I. Introduction

The second part of the coursework is a continuation of the first part [2] , where the Nonlinear Model Predictive Control(NMPC) is translated into a Nonlinear Programming (NLP) problems using the multiple shooting method. To avoid repeated discussions on the same formulation and techniques, the subsequent sections will frequently refer back to Part 1. The main work of Part 2 focuses on system parameter identification through the Extended Kalman Filter and path planning using the 3D A^* algorithm.

II. Control Approach Overview

For the task in Part 2, the optimization is running throughout a non-convex constraint set. The feasible regions for possible trajectories in the three-dimensional space may be very round and narrow. For this reason, it is crucial to minimize the model-plant mismatch so that collisions with obstacles can be avoided. Therefore, the more accurate method Runge-Kutta method of order four (RK-4) is used to forward simulate the system with the set sampling time T_s. It is a matter of fact that the multiple shooting method is sensitive to the initial guess and prone to falling into the local minima. To ameliorate this problem, initial shortest-path planning is applied. Though the shortest path may not be physically plausible or consistent for the actual quadcopter, it indeed helps the optimizer to hop out of some local minima and greatly improve the robustness. More explanations can be found in section VI.

III. Constraints Formulation

The constraints for the outline rectangular box were imposed using the same method as described in Part 1, Equation (2) [2] which supports any orientation of the box. The elliptical cylinder obstacle constraints were added through the logical inequality relation in (1).

$$\begin{aligned} &\frac{(x-x_c)^2}{a^2} + \frac{(y-y_c)^2}{b^2} > 1 \quad or \quad z < z_c \\ &\Rightarrow \; min\left(-\frac{(x-x_c)^2}{a^2} - \frac{(y-y_c)^2}{b^2}) + 1 < 0, \; z - z_c < 0\right) \end{aligned} \tag{1}$$

The boundary condition for the final position is established as $x(t)$ is free, $\forall t \in [T_t, T_t + T_e]$ because by applying the optimal control, we actually want the trajectory to convert in the vicinity of the target point. Hence, hard inequality constraints (2) were added to the position for the last steps during the time interval $[T_t, T_t + T_e]$ as follows:

$$x_{tar} - \epsilon < x < x_{tar} + \epsilon \tag{2}$$

, where x_{tar} is the target point and ϵ is the target point tolerance. This approach adds robustness to the control policy searched by the optimizer because the conditions on the final position are explicitly specified and that is independent of the cost function.

IV. Cost Function

The cost function used in Part 2 is also used to minimize the amount of control effort, serving as a soft constraint for the allowed maximum total thrust. This formulation allows the controller to break this particular constraint and find a feasible solution. However, in the assessed test cases, see Table I, the controller did not break the constraint of total thrust away from hovering. By including the control effort in the cost function reduced the total thrust around 15% to 30%.

Table I: Results after including the control effort in the cost function for three different courses

Cost Function	Total thrust: Course 2	Total thrust: Course 3	Total thrust: Course 4
Exclude thrust	0.013	0.021	0.026
Include thrust	0.008	0.013	0.022

V. Joint Parameter and State Estimation

The state of the original system is $x \in \mathbb{R}^{12}$ which is measurable, and the parameters for the system is $\theta \in \mathbb{R}^9$ which is not known exactly. In this case, an Extended Kalman Filter(EKF) is applied to perform joint parameter and state estimation, which means using the measurement to help estimate the parameter.[3] The parameter θ is viewed as another state, though it is a constant, a process noise $\omega_\theta(t)$ is artificially imposed on the differential equation so that $\dot{\theta} = 0 + \omega_\theta(t)$. Adding the term $\omega_\theta(t)$ is necessary because this allows the estimator to modify the value of θ. In other words, this will allow θ to evolve over time, otherwise, it will not change. The augmented state is then defined as $\tilde{x} \triangleq [x, \theta]^\top \in \mathbb{R}^{21}$ and hence we have,

$$\begin{cases} \dot{\tilde{x}} = \begin{bmatrix} f(x,u,\theta) \\ 0 \end{bmatrix} + \begin{bmatrix} \omega_x \\ \omega_\theta \end{bmatrix} \\ y = h(\tilde{x}) \end{cases} \tag{3}$$

After discretizing (3) and giving an initialization, a standard recursive EKF is constructed by the prediction update and measurement update equation. Though this approach can not guarantee convergence to the true value of θ, roughly speaking, it gives a better estimate of the system parameter. With that, we could use the current estimated parameter in the next NMPC prediction step.

