundamentally

rely on exposed, wireless channels for command and control. Consequently, adversaries are presented with a broad, multi-dimensional attack surface. Primary attack vectors include Global Navigation Satellite System (GNSS) spoofing to induce trajectory deviations, high-power Radio Frequency (RF) jamming to sever the GCS link, and the exploitation of unencrypted, unauthenticated telemetry protocols (e.g., standard MAVLink) for malicious command injection. Beyond external RF manipulation, severe vulnerabilities exist at the system level, where firmware bugs and hardware backdoors have been actively exploited in wild deployments. Furthermore, the integration of deep learning introduces novel vulnerabilities, allowing adversaries to synthesize adversarial perturbations capable of deceiving onboard perception modules or orchestrating prompt injection attacks against LLM-driven cognitive frameworks [37, 32].

2.6.2 Intrusion Detection and Cyber-Physical Mitigation

To counter this hostile landscape, the literature proposes a diverse array of defensive architectures. Conventional approaches predominantly rely on heuristic or signature-based Intrusion Detection Systems (IDS) localized at the network edge. More sophisticated paradigms employ machine learning to model baseline traffic patterns, flagging statistical anomalies in real time. For physical layer attacks, multi-modal sensor fusion—cross-validating GNSS coordinates with IMU and visual odometry data—is frequently utilized to detect spoofing discrepancies. Recently, emerging research has proposed leveraging LLMs to triage IDS alerts and autonomously recommend mitigation strategies. However, these generative systems must be rigorously compartmentalized to prevent adversarial exploitation. A recurrent, critical flaw across this literature is that defensive mechanisms are overwhelmingly implemented as reactive, post-hoc overlays rather than being intrinsically co-designed with the foundational flight control logic.

2.6.3 Vulnerabilities in Generative Autonomy

The deployment of advanced AI within the UAV control stack introduces distinct Adversarial Machine Learning (AML) risks. Data poisoning attacks threaten the integrity of temporal forecasting models during fine-tuning, while adversarial prompt injection, API tool misuse, and systematic jailbreaking critically compromise LLM-based decision modules. The contemporary defense repertoire against these threats is nascent, heavily emphasizing robust adversarial training, rigorous input sanitization pipelines, strict API capability compartmentalization, and cryptographic attestation. Additionally, the maintenance of immutable audit trails is increasingly recognized as vital for post-incident forensic analysis and iterative model refinement [37, 32].

Critical Assessment A pervasive shortcoming in current cybersecurity literature is the siloed nature of threat mitigation. Defensive architectures are frequently evaluated independently of the vehicle's kinematic constraints. Consequently, researchers largely ignore how the computational latency of an IDS destabilizes high-frequency control loops, or how the overhead of cryptographic mitigations directly conflicts with strict SWaP and mission-objective constraints. True cyber-physical resilience requires a unified, co-designed approach that balances security with deterministic flight stability.

2.7 Datasets, Simulators and Benchmarks

High-fidelity simulation environments, such as PX4 Software-in-the-Loop (SITL) integrated with Gazebo and Microsoft AirSim, form the empirical foundation for much of the current literature. Publicly available datasets, notably the EuRoC MAV visual-inertial sequences, remain industry staples for benchmarking state estimation and simultaneous localization and mapping (SLAM). Historically, the literature has suffered from a distinct lack of comprehensive datasets combining synchronized flight telemetry, raw network

traces, and annotated cyber-physical attacks. Consequently, many time-series anomaly detection benchmarks have been aggressively repurposed from macro-meteorological science or electrical grid load forecasting, which completely fail to reflect the high-frequency, multi-modal sensor dynamics of autonomous UAVs. Fortunately, recent efforts in 2024 and 2025 have begun bridging this gap. Novel datasets such as Unmanned Aerial Vehicles-Network Intrusion Detection Dataset (UAV-NIDD)[56] and UAV Network Intrusion Detection (UAVIDS-2025) [57] provide synchronized network intrusion data under simulated MAVLink exploits, while platforms like UAVThreatBench [58] have introduced the first structured benchmarks for evaluating LLM-driven threat identification in aerial robotics.

2.8 Synthesis and Comparative Matrix

A critical synthesis of the contemporary literature reveals several pervasive, unaddressed gaps. Primarily, there remains no widely accepted consensus on the optimal distribution of computational workloads across the edge-fog-cloud continuum, particularly when the UAV's deterministic control loop must close within tens of milliseconds. While recent theoretical frameworks have proposed distributed multi-agent LLM architectures for UAVs, empirical evaluations of these models under strict real-time constraints remain scarce. Furthermore, cybersecurity is overwhelmingly treated as a post-hoc overlay rather than being intrinsically co-designed with the cyber-physical control architecture. Although PEFT methodologies and quantization techniques demonstrate theoretical promise for edge deployment, their safety and latency characteristics are rarely evaluated during active flight control. Most critically, the current literature fails to provide the rigorous empirical evidence—such as deterministic latency bounds, rigorous calibration curves, and joint safety-security metrics—required to satisfy stringent aviation regulatory certification standards.

2.9 Positioning This Dissertation

To address these fundamental limitations, this dissertation proposes a novel, cyber-physically co-designed layered autonomy stack. At the edge, a high-frequency telemetry sentinel utilizing 2D-variation modeling (TimesNet) monitors state dynamics and flags systemic anomalies, while a reprogrammed foundational language model (Time-LLM) provides predictive forecasting of future kinematic states. The outputs of these neural modules continuously feed certified, classical low-level controllers through strictly typed, mathematically auditable interfaces. To ensure the viability of this architecture within severe SWaP constraints, the system heavily leverages quantized adapters and Low-Rank Adaptation (LoRA) for localized onboard inference, while computationally expensive global parameter updates are asynchronously orchestrated in the cloud utilizing the ZeRO optimization framework. Crucially, this architecture intrinsically binds adversarial security monitoring directly into the deterministic flight control loop, ensuring that real-time threat detection actively informs dynamic path planning and operational decisions, rather than merely logging intrusion alerts. To empirically validate the proposed system, this research contributes a new, high-fidelity dataset encompassing multi-variate telemetry, adversarial network injections, and degraded Radio Frequency (RF) conditions. Together, these contributions advance the fundamental science of time-series-driven generative autonomy while providing a highly pragmatic engineering framework for resilient, secure UAV operations.

Table 2.1: Synthesis of representative literature across foundational knowledge domains.

Domain	**Representative Approaches**	**Strengths**	**Limitations / Gaps**
Flight Control & Planning	PID, MPC [33, 14], LADRC [38, 34], Reinforcement Learning [39]	High determinism and mathematical provability; robust nominal trajectory tracking.	Degraded efficacy under non-stationary dynamics; highly vulnerable to unmodeled adversarial perturbations; prohibitive computational overhead for edge deployment.
Time-Series & Anomaly Detection	TFT, Informer, PatchTST [40, 41, 42], TimesNet [29], OmniAnomaly [43]	Long-horizon temporal modeling; multiscale structural feature extraction; strong benchmark performance.	UAV-specific kinematic constraints underexplored; limited empirical validation under active Radio Frequency (RF) or cyber-attack stressors.
LLMs for Autonomy	Tool-use planning (SayCan, RT-2) [44, 45, 46], Time-LLM [28]	Unprecedented zero-shot generalization; instruction following; deep contextual reasoning.	Severe latency mismatch with real-time flight loops; operational safety and prompt-injection vulnerabilities remain systematically unaddressed.
Edge-Fog-Cloud Systems	Cloudlets, Fog computing [47, 48], ZeRO, PEFT, QLoRA [49, 50, 51]	Real-time computational offloading; highly scalable training; memory-efficient localized fine-tuning.	Weak architectural co-design with cybersecurity protocols; limited benchmarking under adversarial load or severe link intermittency.
Cybersecurity & Resilience	GPS spoofing studies [52], UAV security surveys [53], ML-based IDS [54]	Comprehensive threat taxonomies; ML defenses.	Security measures are typically reactive overlays; few joint autonomy-security evaluations; vulnerability to novel AI-specific risks (e.g., data poisoning).

CHAPTER THREE

METHODOLOGY

3.1 Introduction: A Layered Cyber-Physical Security Stack

As established in the literature review, contemporary Unmanned Aerial Vehicles (UAVs) face a critical dichotomy: they require increasingly complex artificial intelligence to execute autonomous missions, yet their reliance on exposed communication channels and resource-constrained onboard hardware leaves them highly vulnerable to cyber-physical attacks. Furthermore, prevailing research frequently treats security as a reactive, post-hoc overlay rather than a fundamental design parameter. To address these systemic vulnerabilities, this dissertation proposes a holistic, cyber-physically co-designed methodology that secures the UAV operational stack from the foundational software layer up to the distributed cloud-edge intelligence network.

This chapter details a comprehensive four-pillar methodology, progressively building a secure autonomy framework:

1. **Foundational Data Generation (HGDAVE):** The synthesis of a high-fidelity dataset capturing nominal flight telemetry alongside simulated crashes and cyber-intrusions.

2. **Generative Threat Emulation (Net-GPT):** The development of an LLM-empowered offensive threat model to execute advanced MITM attacks.

3. **Firmware Resilience (FineObfuscator):** The implementation of context-sensitive control flow obfuscation to protect UAV software from Reverse Engineering Attacks (REAs).

4. **Distributed Generative Defense (Aero-LLM):** The deployment of specialized, fine-tuned LLMs across the edge-fog-cloud continuum for real-time anomaly detection and secure autonomous control.

3.2 Foundational Data Generation and Simulation (HGDAVE)

For robust anomaly detection and state forecasting, this research utilizes advanced Large Language Models (LLMs) uniquely adapted to interpret sequential time-series data. However, to adapt these foundational models for UAV cyber-physical tasks, they must be rigorously fine-tuned using a comprehensive dataset of synchronized sensor telemetry and network traffic. This domain-specific training enables reliable, real-time inference at the tactical edge.

3.2.1 Temporal Large Language Models

While Large Language Models (LLMs) have demonstrated profound efficacy in natural language processing, their underlying architectures must be mathematically reprogrammed to handle continuous, high-frequency numerical data. Frameworks such as Time-LLM [28] and TimesNet [29] are specifically engineered for time-series forecasting and anomaly detection. Time-LLM leverages a frozen transformer backbone to comprehend long-term temporal dependencies by patching continuous data into a discrete token space, enabling it to autoregressively infer future kinematic states. Conversely, TimesNet employs a task-general 2D-variation modeling approach to effectively identify structural anomalies by analyzing intra-period and inter-period data at multiple temporal resolutions. Fine-tuning these temporal architectures on raw UAV telemetry allows the models to internalize the complex aerodynamic constraints and nominal behavioral manifolds of the airframe.

3.2.2 Data Collection Pipeline for Fine-Tuning

Pre-trained foundational models exhibit poor zero-shot generalization when applied to high-frequency UAV telemetry, as their pre-training corpora lack cyber-physical flight data. Consequently, domain-specific fine-tuning is imperative. The PX4 flight stack's telemetry logging infrastructure intrinsically captures micro Object Request Broker (μORB) messaging topics, providing a highly granular record of the vehicle's internal state machine (detailed in Table 3.1).

Topic Name	**Description**
`vehicle_local_position`	UAV's spatial coordinates in the local NED (North-East-Down) frame.
`vehicle_attitude`	UAV's quaternion-based orientation (roll, pitch, yaw).
`sensor_combined`	Aggregated raw sensor arrays (e.g., IMU accelerations, Gyroscope rates).
`actuator_controls`	Low-level normalized control outputs dispatched to ESCs/motors.
`vehicle_command`	High-level state machine directives (e.g., arm, takeoff, land).

Table 3.1: Critical μORB telemetry topics logged for LLM fine-tuning.

Furthermore, PX4's native logging utilities are leveraged to record external MAVLink traffic, encompassing critical trajectory directives such as `SET_POSITION_TARGET_LOCAL_NED`. To facilitate supervised learning, custom logging modules were developed to strictly synchronize the LLM's state variable inputs with its corresponding actuator command outputs. This synchronized telemetry is indispensable

for evaluating the model's inference latency and ensuring the mathematical traceability of its autonomous decision-making processes.

3.2.3 Synthesis of Nominal and Anomalous Flight Profiles (HGDAVE)

To train the anomaly detection models, this research constructs the Heterogeneous Generative Dataset for Unmanned Autonomous Systems (HGDAVE). This dataset mandates the collection of strictly structured nominal flight data, juxtaposed against two distinct classes of anomalous data: kinematic crashes and cybersecurity intrusions.

Nominal Mission Generation: Baseline operational data is synthesized utilizing an automated Python scripting pipeline that iteratively modulates flight parameters across successive missions. During execution, all high-frequency sensory data is continuously logged. The procedural logic governing the generation of nominal mission profiles is detailed in Algorithm 3.1, and the specific aerodynamic parameters altered during data collection are enumerated in Table 3.2.

