Literature Review Synthesis: Ontology-Constrained Reinforcement Learning for Autonomous UAV Systems

Finley Holt

2025-12-25

Table of contents

1 Literature Review Synthesis: Ontology-Constrained Reinforcement Learning for Autonomous UAV Systems

Synthesis of Six Research Papers for Flyby-F11 Platform Development

Date: 2024-12-24

Purpose: Integrate findings from comprehensive literature review to inform ontology-constrained RL architecture for autonomous drone missions in GPS-denied, communications-limited environments

---

1.1 Executive Summary

This synthesis integrates insights from six research papers spanning ontology-driven reinforcement learning, autonomous robot dependability, UAV benchmarks, and hybrid decision-making architectures. The collective body of work provides compelling evidence for a **hybrid approach that combines formal ontological knowledge representation with reinforcement learning** to achieve safe, adaptive, and explainable autonomous UAV systems.

1.1.1 Key Consensus Findings

1. **Hybrid Methods Outperform Pure Approaches**: All papers converge on the superiority of combining knowledge-driven and data-driven methods over pure implementations of either paradigm.

2. **Safety Requires Formal Constraints**: Data-driven approaches alone (LLMs, pure RL) exhibit critical safety failures (30-65% collision rates in UAV-ON benchmark), necessitating formal safety constraints.

3. **Ontologies Enable Dependability**: Formal knowledge representation through ontologies provides explainability, verifiability, and runtime adaptation capabilities essential for safety-critical autonomous systems.
4. **RL Needs Structure**: Reinforcement learning without ontological guidance suffers from poor sample efficiency, safety violations, and difficulty generalizing to unseen situations.
5. **Mission-Intent Interpretation**: High-level semantic mission understanding requires structured knowledge representation that bridges the gap between abstract goals and concrete actions.

1.1.2 Validated Architecture for Flyby-F11

The literature strongly supports an **ontology-constrained reinforcement learning** architecture with:

- **SUMO-based upper ontology** providing foundational concepts
- **IEEE Autonomous Robotics ontology** extending to UAV domain
- **Mission-specific application ontologies** encoding objectives, constraints, and safety rules
- **Multi-agent RL** operating within ontology-defined safe action spaces
- **Runtime reasoning** for situation assessment and goal generation
- **Explainable autonomy** through semantic rule tracing

This approach addresses the "exploration-exploitation" dilemma in RL while maintaining formal safety guarantees and interpretability required for defense/government applications (MCTSSA collaboration).

1.2 Architecture Decision Framework

This section consolidates all literature findings into actionable architecture decisions for Flyby-F11.

1.2.1 Critical Decisions - Implementation Ready

Decision	Supporting Evidence	Confidence	Next Action
Adopt SUMO + IEEE AUR ontology	Papers 03, 04 (standards-based, proven)	HIGH	Download SUMO, study IEEE 1872.2
Use OWL 2 DL (not Prolog)	Paper 03 (open-world assumption better for robots)	HIGH	Install Protégé, OWLREADY2
Implement AGGM for unseen situations	Paper 02 (outperforms baselines, real-time capable)	HIGH	Prototype AGGM stages 1-6
Multi-agent RL with experience sharing	Paper 01 (jump-start training, modular)	HIGH	Set up shared replay buffer

Decision	Supporting Evidence	Confidence	Next Action
Collision avoidance as canonical problem	Papers 04, 05 (critical gap, 37-65% failure rates)	HIGH	Define ontology for spatial safety

1.2.2 Approaches to Validate Before Full Commitment

Approach	Promise	Risk	Validation Plan
Real-time OWL reasoning on Jetson	Papers suggest <100ms possible	Performance unknown on our hardware	Benchmark HermiT on Orin NX
Zero-shot LLM for high-level planning	Paper 05 shows 26% exploration success	45-65% collision rate unacceptable	Test with safety constraints
Hierarchical BT + ontology integration	Paper 03 mentions but limited detail	Integration complexity	Prototype simple BT-ontology bridge
Inverse RL for reward learning	Paper 06 recommends for complex rewards	May require extensive demonstrations	Small-scale test with flight data

1.2.3 Approaches to Avoid (Evidence-Based)

Approach	Why Avoid	Evidence	Alternative
Pure LLM control (like AOA)	65.5% collision rate	Paper 05: UAV-ON benchmark	LLM for planning, ontology for safety
Pure rule-based system	Doesn't scale, overly conservative	Paper 06: "scaling challenge"	Hybrid with RL for optimization
Pure black-box RL	Sample inefficient, unsafe	Papers 02, 05, 06: consistent failures	Ontology-constrained RL
Prolog for robotics	Closed-world assumption problematic	Paper 03: OWL preferred	OWL 2 DL with open-world
VLN paradigm for missions	Requires step-by-step instructions	Paper 05: ObjectNav more autonomous	Semantic goal-driven (ObjectNav)

1.2.4 Open Questions Requiring Investigation

Question	Impact on Architecture	How to Resolve	Timeline
Can HermiT reason at 10Hz on Orin NX?	Critical for real-time control	Benchmark with collision avoidance ontology	Week 2
How to handle sensor uncertainty in ontology?	Affects perception-ontology bridge	Prototype probabilistic extensions	Week 6
Optimal hierarchy: how many ontology levels?	Complexity vs. specificity tradeoff	Iterative testing with collision scenarios	Week 10
Experience sharing: same policy or different?	Multi-agent RL architecture	Literature suggests shared replay, separate policies	Week 8
SLAM integration with semantic ontology?	Indoor navigation without GPS	Research SOMA+SLAM approaches	Week 12

1.3 Thematic Analysis

1.3.1 Theme 1: The Hybrid Imperative

Evidence Across Papers:

Paper 06 (Autonomous Vehicles Survey): - Explicitly identifies hybrid methods as "clear trend and compelling direction for future research" - States pure knowledge-driven methods are "overly conservative" with "scaling challenges" - Notes pure data-driven methods have "poor interpretability" and "safety risks during exploration" - Recommends combining "systematic constraints and guidance" with "adaptability to complex environments"

Paper 03 (Ontology Survey for Robotics): - Demonstrates ontologies improve dependability through "explainability, fault tolerance, adaptability, safety, and verification" - Shows runtime ontology integration enables "alternative actions when primary approach fails" - Identifies challenge: "computational overhead" requires balancing with real-time constraints

Paper 02 (AGGM for Unseen Situations): - Proves ontology-guided goal generation outperforms pure RL (Q-learning, SARSA, DQN) in unseen situations - Demonstrates "real-time adaptation without retraining" - Shows backward reasoning enables "recovery to known states" when novel situations occur

Paper 05 (UAV-ON Benchmark): - Reveals pure learning approaches (LLM-based AOA) achieve only 7.30% success rate with 37-65% collision rates - Demonstrates gap between "exploration capability" (26.30% OSR) and "task completion" (7.30% SR) - Proves unconstrained approaches unsafe for real-world deployment

Paper 01 (Educational Ontology-RL): - Shows multi-agent RL with ontology-structured architecture enables "modular extensibility without retraining entire system" - Demonstrates experience sharing between ontology-guided agents improves training performance - Proves data transformation layer (ontology-based) enables "system flexibility across different platforms"

Synthesis: The unanimous consensus is that **hybrid architectures are not optional but essential** for safety-critical autonomous systems. Pure approaches fail to simultaneously achieve safety, adaptability, efficiency, and interpretability.

