**Movement of Automated Vehicles**

**Problem Statement**

Current autonomous vehicle path planning approaches struggle to balance real-time adaptability, safety, and computational efficiency. While traditional algorithms lack robustness in dynamic environments, learning-based methods like DRL face issues with generalization, interpretability, and real-world deployment. Formal methods using temporal logic ensure safety but are limited by scalability and optimization latency. The lack of an integrated, scalable solution that ensures reliable, safe, and adaptive planning under uncertainty remains a critical challenge. Bridging this gap requires combining learning, optimization, and formal verification to enable autonomous systems to operate safely and efficiently in complex, real-world scenarios.

**✅ 1. Define a Unified Hybrid Framework**

**🎯 Objective:**

Design an **integrated path planning system** that balances:

* **Real-time adaptability**
* **Formal safety assurance**
* **Computational efficiency**

**🛠️ Actions:**

* Combine **Deep Reinforcement Learning (DRL)** or **Learning from Demonstration (LfD)** with **temporal logic-based verification (e.g., STL/LTL)**.
* Use traditional planners (e.g., A\*, RRT\*) for **initial path generation**.
* Apply DRL to refine motion strategies and adapt to dynamic obstacles.
* Employ **Signal Temporal Logic (STL)** to validate paths at runtime.

**✅ 2. Incorporate Real-Time Adaptability**

**🎯 Objective:**

Enable the system to handle changes in environment, traffic, and obstacle behaviors in real time.

**🛠️ Actions:**

* Use **Partially Observable Markov Decision Processes (POMDP)** or **actor-critic DRL architectures** for decision-making.
* Implement **re-planning modules** that adapt the trajectory based on dynamic sensor input or simulation feedback.
* Optimize using **Model Predictive Control (MPC)** for short-horizon planning.

**✅ 3. Improve Generalization and Interpretability**

**🎯 Objective:**

Address the DRL black-box problem and improve trustworthiness of decisions.

**🛠️ Actions:**

* Train DRL agents across **diverse scenarios** (multi-lane, intersections, roundabouts).
* Use **attention-based neural networks** or **explainable AI techniques** to enhance model interpretability.
* Implement **transfer learning** to reduce training time across environments.

**✅ 4. Ensure Safety Through Formal Methods**

**🎯 Objective:**

Use logic-based specifications to enforce safety rules and constraints.

**🛠️ Actions:**

* Define **STL constraints** for traffic rules, obstacle avoidance, and lane discipline.
* Use **robustness metrics** to quantify STL satisfaction.
* Integrate STL checking during and after DRL policy execution for **online validation**.

**✅ 5. Evaluate and Benchmark the System**

**🎯 Objective:**

Validate the system’s safety, adaptability, and performance across different scenarios.

**🛠️ Tools & Metrics:**

* **Simulation Tools**: CARLA, SUMO, IR-SIM
* **Metrics**:
  + Path length, computation time
  + Collision count / obstacle clearance
  + STL robustness score
  + Reward convergence (for DRL)
  + Energy and comfort (if extended to real-world factors)

**✅ 6. Identify and Solve Limitations**

**🎯 Objective:**

Tackle scalability, training time, and real-world deployment issues.

**🛠️ Actions:**

* Incorporate **modular design** for independent updates to each component.
* Explore **edge computing or federated learning** to reduce communication and central training load.
* Test in **multi-agent, multi-lane, and mixed-traffic environments**.

**✅ Final Deliverables**

1. **System Architecture Diagram** (DRL + STL + Planner modules)
2. **Simulation Models** (Code in Python, CARLA APIs, etc.)
3. **Evaluation Report** with quantitative graphs and scenario outcomes
4. **Explained Decisions** (XAI layer for interpretability)
5. **Full Research Report** (Literature, Methodology, Experiments, Results, Conclusion)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **aspect** | **Movement 1.pdf<br>(Two-Layered Planning Model)** | **Movement 2.pdf<br>(Optimized A\* Algorithm)** | **Movement 3.pdf<br>(DRL Survey for Motion Planning)** | **Movement 4.pdf<br>(Temporal Logic for Safe Planning)** | **Common/Key Points Across All Papers** |
| **Actual Idea** | Proposes a two-layered model using Bi-RRT + VFH polynomial in Frenet coordinates to achieve real-time, smooth, safe path planning in dynamic environments. | Enhances A\* with turn penalty, obstacle raster coefficients, and smoothing to improve safety, reduce path length, and handle sparse/dynamic obstacles. | Surveys use of Deep Reinforcement Learning (DRL) for hierarchical motion planning, covering end-to-end and layered strategies with vehicle dynamics. | Uses Learning from Demonstrations (LfD) with temporal logic (STL) to ensure path planning adheres to safety rules, static/dynamic obstacle avoidance, and runtime constraints. | All aim to improve autonomous vehicle path planning with respect to safety, adaptiveness, and real-world constraints using either search, learning, or logic. |
| **Findings** | Bi-RRT improved with steering constraints, VFH enables adaptive goal shifting; outperforms traditional methods in cluttered environments. | 84% reduction in traversed nodes, 39% less turning angle, smoother paths; ensures fewer redundant nodes and better computational efficiency. | DRL can model strategic, motion, and control layers; DQN, DDPG, actor-critic methods applied to car-following, merging, lane-keeping. | Proposed method yields safe trajectories validated against STL rules (e.g., traffic lights, safety zones); works in continuous real-time cycles with partial environment perception. | Advanced methods improve motion planning by ensuring safety, path feasibility, and efficiency. |
| **Survey Conducted / Evaluation Method** | Simulation + real vehicle in complex scenarios using real sensors (GPS, LiDAR); evaluation on various obstacle setups. | Simulation-based comparison with baseline A\*; tested on rasterized real maps; trajectory metrics used for benchmarking. | Literature survey with algorithm taxonomy, simulators, vehicle models, and reward structures; categorized by behavior, motion, and control level. | Evaluated in IR-SIM simulator on valet parking tasks with dynamic/static obstacles and traffic lights; used STL to verify real-time performance over multiple path cycles. | All methods are either simulation-tested or benchmarked against standard environments; validation includes static and dynamic obstacle cases. |
| **Current Progress** | Real-time two-layer path planning combining geometry and dynamic adjustments is viable. | Optimized A\* adapts well to irregular and sparse environments but still lacks real-world deployment. | DRL shows promise but struggles with generalization, real-time inference, and safety guarantees. | Demonstrated real-time STL-based optimization of learned trajectories with robust handling of evolving constraints. | Hybrid approaches combining planning, ML, and verification are trending; however, deployment and standardization are still evolving. |
| **Literature Gaps** | Lacks adaptability in highly dynamic/multi-agent environments; sensitive to map noise and VFH reliability. | Fails to fully adapt to moving obstacle changes in real-time; struggles with large-scale deployment. | Sparse reward signal, model interpretability, and training time are major issues; limited real-world deployments. | Needs real-world testing and expansion to more complex STL constraints; optimization speed still limits scalability. | Need for real-world validation, standard benchmarks, model explainability, and scalability across varying scenarios. |
| **Problems Yet to be Addressed** | Needs better robustness to environment changes; Bi-RRT still suffers in very narrow spaces. | Difficulties in ensuring consistent safety margins across varying environments; lacks dynamic trajectory adaptation. | Safety verification, real-time computation, and policy generalization under uncertainty remain open problems. | Bridging simulation-to-reality gap, dynamic updates to STL rules, and reducing optimization latency. | Real-time adaptability, multi-agent coordination, and safety assurance under uncertain inputs are common unsolved issues. |