Anomalous Mission Generation: The generation of crash-state data is achieved via PGFuzz [59], an advanced fuzzing framework that systematically mutates inputs across command structures, system parameters, and simulated environmental factors. To achieve scalability, a permutation matrix was distributed across n^3 ($n > 1$) parallel Docker containers, maximizing crash-case generation efficiency. Conversely, the cyber-intrusion dataset was curated via dynamic penetration testing, wherein an ethical hacker deployed localized RF jamming, MAVLink command injection, and GNSS spoofing tools against the PX4 autopilot and QGC software. The systemic degradation resulting from these cyber-attacks is visualized in Fig. 3.4.

Algorithm 3.1: Automated Mission Generation and Telemetry Logging

Input: A predefined sequence of spatial coordinates *waypoints*.
Output: Synchronized telemetry log file *log*.

1. *mission_items* $\leftarrow \emptyset$;
2. Establish asynchronous MAVLink connection to the UAS;
3. Await stable telemetry heartbeat and GPS lock;
4. Inject predefined mission parameters (see Table 3.2);
5. **foreach** *waypoint* $\in$ *waypoints* **do**
6. | Instantiate `MISSION_ITEM_INT` for the *waypoint*;
7. | Append to *mission_items*;
8. **end**
9. Compile *mission_items* into an executable flight plan;
10. *Return_to_Launch* $\leftarrow$ *true*;
11. Upload flight plan to flight controller;
12. Engage `ARM` state and initiate `AUTO_MISSION`;
13. **while** *UAS state* == *ARMED* ***and*** *In-Air* **do**
14. | Continuously poll and log μORB telemetry;
15. **end**
16. Terminate asynchronous logging tasks upon successful landing and `DISARM`;
17. **return** *log*;

3.2.4 Digital Twin and Simulation Architectures

High-fidelity simulation, effectively constituting a Digital Twin (DT) of the UAV, is strictly required for the safe generation of adversarial data. Executing kinetic cyber-attacks on physical airframes in uncontrolled real-world environments poses catastrophic safety and financial risks. Consequently, the PX4 Autopilot [4] is utilized to provide robust SITL and HITL capabilities (illustrated in Fig. 3.1).

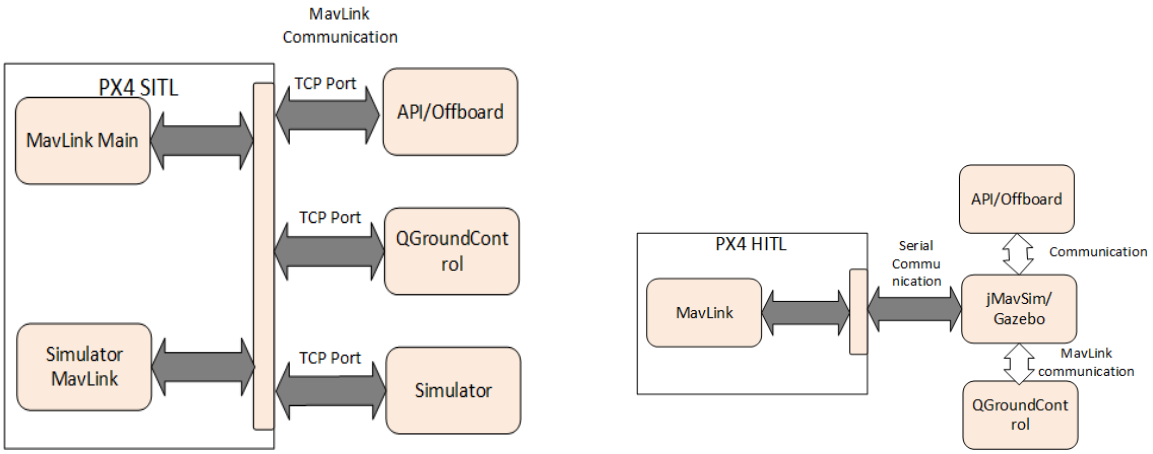
3.2.4.1 SITL Integration In the SITL paradigm, the complete PX4 flight stack executes natively on a centralized workstation, interfacing directly with a physics engine (e.g., Gazebo [60]) that simulates aerodynamic forces and generates virtual sensor telemetry. This configuration eliminates hardware dependencies, facilitating rapid algorithmic

Table 3.2: Parameter variations for Nominal Mission Data Generation (HGDAVE).

Mission	**PX4 Parameter**	**Description**
1	N/A	Baseline nominal mission execution.
2	`MPC_LAND_SPEED`	Modulates the maximum vertical descent velocity.
3	`MPC_TKO_SPEED`	Modulates the maximum vertical ascent velocity.
4	`MIS_TAKEOFF_ALT`	Alters the target altitude for the initial takeoff phase.
5	Multiple Parameters	Executes Missions 1–4 consecutively, applying stochastic parameter variations within a defined temporal window.
6	`MIS_YAW_ERR`	Adjusts the acceptable angular tolerance for yaw alignment.
7	`MPC_YAW_MODE`	Dictates UAV heading behavior (e.g., towards waypoint, towards home, along trajectory).
8	`NAV_ACC_RAD`	Modifies the spatial acceptance radius required to register a waypoint as reached.
9	`MPC_ACC_DOWN_MAX`	Imposes constraints on the maximum allowable downward vertical acceleration.

iteration. Architecturally, this requires compiling the PX4 flight stack for a specific SITL target, initializing the physics engine with a defined environmental world file, and binding QGroundControl via localized UDP streams (see Fig. 3.2). Because the Simulator and the flight stack share the same compute environment, direct, low-latency communication is achieved using ROS2/MAVROS frameworks [61].

3.2.4.2 HITL Integration Conversely, the HITL configuration relies on physical flight controller hardware (the embedded edge device) to execute the firmware. The physical hardware processes simulated sensor data ingested from the workstation's physics engine



(a) Software-In-The-Loop (SITL) (b) Hardware-In-The-Loop (HITL)

Figure 3.1: Digital Twin Simulation Architectures utilized for dataset generation.

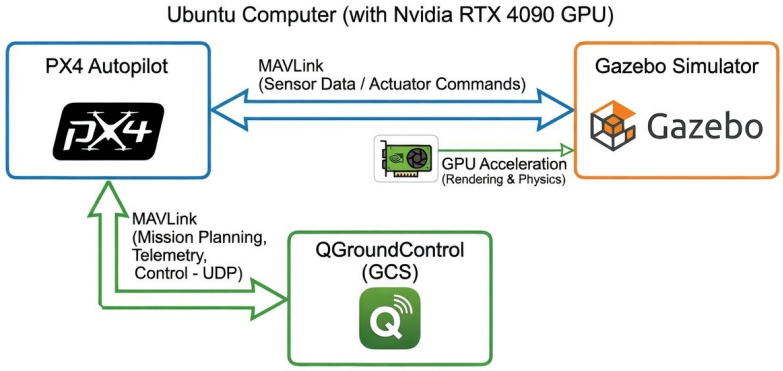


Figure 3.2: Architectural configuration of the PX4 SITL simulation pipeline.

and returns physical actuator Pulse Width Modulation (PWM) commands to the simulation (see Fig. 3.3). This paradigm bridges the "sim-to-real" gap by validating the strict timing and processing constraints of the embedded hardware.

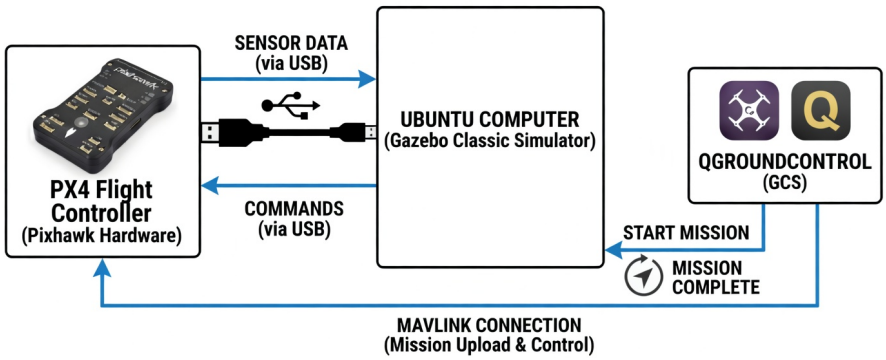


Figure 3.3: Architectural configuration of the PX4 HITL simulation pipeline.

In the HITL setup, the Digital Twin computer communicates with the physical flight controller via serial or robust UDP networking. This setup leaves the deterministic flight control to the microcontroller. By combining the rapid scalability of SITL with the hardware-level fidelity of HITL, this methodology ensures that the generated HGDAVE dataset accurately encapsulates both the theoretical state-space and the tangible processing realities of cyber-physical UAV deployments.

3.3 Generative Adversarial Threat Modeling (Net-GPT)

To establish a rigorous baseline for the proposed defensive framework, it is imperative to formally define the cyber-physical threat landscape. Furthermore, to effectively train and evaluate advanced anomaly detection systems, one must emulate a sophisticated, modern adversary. This section details the comprehensive threat model and introduces an LLM-empowered adversarial framework used to synthesize complex intrusion data.

3.3.1 The Cyber-Physical Threat Landscape

The anomaly detection and state forecasting architectures proposed in this dissertation are engineered to mitigate four primary categories of cyber-physical attack vectors:

1. **Network Intrusions:** Adversarial actions attempting to sever or manipulate the communication channels between the GCS and the UAV, encompassing Radio Frequency (RF) jamming, MAC spoofing, and advanced MITM attacks.

2. **Perception and Sensor Manipulation:** Physical or signal-level injections designed to deceive the UAV's deterministic control loop, such as GNSS/GPS spoofing, acoustic resonance attacks on MEMS gyroscopes, and false data injection.

3. **Firmware Exploitation:** The malicious exploitation of software vulnerabilities within the flight controller stack (e.g., buffer overflows, Reverse Engineering Attacks) to achieve unauthorized root access or arbitrary code execution.

4. **Insider Threats:** Authorized nodes or compromised trusted devices attempting to covertly disrupt operations or exfiltrate sensitive mission telemetry.

To execute these exploits, modern adversaries increasingly leverage high-bandwidth network infrastructure and edge computing. Just as defensive LLMs require specialized AI accelerators, adversaries can utilize proximate edge servers equipped with high-performance GPUs to execute computationally intensive generative attacks. Fig. 3.4 visualizes the operational execution of these attacks, while Table 3.3 categorizes specific exploits against the PX4 autopilot and their corresponding impact on the CIA (Confidentiality, Integrity, Availability) and Privacy triad.

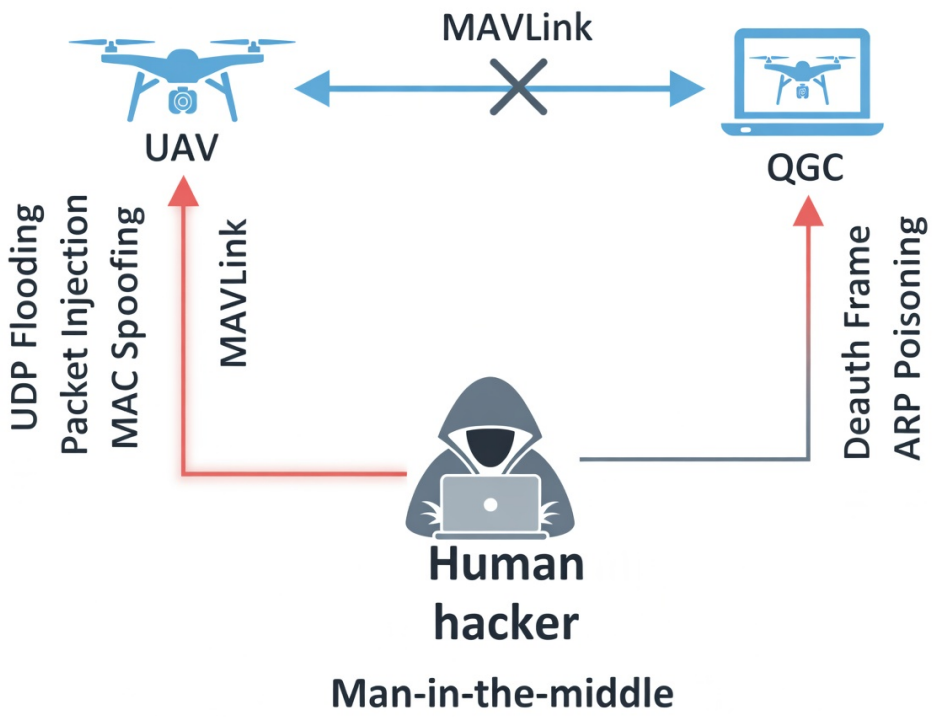


Figure 3.4: Taxonomy and execution pathways of cyber-attacks against autonomous UAVs.

3.3.2 LLM-Empowered Man-In-The-Middle Chatbot

To rigorously evaluate the resilience of the proposed autonomy stack, this methodology introduces Net-GPT, an adversarial Generative AI framework designed to autonomously parse network protocols and orchestrate complex MITM attacks. Net-GPT enables a malicious proxy UAV to autonomously intercept, analyze, and manipulate bidirectional communications between a benign target UAV and its corresponding GCS. By establishing this proxy connection, Net-GPT effectively impersonates the benign vehicle to the GCS, and vice versa.