1.3.2 Theme 2: Ontological Knowledge as Safety Scaffold

Evidence Across Papers:

Paper 04 (Collision Avoidance Ontologies): - Formalizes spatial relationships, regulatory rules, and resolution maneuvers - Provides "safety envelope within which learning-based optimization occurs" - Enables "formal verifiability" and "explainability through semantic rule tracing" - Recommends ontology "eliminates unsafe actions from RL policy's action space"

Paper 03 (Ontology Survey): - Establishes ontologies enable "prediction and monitoring" through internal models - Shows formal reasoning about "preconditions and effects" before action selection - Identifies "belief maintenance" for handling uncertainty and inconsistencies - Emphasizes "execution monitoring to detect failures and trigger replanning"

Paper 02 (AGGM): - Implements concept importance weighting to prioritize "safety-critical observations" - Uses ontology to evaluate "significant change identification" triggering goal adaptation - Demonstrates "state similarity reward function" for safe recovery behaviors - Shows inference rules prevent actions that violate safety constraints

Paper 05 (UAV-ON): - Collision rates of 37-65% demonstrate critical need for safety constraints - Post-processing rules (scene boundary suppression, minimum altitude) reduce but don't eliminate failures - Pure LLM approaches lack "spatial awareness" leading to out-of-bounds violations - Fixed-step control (65.5% collision) worse than variable (45.0%), suggesting context-aware constraints needed

Paper 06 (Autonomous Vehicles): - Rule-based systems provide "compliance with traffic regulations and safety standards" - Knowledge-driven frameworks "guarantee vehicle remains in safe state within predefined range" - Pure RL exhibits "safety risks during exploration phase" and "lack of guaranteed bounds on behavior"

Synthesis: Ontological knowledge provides a **formal safety scaffold** that defines permissible state-action trajectories, eliminating dangerous behaviors a priori while preserving RL's ability to optimize within safe boundaries. This architectural pattern is consistent across all application domains (education, traffic control, robotics, autonomous vehicles, UAVs).

1.3.3 Theme 3: Multi-Level Reasoning and Hierarchical Decomposition

Evidence Across Papers:

Paper 01 (Educational Ontology-RL): - Ontology defines hierarchical knowledge structure: high-level concepts → mid-level topics → low-level primitives - Multi-agent RL assigns specialized agents at appropriate ontology nodes - Example: "linear function", "quadratic function" as mid-level; "domain", "range" as low-level - Enables "specialization vs. generalization" balance

Paper 03 (Ontology Survey): - Establishes three-level abstraction hierarchy: upper-level (SUMO, DOLCE) → domain ontologies → application ontologies - Identifies planning granularity levels: "high-level (mission) → mid-level (task) → low-level (action)" - Describes categorization process: "sensor fusion → classification → abstract recognition" - Shows hierarchical planning enables "multi-scale reasoning from strategic to tactical"

Paper 02 (AGGM): - Implements multi-stage process: observe → evaluate → identify change → reason → generate action → execute - Uses ontology schema with concepts (C) and relations (M) at different abstraction levels - Forward reasoning for high-level goals, backward reasoning for low-level recovery actions - Demonstrates importance weighting creates semantic hierarchy (highest → high → middle → low → lowest)

Paper 06 (Autonomous Vehicles): - Separates "behavior decision-making" (high-level strategies) from "motion planning" (detailed trajectories) - Hierarchical RL architectures recommended for multi-level decomposition - Game-theoretic approaches for multi-agent strategic planning - MDPs/POMDPs for mid-level decision under uncertainty

Paper 04 (Collision Avoidance): - Proposes "hierarchical decomposition: high-level ontological mission planning, low-level RL trajectory optimization" - Spatial reasoning at multiple scales: regions → local areas → objects - Threat assessment hierarchy: imminent collision → near miss → safe passage - Maneuver selection from general (avoidance) to specific (emergency climb, evasive turn)

Synthesis: **Hierarchical decomposition through ontological structure** is a recurring architectural pattern that enables: - Semantic abstraction from low-level sensor data to high-level mission understanding - Computational efficiency through appropriate granularity at each level - Specialization of RL agents to specific subtasks - Interpretability by separating strategic reasoning from tactical execution

1.3.4 Theme 4: Handling Unseen Situations and Generalization

Evidence Across Papers:

Paper 02 (AGGM): - Explicitly addresses "how agents adapt when they experience an unseen situation" - Automatic Goal Generation Model enables "creating new goals to handle unseen situations autonomously" - Backward reasoning allows agents to "take actions that plausibly result in experiencing previously known state" - Outperforms Q-learning, SARSA, DQN in unseen situations while handling seen situations equally well

Paper 05 (UAV-ON): - Test set includes "4 novel environments" and "unseen object categories" to evaluate generalization - OSR metric (26.30% for AOA-V) measures exploration capability across novel scenarios - Instance-level object descriptions require generalization beyond category memorization - Object-scene co-occurrence reasoning demands semantic understanding beyond position memorization

Paper 01 (Educational Ontology-RL): - Experience sharing between agents enables "jump-start" when new topics added - Modular architecture allows "adding educational topics without retraining

entire system" - Online RL ensures "system responds to shifting student behavior trends" - Data transformation layer enables "diverse data sources to interface with standardized architecture"

Paper 03 (Ontology Survey): - Open-world assumption (OWL) better handles "uncertainty and incomplete information in real-world environments" - Belief maintenance for updating knowledge in "dynamic environments" - Abductive reasoning to "infer most likely explanation for observations" - Adaptation through "runtime reconfiguration based on changing conditions"

Paper 06 (Autonomous Vehicles): - Identifies "generalization challenges" as key limitation of pure data-driven methods - "Distribution shift problems when encountering scenarios different from training" - Recommends "transfer learning across domains" and "continual learning without catastrophic forgetting" - Synthetic data generation to augment training with "rare scenarios" and "edge cases"

Paper 04 (Collision Avoidance): - Ontology provides "adaptability: new rules and contexts added without retraining models" - Handles "incomplete knowledge" through open-world reasoning - Recommends "custom inference engines" for contexts not anticipated in ontology - Suggests "continual learning: update ontology from failed episodes"

Synthesis: **Ontological knowledge provides semantic abstractions that enable generalization** beyond specific training scenarios. By encoding concepts, relations, and inference rules at appropriate abstraction levels, systems can: - Reason about novel situations using existing concepts (backward reasoning to known states) - Generate new goals dynamically rather than selecting from fixed sets - Transfer learned policies across contexts via shared ontological representations - Update knowledge bases incrementally without full retraining

1.3.5 Theme 5: Explainability and Verification for Safety-Critical Systems

Evidence Across Papers:

Paper 03 (Ontology Survey): - Identifies explainability as core benefit: "knowledge can be queried so humans understand why robot acts in certain way" - Formal reasoning enables "verification of system properties and behaviors" - Ontology-based decisions "can be traced and explained to operators" - Distinguishes "design-time verification" from "runtime monitoring"

Paper 04 (Collision Avoidance): - Emphasizes "formal verifiability: safety rules encoded declaratively can be formally verified" - Provides "explainability through semantic rule tracing" - Enables "regulatory alignment: direct encoding of aviation regulations ensures compliance by construction" - Recommends "runtime verification: monitor compliance during flight"

Paper 02 (AGGM): - Semantic Web Rule Language (SWRL) expresses inference rules transparently - Forward and backward reasoning chains provide audit trail - Ontology structure makes decision process "interpretable and verifiable" - Concept importance weights explicitly represent prioritization logic

Paper 06 (Autonomous Vehicles): - Lists "explainable AI (XAI)" as future research priority - Identifies "poor interpretability ('black box' problem)" as limitation of pure data-driven methods - Recommends "interpretable decision-making" and "human-understandable explanations" - Emphasizes need for "transparent neural networks" and "causal reasoning"

Paper 01 (Educational Ontology-RL): - Ontology provides "semantic organization" for "interpretable structure" - Rule-based assistance generation enables "transparency in content selection" -

Action vector components explicitly represent different assistance dimensions - System behavior traceable through ontology nodes and agent assignments

Paper 05 (UAV-ON): - LLM-based AOA generates "semantic action commands" that are interpretable - Structured prompts provide transparency into decision process - Collision failures "diagnosable" when ontological constraints are explicit - Contrast with "black box" neural policies that lack interpretability

Synthesis: **Formal ontological knowledge enables explainability and verification** essential for: - Safety-critical applications requiring auditable decision-making (defense, healthcare, transportation) - Regulatory compliance (FAA Part 107, NDAA requirements for government applications) - Debugging and failure analysis (understanding why system made specific decisions) - Human trust and acceptance (operators can understand and predict system behavior) - Formal verification (proving safety properties hold across all reachable states)

This capability is particularly critical for **flyby-f11's MCTSSA collaboration** where mission-intent interpretation and communications-denied operations require transparent, trustworthy autonomous decision-making.

