Optimization-Based Trajectory Design Through Waypoints

Qiyuan Wu

July 2025

GitHub Repository

Contents

1 Setup

This report explores trajectory planning using Richter et al.'s methodology [3]. The objective is to minimize snap and time through a trajectory with three waypoints. The true objective is to minimize the maximum motor force; however, as a compromise, we minimize jerk/snap and hope to minimize force. The drone starts and ends hovering while passing through a gate at 90 degrees in the middle. The waypoints are the same as in my MAE 157A capstone project [2], so we can examine the benefits of constrained/unconstrained optimization over simple polynomial fitting.

Suppose we have a trajectory P(t). We want a minimum jerk trajectory, so we set

$$J = \int \frac{\partial^3 P}{\partial t^3} dt \tag{1}$$

We can also write

$$J = p^T Q(T) p \tag{2}$$

where p is the vector of polynomial coefficients and Q is the Hessian matrix. Additionally, we can write constraints in the form of

$$Ap = d \tag{3}$$

Alternatively, we can turn this constrained quadratic problem into an unconstrained QP by writing

$$p = A^{-1} d \tag{4}$$

and thus

$$J = d^T A^{-T} Q A^{-1} d \tag{5}$$

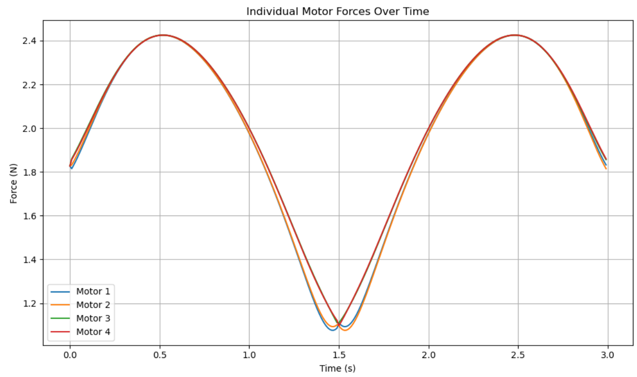


Figure 1: Motor forces with minimum jerk trajectory.

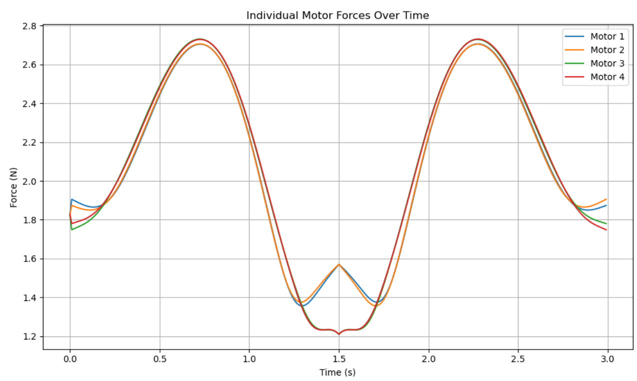


Figure 2: Motor forces with minimum snap trajectory.

2 Unconstrained QP (fixed time)

The initial and final states are fixed. At the gate, all states are fixed to geometry and the drone's orientation, except that x-velocity and y-acceleration are only required to be continuous. The time of each segment is fixed at 1.5s. The minimum snap trajectory yields similar results in motor forces compared to my capstone. It is surprising that the minimum jerk trajectory yields smaller motor force (2.4N) than the minimum snap trajectory (2.7N), contrary to the conclusion in [1].

The trajectory itself looks similar to that shown in [2], so it is not reproduced.

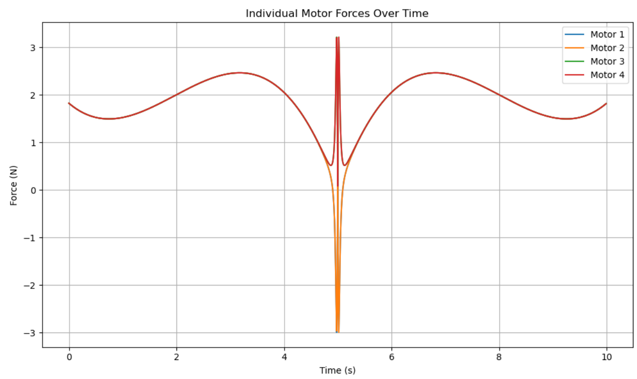


Figure 3: Motor forces blowup at the gate when T = 5s for each segment.

3 Unconstrained QP (time as free variable)

The cost function is shown below.

$$J = p^T Q p + k_T \sum_i T_i \tag{6}$$

Note that both p and Q are functions of T_i. If $k_T = 0$, then we solve for the minimum snap trajectory. Otherwise, we factor in the total time.

We use gradient descent to determine the optimal proportion of time segments and manually adjust the total time to fit the actuator constraints. Specifically,

$$T_i^{(k+1)} = T_i^{(k)} - \alpha \frac{\partial J}{\partial T_i} \tag{7}$$

The partial derivative is computed numerically.

