EN.530.603 Applied Optimal Control Fall 2021

Quadcopter Optimal Trjectory Generation and Obstacle Avoidance

Saicharan Balamurali and Bhanu Pranav Addepalli

Johns Hopkins University

December 23, 2021

Abstract

Quadcopters have become a common sight today due to their diverse functionality and convenience when it comes to certain activities. They are generally used for surveillance and as personal cameras but are finding applications in other environments such as search and rescue, exploration and assistance and construction. Such use case scenarios require the quadcopter to maneuver around obstacles and find the optimal path to the target location while minimising power usage. This report explores two optimal control techniques namely the Linear Quadratic Regulator (LQR) and the Differential Dynamic Programming (DDP) approach to find the optimal path for a quadcopter and discusses the results of either approach.

1 Introduction

The report explores the use of the LQR controller and the DDP approach to optimise the controls of a quadcopter. The general nonlinear dynamics of a quadcopter are used along with the control vector which includes the thrust acting on the quadcopter and the 3 components of the torque at the center of the quadcopter. The LQR controller uses a quadratic cost term to regulate the flight of the quadcopter. The aim of the controller is to minimise this quadratic cost to find an optimal control policy to drive the quadcopter to the target location. The DDP approach works by optimising the trajectory. The algorithm first starts by performing a backward pass iteratively on a nominal trajectory while minimizing a quadratic cost to generate a new control sequence. This control sequence is then used to generate a new trajectory which is then used for the next backward pass until the optimal trajectory is found.

2 Materials & Methods

For this project, we used MATLAB to simulate and visualize the trajectory of our quadcopter. We appended Dr. Kobilarov's existing starter code for differential dynamic programming shared for the purpose of this class. The template code was initially implemented for a 2-D car model. We extended the state model to work in 3 dimensions by adding more state variables and plotted the generated trajectory in 3 dimensions. Furthermore, we added and moved around obstacles that the quadcopter is programmed to evade. We achieved this by starting with a simple single obstacle map and cumulatively challenging the robot to avoid a larger number of obstacles.

In terms of the algorithms we used, we started off by applying a Linear Quadratic Regulator optimization to generate the controls required to navigate the quadcopter in a simple, obstacle-free environment. The nonlinear dynamics of the quadcopter are linearized about the defined final state of the quadcopter.We subsequently added a single obstacle and tuned the running cost matrices that penalize the state and controls i.e. the Q and R matrices to try to avoid the obstacle present in that particular configuration. The next step was to apply Differential Dynamic Programming to our problem and study how different initial conditions, final times and cost functions affect the trajectory of our quadcopter. In further sections of this report, we discuss the mathematical model used, effect of different parameters on the optimal trajectory and compare the two algorithms - LQR and DDP.

3 Mathematical Model

The state vector is denoted by the variable s in our model shown below:

$$s = [x \;\; y \;\; z \;\; \phi \;\; \theta \;\; \psi \;\; \dot{x} \;\; \dot{y} \;\; \dot{z} \;\; p \;\; q \;\; r]^T$$

The state variables as shown in the vector above are respectively the x, y, z coordinates, the roll, pitch, yaw angles, and their corresponding variations with time. The variation of the state with respect to the existing state and the control inputs are depicted as per the equation shown below:

$$\dot{s} = f(x) + g(x)u$$

$$f(x) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ p + q\sin(\phi)\tan(\theta) + r\cos(\phi)\tan(\theta) \\ q\cos(\phi) - r\sin(\phi) \\ q\frac{\sin(\phi)}{\cos(\theta)} + r\frac{\cos(\phi)}{\cos(\theta)} \\ 0 \\ 0 \\ g \\ \frac{(I_y - I_z)}{I_x}qr \\ \frac{(I_z - I_x)}{I_y}pr \\ \frac{(I_x - I_y)}{I_z}pq \end{bmatrix}$$

$$g(x) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ g_{71} & 0 & 0 & 0 \\ g_{81} & 0 & 0 & 0 \\ g_{91} & 0 & 0 & 0 \\ 0 & \frac{1}{I_x} & 0 & 0 \\ 0 & 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & 0 & \frac{1}{I_z} \end{bmatrix}$$