1.3.6 Theme 6: Computational Efficiency and Real-Time Constraints

Evidence Across Papers:

Paper 03 (Ontology Survey): - Identifies "computational overhead" as key challenge for real-time systems - Recommends "cached inferences for time-critical decisions" and "incremental reasoning for efficiency" - Suggests "separate design-time and runtime reasoning" to manage computational load - Notes description logic provides "balance of expressiveness and computational tractability"

Paper 02 (AGGM): - Operates within 5-second action intervals in traffic control case study - SWRL evaluation exhibits "efficient rule-based reasoning (<100ms typical latency)" - Ontology size "typically <10 MB" much smaller than DNN weights - Memory footprint: ontology + rules "< 1 MB" compared to replay buffers

Paper 04 (Collision Avoidance): - Targets "<100ms per reasoning cycle for real-time control" - Notes Jetson Orin NX 16GB "allows loading full ontology + neural policy in unified memory" - Recommends "custom rule engine: RETE-based for real-time performance (may sacrifice completeness for speed)" - Suggests "pre-compile inference rules" and "parallel reasoning for independent goal candidates"

Paper 01 (Educational Ontology-RL): - Multi-agent architecture distributes computational load across specialized agents - Experience sharing reduces training time by providing "jump-start" to new agents - Data transformation happens once, enabling efficient reuse across agents - Policy gradient methods (DDPG, TD3, SAC) support continuous action spaces efficiently

Paper 05 (UAV-ON): - AOA-V achieves 85.65 average steps (longest exploration) demonstrating computational headroom - CLIP-H visual similarity matching is "efficient path following" - Multi-modal LLM (GPT-4o mini) operates at interactive speeds despite complexity - Fixed-step control (AOA-F) simplifies computation but reduces adaptability

Paper 06 (Autonomous Vehicles): - Identifies "computational resource constraints" and "real-time performance requirements" as challenges - Sampling-based planning (RRT, PRM) "effective in high-dimensional spaces" - Hybrid approaches require "managing computational resources for

both components" - Hierarchical architectures enable "clear separation" reducing per-component complexity

Synthesis: **Ontological reasoning is computationally efficient** compared to alternatives: - Ontology + inference rules have smaller memory footprint than deep neural networks - Rule-based reasoning operates in milliseconds, suitable for control loops - Hierarchical decomposition reduces complexity through appropriate granularity - Incremental reasoning and caching optimize repeated queries - Multi-agent specialization distributes computational load

For **edge-based autonomy on Jetson platforms** (Orin Nano Super 8GB for project-drone, Orin NX 16GB for flyby-f11), ontological reasoning fits within real-time constraints while providing formal guarantees that pure learning approaches cannot.

1.4 Cross-Paper Connections and Complementary Insights

1.4.1 Connection 1: Ontology Standards and Foundations

Paper 03 (Ontology Survey) provides comprehensive analysis of upper-level ontologies: - **SUMO**: Largest open-source ontology with expressive formal definitions, extensive domain coverage - **DOLCE/DUL**: Captures ontological categories underlying natural language and common sense - **IEEE CORA**: Core ontology for robotics and automation (IEEE Standard 1872-2015) - **IEEE AUR**: Extension for autonomous robots (aerial, ground, surface, underwater, space)

Paper 01 (Educational Ontology-RL) demonstrates practical ontology use: - Directed acyclic graph: G = C, E, , D, - Concepts (C), Relations (E with properties), Attributes ( ), Database (D), Agents ( ) - Shows how to assign RL agents to ontology nodes for specialization

Paper 02 (AGGM) extends to runtime reasoning: - Builds on Semantic Sensor Network (SSN) ontology - Uses SWRL for inference rules - Implements forward reasoning (state → goal) and backward reasoning (goal → required state)

Paper 04 (Collision Avoidance) applies to UAV safety domain: - Recommends building on SUMO upper-level ontology - Extending with IEEE Autonomous Robotics standard - Developing UAV-specific domain ontology for spatial relations, flight phases, regulatory rules

Integration for Flyby-F11:

```
Upper Level: SUMO (foundational concepts)
    ↓
Domain Level: IEEE AUR + UAV Domain Ontology
    ↓ (spatial relations, flight modes, mission types)
Application Level: Flyby-F11 Mission Ontology
    ↓ (specific objectives, constraints, environments)
Instance Level: Runtime state (sensor observations, current mission)
```

1.4.2 Connection 2: Multi-Agent RL with Ontological Structure

Paper 01 (Educational Ontology-RL) introduces multi-agent architecture: - Each RL agent assigned to specific ontology topic/concept - Experience sharing between agents accelerates learning - MDP formulation: S, A, R, P, per agent - Agents observe state (ontology-structured), select actions, receive rewards

Paper 02 (AGGM) extends to automatic goal generation: - Agents use ontology for situation assessment (observe → evaluate) - Forward reasoning selects predefined goals from goal-set - Backward reasoning creates new goals for unseen situations - Priority function balances problem-specific reward (B) with state-similarity reward (J)

Paper 06 (Autonomous Vehicles) validates approach: - Multi-agent MDPs "facilitate coordinated strategies in multi-vehicle environments" - Game-theoretic approaches for cooperative/competitive scenarios - Hierarchical RL architectures for multi-level decomposition

Integration for Flyby-F11:

```
High-Level Agent (Mission Planning):
- Ontology: Mission objectives, constraints, airspace
- RL: Optimize mission sequence, waypoint selection
- State: Mission progress, resource status
- Actions: Select next mission phase, adapt mission plan

Mid-Level Agent (Behavior Selection):
- Ontology: Flight phases, behavioral primitives, safety rules
- RL: Optimize behavior transitions, parameter tuning
- State: Environmental context, vehicle state
- Actions: Select behavior (navigate, loiter, land), set parameters

Low-Level Agents (Control):
- Ontology: Control modes, actuator constraints, sensor capabilities
- RL: Optimize trajectory, control inputs
- State: Sensor observations, vehicle dynamics
- Actions: Velocity commands, control surface deflections

Experience Sharing: Knowledge transfer between mission scenarios, behavior contexts, control re
```

1.4.3 Connection 3: Handling Uncertainty and Incomplete Information

Paper 03 (Ontology Survey) establishes open-world assumption: - OWL (open-world): "statements can be true whether known or not" - Better suited for "incomplete, evolving knowledge in robotic applications" - Belief maintenance for "resolving inconsistencies" and "handling uncertainty"

Paper 02 (AGGM) implements uncertainty handling: - State distance metric (D) captures unexpected environmental changes - Q-value discrepancy detection identifies reward anomalies - Importance weighting prioritizes critical observations - Three triggering cases for significant change identification

Paper 06 (Autonomous Vehicles) addresses POMDPs: - Partially Observable MDPs for "scenarios with hidden road users and occluded conditions" - Model decision problems when "not all variables are fully observable" - Enable vehicles to "evaluate potential actions based on probabilistic models"

Paper 05 (UAV-ON) demonstrates practical challenge: - No GPS, global maps, or privileged information - Egocentric visual observations with depth uncertainty - Instance-level object identification requires semantic disambiguation - Dynamic environments with moving obstacles

Integration for Flyby-F11:

```
Perception Layer:
- Sensor fusion (T265 visual odometry, D455 depth) → ontological concepts
- Uncertainty propagation through belief maintenance
- Confidence scores for ontological assertions

Reasoning Layer:
- POMDP formulation for GPS-denied navigation
- Ontology defines observable vs. hidden state variables
- Belief state represented as probability distribution over ontological concepts

Decision Layer:
- Action selection considers observation uncertainty
- Conservative strategies when confidence low
- Information-gathering actions to reduce uncertainty
```