The cognitive engine of Net-GPT is driven by a fine-tuned LLM deployed on a proximate adversarial edge server. The foundational model is initially fine-tuned on public

Table 3.3: Taxonomy of specific attacks launched against the PX4 autopilot. Security criteria impacted: Confidentiality (C), Privacy (P), Integrity (I), and Availability (A).

Attack Method	**Tools**	**Unauth. Access**	**Man-in-the-Middle**	**Denial of Service**	**Affected Criteria**
MAC Spoofing	MAC Changer	✓	✓	✗	C, P, I
Session Hijacking	aireplay-ng Rogue QGC	✓	✓	✗	C, P, I
ARP Poisoning	Ettercap arpspoof	✓	✓	✗	C, P, I
Packet Injection	airodump-ng	✓	✓	✗	C, P, I
Flooding Attack	LOIC hping3	✗	✗	✓	A
Deauthentication	aireplay-ng	✗	✓	✓	A

network traffic repositories to deeply internalize Transmission Control Protocol (TCP)/User Datagram Protocol (UDP) and MAVLink protocol structures. Leveraging the profound zero-shot reasoning capabilities of foundational LLMs, Net-GPT dynamically synthesizes highly contextualized, protocol-compliant payloads that seamlessly mimic the existing communication flow. This capability allows the adversary to inject malicious kinematic waypoints or false telemetry without triggering heuristic, signature-based anomaly detectors. The systemic architecture of this generative threat model is illustrated in Fig. 3.5.

3.3.3 Hijacking and Deceiving Benign UAVs

The execution of this generative MITM attack follows a two-phase adversarial kill-chain: (1) target acquisition and session hijacking, followed by (2) generative deception.

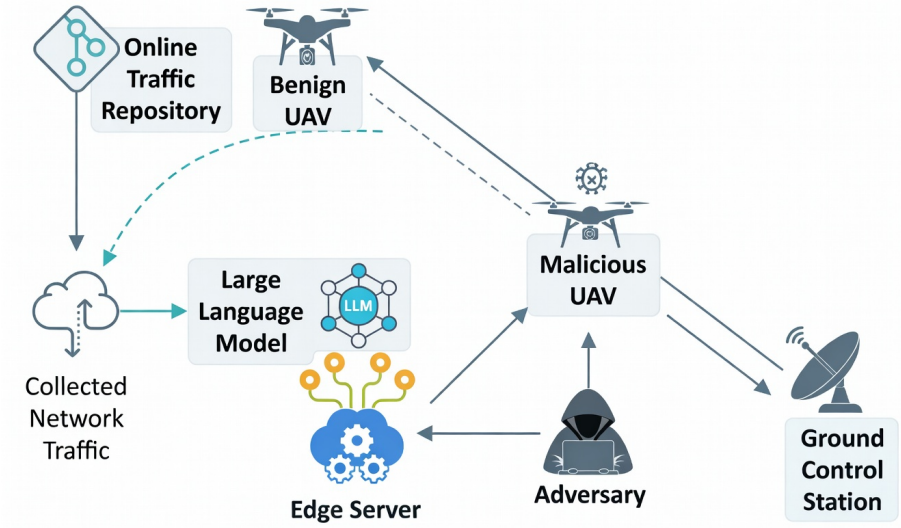


Figure 3.5: Architectural workflow and proxy mechanism of the Net-GPT generative adversary.

To initially hijack the benign UAV, the methodology employs three distinct attack vectors (as visualized in Fig. 3.4): *ARP Poisoning*, *Semi-Session Hijacking*, and *Full Session Hijacking*. During the acquisition phase, the adversary utilizes tools such as `aireplay-ng` to broadcast spoofed IEEE 802.11 de-authentication frames to the GCS. This abruptly forces the GCS to drop its connection. Before the benign UAV can initiate a failsafe "Return-to-Launch" (RTL) protocol, the malicious Net-GPT proxy assumes the MAC and IP identity of the GCS, establishing a rogue session. Once the hijack is successful, Net-GPT relies on its edge-deployed LLM to continuously generate context-aware MAVLink control messages, actively steering the hijacked UAV while feeding synthesized, nominal telemetry back to the genuine GCS to mask the intrusion. The complex network traces and asynchronous kinematic telemetry generated during these dynamic exploits are subsequently captured and cataloged within the HGDAVE

dataset, providing highly realistic, adversarial training data for the subsequent defensive fine-tuning phases.

3.4 Firmware Resilience and Obfuscation (FineObfuscator)

Prior to the deployment of advanced generative AI models at the network edge, the foundational software executing on the UAV microcontroller and GCS must be cryptographically secured. This section details the methodology for protecting the core flight stack from adversarial static analysis and intellectual property theft, ensuring that the underlying deterministic control loops remain uncompromised.

3.4.1 Vulnerabilities to Reverse Engineering Attacks (REAs)

In the domain of cyber-physical systems, software reverse engineering is heavily leveraged by adversaries to identify zero-day vulnerabilities and hidden logical functionalities without requiring access to the original source code. The unprotected dissemination of compiled flight controller binaries (e.g., PX4 firmware) and GCS software facilitates proprietary algorithmic theft and allows adversaries to map the exact memory architecture required for malicious payload injection. These exploits are formally categorized as Reverse Engineering Attacks (REAs).

Traditionally, commercial obfuscation tools attempt to thwart REAs by applying cryptographic transformations and control-flow flattening across the entire monolithic software package. However, within the context of autonomous UAVs, this brute-force approach introduces unacceptable latency, memory bloat, and execution overhead, directly violating the strict SWaP constraints of embedded flight controllers. Consequently, a highly optimized, resource-efficient obfuscation paradigm is required.

3.4.2 Context-Sensitive Control Flow Obfuscation

To neutralize REAs without violating the strict temporal constraints of flight management systems, this methodology integrates FineObfuscator [62], a novel, context-sensitive security framework. Rather than encrypting the entire flight stack, FineObfuscator isolates and protects only a mathematically minimized subset of highly critical software components, thereby maintaining optimal runtime efficiency.

Architecturally, FineObfuscator executes a rigorous static analysis pipeline across the compiled flight software. During this phase, it parses the binary to extract localized Control Flow Graphs (CFGs) and global Inter-Procedural Control Flow Graphs (ICFGs). By mapping the computational pathways, the system systematically identifies the most critical functional components and proprietary code snippets (e.g., cryptographic key generation, proprietary state-estimation algorithms, and secure MAVLink parsing logic).

Once the critical ICFG nodes are isolated, FineObfuscator applies targeted structural obfuscation policies—such as opaque predicate insertion, basic block splitting, and instruction substitution—exclusively to these high-value logic blocks. This highly granular, context-sensitive approach effectively breaks the semantic continuity required by adversarial decompilers (e.g., IDA Pro, Ghidra), successfully defeating the REA process. By obfuscating the flight stack at this fine-grained level, the framework achieves maximum cyber-resilience while requiring significantly fewer computational resources than existing whole-binary obfuscation tools.

3.5 Task-Specific LLM Fine-Tuning

Pre-trained foundational Large Language Models (LLMs) inherently lack the domain-specific contextual awareness required to parse cyber-physical telemetry or detect network-level intrusions. Consequently, rigorous fine-tuning is required to adapt these models to the UAV operational domain. To train the models for network anomaly

detection, a vast corpus of nominal and adversarial network traffic was captured via packet sniffing (e.g., Wireshark [63]). To prevent algorithmic bias and over-fitting, the dataset was carefully diversified and constrained to a highly representative scale, encompassing thousands of multi-turn TCP/UDP sessions.

3.5.1 Network Traffic Feature Extraction (TCP/UDP)

UAV GCS, such as QGroundControl, communicate with the flight controller utilizing standard Transmission Control Protocol/Internet Protocol (TCP/IP) and UDP protocols. Rather than processing the entire variable-length payload—which introduces unnecessary computational overhead—the methodology extracts a deterministic subset of critical header features. For TCP (illustrated in Fig. 3.6) and UDP (illustrated in Fig. 3.7), the LLM is trained on specific byte-level sequences to mathematically model nominal traffic flow and identify malicious deviations, such as sequence hijacking or volumetric flooding.

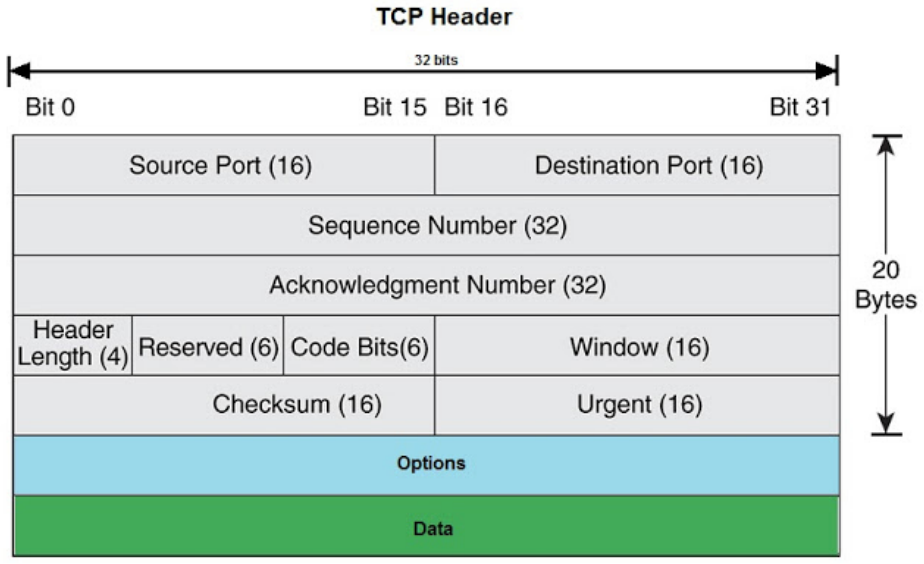


Figure 3.6: Standard TCP Header Architecture.

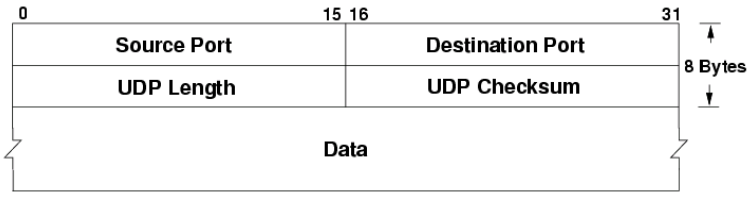


Figure 3.7: Standard UDP Header Architecture.

The specific network header fields isolated for the tokenization and fine-tuning pipeline are detailed in Table 3.4. By analyzing the relationships between `sport`, `dport`, `seq`, and specific control `flags` (e.g., SYN, ACK, RST), the generative model learns to predict the structural validity of the subsequent packet in a session.

Table 3.4: Selected TCP/UDP header features extracted for LLM fine-tuning.

Feature Field	**Size (Bytes)**
Source Port (`sport`)	2
Destination Port (`dport`)	2
Control Flags (`flags`)	1
Sequence Number (`seq`)	4
Acknowledgment Number (`ack`)	4
Payload Length (`length`)	4

For the LLM to process this continuous network traffic, the extracted data is structured into a rigorous prompt-completion format (illustrated in Fig. 3.8). Each training entry is segmented into three distinct context blocks: `#Previous_Packet`, `#Predicted_Packet`, and `#Context`, utilizing a strictly enforced <key:value> pair syntax.

```
{
    #Previous_Packet: {
          #BLOCK {}
    },
    #Predicted_Packet: {
        #BLOCK {}
    },
    #Context: [
        {
            #BLOCK {}
        },
        ......
        {
            #BLOCK {}
        }
    ]
}
```

(a) The structure of one training data entry

```
#BLOCK {
    $source:        $Value,
   $destination:    $Value,
   $sport:          $Value,
   $dport:          $Value,
   $flags:          $Value,
   $seq:            $Value,
   $ack:            $Value,
   $length:         $Value
}
```

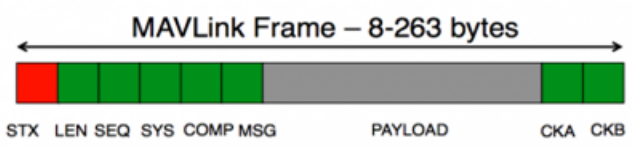
(b) The structure of the `#BLOCK` section

Figure 3.8: Structured data format for autoregressive network anomaly fine-tuning.

3.5.2 MAVLink Telemetry Processing

Parallel to network traffic, the UAV's internal sensors generate continuous, asynchronous time-series data. The PX4 flight controller publishes this telemetry via μORB topics, which are subsequently encapsulated into MAVLink messages for external

transmission. To fine-tune the temporal models (TimesNet and Time-LLM), raw μlog data is extracted, cleaned, and serialized into formatted CSV structures. The architecture of a standard MAVLink frame is illustrated in Fig. 3.9, and critical telemetry messages utilized for state forecasting are enumerated in Table 3.5.