The authors noted that, as $T \to \infty$, "the quadrotor states along the trajectory converge to hover [3]." However, if we impose a fixed boundary condition on acceleration (for instance, free fall at the gate), the quadrotor states do not converge to hover. Instead, increasing the time risks exceeding the position or actuator constraints as shown in Figure 3. This suggests that we cannot simply increase T to reduce motor forces.

In our case, the gradient descent algorithm suggests that $T_1 = T_2$. This is true even if $T_1 \neq T_2$ in our initial guess, as shown in Figure 4. The paper tells us that this is the optimal proportion of time segments between waypoints regardless of k_T. Therefore, to reduce the maximum motor forces, we gradually increase T until we hit large oscillations in motor forces as shown in Figure 5. The maximum motor force is much lower than that in the capstone (2.3N vs 2.7N). To reduce time while minimizing snap, we gradually decrease T while staying under maximum motor force as shown in Figure 6.

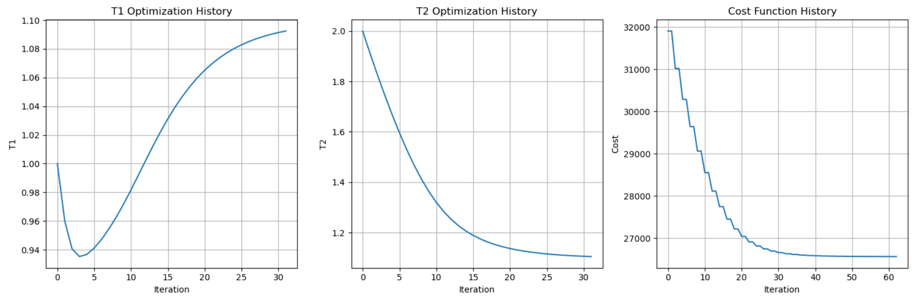


Figure 4: Convergence history for time and total cost. $k_T = 10000$ to optimize time as well as minimizing jerk.

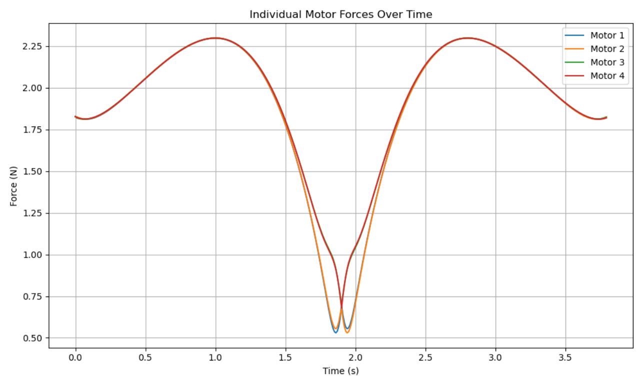


Figure 5: Minimum jerk trajectory with smallest maximum motor forces with T = 1.9s for each segment.

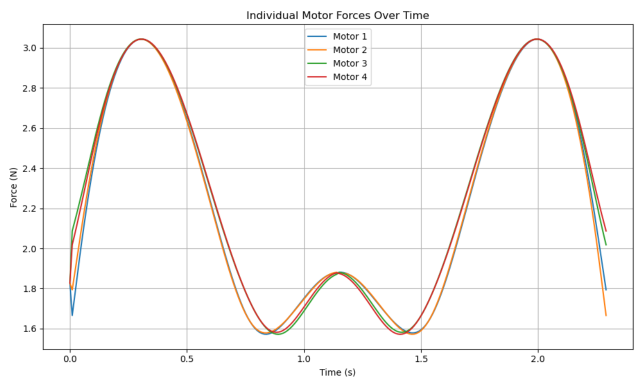


Figure 6: Minimum jerk trajectory with smallest time while under motor force constraints with T = 1.15s for each segment.

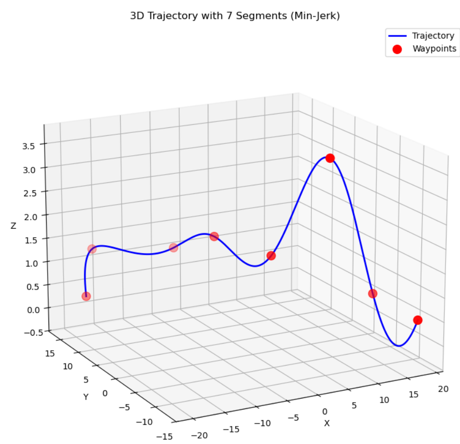


Figure 7: Eight-waypoint trajectory generated with initial guess of time proportions based on average speed.

4 Eight-waypoint minimum jerk and time trajectory

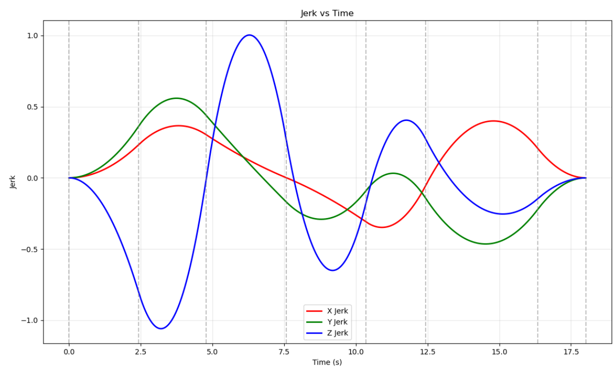
Now, we use the approach in section 3 on eight waypoints, or equivalently, seven trajectory segments. We use the cost function [6] and enforce the velocity and acceleration to be continuous at the intermediate waypoints.