1.4.4 Connection 4: Semantic Grounding and Object-Scene Co-occurrence

Paper 05 (UAV-ON) introduces object-scene reasoning: - LLM-generated co-occurrence priors: benches/trash bins → parks, bicycles → roads, boats → water - Instance-level descriptions require semantic understanding beyond category labels - Object-scene associations guide exploration strategy

Paper 02 (AGGM) formalizes through ontology relations: - Traffic Signal Control Ontology: isOn(Vehicle, Road), atIntersection(Road, Intersection) - Inference rules encode domain knowledge: ambulance at intersection → prioritize ambulance - Concept importance weighting reflects semantic significance

Paper 01 (Educational Ontology-RL) demonstrates content organization: - Ontology semantic relationships inform assistance generation - Database (D) organized by ontology concepts: each concept c_i has subset D_ci of related materials - Example: "linear function" links to tutorials, videos, problems on that topic

Paper 06 (Autonomous Vehicles) applies to behavior prediction: - Enhanced accuracy through "traffic participant predictions" and "intent recognition" - Scenario simulation leverages scene understanding - Multi-sensor fusion integrates diverse modality semantics

Integration for Flyby-F11:

```
Spatial Ontology:
- NoFlyZone, LandingZone, Waypoint, Obstacle, Clearance
- Relations: isNear, hasObstacle, canLand, requiresClearance

Environmental Ontology:
- Weather, Wind, Visibility, GNSSAvailability, Terrain
- Relations: affects, interferesWith, enables, restricts

Mission Ontology:
- Survey, Delivery, Inspection, Search
- Co-occurrence: Survey → OpenArea, Inspection → Structure, Delivery → LandingZone
- Guides exploration: if mission=Inspection, prioritize regions with structures
```

```
Semantic Grounding:
- Vision models detect objects → classify via ontology (is-a relations)
- Depth sensors measure distances → spatial relations (isNear, isFar)
- Mission goals map to object-scene pairs → guide navigation strategy
```

1.4.5 Connection 5: Safety Constraints and Action Space Filtering

Paper 04 (Collision Avoidance) formalizes safety constraints: - Ontology defines "safe state space and action preconditions" - Example: `canExecute(moveForward, state) ← minDepth(state.frontView) > safetyThreshold` - "Hard constraints prevent collision-prone actions a priori" - "Ontology eliminates unsafe actions from RL policy's action space"

Paper 02 (AGGM) implements constraint checking: - Inference rules determine valid goals given current state - State similarity reward function encourages safe recovery behaviors - Priority function (J over B) ensures safety takes precedence

Paper 05 (UAV-ON) reveals consequences of unconstrained policies: - Collision rates: Random 37.9%, CLIP-H 51.4%, AOA-V 45.0%, AOA-F 65.5% - Post-processing rules (boundary suppression, altitude enforcement) reduce but don't eliminate failures - LLM spatial reasoning insufficient for safety guarantees

Paper 06 (Autonomous Vehicles) emphasizes formal guarantees: - Rule-based systems "guarantee vehicle remains in safe state within predefined range" - Pure RL has "safety risks during exploration" and "lack of guaranteed bounds on behavior" - Hybrid methods provide "systematic constraints and guidance" alongside learning

Paper 01 (Educational Ontology-RL) demonstrates action filtering: - Action vector a* = {a , a , ..., a } constrained by ontology properties - Database filtering based on action vector ensures valid content selection - Assistance generation module validates actions before execution

Integration for Flyby-F11:

```
Pre-Flight Safety Ontology:
- Battery level constraints: canTakeOff ← chargeLevel > minimumReserve
- Weather constraints: canOperate ← windSpeed < maximumWind  visibility > minimumVisibility
- Airspace constraints: canEnter(zone) ← ¬isRestricted(zone)  hasAuthorization(zone)

In-Flight Safety Ontology:
- Collision avoidance: canExecute(action) ←  obstacle: distance(self, obstacle) > safetyMargin
- Geofence: canExecute(moveToward(position)) ← isWithin(position, geofence)
- Energy management: canContinueMission ← estimatedEnergyRemaining > energyToReturnHome

RL Action Space Filtering:
1. RL policy generates candidate action
2. Ontology reasoner validates preconditions
3. If valid: execute action
4. If invalid: sample alternative from valid action subset
5. Update RL policy with validity feedback (constraint violation penalty)

Runtime Monitoring:
```

```
- Continuously evaluate state against safety ontology
- Trigger emergency behaviors (land, return-to-home) when constraints violated
- Log violations for post-flight analysis and ontology refinement
```

1.4.6 Connection 6: Benchmark Evaluation and Validation

Paper 05 (UAV-ON) establishes rigorous evaluation framework: - Standardized metrics: SR (success rate), OSR (oracle success rate), DTS (distance to success), SPL (success-weighted path length) - Diverse environments: 14 Unreal Engine scenes (urban, natural, mixed) - Generalization testing: novel environments, unseen object categories - Safety analysis: collision rate tracking, termination type breakdown

Paper 02 (AGGM) validates against baselines: - Comparison with Q-learning, SARSA, DQN - Performance metrics: reward accumulation, convergence speed, adaptation to unseen situations - Traffic simulation: SUMO (Simulation of Urban MObility) with 16 intersections

Paper 01 (Educational Ontology-RL) demonstrates prior validation: - Positive impact on student performance [17] - Experience sharing improves agent training [16] - Successful application in educational serious game

Paper 06 (Autonomous Vehicles) emphasizes simulation importance: - "Essential for transitioning algorithms from theory to practice" - "Enable comprehensive testing across diverse conditions" - "Support validation of safety-critical systems" - "Facilitate comparison of different approaches"

Paper 03 (Ontology Survey) identifies validation gaps: - "Relatively few real-world, long-duration deployments discussed" - "Limited quantitative analysis: few performance comparisons between approaches" - Recommends "field deployments", "long-duration autonomy demonstrations", "comparison with non-ontological approaches"

Integration for Flyby-F11:

```
Phase 1: Simulation Validation (Gazebo + PX4 SITL)
- Metrics: Mission success rate, safety violation count, trajectory efficiency
- Scenarios: GPS-denied navigation, dynamic obstacle avoidance, communications-
denied operations
- Baselines: Pure RL (SAC, PPO), pure rule-based, LLM-based (AOA-style)
- Environments: Indoor (project-drone simulation), outdoor (flyby-f11 simulation)

Phase 2: Hardware-in-Loop Testing (Jetson platforms)
- Real-time performance: reasoning latency, control loop frequency
- Resource utilization: CPU/GPU load, memory footprint, power consumption
- Sensor integration: T265/D455 data → ontological concepts
- Edge case handling: sensor failures, unexpected obstacles, mission changes

Phase 3: Flight Testing (Progressive complexity)
- Controlled environment: indoor with safety nets
- Structured outdoor: marked waypoints, known obstacles
- Unstructured outdoor: GPS-denied, unknown obstacles, dynamic conditions
- Mission scenarios: survey, inspection, delivery (flyby-f11 payload capacity)

Phase 4: Comparative Analysis
```

```
- Ontology-constrained RL vs. pure RL vs. pure rule-based vs. LLM-based
- Quantify safety (collision rates), efficiency (path length, energy), success (completion rat
- Measure explainability (decision trace clarity), adaptability (novel scenario handling)
- Document failure modes and ontology refinements
```

1.5 Validated Approaches for Flyby-F11

Based on convergent evidence across all six papers, the following architectural decisions are strongly supported:

1.5.1 1. Upper-Level Ontology: SUMO

Rationale: - **Paper 03**: "Largest open-source ontology with expressive formal definitions", "extensive domain coverage" - **Paper 04**: "Recommends building on SUMO upper-level ontology" - Industry-standard foundation ensures compatibility and reusability

Implementation: - Use SUMO as foundational vocabulary (Object, Process, Event, Agent, Attribute, Relation) - Extend with IEEE 1872.2-2021 (Autonomous Robotics ontology) for UAV-specific concepts - Develop flyby-f11 domain ontology for mission types, spatial constraints, sensor capabilities