Byte Index	Content	Value	Explanation
0	Packet start sign	v1.0: 0xFE (v0.9: 0x55)	Indicates the start of a new packet.
1	Payload length	0 - 255	Indicates length of the following payload.
2	Packet sequence	0 - 255	Each component counts up his send sequence. Allows to detect packet loss
3	System ID	1 - 255	ID of the SENDING system. Allows to differentiate different MAVs on the same network.
4	Component ID	0 - 255	ID of the SENDING component. Allows to differentiate different components of the same system, e.g. the IMU and the autopilot.
5	Message ID	0 - 255	ID of the message - the id defines what the payload "means" and how it should be correctly decoded.
6 to (n+6)	Data	(0 - 255) bytes	Data of the message, depends on the message id.
(n+7) to (n+8)	Checksum (low byte, high byte)	ITU X.25/SAE AS-4 hash, **excluding packet start sign, so bytes 1..(n+6)** Note: The checksum also includes MAVLINK_CRC_EXTRA (Number computed from message fields. Protects the packet from decoding a different version of the same packet but with different variables).	

Figure 3.9: MAVLink Packet Structure and Field Descriptions.

Table 3.5: Critical PX4 MAVLink Sensor Data Messages utilized for state forecasting.

MAVLink Message	**Category**	**Purpose / Data Provided**
SYS_STATUS	System Status	Sensor health, battery voltage, hardware flags.
HIGHRES_IMU	IMU (High-rate)	High-precision accelerometer, gyroscope, magnetometer.
ATTITUDE_QUATERNION	IMU (Standard)	Spatial orientation (quaternion) and angular rates.
GPS_RAW_INT	GNSS / Position	Latitude, longitude, altitude, HDOP, fix type.
VIBRATION	Vibration	Mechanical vibration levels across x, y, z axes.

The telemetry ingestion pipeline operates as follows (visualized in Fig. 3.10): The GCS requests high-frequency streams (e.g., HIGHRES_IMU) via the MAV_CMD_SET_MESSAGE_INTERVAL directive. The PX4 receiver validates the System ID and cryptographic checksum (CRC) of incoming packets before asynchronously broadcasting the requested telemetry back to the GCS. This continuous stream provides the highly dynamic sequence data necessary for the temporal LLMs to forecast future kinematic states.

3.5.3 Advanced Optimization Techniques (PEFT, QLoRA, RLHF)

To execute the fine-tuning of the selected foundational models (detailed in Table 3.6), the methodology leverages the DeepSpeed optimization framework [64], implementing a rigorous three-stage alignment pipeline:

1. **Supervised Fine-Tuning (SFT):** An actor model is trained on the labeled dataset to accurately predict subsequent packets and telemetry states.

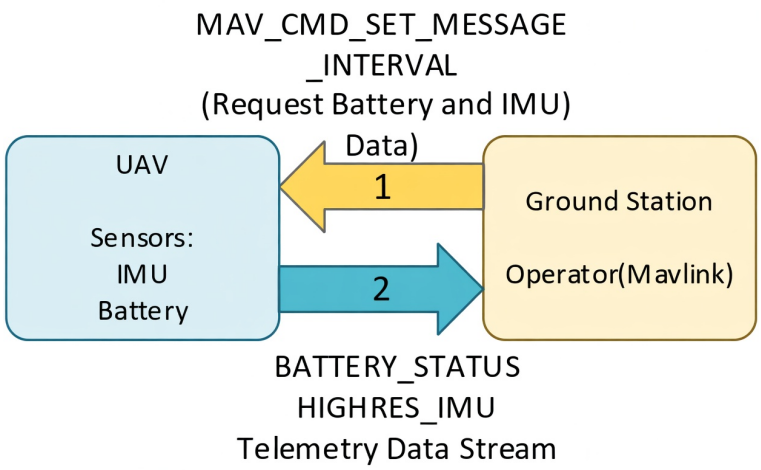


Figure 3.10: MAVLink Telemetry Request and Ingestion Flow.

2. **Critic Model Training:** A secondary reward model is trained via human feedback cycles to assess and score the accuracy of the actor's generative predictions.

3. **Reinforcement Learning from Human Feedback (RLHF):** The RLHF framework recursively refines the SFT model, forcing the generative output to align with strict safety and accuracy thresholds based on the critic's reward function.

Table 3.6: Foundational LLMs utilized for Network Anomaly Detection.

Model Name	**HuggingFace Sub-version**	**Parameter Count**
Llama-2-7B	*llama-2-7b-hf* [65]	7 Billion
Llama-2-13B	*llama-2-13b-hf* [66]	13 Billion
GPT-2	*gpt2* [67]	137 Million
Distil-GPT-2	*distilgpt2* [68]	82 Million

To ensure these massive models remain deployable on constrained edge hardware, the training process utilizes Quantized Low-Rank Adaptation (QLoRA) [51]. QLoRA freezes the 4-bit quantized foundational weights and trains only a minimal set of low-rank adapter matrices. The precise hyperparameters utilized for the QLoRA optimization are provided in Table 3.7.

Table 3.7: Hyperparameters configured for the QLoRA fine-tuning pipeline.

Parameter	**Value**	**Parameter**	**Value**
Quantization	NF4	**Logging steps**	10
Batch size/device	8	**Learning rate**	2e-4
Gradient steps	12	**Global grad norm**	0.3
Paged Optimizer	paged_adamw	**Warm-up ratio**	0.03
Scheduler type	constant	**Epoch(s)**	1 - 10

For kinematic anomaly detection and state forecasting, the temporal architectures—TimesNet [29] and Time-LLM (utilizing Llama-2-7B as the backbone)—undergo specialized training (see Table 3.8 and Table 3.9). The telemetry data is pre-processed (imputation, normalization, and scaling) before being tokenized. TimesNet identifies structural anomalies across temporal resolutions, while Time-LLM predicts future kinematic degradation, evaluated strictly via Mean Absolute Error (MAE) and Mean Squared Error (MSE).

3.6 The Distributed Defensive Architecture (Aero-LLM)

The culmination of this methodology is Aero-LLM, a novel distributed framework explicitly designed to resolve the computational and operational challenges of deploying

Table 3.8: Temporal architectures utilized for MAVLink Anomaly Detection and Forecasting.

Architecture	**Backbone / Sub-version**	**Parameter Count**
TimesNet	*TimesNet* [29]	N/A (Task-General)
Time-LLM	*llama-2-7b* [65]	7 Billion

Table 3.9: Hyperparameters for fine-tuning the temporal state-forecasting models.

Training Parameter	**TimesNet**	**Time-LLM**
Epochs	3	1
Learning Rate	0.02	0.02
Llama Layers Utilized	N/A	8
Batch Size	128	4
Model Dimension (d_{model})	64	32
FCN Dimension (d_{ff})	64	128

Generative AI on autonomous vehicles. To reconcile the dichotomy between resource-intensive AI and strict real-time flight constraints, Aero-LLM deploys specialized models across an Edge-Fog-Cloud continuum (illustrated in Fig. 3.11):

- **Onboard Systems (The Edge):** The foundational flight stack (PX4) interfaces directly with a companion computer, specifically an NVIDIA Jetson Nano (ARM CPU + embedded GPU) operating within a strict 10–25 W power envelope. To ensure deterministic execution under 5–30 ms latency bounds, this tier executes highly quantized, lightweight models (e.g., OPT-125m, OPT-350m, and the

TimesNet Sentinel). These models are strictly responsible for real-time anomaly detection and immediate kinematic command vetting.

- **Edge Servers (The Fog):** To handle medium-latency tasks (100–500 ms)—including complex sensor fusion, dynamic mission reconfiguration, and intermediate Net-GPT adversarial triage—computation is offloaded to proximate GCS edge servers. Architecturally, these nodes utilize Intel Xeon processors running Ubuntu 22.04 LTS, accelerated by NVIDIA RTX 4090 GPUs. This tier supports the deployment of medium-scale models, such as OPT-1.3B, OPT-6.7B, and the Time-LLM forecaster.

- **Cloud Platforms (The Cloud):** Highly scalable centralized infrastructure (e.g., Google Cloud Platform or PaperSpace computing instances with massive RAM and GPU clusters) is reserved for computationally intensive, offline/nearline operations. This includes global policy fine-tuning (via DeepSpeed/ZeRO), deep historical analytics, and the hosting of massive foundational models (e.g., Llama-2-7B and Llama-2-13B) for collaborative multi-UAV swarm planning.

3.6.1 Subsystems Architecture and Workflow

The Aero-LLM framework operates as a continuous-learning, closed-loop distributed system comprised of four primary subsystems: the *Data Collector*, the *Fine-Tuner (with Pre-processing)*, the *Online Controller*, and the *Offline Processor*.

1. Data Collector: This subsystem continuously ingests heterogeneous data streams and consolidates them into a time-synchronized, provenance-rich corpus. It aggregates inputs from SITL/HITL simulators, physical ethical-hacking sessions, PGFuzz protocol perturbations, and adversarial network exchanges synthesized by Net-GPT. It

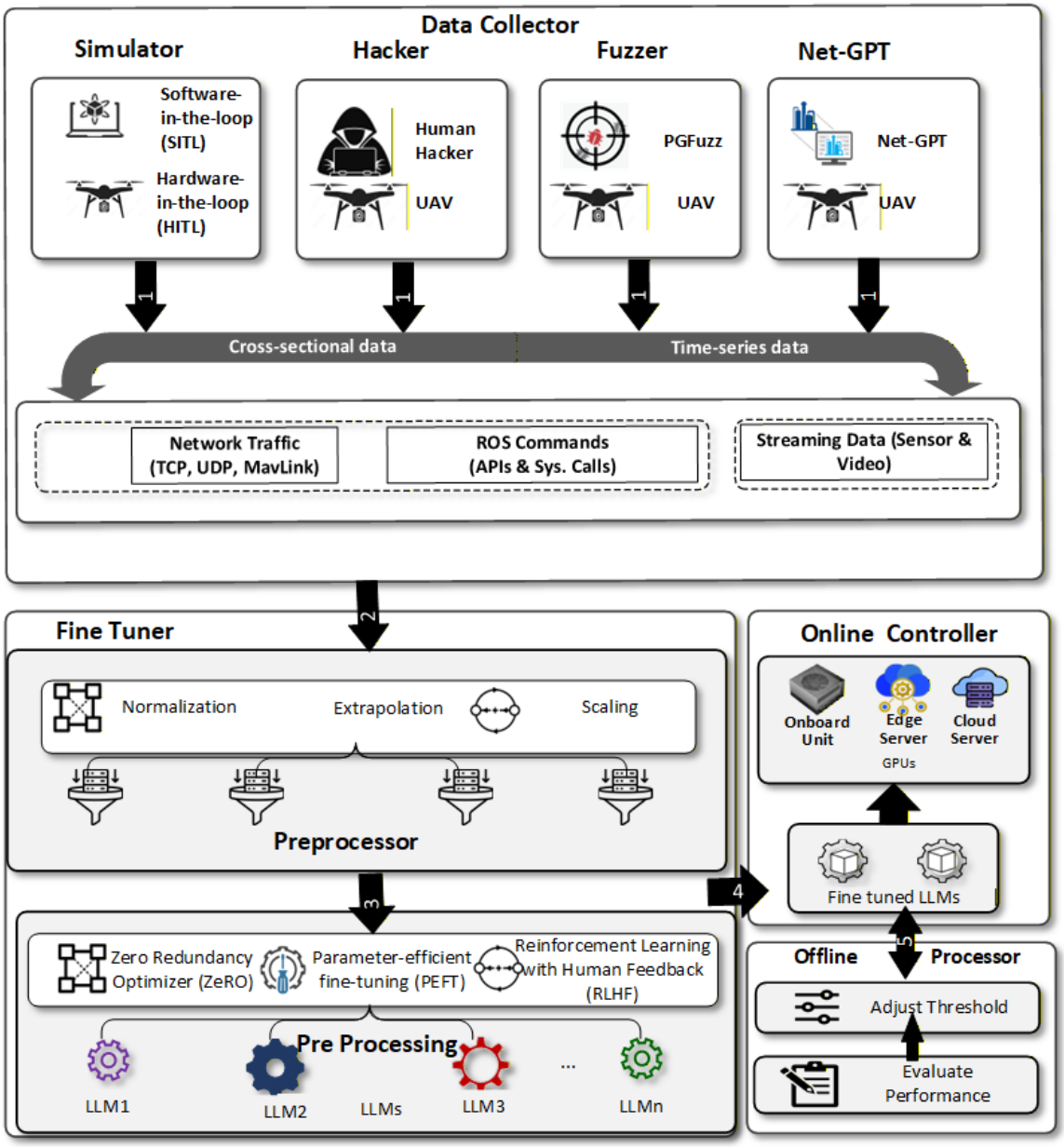


Figure 3.11: System Architecture of the Aero-LLM Distributed Framework.

ensures schema-validated storage and immutable audit logging, producing curated episode manifests.

2. Fine-Tuner with Pre-processing: This module transforms raw telemetry traces into training-ready supervised data. Pre-processing scales and aligns asynchronous modalities (sensor streams, ROS commands, network captures) via causal extrapolation. The Fine-Tuner then executes memory-efficient distributed optimization (DeepSpeed/ZeRO) to produce parameter-efficient adapters (PEFT/QLoRA), rigorously aligned via RLHF. Its outputs are version-controlled model artifacts pushed to a secure registry.