We want a maximum velocity of 5 m/s. To get our initial guess of the time segments, we divide the distance of each trajectory segment by an average speed of 3 m/s. Using this initial guess, we can generate a trajectory as shown in Figure 7.

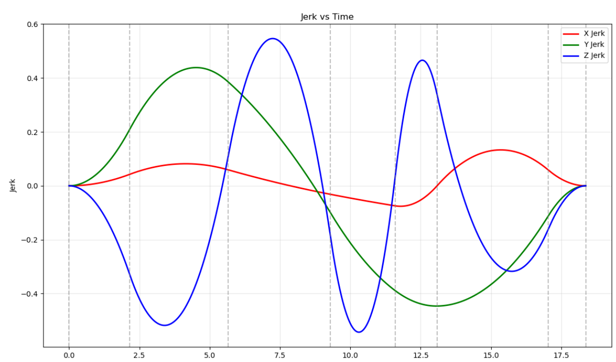
Next, we use the gradient descent algorithm as in [7] to optimize the time proportions. We use $k_T = 100$, a learning rate $\alpha = 10^{-4}$, a maximum iteration of 1000, and an error tolerance of 10^{-3}. To satisfy the maximum velocity "motor constraints," we iteratively increase the total time while keeping the trajectory segments' time proportions the same.

Figure 8 shows the jerk along the trajectory with the time proportions from the initial guess and after applying the gradient descent. The maximum jerk reduces from about 1.0 to 0.6. Although we only enforce the velocity and acceleration to be continuous at the intermediate waypoints, the jerk and motor forces (not shown) along the trajectory turn out to be smooth, and the snap (not shown) is continuous. As the QP minimizes jerk, it likely makes use of the free constraints to automatically make jerk smooth and snap continuous at the waypoints.

Right after applying the gradient descent, we generate a trajectory that minimizes jerk and total time as shown in Figure 9a. However, this trajectory

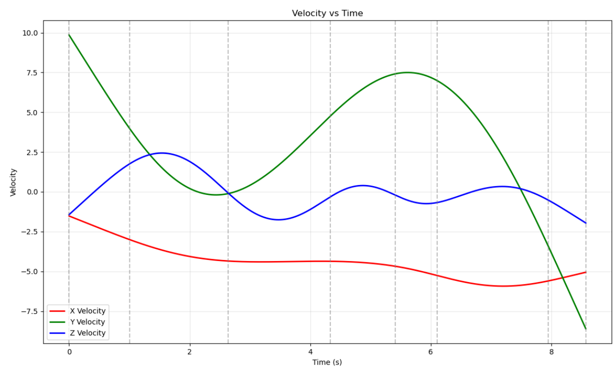


(a) Time proportions from average speed.

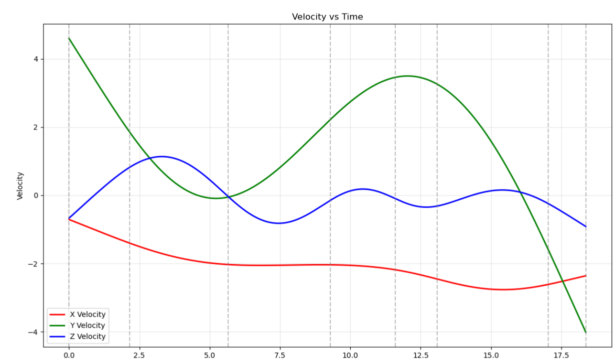


(b) Optimized time proportions with gradient descent.

Figure 8: Jerk along trajectory.



(a) Time proportions with gradient descent.



(b) Total time adjusted to satisfy velocity constraint.

Figure 9: Velocity along trajectory.

does not necessarily satisfy the motor constraints. Here, the velocity reaches 10 m/s above the maximum velocity of 5 m/s. Thus, we iteratively increase the total time, keeping the time ratios the same, until the maximum velocity falls below 5 m/s, shown in Figure 9b.

5 Future work

- Incorporate motor force constraints directly into optimization rather than adjusting time afterwards.
- Optimize code to run faster; write in c++.

References

[1] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation*, pages 2520–2525, 2011.

[2] Jonathan Morris, Santoshi Subramanian, Ryan Teoh, Ben Whitley, and Wu Qiyuan. Stokes Drifter MAE 157A: Github repository. `https://github.com/qiyuanbillwu/Stokes-Drifter-MAE-157A`, 2025. Accessed: 2025-07-02.

[3] Charles Richter, Adam Bry, and Nicholas Roy. *Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments*, pages 649–666. Springer International Publishing, Cham, 2016.