1.5.2 2. Knowledge Representation Language: OWL 2 DL

Rationale: - **Paper 03**: "Open-world assumption better suited for real-world robotics", "description logic provides balance of expressiveness and tractability" - **Paper 02**: Demonstrates SWRL (Semantic Web Rule Language) for efficient inference - **Paper 04**: "OWL 2 DL profile for decidable reasoning"

Implementation: - OWL 2 DL for ontology definition (SROIQ(D) expressivity) - SWRL for inference rules (forward and backward reasoning) - Reasoner: HermiT or Pellet with caching for real-time performance - Python integration: OWLREADY library for ROS 2 compatibility

1.5.3 3. Multi-Agent Reinforcement Learning

Rationale: - **Paper 01**: Demonstrates specialized agents per ontology concept with experience sharing - **Paper 02**: Shows multi-stage agent process (observe → evaluate → reason → act) - **Paper 06**: Validates "multi-agent RL" for coordinated strategies

Implementation:

```
Agent Hierarchy:

Mission Planner Agent:
- Ontology Level: Mission objectives, constraints, priorities
- State Space: Mission progress, resource availability, environmental conditions
- Action Space: Select next mission phase, adapt mission parameters, abort mission
- Reward: Mission completion, efficiency, safety margins
- Algorithm: SAC (Soft Actor-Critic) for continuous action spaces
```

```
Behavior Selector Agent:
- Ontology Level: Flight phases, behavioral primitives, transitions
- State Space: Vehicle state, environmental context, active mission phase
- Action Space: Navigate, Loiter, Land, Avoid, Return-to-Home with parameters
- Reward: Behavior appropriateness, smooth transitions, constraint satisfaction
- Algorithm: PPO (Proximal Policy Optimization) for stable learning

Trajectory Optimizer Agent (per behavior):
- Ontology Level: Control modes, actuator constraints, sensor capabilities
- State Space: Sensor observations (T265, D455), vehicle dynamics, setpoints
- Action Space: Velocity commands [vx, vy, vz, yaw_rate]
- Reward: Trajectory smoothness, energy efficiency, obstacle clearance
- Algorithm: TD3 (Twin Delayed DDPG) for low-level control

Experience Sharing:
- Shared replay buffer across similar behaviors (different Navigate contexts)
- Policy distillation from expert agent to new agent (transfer learning)
- Meta-learning across mission types to improve generalization
```

1.5.4 4. Automatic Goal Generation Model (AGGM)

Rationale: - **Paper 02**: Proves AGGM outperforms pure RL in unseen situations - **Paper 05**: Demonstrates need for dynamic goal adaptation (OSR-SR gap) - **Paper 06**: Recommends "continual learning" and "online adaptation"

Implementation:

```
Observe Stage:
- Multi-view RGB-D cameras (front, left, right, down) → ontological observations
- T265 visual odometry → pose estimate
- MAVSDK telemetry → vehicle state
- Ontology schema: L^t = {C^t, M^t} (concepts, relations)

Evaluate Stage:
- Q-value: from RL policy for current state-action
- State distance: |s^t - s^(t-1)| using ontological similarity metric
- Observation importance: concept weighting iw_c(x) based on safety criticality

Significant Change Identification (triggers):
- Case 1: Q-value discrepancy (unexpected reward)
- Case 2: State distance threshold (significant environmental change)
- Case 3: High-importance concept detected (e.g., obstacle, human, restricted airspace)

Reasoning Stage:
- Forward reasoning: infer predefined goal from goal-set using SWRL rules
- Backward reasoning: create new goal to return to known safe state
- Priority function: F(B, J) balances mission reward (B) with state-similarity reward (J)

Generate Action Stage:
```

```
- Ontology-constrained action space (only valid actions)
- RL policy selects action within constraints
- Multi-agent coordination if needed

Execute Action Stage:
- Translate high-level action to MAVSDK offboard commands
- Monitor execution through PX4 telemetry
- Update belief state for next cycle
```

1.5.5 5. Hierarchical Planning with Ontological Decomposition

Rationale: - **Paper 03**: "Hierarchical planning: multi-level decomposition (mission → task → action)" - **Paper 06**: "Hierarchical RL architectures" enable "clear separation" of strategic and tactical - **Paper 04**: "High-level ontological mission planning, low-level RL trajectory optimization"

Implementation:

```
Level 1 - Mission Planning (10-second horizon):
- Ontology: Mission types (Survey, Inspection, Delivery), objectives, priorities
- Planning: Decompose mission into waypoint sequence, assign timing, allocate resources
- Decision: Adapt mission based on environmental changes, resource consumption
- Output: Ordered list of waypoints with associated behaviors

Level 2 - Behavior Planning (1-second horizon):
- Ontology: Flight phases (Transit, Loiter, Land), behaviors, transitions
- Planning: Select appropriate behavior for current waypoint and context
- Decision: Trigger behavior transitions based on events, constraints
- Output: Behavior command with parameters (speed, altitude, duration)

Level 3 - Trajectory Planning (100-ms horizon):
- Ontology: Control modes, actuator limits, safety margins
- Planning: Generate smooth, collision-free trajectory to satisfy behavior command
- Decision: Real-time obstacle avoidance, constraint satisfaction
- Output: Velocity setpoint [vx, vy, vz, yaw_rate]

Level 4 - Control Execution (10-ms horizon):
- Ontology: Not directly involved (PX4 handles low-level control)
- Planning: PX4 cascaded controllers (position → velocity → attitude → rate)
- Decision: Actuator mixing, failsafe triggers
- Output: Motor commands via MAVLink
```

1.5.6 6. Safety-Critical Constraint Enforcement

Rationale: - **Paper 05**: 37-65% collision rates demonstrate critical need for formal safety - **Paper 04**: "Ontology eliminates unsafe actions from RL policy's action space" - **Paper 06**: "Guarantee vehicle remains in safe state within predefined range of rules"

Implementation:

```
Spatial Safety Constraints:
```

```
hasSafeSeparation(UAV, Object) ←
  distance(UAV, Object) > minimumSeparation(objectType(Object))

canExecute(moveToward(Position)) ←
  hasSafeSeparation(UAV, Position)
  isWithinGeofence(Position)
  hasLineOfSight(currentPosition, Position)

Flight Phase Constraints:
canTakeOff ←
  batteryLevel > minimumForMission
  gpsQuality > minimumAccuracy
  ¬hasActiveWarning

canLand ←
  isLandingZone(currentPosition)
  horizontalVelocity < maxLandingVelocity
  altitude < landingInitiationAltitude

Energy Constraints:
canContinueMission ←
  estimatedEnergyRemaining >
    (energyToReturnHome + safetyReserve)

mustReturnToHome ←
  batteryLevel < criticalThreshold
  estimatedTimeToDepletion < timeToReturnHome

Sensor Constraints:
requiresVisualNavigation ←
  gpsQuality < minimumAccuracy

canNavigateVisually ←
  hasWorkingSensor(T265)
  hasWorkingSensor(D455)
  lighting > minimumLighting

Regulatory Constraints (NDAA compliance, FAA Part 107):
canOperateInAirspace(Airspace) ←
  ¬isRestricted(Airspace)   hasWaiver(Airspace)

maxAltitude(Height) ← Height   400  # feet AGL

requiresVisualObserver ←
  operationDistance > visualLineOfSightRange
```

1.5.7 7. Semantic Sensor Fusion

Rationale: - **Paper 02**: "SSN ontology bridges low-level data streams and high-level concepts" - **Paper 03**: "Categorization: combine information to create picture of what is happening" - **Paper 04**: "Sensor data (vision, radar, ADS-B) feeds semantic object detection"