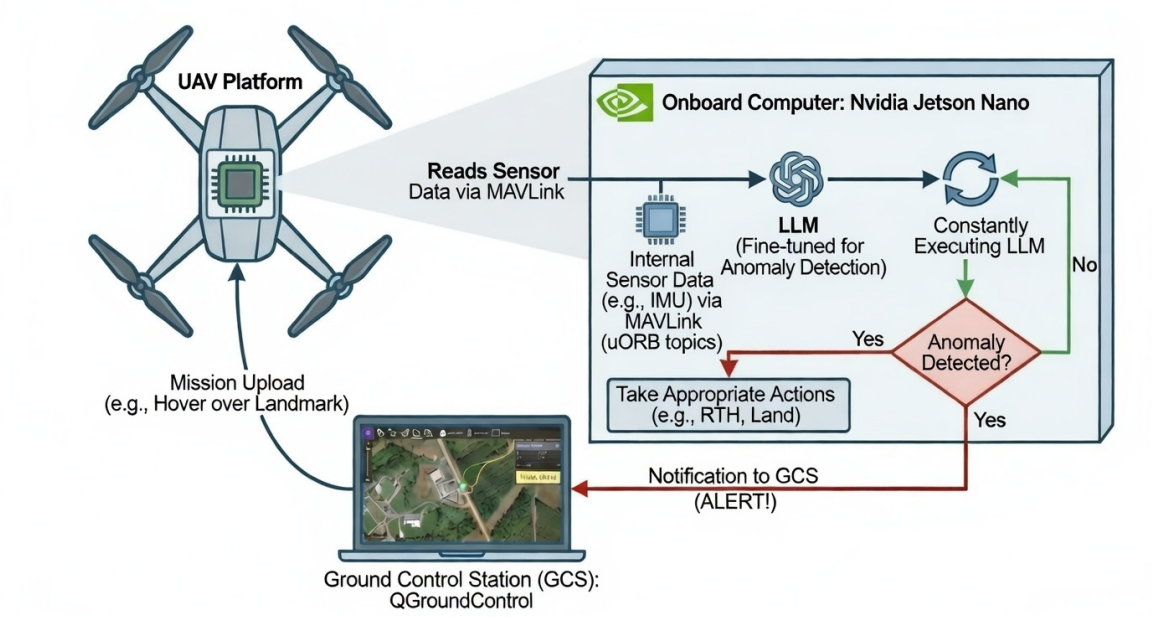


Figure 3.12: Deployment of the Online Controller on the onboard NVIDIA Jetson Nano edge unit.

3. Online Controller: As the live inference engine, the Online Controller enforces mission safety under strict latency bounds. The onboard unit (illustrated in Fig. 3.12) acts as a low-latency gatekeeper, vetting MAVLink commands and detecting immediate

anomalies. A policy-driven router dynamically escalates computationally ambiguous threats to the Fog/Cloud tiers when network links permit, while gracefully degrading to strict onboard enforcement during link loss. Every decision (allow, block, escalate) is serialized to a mission log bus.

4. Offline Processor: This subsystem closes the learning loop by translating post-mission logs into calibrated algorithmic improvements. It evaluates key performance metrics—including command-vetting precision, false-positive rates under attack, and end-to-end latency—to optimize operational thresholds. Misclassified edge cases are distilled into a "hard-example" buffer that refreshes the training dataset for the subsequent fine-tuning round.

3.6.2 Contracts, Trust Boundaries, and Systemic Capabilities

To ensure structural integrity, strict trust boundaries are mathematically enforced across the framework. All inbound sensor and network telemetry is treated as untrusted until formally normalized by the Data Collector. The Online Controller executes only the model artifacts and threshold bundles retrieved from the registry. Ultimately, this architecture guarantees bounded per-decision latency on the UAV while providing an automated, measurement-driven pathway for continual algorithmic improvement. By intrinsically binding LLM-driven anomaly detection into the deterministic control loop, Aero-LLM ensures that defensive insights directly influence dynamic path planning and operational survival.

CHAPTER FOUR

RESULTS

The preceding chapters established a comprehensive, multi-tiered methodology for securing Unmanned Aerial Vehicles (UAVs) via generative artificial intelligence, contextual firmware obfuscation, and distributed edge computing. This chapter presents the empirical validation of the proposed architectures. The evaluation is systematically structured to assess each core contribution: the synthesis of the experimental dataset, the mathematical metrics governing system performance, the resilience of the obfuscated firmware, the generative efficacy of the adversarial threat model (Net-GPT), and the real-time anomaly detection and forecasting capabilities of the distributed defense framework (Aero-LLM).

4.1 Experimental Dataset and Pre-Processing Pipeline

As established in Chapter 3, all empirical evaluations in this dissertation are conducted utilizing the Heterogeneous Generative Dataset for Unmanned Autonomous Systems (HGDAVE) [31]. Rather than relying on repurposed macro-meteorological or static IT network datasets, HGDAVE provides a high-fidelity, domain-specific foundation required to rigorously benchmark autonomous cyber-physical systems.

4.1.1 Dataset Composition

To guarantee statistical relevance and algorithmic robustness, the HGDAVE dataset curated for this evaluation encompasses three distinct operational classifications:

1. **Nominal Flight Trajectories:** Baseline telemetry and network traffic derived from script-generated autonomous missions (e.g., waypoint navigation, dynamic altitude adjustments) executed within a Digital Twin (DT) environment [69].

2. **Kinematic and System Failures:** Crash-state telemetry and critical fault logs generated via aggressive software fuzzing against the PX4 flight stack's API and command interfaces.

3. **Adversarial Cyber-Intrusions:** Highly structured attack vectors synthesized by ethical hackers and the Net-GPT framework, specifically targeting the UAV-GCS communication link.

The finalized evaluation corpus utilized in subsequent sections consists of over 100,000 synchronized network packets spanning approximately 7,000 multi-turn TCP/UDP sessions. This network data is strictly synchronized with thousands of high-frequency MAVLink telemetry sequences, providing ground-truth labels for complex cyber-attacks, including MAC Spoofing, Session Hijacking, ARP Poisoning, and False Data Injection.

4.1.2 Multivariate Temporal Pre-Processing

Prior to supervised fine-tuning and real-time inference, the raw HGDAVE telemetry must be mathematically normalized. Because multi-modal telemetry streams are intrinsically asynchronous—with distinct hardware sensors (e.g., IMU, Barometer, GNSS) polling at varying frequencies—the raw data naturally exhibits missing values, disparate sampling rates, and severe temporal misalignment.

To rectify this and ensure compatibility with the strict sequential input constraints of TimesNet and Time-LLM, a rigorous multivariate temporal alignment pipeline was implemented. All asynchronous MAVLink streams were resampled to a uniform, common frequency baseline. Missing data points and temporal gaps resulting from stochastic network packet loss were resolved utilizing advanced interpolation techniques [70]. Furthermore, redundant values caused by sensor oversampling were aggressively filtered,

and all kinematic features (e.g., velocities, quaternion attitudes) were subjected to Min-Max scaling to prevent high-magnitude variables from dominating the gradient updates during LLM fine-tuning.

For the primary evaluations conducted in this chapter, the dataset is strictly partitioned, with a concentrated focus on the **cybersecurity intrusion subset**. These vectors represent subtle, dynamic cyber-physical anomalies that are historically difficult for traditional heuristic or threshold-based intrusion detection systems to reliably identify.

4.2 Evaluation Metrics

Because cyber-physical anomaly detection fundamentally operates as a binary classification problem (Nominal vs. Anomalous), the foundational evaluation relies on standard confusion matrix metrics. However, given the continuous, high-frequency nature of UAV kinematics, traditional point-wise metrics are overly punitive and fail to reflect operational reality. Consequently, this evaluation implements a **Time-Window Tolerance** paradigm ($\delta = 2s$): if the temporal model flags an anomaly within two seconds of the ground-truth attack initiation, the prediction is classified as a True Positive (TP). Systemic performance is quantified using Precision, Recall, and the harmonic mean (F1-Score):

$$\text{Precision} = \frac{TP}{TP+FP}, \quad \text{Recall} = \frac{TP}{TP+FN}, \quad F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.1}$$

Furthermore, to validate the deployability of the distributed architecture, we strictly monitor **Inference Latency** (L_{inf}). This is mathematically defined as the temporal delta (Δt) between the ingestion of raw sensor telemetry at the edge and the algorithmic generation of the subsequent control or alert signal.

4.3 Firmware Resilience Evaluation (FineObfuscator)

The efficacy of the FineObfuscator framework (detailed in Section 3.4) was evaluated against two primary constraints: the successful mitigation of Reverse Engineering Attacks (REAs) and the preservation of deterministic execution speeds on the flight controller.

By strategically limiting Control Flow Graph (CFG) obfuscation to isolated, context-sensitive security modules, FineObfuscator successfully thwarted standard decompilation pipelines (e.g., IDA Pro, Ghidra) deployed during the red-team evaluation. Crucially, unlike traditional whole-binary packers which introduce prohibitive memory bloat, the context-sensitive approach introduced negligible computational overhead. From the observed results, FineObfuscator, is on average 25.6% faster than full obfuscation.

4.4 Generative Threat Emulation Results (Net-GPT)

As defined in Section 3.3, the Net-GPT framework was deployed to synthetically generate MITM network payloads, acting as an advanced adversary against the UAV-GCS link. The evaluation of this generative adversary focuses on its payload prediction accuracy—its ability to seamlessly spoof protocol-compliant TCP/UDP sequences.

Experimental findings underscore the severe threat posed by generative AI in offensive cybersecurity. When imitating real, dynamic UAV-GCS interactions, the Llama-2-13B and Llama-2-7B backbones achieved remarkable payload generation accuracies of 95.30% and 94.09%, respectively. Highly compressed, fine-tuned models such as GPT-2 retained 74.16% of the predictive accuracy of the 13B model while operating at a fraction of the computational cost, rendering them highly dangerous for edge-deployed spoofing attacks.

Empirical findings (detailed in Table 4.1 and Table 4.2) indicate that scaling the fine-tuning dataset volume (from 20k to 100k packets) yields marginal improvements in

overall generative accuracy. However, dataset scaling significantly enhances the prediction of highly dynamic network parameters, specifically Sequence (`Seq`) and Acknowledgment (`Ack`) numbers. This precision renders the resulting MITM attacks highly resistant to traditional heuristic network monitors.

Table 4.1: Generative payload accuracy of Net-GPT (Llama-2-7B, 1 epoch) across varying dataset sizes (%).

Model Config.	**Src_IP**	**Dst_IP**	**Src_Port**	**Dst_Port**	**Flag**	**Seq#**	**Ack#**	**Length**	**Overall Avg.**
Llama-2-7B (20k)	92.50	95.00	93.75	95.00	83.75	92.50	80.00	93.75	90.78
Llama-2-7B (40k)	100.00	100.00	100.00	100.00	85.54	95.18	90.36	98.79	96.23
Llama-2-7B (60k)	100.00	100.00	100.00	100.00	86.25	98.75	93.75	97.50	97.03
Llama-2-7B (80k)	96.29	97.53	96.29	97.53	80.24	93.82	88.88	95.06	93.20
Llama-2-7B (100k)	97.53	97.53	97.53	97.53	74.07	96.29	86.42	98.76	93.21
Mean	97.26	98.01	97.51	98.01	81.97	95.31	87.88	96.77	94.09
Std. Deviation	3.11	2.08	2.64	2.08	4.98	2.39	5.14	2.27	2.53

Table 4.2: Generative payload accuracy of Net-GPT (Llama-2-13B, 1 epoch) across varying dataset sizes (%).

Model Config.	**Src_IP**	**Dst_IP**	**Src_Port**	**Dst_Port**	**Flag**	**Seq#**	**Ack#**	**Length**	**Overall Avg.**
Llama-2-13B (20k)	98.85	98.85	98.85	98.85	88.51	96.55	87.36	100.00	95.98
Llama-2-13B (40k)	97.59	97.59	97.59	97.59	85.54	96.39	87.95	98.80	94.88
Llama-2-13B (60k)	98.82	98.82	98.82	98.82	84.71	97.65	95.29	98.82	96.47
Llama-2-13B (80k)	97.68	97.68	97.68	97.68	77.61	92.54	91.05	98.51	93.47
Llama-2-13B (100k)	98.85	98.85	98.85	98.85	80.46	97.70	93.10	98.85	95.69

4.5 Distributed Defense Evaluation (Aero-LLM)

To counter adversarial exploits, the Aero-LLM framework utilizes distributed temporal models to detect structural network and kinematic anomalies. The real-time evaluation logic governing the TimesNet sentinel is formalized in Algorithm 4.1.

