AER1217 Final Project Report:

Autonomous Drone Racing

Anton Zheng 1005790315

Souren Pashangpour 1006324611

Gibran Rajput 1005310619

1. Introduction

Autonomous drone racing demands rapid trajectory planning and precise control under uncertainty. This project tackles these challenges by developing an algorithm to navigate a Crazyflie quadrotor through sequenced gates while avoiding obstacles. Building on prior labs — geometric control (Lab 2), target localization (Lab 3), and visual odometry (Lab 4) — the solution combines search-based path planning, spline smoothing, and low-level control. The two-phase approach first validates the algorithm in a PyBullet simulation, then transfers it to a physical drone, emphasizing sim2real compatibility and robustness against environmental uncertainties.

2. Algorithm Overview

The core solution uses a 3D grid-based A* algorithm with Euclidean heuristic for global path planning. Obstacles are inflated by $\pm 0.2m$ to account for positional noise, ensuring

collision avoidance in worst-case scenarios. Predefined gate waypoints guide segmented path generation, with cubic splines interpolating waypoints into smooth trajectories to minimize jerk. For sparse paths, polynomial fitting ensures continuity. Low-level control executes via cmdFullState commands, sending position, velocity, and acceleration references to the Crazyflie firmware. A* was prioritized over sampling-based methods for deterministic performance in structured environments, while sequenced commands (takeoff, trajectory tracking, landing) ensure operational safety. The design balances computational efficiency with dynamic feasibility, leveraging prior lab insights on PD/velocity control tuning to address real-world execution challenges.

Key Components:

1. **Environment Discretization**
    a. The workspace is discretized into a 3D grid with resolution *0.1m*.
    b. Obstacles are inflated by *±0.2m* to account for positional uncertainty *U (−0.2, 0.2)*.
2. ***A**** **Path Planning**
    a. 3D grid-based A* with 6-directional connectivity.
    b. Heuristic: Euclidean distance.
    c. Predefined waypoints guide the drone through gates (e.g., *start → Gate 1 → Gate 3 → Gate 4 → Gate 2 → Gate 1→ Gate 4 → land*).

3. **Trajectory Smoothing**

    a. Cubic splines interpolate waypoints to generate smooth, continuous trajectories.

    b. Polynomial fitting ensures dynamic feasibility.

4. **Control Execution**

    a. cmdFullState sends position, velocity, and acceleration references to the firmware.

    b. Takeoff, trajectory tracking, and landing commands are sequenced using timestep counters.

3. Key Equations and Implementation

3.1 A* Algorithm

- **Heuristic Function**:

$$h(n) = \sqrt{(x_{\text{goal}} - x_n)^2 + (y_{\text{goal}} - y_n)^2 + (z_{\text{goal}} - z_n)^2}$$

- **Cost Function**:

$$f(n) = g(n) + h(n)$$

where $g(n)$ is the cumulative travel cost from the start.

3.2 Spline Interpolation

The cubic spline interpolation in the code (using *splprep* with $k=3$) creates smooth trajectories by fitting piecewise third-order polynomials through the planned waypoints. This approach ensures continuous acceleration while precisely passing through each navigation point, eliminating jerky movements that could destabilize the drone. The spline is sampled at the control frequency (*30Hz*) to generate position references that respect the drone's dynamic limits. For cases with too few waypoints, the code falls back to high-degree polynomial fitting to maintain trajectory smoothness. This implementation provides an optimal balance between motion quality and computational efficiency for aggressive racing maneuvers.

3.3 Obstacle Uncertainty

- **Effective Obstacle Radius**:

$$r_{eff} = r_{base} + \text{safety margin} = 0.06 + 0.20 = 0.26m$$

4. Results

4.1 Simulation

- **Success Criteria**
    - Navigated through all six gates.

- Reached the target point at *(−0.5, 2.0, 0)* without collisions.
- Achieved smooth landing after trajectory completion.

- **Performance Metrics**
  - Flight Time: *~30* seconds (varies with target speed tuning).
  - Trajectory Smoothness: Minimal oscillations due to spline interpolation.
  - Collision Avoidance: No collisions observed in nominal scenarios.

4.2 Physical Testing

Three trials were conducted to validate real-world performance. The first trial suffered from improper waypoint sequencing near Gate 4, causing the drone to turn back and revisit Gates 4 and 2, resulting in only 3 gates passed due to controller instability. The second trial corrected the path planning issue, successfully navigating all gates, but failed during landing due to an overly aggressive descent duration. The final trial adjusted the landing parameters and increased speed to 1.6 m/s (simulation-tested), but physical hardware limitations of the Crazyflie prevented stable execution, highlighting discrepancies between simulated and real-world dynamics.

5. Improvements and Future Work

1. **Dynamic Gate Waypoint Generation**

Replace hardcoded waypoints with an automated system that reads gate coordinates from the instruction. Programmatically determine entry/exit points for each gate based on their orientation and position, enabling adaptive path planning for any gate sequence.

2. **Speed and Trajectory Optimization**

To maximize performance while ensuring stability, the target speed should be systematically fine-tuned. Starting with conservative values to identify the drone's operational limits, then incrementally increasing for agility. Adaptive velocity profiles should be implemented for turns, dynamically adjusting speed to minimize overshooting while maintaining efficiency. This approach must extend to landing parameters, where descent rates and timing require refinement to ensure consistent real-world performance. By balancing simulated speed targets with physical validation, the system can achieve reliable high-speed navigation without exceeding the drone's dynamic capabilities.

3. Algorithm Efficiency

Optimize the A* heuristic to minimize redundant waypoints, improving path smoothness and computational speed. Introduce lightweight replanning to handle minor deviations without full recalculations.

6. Conclusion

This project developed an autonomous drone racing system using A* path planning and spline-based trajectory optimization, successfully navigating gates in simulation while avoiding uncertain obstacles. Real-world testing revealed challenges in waypoint sequencing and speed control, highlighting the gap between simulation and physical execution. The results demonstrate that while search-based planning works reliably in structured environments, achieving robust high-speed performance requires adaptive gate detection and hardware-aware speed tuning. This work provides a foundation for future improvements in dynamic racing scenarios.