Implementation:

```
Perception Pipeline → Ontological Concepts:

T265 Visual Odometry:
- Raw: pose [x, y, z, qw, qx, qy, qz], confidence
- Ontology: hasPosition(UAV, Position), hasPoseEstimate(UAV, Pose, Confidence)
- Reasoning: lowConfidence(Pose) ← confidence < threshold → requireAlternativeLocalization

D455 Depth Camera:
- Raw: depth image, point cloud
- Ontology: hasObstacle(Direction, Distance), hasClearance(Direction, Distance)
- Reasoning: tooClose(Direction) ← distance < safetyMargin → triggerAvoidance(Direction)

Vision Models (YOLO, etc.):
- Raw: bounding boxes, class labels, confidences
- Ontology: detectedObject(Type, Position, Size), objectType(Object, Category)
- Reasoning: isObstacle(Object) ← objectType(Object, StaticObject)   objectType(Object, Human)

MAVSDK Telemetry:
- Raw: GPS, IMU, battery, mode, armed state
- Ontology: hasGPSQuality(UAV, Quality), hasBatteryLevel(UAV, Level), inFlightMode(UAV, Mode)
- Reasoning: gpsCompromised ← hasGPSQuality(UAV, Poor) → switchToVisualNavigation

Semantic Fusion:
- Combine observations from multiple sensors via ontology
- Example: T265 says "moving forward 2 m/s", D455 says "obstacle at 5m", Vision says "detected
  → Ontology infers: isApproachingPerson, distance=5m, timeToCollision=2.5s → triggerStopManeu
```

1.6 Open Questions and Research Gaps

Despite strong convergent evidence for ontology-constrained RL, several important questions remain:

1.6.1 1. Ontology Granularity and Scalability

Question: What is the optimal level of detail for UAV mission ontologies? How do we balance expressiveness with reasoning performance?

Evidence from Papers: - **Paper 01**: Discusses "granular enough for specialized assistance while broad enough to avoid requiring massive numbers of agents" - **Paper 03**: Notes "computational overhead" as key challenge, recommends "cached inferences" - **Paper 04**: Asks "how detailed should collision avoidance rules be? (trade-off: specificity vs. reasoner performance)"

Gap: No quantitative analysis of ontology size vs. reasoning latency for UAV control loops (10-100 Hz). Need empirical study on Jetson platforms.

Proposed Investigation: - Develop ontologies at varying granularity levels (coarse, medium, fine) - Benchmark reasoning performance (HermiT, Pellet, custom RETE engine) - Measure impact on RL training (sample efficiency, convergence speed) - Identify optimal granularity for different hierarchy levels (mission vs. behavior vs. trajectory)

1.6.2 2. Learning-to-Reason: Updating Ontologies from Experience

Question: How can the ontology itself be refined through RL experience? Can we automate ontology updates without manual knowledge engineering?

Evidence from Papers: - **Paper 02**: AGGM adapts goals but ontology structure is static - **Paper 06**: Recommends "continual learning" and "learning from deployment experience" - **Paper 03**: Identifies "limited work on combining ontologies with machine learning" - **Paper 04**: Suggests "continual learning: update ontology from failed episodes"

Gap: No framework for ontology learning from failed missions, unexpected events, or novel situations. All papers assume ontologies are manually engineered.

Proposed Investigation: - Inverse reinforcement learning to infer new inference rules from successful mission traces - Clustering failed episodes to identify missing ontology concepts - Active learning: query human expert when ontology insufficient - Meta-learning across mission types to discover generalizable patterns - Version control and validation for ontology updates (prevent knowledge corruption)

1.6.3 3. Multi-UAV Coordination with Shared Ontology

Question: How do multiple UAVs with independent RL policies coordinate using a shared ontology? What communication protocols are required?

Evidence from Papers: - **Paper 06**: "Multi-agent MDPs facilitate coordinated strategies in multi-vehicle environments" - **Paper 03**: "Multi-robot systems" mentioned as future direction - **Paper 01**: Experience sharing between agents, but single-UAV context

Gap: No discussion of distributed ontology reasoning, communication bandwidth constraints, or conflict resolution for multi-UAV scenarios.

Proposed Investigation: - Shared ontology vs. local ontologies with synchronization - Consensus mechanisms for distributed reasoning - Bandwidth-limited communication: what ontological information to share? - Game-theoretic approaches for cooperative mission planning - Swarm intelligence with ontological constraints

1.6.4 4. Uncertainty Quantification in Ontological Reasoning

Question: How do we represent and reason about uncertainty in ontological assertions derived from noisy sensors?

Evidence from Papers: - **Paper 03**: "Open-world assumption" handles incomplete knowledge, but not uncertainty - **Paper 06**: "POMDPs for scenarios with hidden road users" but requires probabilistic models - **Paper 02**: State distance and Q-value metrics capture change but not uncertainty - **Paper 05**: Depth cameras, visual detections have inherent uncertainty

Gap: OWL reasoning is crisp (true/false); doesn't natively handle probabilistic assertions. Need integration with probabilistic reasoning.

Proposed Investigation: - Probabilistic OWL extensions (PR-OWL, BayesOWL) - Fuzzy description logics for graded truth values - Dempster-Shafer theory for evidence combination - Confidence-weighted ontological assertions from perception pipeline - Bayesian networks integrated with ontology structure

1.6.5 5. Real-Time Reasoning Performance on Edge Hardware

Question: Can ontology reasoners achieve <100ms latency for UAV control loops on Jetson platforms? What optimizations are required?

Evidence from Papers: - **Paper 02**: "SWRL evaluation <100ms typical latency" but for traffic control (5-second intervals) - **Paper 03**: "Cached inferences for time-critical decisions", "incremental reasoning" - **Paper 04**: "Target <100ms per reasoning cycle for real-time control"

Gap: No empirical benchmarks of OWL reasoners (HermiT, Pellet) on ARM-based edge platforms (Jetson Orin) for UAV-scale ontologies.

Proposed Investigation: - Benchmark study: HermiT, Pellet, FaCT++, custom RETE engine on Jetson Orin Nano Super (8GB) and Orin NX (16GB) - Measure latency vs. ontology size, query complexity, reasoning mode (classification, consistency, instance checking) - Optimize reasoning: pre-compilation, materialization, partitioning (static vs. dynamic knowledge) - Hybrid approach: critical safety queries via fast rule engine, complex reasoning offline - Profile CPU/GPU utilization, memory footprint, power consumption

1.6.6 6. Sim-to-Real Transfer with Ontological Abstractions

Question: Do ontological abstractions improve sim-to-real transfer by providing domain-invariant representations?

Evidence from Papers: - **Paper 05**: "Sim-to-real transfer" identified as future direction but not investigated - **Paper 06**: "Transfer learning across domains" recommended - **Paper 01**: Data transformation layer enables "interface with different systems" but only in simulation

Gap: No validation that ontology-structured policies transfer better from simulation (Gazebo) to real UAVs than end-to-end learned policies.

Proposed Investigation: - Train RL policies in simulation with ontology-structured state/action spaces - Train baseline end-to-end policies (raw sensor inputs → actions) - Deploy both to real hardware (project-drone, flyby-f11) - Measure performance degradation: success rate, safety violations, trajectory quality - Hypothesis: Ontological abstractions reduce sim-to-real gap by filtering irrelevant details

1.6.7 7. Explainability Trade-offs: Performance vs. Interpretability

Question: Does enforcing ontological constraints reduce RL performance? How much efficiency do we sacrifice for explainability?

Evidence from Papers: - **Paper 06**: Discusses "balancing control between knowledge-driven and data-driven components" - **Paper 05**: Pure learning (AOA-V) achieves higher OSR (26.30%) than

constrained approaches - **Paper 04**: "Hybrid integration: ontology provides safety envelope within which learning-based optimization occurs"

Gap: No quantitative analysis of the performance trade-off. Is constrained RL suboptimal compared to unconstrained, and by how much?

Proposed Investigation: - Ablation study: RL with full ontology constraints vs. partial constraints vs. no constraints - Measure: sample efficiency (training steps to convergence), final performance (success rate), safety (violation count) - Pareto frontier: efficiency vs. safety vs. explainability - Determine if constraints guide exploration (improve sample efficiency) or limit expressiveness (reduce ceiling performance)

1.6.8 8. Formal Verification of Ontology-Constrained Policies

Question: Can we formally prove that ontology-constrained RL policies satisfy safety properties?