Algorithm 4.1: Real-Time Temporal Anomaly Detection Pipeline

Input: $\mathscr{M}$: Fine-Tuned Temporal Model (e.g., TimesNet)
$\mathscr{D}$: Real-Time Telemetry/Network Stream
$\mathscr{A}$: Defined Anomaly Threshold Ratio
Output: α: Accuracy, π: Precision, ρ: Recall, ϕ: F1-Score

1. $\mathscr{P} \leftarrow \emptyset$; $\mathscr{G} \leftarrow \emptyset$
2. **foreach** $window \in \mathscr{D}$ **do**
3. $\quad x_{seq}, y_{true} \leftarrow window$
4. $\quad y_{pred} \leftarrow \mathscr{M}.predict(x_{seq})$
5. $\quad$ Append y_{pred} to $\mathscr{P}$; Append y_{true} to $\mathscr{G}$
6. **end**
7. $threshold \leftarrow \text{percentile}(\text{MSE\_Loss}, 100 - \mathscr{A})$
8. $\mathscr{P}_{flagged} \leftarrow (\text{MSE}(\mathscr{P} - \mathscr{G}) > threshold)$

`/* Calculate Windowed Evaluation Metrics */`

9. $TP \leftarrow |\mathscr{P}_{flagged} \cap \mathscr{G}_{anomalous}|$
10. $FP \leftarrow |\mathscr{P}_{flagged} \setminus \mathscr{G}_{anomalous}|$
11. $FN \leftarrow |\mathscr{G}_{anomalous} \setminus \mathscr{P}_{flagged}|$
12. $\alpha \leftarrow \frac{TP+TN}{Total}$; $\pi \leftarrow \frac{TP}{TP+FP}$; $\rho \leftarrow \frac{TP}{TP+FN}$
13. $\phi \leftarrow 2 \cdot \frac{\pi \cdot \rho}{\pi + \rho}$ `/* F1-Score */`
14. **return** α, π, ρ, ϕ

4.5.1 Defensive Network Triage Performance

For defensive network screening, lightweight models (OPT-350M and OPT-1.3B) were deployed at the edge to parse incoming packets and predict legitimate sequence

flows, successfully identifying adversarial MITM deviations. As evidenced in Table 4.3, both models achieved near-perfect accuracy (100%) in verifying source and destination ports, and over 98% accuracy in sequence validation. The deployment of OPT-350M proves particularly viable for onboard execution, achieving zero-error processing on 53.23% of complex network frames while minimizing computational latency.

Table 4.3: Accuracy of Predicted Fields (in %)

Field	**SFT & RLHF**
OPT-350M	
sport	100.00
dport	100.00
flags	99.85
seq	98.45
ack	48.10
length	99.95
0 errors: 53.23, 1 error: 45.29,	2 errors: 1.11; 3 errors: 0.37, 4+ errors: 0
OPT-1.3B	
sport	100.00
dport	100.00
flags	99.45
seq	98.34
ack	53.97
length	99.63
0 errors: 47.30, 1 error: 51.80,	2 errors: 0.85; 3 errors: 0.05, 4+ errors: 0

4.6 Qualitative Analysis: The "Why" Factor

Beyond purely quantitative metrics, the foundational advantage of utilizing Large Language Models within the Aero-LLM framework is their inherent capacity for semantic explainability. Traditional Deep Neural Networks (DNNs) employed for anomaly detection operate as opaque "black boxes"; they successfully flag statistical outliers but fail to contextualize the root cause of the system failure.

In contrast, the instruction-tuned models within Aero-LLM decode complex multivariate deviations into human-readable diagnostic rationales. For example, rather than merely outputting a binary anomaly flag, the system contextualizes the event: *"Anomaly detected: GPS coordinate deviation strongly correlates with external RF signal degradation, highly indicative of active GNSS spoofing rather than nominal sensor drift."* This qualitative "Why" factor fundamentally transforms the role of AI in autonomous aviation, evolving it from a passive intrusion detection mechanism into an active, transparent decision-support system capable of guiding deterministic, failsafe recovery protocols.

4.6.1 Kinematic Anomaly Detection Resilience (TimesNet)

While edge-deployed LLMs screen the network layer, the TimesNet model continuously monitors the physical kinematic layer via MAVLink μORB topics. To rigorously validate the cyber-physical resilience of this temporal sentinel, the baseline MAVLink telemetry was subjected to four classes of synthetic adversarial transformations designed to mimic realistic sensor degradation and spoofing:

1. **Deterministic Periodic Injection:** Forcing anomalies at fixed temporal intervals (e.g., every n-th telemetry frame, where $n = 5$) to simulate cyclic sensor faults.

2. **Targeted Feature Manipulation:** Implementing controlled, random modifications to specific, highly correlated sensor axes.

3. **Distributional Shifts:** Systematically varying the underlying dataset distribution to simulate severe operational deviations (e.g., abrupt weather changes).

4. **Poisson-Distributed Injection:** Incorporating realistic, stochastically distributed anomaly occurrences mirroring the unpredictable nature of active cyber-attacks.

As defined computationally in Algorithm 4.1, anomalies are mathematically flagged when the Mean Squared Error (MSE) of the prediction exceeds a dynamically calculated percentile threshold:

$$\text{Loss}_{MSE} = \text{MSE}(\mathbf{X}_{predicted} - \mathbf{X}_{groundtruth}) \tag{4.2}$$

$$\text{Threshold} = \text{Percentile}(\text{Loss}_{MSE}, 100 - \text{Anomaly\_Ratio}) \tag{4.3}$$

When subjected to the deterministic periodic injection ($n = 5$), the TimesNet architecture accurately flagged the deviations the moment the reconstruction loss exceeded the threshold (illustrated in Fig. 4.1 and Fig. 4.2). Furthermore, subjecting specific high-frequency kinematic axes (e.g., `accelerometer_m_s2_2`) to severe stochastic variance to simulate environmental noise and sensor degradation demonstrated remarkable algorithmic resilience. As evidenced by the stability of the metrics plotted in Fig. 4.3b, the structural variance induced negligible degradation across overall Precision, Recall, and F1-Scores.

4.6.2 Inference Latency and Hardware Optimization

To satisfy the strict SWaP constraints of the onboard NVIDIA Jetson Nano edge unit, the relationship between inference batch size and computational latency was empirically quantified.

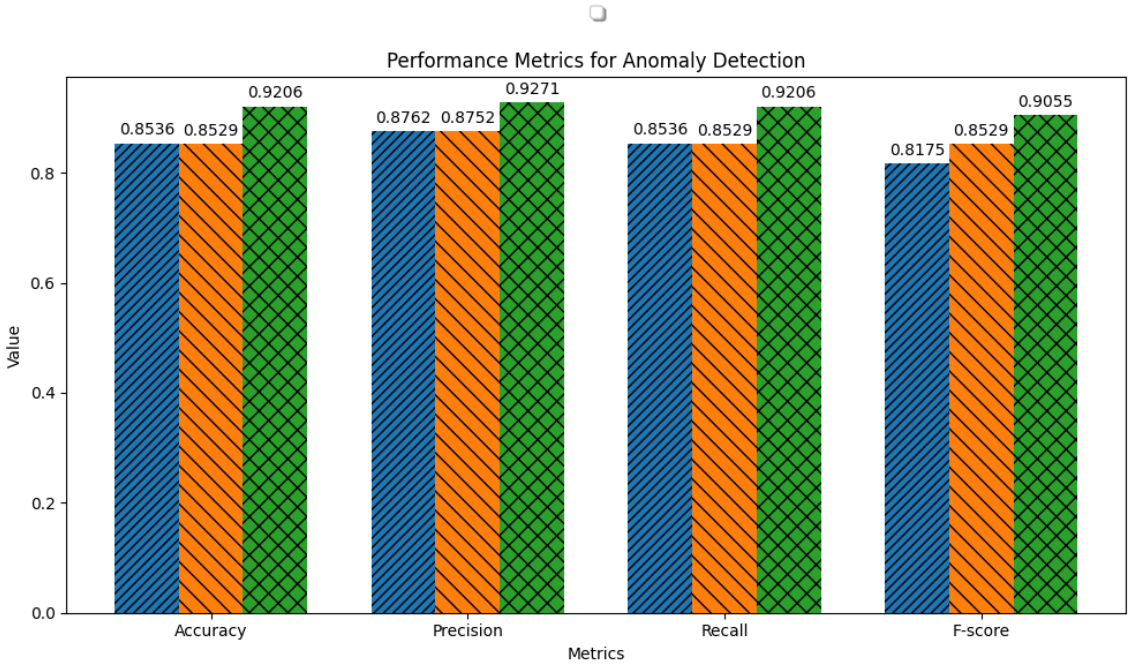


Figure 4.1: TimesNet Anomaly Detection Metrics under periodic injection ($n = 5$).

We executed the fine-tuned TimesNet model on Nvidia Jetson Nano and a Ubuntu Server equipped with RTX 4090 GPU. The elapsed time taken to infer and the runtime parameters are recorded in Table 4.4.

Table 4.4: TimesNet Inference Latency.

Hardware	**Input Records**	**Elapsed Time (seconds)**	**Latency (msec)**
NVidia Jetson Nano	21480	2630.40	122.46
Ubuntu Server w/RTX 4090	21480	534.00	24.86

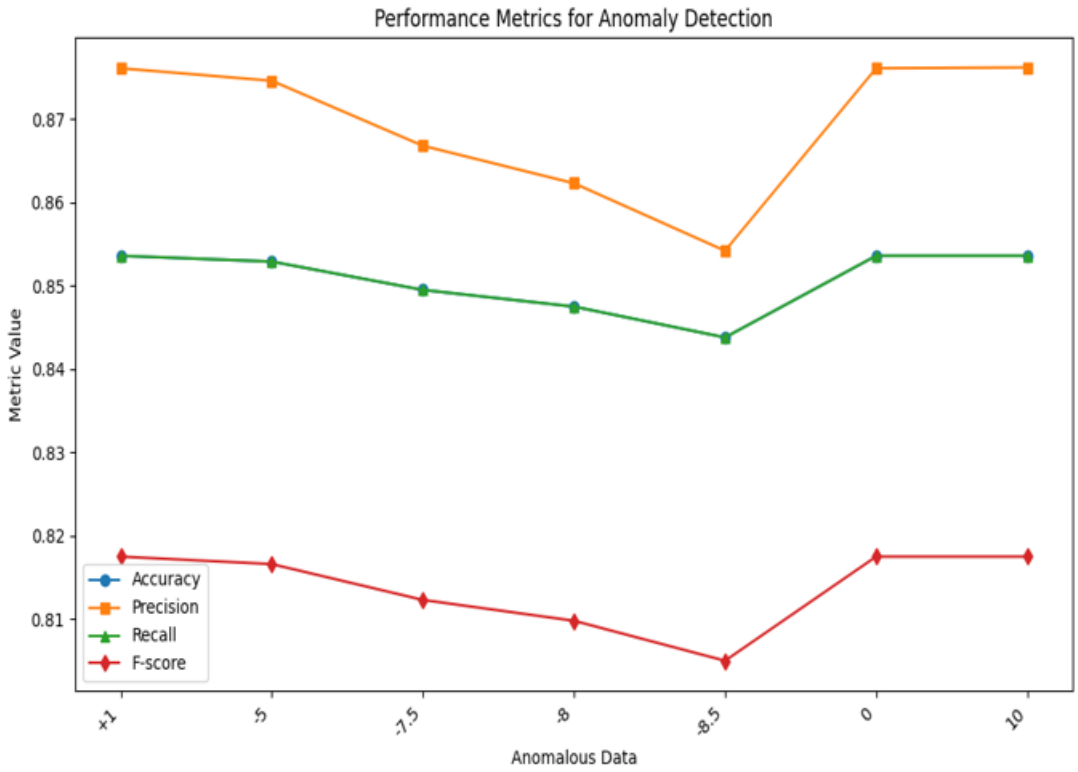


Figure 4.2: Correlation between anomalous data injection volume and resultant detection metrics.

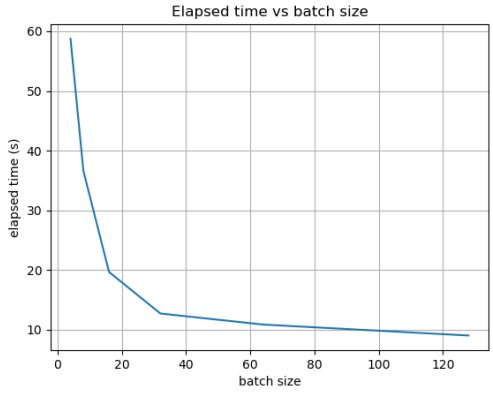
In the next experiment, an arbitrary adversarial value (-8.5) was injected into a specific sensor channel, and the anomaly detection pipeline was executed across varying batch sizes while preserving all other hyperparameters. Regardless of the batch size, the model maintained strict deterministic accuracy (Accuracy: 0.8438, Precision: 0.8542, Recall: 0.8438, F1-Score: 0.8050).

However, as detailed in Table 4.5 and visualized in Fig. 4.3a, the batch size played a critical role in system latency.