VI. Path Planning

As mentioned in section II, the goal is to guide the quadrotor through an optimal trajectory that avoids all the obstacles in the known environment. As this is done through an optimization approach, the optimizer could be initialized intelligently by using a planned path, (i.e. a geometric optimal path) in a way that helps the optimizer to converge. In the final controller, 3D A^* planner [1] provides an initial path guess to the optimizer in the first iteration. The constraints were represented by an occupation map, which is a logical matrix where the free cells are `false` or `0` and the obstacles are `true` or `1`. The occupation map along with the starting and end coordinates are then fed into a 3D $A^{\star}$ path planning algorithm which returns the shortest route. This route is resampled and divided into N equidistant way-points, ω. Finally, the initial guess X_0 for the optimizer is formed as:

$$X_0 = \begin{bmatrix} w_1^x & w_1^y & w_1^z & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ w_N^x & w_N^y & w_N^z & \cdots & 0 & 0 \end{bmatrix} \tag{4}$$

, where $X_0 \in \mathbb{R}^{N\times 12}$, N is the prediction horizon. This heuristic approach may increases the computational time significantly because the geometrical optimal path may not be physically optimal. However, one stand-out advantage of path planning is that it increases the recursive feasibility or reliability of the controller. Without planning, approximately 80% of the test courses can not be accomplished by the solver. One example is shown in Fig. 2. After providing the planned route, most of them were solved successfully, except that for certain shortest paths posing physical challenges to the quadcopter would incur collision with the obstacles.

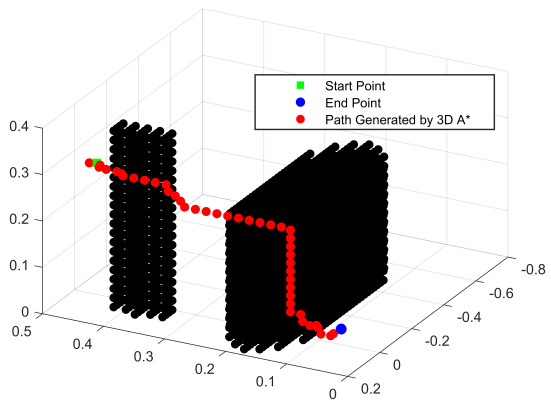

Fig. 1. The occupation map and path generated by the 3D $A^{\star}$.

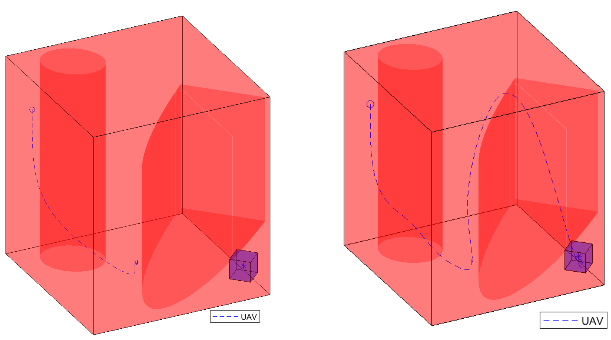

Fig. 2. Left: Without using a heuristic path guess results in failure to find a solution. Right: Using a heuristic path guess leads to a successful trajectory.

VII. Robustness

Besides the two methods mentioned in section V and VI: EKF and path planning, extra techniques were added to further enhance the controller's robustness. Firstly, the geometrical constraints of the elliptical cylinder were enlarged by a small factor $\rho = \frac{1}{3} \cdot d_{min}$, where d_{min} is the minimum allowed space between obstacles. This approach allows a wider margin for modelling error but may incur infeasibility of course. Secondly, since the NMPC controller has an entire feedback loop of current states, it will adjust the plan based on the current measurement. In other words, this online NMPC inherently can reject small modelling errors. Therefore, by setting a shorter sampling time $T_s = 0.1s$ could increase the ability to respond to disturbance (e.g. model inaccuracy and input perturbation) adequately fast. Additionally, T_t is decreased by $5 \cdot T_s$ to ensure the quadcopter settles in time.

VIII. Computation Time Reduction

To reduce the computation time, warm start and decreasing prediction horizon were included as the same in Part 1 [2]. This can be further improved, in most cases but not all cases, by providing a good initial guess to the optimization solver. This is because the heuristic initial guess directly puts the solver in a different area of search space which greatly helps it to find the control policy with less time.

In this NLP problem, one of the difficulties for the online optimizer is to identify the global minimum within a reasonable time. Consequently, to speed up computation the optimizer settings were tweaked so that a sub-optimal solution is permitted based on certain conditions shown in Table II. Although this reduces the optimality and accuracy of the solution, the solver overcomes the "limitation of $90s$ execution time ". Different from Part 1, instead of using sequential quadratic programming (SQP), the interior point method is selected for the solver, which turns out it could handle the problem more efficiently (see Table III).

Table II: Parameter settings applied to the optimizer

Parameter	Value
Function Tolerance	0.001
Step Tolerance	0.001
Max Iterations	45
Constraint Tolerance	0.001

Table III: Computation for different optimization algorithms for the course shown in Fig. 2

Optimization Algorithm	Computation Time (sec)
Interior Point	79.39
SQP	332.94

IX. Verification

The final online NMPC was tested under a wide range of scenarios with up to 10 elliptical cylinder obstacles and 20% random perturbations of system parameters. In all cases, the controller completes the task successfully.

References

[1] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[2] HuaYi Sheng. Predictive control core assignment part 1 spring. Coursework report, 2024.

[3] Wei Zhang. ME424 Modern Control and Estimation Lecture Note 7: Kalman Filter - Extended Kalman Filter. Fall 2021.