Evidence from Papers: - **Paper 04**: "Formal verifiability: safety rules encoded declaratively can be formally verified" - **Paper 03**: "Formal reasoning about system properties and behaviors" - **Paper 06**: "Verification" identified as challenge for learned systems

Gap: Ontologies enable formal verification of rules, but RL policies are stochastic. How do we verify probabilistic satisfaction of safety properties?

Proposed Investigation: - Model checking: verify ontology inference rules guarantee safety invariants - Statistical model checking: estimate probability policy violates constraints - Shield synthesis: automatically generate safety shields from ontology - Runtime verification: monitor policy execution against ontological contracts - Certification frameworks for autonomous systems with learned components

---

1.7 Recommended Next Steps

Based on the comprehensive literature synthesis, the following development roadmap is recommended for flyby-f11:

1.7.1 Phase 1: Ontology Foundation (Weeks 1-4)

Objective: Establish formal knowledge representation infrastructure

Tasks: 1. **Select and Install Upper-Level Ontology** - Download SUMO ontology from ontologyportal.org - Install IEEE 1872.2-2021 Autonomous Robotics ontology - Set up Protégé editor for ontology development - Configure OWLREADY Python library for ROS 2 integration

2. **Develop UAV Domain Ontology**
   - Extend IEEE AUR with UAV-specific concepts:
     - Flight phases: Preflight, Takeoff, Transit, Loiter, Landing, Emergency
     - Spatial relations: Above, Below, Near, Far, Inside, Outside, Approaching, Receding
     - Environmental: Weather, Wind, Visibility, Terrain, Airspace, NoFlyZone
   - Define properties with domain/range constraints
   - Implement concept importance weighting (safety criticality)
3. **Create Flyby-F11 Application Ontology**
   - Mission types: Survey, Inspection, Delivery (3kg payload)

- Mission constraints: Geofence, energy budget, time limit, airspace restrictions
- Sensor capabilities: T265 visual odometry range, D455 depth range, camera FOV
- NDAA compliance constraints for government applications

4. **Develop SWRL Inference Rules**
   - Safety rules: collision avoidance, geofence enforcement, energy management
   - Mission rules: goal selection, behavior transitions, emergency protocols
   - Sensor rules: data fusion, quality assessment, failure detection
   - Test rules using Pellet reasoner in Protégé

Deliverables: - `/flyby-f11/ontology/sumo_base.owl` (upper-level) - `/flyby-f11/ontology/uav_domain.owl` (domain) - `/flyby-f11/ontology/flyby_mission.owl` (application) - `/flyby-f11/ontology/rules/safety_rul` - Validation report: ontology consistency check, rule coverage analysis

1.7.2 Phase 2: ROS 2 Integration (Weeks 5-8)

Objective: Connect ontological reasoning with perception and control systems

Tasks: 1. **Create Ontology Interface Package** - New package: `/flyby-f11/ros2_ws/src/ontology_interface` - Nodes: - `ontology_reasoner`: Load ontology, execute SWRL rules, answer queries - `perception_mapper`: Convert sensor data → ontological observations - `action_validator`: Filter RL actions through ontology constraints - Messages: Custom message types for ontological assertions, queries, constraints

2. **Implement Semantic Sensor Fusion**
   - Subscribe to:
     - `/t265/odom` (visual odometry pose)
     - `/d455/depth/image_raw` (depth camera)
     - `/yolo/detections` (vision model bounding boxes)
     - `/mavros/state`, `/mavros/battery` (MAVSDK telemetry)
   - Publish:
     - `/ontology/observations` (ontological concepts from sensors)
     - `/ontology/constraints` (active safety constraints)
     - `/ontology/goal` (current ontological goal state)
3. **Integrate with Behavior Trees**
   - Custom BT nodes:
     - `OntologyCondition`: Check ontological predicate (e.g., hasSafeSeparation)
     - `OntologyAction`: Execute ontology-validated action
     - `GoalReasoner`: Use AGGM to select/generate goals
   - Modify existing BTs to query ontology for action preconditions
4. **Develop Action Filtering Layer**
   - RL policy outputs candidate action
   - Ontology reasoner evaluates `canExecute(action)` using SWRL rules
   - If valid: send to MAVSDK bridge
   - If invalid: request alternative from RL policy or use ontology-suggested safe action

Deliverables: - `ontology_interface` package with nodes, messages, launch files - Updated `behavior_trees` package with ontology-aware BT nodes - Integration tests in simulation (Gazebo + PX4 SITL) - Performance benchmarks: reasoning latency, CPU/memory usage

1.7.3 Phase 3: Multi-Agent RL Implementation (Weeks 9-14)

Objective: Develop hierarchical RL agents constrained by ontology

Tasks: 1. **Define MDP Formulations** - Mission Planner Agent: - State: mission progress, resources, environment (ontological abstractions) - Actions: select waypoint, adapt mission, abort - Reward: mission completion, efficiency, safety margins - Behavior Selector Agent: - State: vehicle state, context, mission phase (ontological) - Actions: navigate, loiter, land, avoid, RTH with parameters - Reward: behavior appropriateness, smooth transitions - Trajectory Optimizer Agent: - State: sensor observations, dynamics, setpoints - Actions: velocity commands [vx, vy, vz, yaw_rate] - Reward: smoothness, energy, obstacle clearance

2. **Implement AGGM for Each Agent**
   - Observe: collect state from sensors → ontological representation
   - Evaluate: compute Q-value, state distance, observation importance
   - Identify Change: check triggering conditions (Cases 1-3)
   - Reason: forward reasoning for goal selection, backward reasoning for new goals
   - Generate Action: ontology-constrained action from RL policy
   - Execute: send to subordinate agent or actuators
3. **Develop Training Infrastructure**
   - Environments:
     - Simple: single waypoint navigation with static obstacles
     - Medium: multi-waypoint mission with dynamic obstacles
     - Complex: GPS-denied navigation in cluttered environment
   - Reward shaping:
     - Task reward (mission completion, waypoint reaching)
     - Safety penalty (constraint violations, collisions)
     - Efficiency reward (energy, time, smoothness)
   - Algorithms: SAC (mission), PPO (behavior), TD3 (trajectory)
4. **Implement Experience Sharing**
   - Shared replay buffer across similar contexts
   - Policy distillation from expert agents to novice agents
   - Meta-learning for cross-mission generalization

Deliverables: - `ontology_rl` package with multi-agent training framework - Trained policies for mission planner, behavior selector, trajectory optimizer - Training curves: sample efficiency, convergence, safety violations - Ablation study: with/without ontology constraints, with/without experience sharing

1.7.4 Phase 4: Benchmark Evaluation (Weeks 15-18)

Objective: Validate ontology-constrained RL against baselines and UAV-ON metrics

Tasks: 1. **Develop Benchmark Scenarios** - Scenario 1: Waypoint navigation (seen environments) - Scenario 2: Obstacle avoidance (dynamic obstacles) - Scenario 3: GPS-denied navigation (T265-only localization) - Scenario 4: Mission adaptation (unseen obstacles requiring replanning) - Scenario 5: Communications-denied (no operator input for entire mission)

2. **Implement Baseline Methods**
   - Pure RL (SAC without ontology constraints)
   - Pure rule-based (if-then logic, no learning)

- LLM-based (AOA-style with GPT-4o mini for comparison)
- Hybrid IL+Rules (imitation learning with safety post-processing)

3. **Collect Metrics**
   - Success metrics: SR (success rate), OSR (oracle success rate), DTS (distance to success), SPL (success-weighted path length)
   - Safety metrics: collision rate, constraint violation count, geofence breaches
   - Efficiency metrics: energy consumption, trajectory length, mission time
   - Explainability: decision trace clarity (qualitative), rule coverage (quantitative)
4. **Statistical Analysis**
   - Repeated trials (50+ per scenario per method)
   - Statistical significance testing (t-tests, ANOVA)
   - Performance variance (robustness measure)
   - Pareto frontier: safety vs. efficiency vs. success