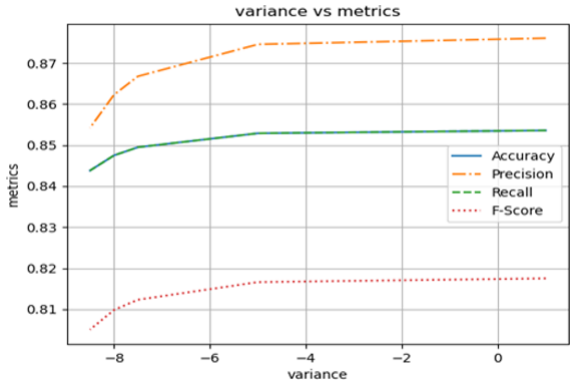
The empirical data indicates a logarithmic decay in latency improvements. Increasing the batch size to 48 and above significantly reduces the elapsed time required to

Table 4.5: Impact of Batch Size on TimesNet Inference Latency.

Batch Size	Average Elapsed Time (seconds)
128	9.08
64	10.89
32	12.71
16	18.21
8	34.29
4	61.91



(a) Inference Latency vs. Batch Size



(b) Detection Metrics under High Variance

Figure 4.3: Comparative analysis of TimesNet hardware performance and cyber-physical resilience.

process the telemetry windows. However, scaling the batch size beyond 64 yields heavily diminishing returns regarding latency reduction, while continuing to consume strictly limited onboard GPU memory. Consequently, a calibrated batch size of 64 provides the optimal equilibrium between rapid real-time detection and edge-memory preservation.

4.6.3 Predictive State Forecasting (Time-LLM)

Beyond real-time anomaly detection, anticipating future cyber-physical degradation is critical for autonomous survivability. To evaluate the predictive capabilities of the autonomy stack, the fine-tuned Time-LLM model was tasked with forecasting future kinematic states based on real-time mission sequences derived from the HGDAVE test dataset.

The forecasting architecture demonstrated exceptional proficiency in modeling nominal flight trajectories over a projected horizon. The evaluation yielded a highly favorable Test Loss of *0.1068* and a strictly bounded Mean Absolute Error (MAE) of *0.2587*. Overall, the Time-LLM model achieved a robust predictive accuracy exceeding 82%, validating the efficacy of the Aero-LLM framework in reliably estimating future states to preemptively trigger failsafe (e.g., Return-to-Launch) protocols before critical system failures occur.

CHAPTER FIVE

CONCLUSION AND FUTURE WORK

5.1 Conclusion

The integration of artificial intelligence into Unmanned Aerial Vehicles (UAVs) presents a critical dichotomy: while autonomous capabilities are strictly required for complex missions, the reliance on exposed communication channels and heavily constrained onboard computing renders these vehicles highly vulnerable to cyber-physical threats. To address these systemic vulnerabilities, this dissertation introduces Aero-LLM, a novel, cyber-physically co-designed framework that embeds Large Language Models (LLMs) and advanced temporal architectures directly into the distributed flight control stack.

By leveraging a tiered Edge-Fog-Cloud distributed architecture, this research successfully resolves the computational bottleneck of deploying massive generative models on SWaP-constrained microcontrollers. Comprehensive empirical evaluations confirm that the Aero-LLM framework—augmented by PEFT and context-sensitive obfuscation—achieves state-of-the-art performance in real-time anomaly detection, kinematic forecasting, and network triage. Ultimately, this research overcomes traditional threshold-based detection limitations, establishing a scalable, intelligent, and highly resilient control architecture for the next generation of autonomous flight operations.

5.2 Summary of Contributions

This dissertation successfully fulfilled the foundational research objectives defined in Section 1.9, resulting in the following original contributions to the field of computer science and autonomous systems:

- **Synthesis of the HGDAVE Cyber-Physical Dataset:** This research contributed a highly curated, multi-modal dataset bridging the gap between kinematic flight telemetry and cyber-security. By utilizing Software-in-the-Loop (SITL), Hardware-in-the-Loop (HITL), protocol fuzzing, and ethical hacking exploits, HGDAVE provides a rigorous foundation for training and benchmarking UAV anomaly detection systems.

- **Generative Adversarial Threat Modeling (Net-GPT):** By deploying fine-tuned LLMs at the network edge, this research successfully modeled advanced zero-day MITM attacks. This generative framework proved that adversaries can autonomously synthesize and inject protocol-compliant network packets to deceive traditional intrusion detection systems.

- **Domain-Specific LLM Optimization:** The methodology successfully adapted foundational language models for complex cyber-physical tasks. By applying PEFT, Quantized Low-Rank Adaptation (QLoRA), and RLHF, massive models were optimally compressed without sacrificing deterministic accuracy.

- **Firmware Resilience and Obfuscation:** The integration of FineObfuscator successfully secured the underlying flight software. By applying context-sensitive control flow graph (CFG) obfuscation, the methodology thwarted Reverse Engineering Attacks (REAs) while preserving the deterministic execution speed required by the flight controller.

- **Empirical Validation of Edge Deployability:** Through rigorous latency profiling and batch-size optimization, the Aero-LLM framework empirically proved that high-fidelity temporal anomaly detection (TimesNet) and state forecasting

(Time-LLM) can be executed within strict sub-second latency bounds on resource-constrained edge hardware (e.g., NVIDIA Jetson Nano).

5.3 Future Research Trajectories

While the Aero-LLM framework establishes a robust foundation for secure autonomous flight, the rapid evolution of generative artificial intelligence presents several promising vectors for future research:

Autonomous Policy Composition and Tool-Use: Currently, the Aero-LLM architecture operates primarily as a diagnostic sentinel and state forecaster. Future iterations will integrate an instruction-following LLM acting as a semantic policy composer (analogous to the Toolformer paradigm). Upon detecting a cyber-physical anomaly, this composer would autonomously invoke verified system application programming interfaces (APIs)—such as `recalculate_trajectory()`, `isolate_sensor_bus()`, or `initiate_secure_RTL()`—transitioning the system from passive detection to active, autonomous threat mitigation.

Multi-Modal Vision-Language Integration: The current anomaly detection pipeline relies strictly on numerical time-series telemetry (MAVLink/μORB) and network packet headers. Extending this framework to support multi-modal sensory inputs would profoundly enhance environmental context. Future research should investigate fusing high-frequency kinematic telemetry with real-time optical and infrared camera feeds utilizing Vision-Language Models (VLMs). This would allow the LLM to cross-validate sensor drift against visual odometry, thereby identifying sophisticated GNSS spoofing attacks with unprecedented accuracy.

Multi-Agent Swarm Collaboration: Future research aims to scale the Aero-LLM framework from single-vehicle operations to multi-UAV swarm intelligence. By deploying localized, specialized LLMs on individual drones within a swarm, the vehicles could collaboratively execute distributed terrain mapping, decentralized threat analysis, and dynamic mission reallocation in highly contested, communication-denied environments.

Agentic Workflows for Failsafe Execution: Finally, future architectures should explore fully autonomous agentic loops. Rather than operating strictly as a sequential inference engine, an agentic LLM workflow would allow the UAV to continuously reason, plan, execute, and self-reflect upon its actions. By developing specialized LLM agents capable of autonomous triage and safe-landing execution, the operational autonomy and survivability of unmanned aerial systems will be significantly advanced.

APPENDIX A

TIME-LLM FINE TUNING PROMPT

The following JSON shows the prompt to the LLM:

"prompt":

"task Provided below is a question detailing key attributes of a tcp packet. Preceding this question is a series of tcp packets, which together form the context. Please continue the sequence by providing the subsequent tcp packet that follows the question, serving as the answer. Formulas: Next Expected Sequence Number: Next 'seq' = Current 'ack'. Next 'ack' = Current 'seq' + Current 'len'. If 'len' = 0:Next 'ack' = Current 'seq' + 1

Context ['sport': None, 'dport': None, 'ack': None, 'flags': None, 'length': None, 'seq': None, 'sport': None, 'dport': None, 'ack': None, 'flags': None, 'length': None, 'seq': None, 'sport': None, 'dport': None, 'ack': None, 'flags': None, 'length': None, 'seq': None]

Question 'sport': 53970, 'dport': 4560, 'ack': 1, 'flags': 'A', 'length': 0, 'seq': 1",

"chosen": "'sport': 53970, 'dport': 4560, 'ack': 1, 'flags': 'AP', 'length': 93, 'seq': 1",

"rejected": "bad answer"

APPENDIX B

HARDWARE SPECIFICATIONS

The table B.1 offers a comparative analysis of the hardware specifications and designated architectural roles across three distinct computing platforms utilized in the Aero-LLM framework.

Table B.1: Hardware Specifications Comparison

	Pixhawk FC	**Jetson Nano Orin**	**Ubuntu + RTX 4090**
GPU	None	512-core Ampere, 16 Tensor Cores	16,384 CUDA Cores
VRAM	2MB/1MB	4 GB	24 GB
CPU	Cortex-M7 480MHz	Quad-core A57 1.7GHz	Xeon W 3.5GHz
Power	<5W	5–10W	∼450W
Role	Flight Controller	Onboard LLM (Tier 1)	Edge/Cloud (Tier 2/3)

APPENDIX C

DRONE MISSION WORKFLOW

A mission is a set of way points and UAV executes the mission as shown in section 1.3.2 and illustrated in Figure C.1.

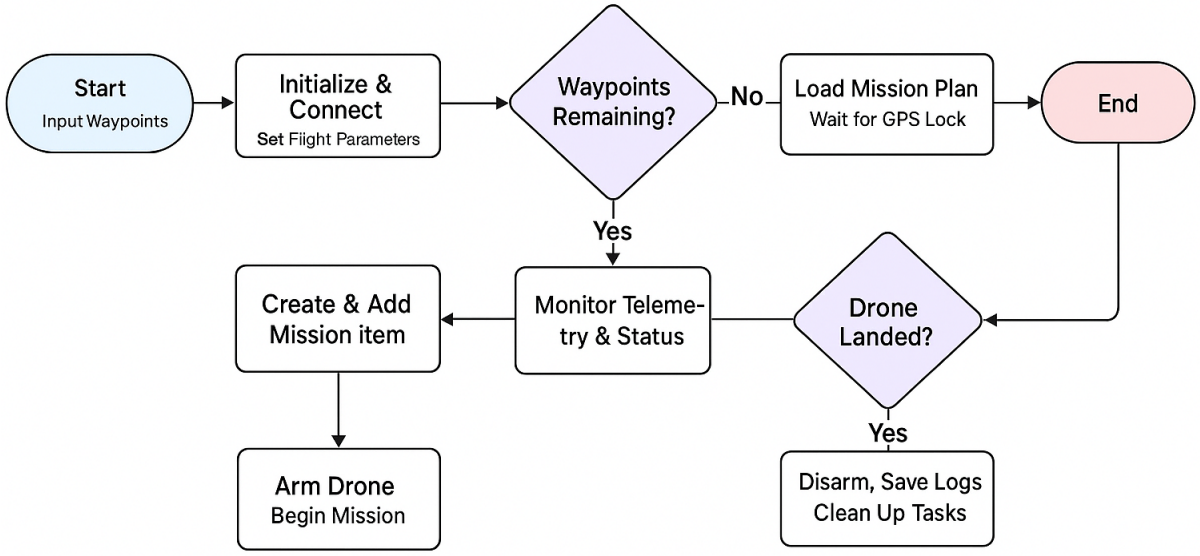


Figure C.1: Mission Flowchart.

Mission Categories for data collection: We categorize the missions in the table below and generate sensor data accordingly.

Category	Purpose
Baseline	Standard operation with default parameters
Speed Control	Test landing and takeoff speed variations
Altitude Control	Test takeoff altitude variations
Combined Parameters	Stress test with multiple randomized parameters
Orientation/Heading	Test yaw error tolerance and heading modes
Navigation Precision	Test waypoint acceptance radius
Acceleration Limits	Test vertical acceleration constraints

Figure C.2: Mission Categories.

APPENDIX D

PUBLICATIONS

I published the following research papers in various conferences.

1. **B. Dharmalingam**, B. Piggott, G. Feng, R. Mukherjee, and A. Liu. Aero-LLM: A Distributed Framework For Secure UAV Communication and Intelligent Decision-making. In Proceedings of the 33rd IEEE International Conference on Computer Communications and Networks Conference (IEEE ICCCN'24).

2. B. Piggott, S. Patil, G. Feng, I. Odat, R. Mukherjee, **B. Dharmalingam**, and A. Liu. Net-GPT: A LLM-empowered Man-in-the-middle Chatbot for Unmanned Aerial Vehicle. In 2023 IEEE/ACM Symposium on Edge Computing (IEEE/ACM SEC'23).

3. **B. Dharmalingam**, I.Odat, R.Mukherjee, B.Piggott, and A.Liu. HGDAVE: Heterogeneous Generative Dataset for UASes. In Proceedings of the 1st IEEE Conference on Mobility: Operations, Services, and Technologies (IEEE MOST'23).

4. **B. Dharmalingam**, A. Liu, S. Ganesan, and S. Roy. FineObfuscator: Defeating Reverse Engineering Attacks with Context-sensitive and Cost-efficient Obfuscation for Android apps. In 22nd Annual IEEE International Conference on Electro Information Technology (IEEE EIT'22)

5. A.Liu, A.Ali, M.Hua, **B.Dharmalingam**. "IoTVerif: Automatic Verification of SSL/TLS Certificate for IoT Applications". IEEE Access

REFERENCES

[1] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, "Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges," *IEEE Access*, vol. 7, pp. 48572–48634, 2019.

