Flyby F-11 System Constraints

Finley Holt

2025-12-26

Table of contents

1 Flyby F-11 System Constraints

1.1 Hardware Platform

Aircraft: Flyby Robotics F-11 Developer Quadcopter - 3kg payload capacity - NDAA-compliant (government/defense certified) - Open flight controller and GPU access

Compute: NVIDIA Jetson Orin NX 16GB - 50 TOPS (AI performance) - 1,024 CUDA cores, 32 Tensor cores (Ampere architecture) - 16GB unified memory (GPU/CPU shared)

Flight Controller: ArduPilot firmware - MAVLink protocol (similar to PX4, different parameter structure) - Uses `.parm` files for configuration - ArduPilot SITL available for simulation

Primary Control Interface: MQTT - Publish/subscribe architecture - Topic-based communication (e.g., `flyby/command/goto`) - Integrates with F-11 onboard systems

ISR Sensor Payloads (mission-configurable, one at a time):

- **Gremsy VIO**: 640p FLIR thermal + 4K RGB + 20x optical zoom
  - Primary use: ISR surveillance, target identification, day/night operations
- **RESEPI LiDAR**: 3D point cloud mapping system
  - Primary use: Terrain mapping, volumetric survey, obstacle detection
- **NextVision Raptor**: EO-IR gimbal with 1280×720p thermal
  - Primary use: Reconnaissance, target tracking, thermal detection

Note: Unlike project-drone (which uses T265/D455 RealSense cameras), the F-11 uses mission-specific ISR payloads optimized for military surveillance applications.

1.2 Software Stack

Autonomy Framework: ROS 2 Humble - Inter-component communication - Standard message types for sensor data, commands, telemetry

Flight Control Bridge: ArduPilot MAVLink + MQTT - Not using PX4/MAVSDK (unlike project-drone) - Requires `ardupilot_interface` and `mqtt_interface` packages

Simulation Environment: Gazebo + ArduPilot SITL - Docker containerized for reproducibility - GPU passthrough for vision model acceleration

Training Infrastructure: Naval Postgraduate School (NPS) Computing Cluster - Off-board training (not on Jetson) - Deploy frozen policies to Jetson for inference

1.3 Learning Approach

Paradigm: Reinforcement Learning constrained by SUMO ontology - Ontology provides formal knowledge representation (safety, semantics, mission structure) - RL provides adaptability and optimization - Integration points: state abstraction, reward shaping, action filtering

Ontology: SUMO (Suggested Upper Merged Ontology) subset - Full SUMO too large; curated subset for UAV domain - Extensions for drone-specific concepts (flight phases, spatial relations, safety bounds) - Advisor: Adam Pease (SUMO creator)

1.4 Development Constraints

Hardware Access: Via MCTSSA collaboration (timeline TBD) - Physical drone available for validation testing - Primary development in simulation until hardware access

Compute Budget: - Training: NPS cluster (no local GPU limitations) - Inference: Jetson Orin NX (real-time constraints, 16GB memory)

Simulation-to-Real: Critical requirement - Must validate in ArduPilot SITL before hardware deployment - Domain randomization / sim-to-real transfer techniques needed

1.5 Technical Decisions

RL Algorithm: TBD (likely PPO or SAC for continuous control)

Ontology Reasoning Engine: TBD (Pellet, HermiT, or custom - discuss with Adam Pease)

Policy Architecture: TBD (end-to-end neural, hierarchical, hybrid symbolic-learning)

Training Framework: TBD (Stable-Baselines3, RLlib, custom Gymnasium environment)

1.6 Success Criteria

1. Formally specified canonical problems (mission scenarios)
2. Domain vocabulary grounded in SUMO ontology
3. Trained policies that respect ontological constraints
4. Validated in ArduPilot SITL simulation
5. Successful deployment on physical F-11 hardware
6. Explainable behaviors (ontology provides interpretability)