Deliverables: - Benchmark suite: scenarios, evaluation scripts, visualization tools - Comparative results: tables, plots, statistical tests - Failure analysis: categorize failure modes, identify ontology gaps - Publication draft: technical report or conference paper

1.7.5 Phase 5: Hardware Validation (Weeks 19-24)

Objective: Deploy and validate on project-drone and flyby-f11 platforms

Tasks: 1. **Project-Drone Testing (Weeks 19-21)** - Platform: Jetson Orin Nano Super 8GB, T265 + D455 - Environment: Indoor controlled space with safety nets - Missions: - Week 19: Waypoint navigation, static obstacle avoidance - Week 20: Dynamic obstacle avoidance (moving objects) - Week 21: GPS-denied navigation using T265 visual odometry - Metrics: Real-time performance (reasoning latency, control frequency), safety (no collisions), mission success

2. **Flyby-F11 Integration (Weeks 22-23)**
   - Platform: Jetson Orin NX 16GB, flyby sensor suite
   - Environment: Outdoor structured area (marked waypoints)
   - Missions:
     - Week 22: Survey mission (autonomous flight pattern)
     - Week 23: Inspection mission (structure proximity flying)
   - Metrics: Mission-specific (coverage, image quality), safety, efficiency (energy, time)
3. **Stress Testing (Week 24)**
   - Sensor failures: GPS denial, camera occlusion
   - Environmental challenges: wind gusts, lighting changes
   - Mission changes: dynamic waypoint updates, abort scenarios
   - Edge cases: unexpected obstacles, airspace restrictions
   - Metrics: Graceful degradation, recovery behaviors, failure logging
4. **Iterative Refinement**
   - Analyze flight logs for ontology gaps (missing concepts, insufficient rules)
   - Update ontology based on failure modes
   - Retrain RL policies with updated constraints
   - Repeat testing to validate improvements

Deliverables: - Flight test videos with ontology decision overlays - Hardware performance benchmarks (Jetson resource utilization) - Ontology refinements based on real-world failures - Safety assessment report (collision-free hours, constraint compliance)

1.7.6 Phase 6: Documentation and Dissemination (Weeks 25-26)

Objective: Document findings and share with research community and MCTSSA

Tasks: 1. **Technical Documentation** - System architecture diagram with ontology-RL integration - Ontology reference: concepts, relations, rules with justifications - ROS 2 API documentation for ontology interface - Deployment guide for flyby-f11 missions

2. **Research Outputs**
   - Conference paper: "Ontology-Constrained Reinforcement Learning for Autonomous UAV Missions"
   - Target venues: ICRA, IROS, RSS (robotics), AIAA GNC (aerospace)
   - Open-source release: GitHub repository with code, ontologies, trained policies
   - Benchmark contribution: Extend UAV-ON with ontology-constrained baselines
3. **MCTSSA Demonstration**
   - Prepare demonstration scenarios showcasing:
     - Mission-intent interpretation (high-level goal → autonomous execution)
     - Communications-denied operation (no operator input)
     - Explainable decisions (ontology rule trace)
     - Safety guarantees (formal constraint enforcement)
   - Documentation for government/defense applications:
     - NDAA compliance verification
     - Safety case arguments using ontological verification
     - Failure mode analysis and mitigation strategies

Deliverables: - Complete system documentation (architecture, API, deployment) - Draft research paper with experimental results - Open-source repository with README, tutorials, examples - MCTSSA demonstration package (videos, slides, technical brief)

1.8 Conclusion

This comprehensive synthesis of six research papers provides overwhelming evidence that **ontology-constrained reinforcement learning** represents the most promising architectural approach for autonomous UAV systems operating in GPS-denied, communications-limited environments. The convergent findings across diverse domains (education, traffic control, robotics, autonomous vehicles, UAV benchmarks) establish that:

1. **Hybrid methods combining knowledge-driven and data-driven approaches outperform pure implementations** in safety, adaptability, efficiency, and interpretability.
2. **Formal ontological knowledge provides the safety scaffold** that constrains RL exploration while preserving the flexibility to discover optimal policies.
3. **Hierarchical decomposition through ontological structure** enables multi-level reasoning from strategic mission planning to tactical trajectory optimization.
4. **Semantic abstractions enable generalization** to unseen situations through ontology-guided goal generation and backward reasoning.
5. **Explainability and verification** are achievable through formal knowledge representation, critical for safety-critical and defense applications.

The flyby-f11 development roadmap synthesizes validated approaches from all six papers: - SUMO upper-level ontology (Paper 03, 04) - OWL 2 DL with SWRL inference rules (Paper 02, 03, 04) - Multi-agent RL with experience sharing (Paper 01, 06) - Automatic Goal Generation Model (Paper 02) - Hierarchical planning decomposition (Paper 01, 03, 04, 06) - Safety-critical constraint enforcement (Paper 04, 05, 06) - Semantic sensor fusion (Paper 02, 03, 04)

This architecture directly addresses the critical safety failures identified in UAV-ON benchmark (Paper 05) where unconstrained learning approaches achieved only 7.30% success with 37-65% collision rates. By integrating formal ontological constraints, we expect to: - **Reduce collision rates** from 37-65% → <10% through hard safety constraints - **Improve task success** from 7.30% → >50% through semantic grounding and goal reasoning - **Enable explainability** for MCTSSA collaboration through ontological decision traces - **Achieve edge-based autonomy** on Jetson Orin platforms through efficient ontological reasoning

The literature synthesis validates not only the technical approach but also the development methodology: simulation validation → hardware-in-loop → flight testing → iterative refinement. This systematic progression, grounded in empirical evidence from multiple research communities, provides a robust foundation for advancing autonomous UAV capabilities while maintaining the safety guarantees and interpretability required for real-world deployment.

1.9 References

1.9.1 Primary Papers Synthesized

1. Hare, R., & Tang, Y. (2024). Ontology-driven reinforcement learning for personalized student support. *IEEE Systems, Man, and Cybernetics Conference*. arXiv:2407.10332v2.

2. Ghanadbashi, S., & Golpayegani, F. (2022). Using ontology to guide reinforcement learning agents in unseen situations: A traffic signal control system case study. *Applied Intelligence*, 52, 1808-1824. https://doi.org/10.1007/s10489-021-02449-5

3. Aguado, E., Gomez, V., Hernando, M., Rossi, C., & Sanz, R. (2024). A survey of ontology-enabled processes for dependable robot autonomy. *Frontiers in Robotics and AI*, 11:1377897. https://doi.org/10.3389/frobt.2024.1377897

4. [Collision Avoidance Ontologies for UAS] - Domain application ontology review (internal document)

5. Xiao, J., Sun, Y., Shao, Y., Gan, B., Liu, R., Wu, Y., Guan, W., & Deng, X. (2025). UAV-ON: A benchmark for open-world object goal navigation with aerial agents. arXiv:2508.00288v4. https://github.com/Kyaren/UAV_ON

6. Hu, J., Wang, Y., Cheng, S., Xu, J., Wang, N., Fu, B., Ning, Z., Li, J., Chen, H., Feng, C., & Zhang, Y. (2025). A survey of decision-making and planning methods for self-driving vehicles. *Frontiers in Neurorobotics*, 19:1451923. https://doi.org/10.3389/fnbot.2025.1451923

1.9.2 Supporting References

- Avižienis, A., Laprie, J. C., Randell, B., & Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1), 11-33.

- IEEE SA (2015). *IEEE Standard Ontologies for Robotics and Automation* (IEEE Std 1872-2015).
- IEEE SA (2022). *IEEE Standard for Autonomous Robotics (AuR) Ontology* (IEEE Std 1872.2-2021).
- Niles, I., & Pease, A. (2001). Towards a standard upper ontology. *Proceedings of the International Conference on Formal Ontology in Information Systems*, 2001, 2-9.
- Pease, A. (2011). *Ontology: A Practical Guide*. Articulate Software Press.
- Prestes, E., et al. (2013). Towards a core ontology for robotics and automation. *Robotics and Autonomous Systems*, 61(11), 1193-1204.