[2] H. Xu, W. Yu, D. Griffith, and N. Golmie, "A Survey on Industrial Internet of Things: A Cyber-Physical Systems Perspective," *IEEE Access*, vol. 6, pp. 78238–78259, 2018.

[3] P. Radanliev, D. De Roure, R. Nicolescu, M. Huth, and O. Santos, "Artificial intelligence and the internet of things in industry 4.0," *CCF Transactions on Pervasive Computing and Interaction*, vol. 3, no. 3, pp. 329–338, 2021.

[4] PX4, "PX4 Auto Pilot." `https://px4.io/`, 2025. Accessed: August 10, 2025.

[5] Apache, "NuttX." `https://nuttx.apache.org/docs/latest/`, 2025. Accessed: February 7, 2026.

[6] Mavlink, "MAVSDK." `https://mavsdk.mavlink.io/main/en/index.html`, 2026. Accessed: Feb 28, 2026.

[7] W. Lee, B. Alkouz, B. Shahzaad, and A. Bouguettaya, "Package Delivery Using Autonomous Drones in Skyways," in *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*, UbiComp/ISWC '21 Adjunct, (New York, NY, USA), p. 48–50, Association for Computing Machinery, 2021.

[8] A. Tahir, J. Böling, M.-H. Haghbayan, H. T. Toivonen, and J. Plosila, "Swarms of Unmanned Aerial Vehicles — A Survey," *Journal of Industrial Information Integration*, vol. 16, p. 100106, 2019.

[9] Mavlink, "Mavlink Protocol Overview." `https://mavlink.io/en/about/overview.html`, 2025. Accessed: Sep 22, 2025.

[10] Wikipedia, "World Geodetic System." `https://en.wikipedia.org/wiki/World_Geodetic_System`, 2026. Accessed: February 22, 2026.

[11] D. Bank, N. Koenigstein, and R. Giryes, *Autoencoders*, pp. 353–374. Cham: Springer International Publishing, 2023.

[12] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

[13] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[14] D.-N. Bui, T. H. Khuat, M. D. Phung, T.-H. Tran, and D. L. Tran, "Model Predictive Control for Optimal Motion Planning of Unmanned Aerial Vehicles," 2024.

[15] Z. Xu, Y. Zhang, H. Chen, Y. Hu, and L. Wang, "Research on precise route control of unmanned aerial vehicles based on physical simulation systems," *Results in Physics*, vol. 56, p. 107200, 2024.

[16] V. Beliaev, N. Kunicina, A. Ziravecka, M. Bisenieks, R. Grants, and A. Patlins, "Development of Adaptive Control System for Aerial Vehicles," *Applied Sciences*, vol. 13, no. 23, 2023.

[17] A. Montaruli, R. Patriarca, and D. Taurino, "How Cyber-Resilient Are Unmanned Aircraft Systems? A Systematic Meta-Review," *Aerospace*, vol. 13, no. 2, 2026.

[18] LLM, "Llm." `https://en.wikipedia.org/wiki/Large_language_model`, 2025. Accessed: August 24, 2025.

[19] OpenAI, "Gpt-5." `https://openai.com/index/introducing-gpt-5/`, 2025. Accessed: August 26, 2025.

[20] DeepSeek-AI, "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning," 2025.

[21] xAI, "Grok 4." `https://docs.x.ai/docs/introduction`, 2025. Accessed: August 26, 2025.

[22] Google, "Genie 3." `https://genie3.ai/`, 2025. Accessed: August 26, 2025.

[23] Meta, "Llama 4." `https://www.llama.com/docs/model-cards-and-prompt-formats/llama4/`, 2025. Accessed: August 26, 2025.

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2019.

[25] OpenAI, "GPT." `https://openai.com/index/language-unsupervised/`, 2025. Accessed: August 26, 2025.

[26] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[27] S. Javaid, H. Fahim, B. He, and N. Saeed, "Large Language Models for UAVs: Current State and Pathways to the Future," *IEEE Open Journal of Vehicular Technology*, vol. 5, pp. 1166–1192, 2024.

[28] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan, and Q. Wen, "Time-LLM: Time Series Forecasting by Reprogramming Large Language Models," in *International Conference on Learning Representations (ICLR)*, 2024.

[29] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis," in *International Conference on Learning Representations (ICLR)*, 2023.

[30] J. Whelan, T. Sangarapillai, O. Minawi, A. Almehmadi, and K. El-Khatib, "UAV Attack Dataset," 2020.

[31] B. Dharmalingam, I. Odat, R. Mukherjee, B. Piggott, and A. Liu, "Heterogeneous generative dataset for uases," in *2023 IEEE International Conference on Mobility, Operations, Services and Technologies (MOST)*, pp. 229–230, 2023.

[32] B. Dharmalingam, B. Piggott, G. Feng, R. Mukherjee, and A. Liu, "Aero-LLM: A Distributed Framework for Secure UAV Communication and Intelligent Decision-Making," in *The 33rd International Conference on Computer Communications and Networks (ICCCN 2024)*, 2024.

[33] I. Lopez-Sanchez and J. Moreno-Valenzuela, "PID control of quadrotor UAVs: A survey," *Annual Reviews in Control*, vol. 56, p. 100900, 2023.

[34] S. Rezwan and W. Choi, "Artificial Intelligence Approaches for UAV Navigation: Recent Advances and Future Challenges," *IEEE Access*, vol. 10, pp. 26320–26339, 2022.

[35] M. AI, "Llama 2: Enhancing large language models." `https://huggingface.co/blog/llama2`, 2023. Accessed: March 28, 2024.

[36] A. Dadheech and M. Bhavsar, "AirNav: A Computation-driven Cloud-based Solution for the Drone Path Planning Utilizing Machine Learning," *Procedia Computer Science*, vol. 258, pp. 1688–1697, 2025.

[37] S. Yang, W. Yu, Z. Liu, and F. Ma, "A Robust Hybrid Iterative Learning Formation Strategy for Multi-Unmanned Aerial Vehicle Systems with Multi-Operating Modes," *Drones*, vol. 8, no. 8, 2024.

[38] C.-P. Wu, N.-K. Hsieh, L.-R. Chen, and V. E. Balas, "Enhancing Longitudinal Flight Stability of Electric-powered Micro Unmanned Aerial Vehicles Using Fuzzy Proportional-integral-derivative Control," *International Journal of Control, Automation and Systems*, vol. 23, no. 7, pp. 2132–2141, 2025.

[39] G. Tong, N. Jiang, L. Biyue, Z. Xi, W. Ya, and D. Wenbo, "UAV navigation in high dynamic environments: A deep reinforcement learning approach," *Chinese Journal of Aeronautics*, vol. 34, no. 2, pp. 479–489, 2021.

[40] B. Lim, J. Arús-Pous, and S. Zohren, "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting," *International Journal of Forecasting*, vol. 37, no. 4, pp. 1748–1764, 2021.

[41] H. Zhou, S. Zhang, J. Peng, S. Zhang, and J. Li, "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting," in *AAAI*, pp. 11106–11115, 2021.

[42] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," 2022.

[43] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, and Q. Zhang, "Time-Series Anomaly Detection Service at Microsoft," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, (New York, NY, USA), p. 3009–3017, Association for Computing Machinery, 2019.

[44] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng, "Do As I Can, Not As I Say: Grounding Language in Robotic Affordances," 2022.

[45] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "RT-1: Robotics Transformer for Real-World Control at Scale," 2023.

[46] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, Q. Vuong, V. Vanhoucke, H. Tran, R. Soricut, A. Singh, J. Singh, P. Sermanet, P. R. Sanketi, G. Salazar, M. S. Ryoo, K. Reymann, K. Rao, K. Pertsch, I. Mordatch, H. Michalewski, Y. Lu, S. Levine, L. Lee, T.-W. E. Lee, I. Leal, Y. Kuang, D. Kalashnikov, R. Julian, N. J. Joshi, A. Irpan, B. Ichter, J. Hsu, A. Herzog, K. Hausman, K. Gopalakrishnan, C. Fu, P. Florence, C. Finn, K. A. Dubey, D. Driess, T. Ding, K. M. Choromanski, X. Chen, Y. Chebotar, J. Carbajal,

N. Brown, A. Brohan, M. G. Arenas, and K. Han, "RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control," in *Proceedings of The 7th Conference on Robot Learning* (J. Tan, M. Toussaint, and K. Darvish, eds.), vol. 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183, PMLR, 06–09 Nov 2023.

[47] M. Satyanarayanan, P. Bahl, R. Cáceres, and N. Davies, "The Case for VM-Based Cloudlets in Mobile Computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[48] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog Computing and Its Role in the Internet of Things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, pp. 13–16, 2012.

[49] S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He, "ZeRO: Memory Optimizations Toward Training Trillion Parameter Models," in *SC '20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020.

[50] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," 2021.

[51] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs," 2023.

[52] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys, "Unmanned Aircraft Capture and Control via GPS Spoofing," *Journal of Field Robotics*, vol. 31, no. 4, pp. 617–636, 2014.

[53] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. Othman, A. Khreishah, and M. Guizani, "Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges," *IEEE Access*, vol. 7, pp. 48572–48634, 2019.

[54] Y.-D. Lin, W.-H. Chan, Y.-C. Lai, C.-M. Yu, Y.-S. Wu, and W.-B. Lee, "Enhancing CAN security with MLbased IDS: Strategies and efficacies against adversarial attacks," *Computers & Security*, vol. 151, p. 104322, 2025.

[55] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 737–744, 2020.

[56] H. J. Hadi, Y. Cao, M. K. Khan, N. Ahmad, Y. Hu, and C. Fu, "UAV-NIDD: A Dynamic Dataset for Cybersecurity and Intrusion Detection in UAV Networks," *IEEE Transactions on Network Science and Engineering*, vol. 12, no. 4, pp. 2739–2757, 2025.

[57] Q. Zeng, A. Bashir, and F. Nait-Abdesselam, "UAVIDS-2025: A Benchmark Dataset for Intrusion Detection in UAV Networks Using Machine Learning Techniques," in *2025 IEEE Conference on Communications and Network Security (CNS)*, pp. 1–9, 2025.

[58] P. Iyenghar, "UAVThreatBench: A UAV Cybersecurity Risk Assessment Dataset and Empirical Benchmarking of LLMs for Threat Identification," *Drones*, vol. 9, no. 9, 2025.

[59] H. Kim, M. O. Ozmen, A. Bianchi, Z. B. Celik, and D. Xu, "PGFUZZ: Policy-Guided Fuzzing for Robotic Vehicles," in *Network and Distributed System Security Symposium (NDSS)*, pp. 1–18, 2021.

[60] Gazebo, "Gazebo simulation." `https://gazebosim.org/docs/all/getstarted/`, 2025. Accessed: August 10, 2025.

[61] ROS2, "Ros2 kilted." `https://docs.ros.org/en/kilted/index.html`, 2025. Accessed: August 10, 2025.

[62] B. Dharmalingam, A. Liu, S. Ganesan, and S. Roy, "FineObfuscator: Defeating Reverse Engineering Attacks with Context-sensitive and Cost-efficient Obfuscation for Android Apps," in *2022 IEEE International Conference on Electro Information Technology (eIT)*, pp. 368–374, 2022.

[63] Wireshark, "Wireshark Network Packet Analyzer." `https://www.wireshark.org/about`, 2026. Accessed: Feb 18, 2026.

[64] R. Yazdani Aminabadi, S. Rajbhandari, M. Zhang, A. A. Awan, C. Li, D. Li, E. Zheng, J. Rasley, S. Smith, O. Ruwase, and Y. He, "DeepSpeed Inference: Enabling Efficient Inference of Transformer Models at Unprecedented Scale," *arXiv preprint arXiv:2207.00032*, 2022.

[65] Meta, "Llama-2-7B." `https://huggingface.co/meta-llama/Llama-2-7b-hf`, 2023. Accessed: August 19, 2023.

[66] Meta, "Llama-2-13B." `https://huggingface.co/meta-llama/Llama-2-13b-hf/`, 2023. Accessed: August 19, 2023.

[67] OpenAI, "GPT 2." `https://huggingface.co/gpt2`, 2023. Accessed: August 19, 2023.

[68] OpenAI, "distilgpt2." `https://huggingface.co/distilgpt2`, 2023. Accessed: August 19, 2023.

[69] S. Mihai, M. Yaqoob, D. V. Hung, W. Davis, P. Towakel, M. Raza, M. Karamanoglu, B. Barn, D. Shetve, R. V. Prasad, H. Venkataraman, R. Trestian, and H. X. Nguyen, "Digital Twins: A Survey on Enabling Technologies, Challenges, Trends and Future Prospects," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2255–2291, 2022.

[70] Wikipedia, "Interpolation." `https://en.wikipedia.org/wiki/Interpolation`, 2026. Accessed: February 22, 2026.