Here the quadcopter parameters I_x, I_y, I_z represent its moment of inertia about the x, y and z axes respectively. The coefficients g_{71}, g_{81}, g_{91} are shown below:

$$g_{71} = -\frac{1}{m}(\sin(\phi)\sin(\psi) + \cos(\phi)\cos(\psi)\sin(\theta))$$
$$g_{81} = -\frac{1}{m}(\cos(\psi)\sin(\phi) - \cos(\phi)\sin(\psi)\sin(\theta))$$
$$g_{91} = -\frac{1}{m}(\cos(\phi)\cos(\theta))$$

The control inputs u in our model is a column vector of the vertical thrust, and the torques about the x, y and z axes.

$$u = [u_1 \;\; u_2 \;\; u_3 \;\; u_4]$$

In our implementation, the quadcopter was set to finish its trajectory at the origin. The obtained linearized matrices A and B are shown below:

$$A = \text{zeros}(12, 12)$$
$$A_{7:12,7:12} = I_6$$
$$A_{7,5} = -49.05$$
$$A_{8,4} = -49$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -5 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 6.67 \end{bmatrix}$$

4 Results

For our first trial, we generated a trajectory by using a discrete-time LQR to obtain the optimal control values at each time step and generated the trajectory for an obstacle-free environment as shown in Fig 1. We linearized the nonlinear dynamics of the quadcopter around the final point in order to be able to use the LQR algorithm as a linear state-space model is necessary. Upon introducing an obstacle, we found that tuning the cost function in order to avoid obstacles is an inefficient and often ineffective method of trajectory optimization.

Subsequently, we applied the Differential Dynamic Programming algorithm and used the cost function parameters shown below:

$$Q = 0$$
$$Q_f = 0.1 * diag([2, 2, 2, ones(1, 9)])$$
$$R = 0.05 * diag([10, 5, 5, 5])$$

Using the cost function parameters shown above, we show the generated trajectory for three different cases.

- 2 obstacles
- 2 obstacles with a greater cost on controls
- 3 obstacles
- 3 obstacles with one of the obstacles moved into the path of the previously generated optimal trajectory

We chose to not apply a running cost on the states as we observed better results without them.

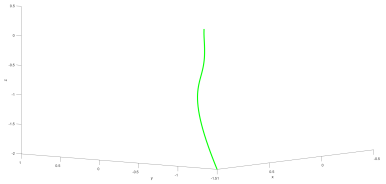

Figure 1: LQR Controller

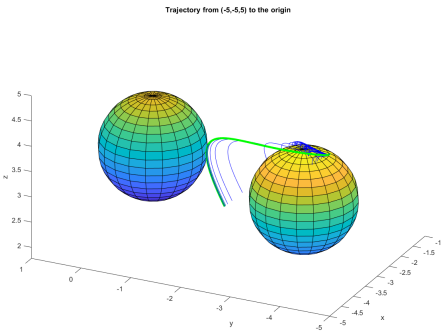

Figure 2: 2 obstacles

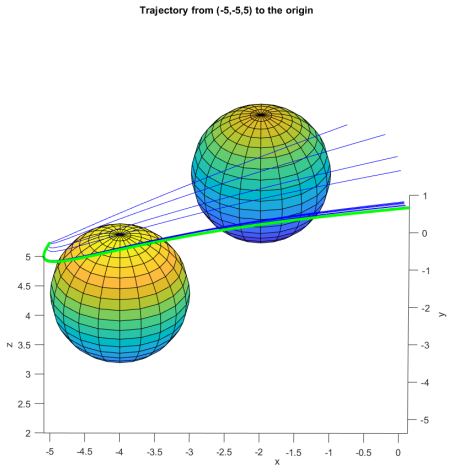

Figure 3: 2 obstacles with a greater running cost on controls

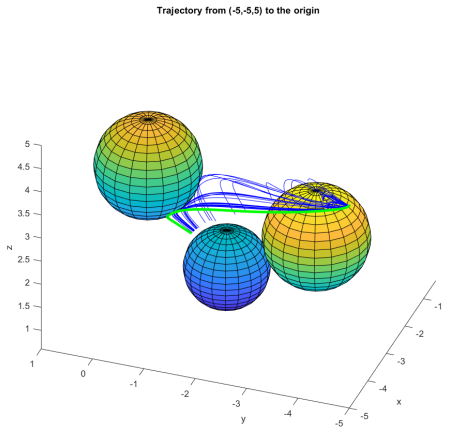

Figure 4: 3 obstacles

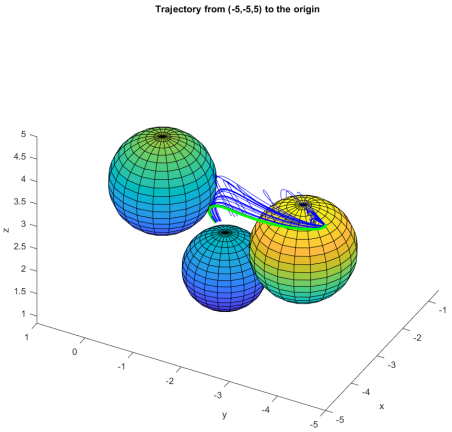

Figure 5: 3 obstacles with one obstacle's position changed

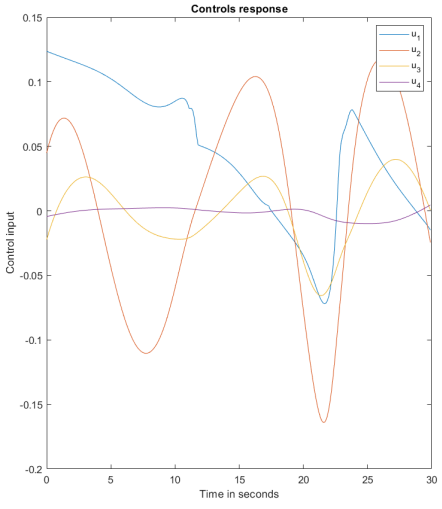

Figure 6: Control Inputs Variation

5 Discussion

As can be seen from the results above, we confirmed that the LQR controller can be used to obtain an optimal trajectory in an obstacle free environment. However, the LQR approach does not give good results when there are obstacles in the environment.

Looking at the results from DDP, it is evident that the optimal trajectory responds to the cumulatively increasing complexity portrayed in our test cases. In figure 2, an optimal trajectory is generated by circumventing 2 obstacles. This trajectory updates itself when the control is penalized further by generating a trajectory that has a smaller control input throughout the flight time of the quadcopter as shown in figure 3.

With the introduction of a third obstacle, the algorithm yet again finds a way to navigate around the map to get to its final position as depicted in figure 4. To prove that the trajectory is updated upon challenging the existing optimal control, we moved one of the obstacles further down such that it interferes with the existing optimal trajectory. Figure 5 shows the result of running the algorithm with this updated position where the trajectory yet again finds a way around the obstacles to its end point.

Moreover, the control inputs corresponding to the first case with 3 obstacles has been shown for reference. An important point to note here is that the control input u_1 shown here is the input in addition to gravity. Hence, the actual input will be $g + u_1$.

An interesting point that we discovered while running our algorithm was that DDP was really insensitive to the softness of the constraint. The constraints in DDP are handled by simply augmenting the cost function L and the coefficient of each constraint added to this cost was set to a large value to force the trajectory away further away from obstacles.

6 Future Developments

As we saw, Differential Dynamic Programming proves to be an effective algorithm for handling simple constraints imposed by interfering obstacles for our problem. A possible application of this is to use a quadcopter for exploration or maintenance. Such an application would require regular updates of the environment as well with respect to the quadcopter. Such a task can be achieved by integrating the above Path Optimization with Optimal Estimation of the state of the environment and the quadcopter, that is using an Extended Kalman Filter to iteratively predict and correct the state of the quadcopter and use this estimation to develop a new optimal trajectory with the already present DDP algorithm. We expect that doing this might allow the quadcopter to robustly avoid dynamically changing ostacles.

Acknowledgements

We would like to thank Dr. Marin Kobilarov for the opportunity and the resources to conduct this project. In addition, the teaching assistants for Applied Optimal Control, Peiyao Zhang and Brian Kim, have been really helpful throughout the duration of this course.

7 References

(1) Lysukho, G.V. Maslennikov, Andrey. (2020). Quadcopter: dynamics and control. Politechnical student journal. 10.18698/2541-8009-2020-5-604.;

(2) G. Garimella, M. Kobilarov, Towards Model-predictive Control for Aerial Pick-and-Place, In IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 4692-4697.;

(3) Francesco Sabatino, Quadcopter Control: Modelling, Nonlinear Control and Simulation

(4) Z. Xie, C. K. Liu and K. Hauser, "Differential dynamic programming with nonlinear constraints," 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 695-702, doi: 10.1109/ICRA.2017.7989086.;

(5) M. Kobilarov, Applied Optimal Control (Course Id: EN.530.603), Autonomous Systems, Control and Optimization Lab, Johns Hopkins University.;