# Project 4: Quadrotor Tracking Control

A. Srigopal*

**The goal of this project is to create a stable controller and observer that controls a drone to race through rings from start to finish without crashing. Additionally, I was required to assign equilibrium conditions to optimize the drone motion through the desired positions from tracking rings. This document outlines my process, results, and analysis of the quadrotor while developing a controller and observer.**

## I. Nomenclature

| | | |
|---|---|---|
| $p_x$ | = | $x$ position (m) |
| $p_y$ | = | $y$ position (m) |
| $p_z$ | = | $z$ position (m) |
| $\phi$ | = | Roll Angle (rad) |
| $\theta$ | = | Pitch Angle (rad) |
| $\psi$ | = | Yaw Angle (rad) |
| $v_x$ | = | Linear Velocity along the $x$ axis (m/s) |
| $v_y$ | = | Linear Velocity along the $y$ axis (m/s) |
| $v_z$ | = | Linear Velocity along the $z$ axis (m/s) |
| $w_x$ | = | Angular Velocity about the body-fixed $x$ axis (rad/s) |
| $w_y$ | = | Angular Velocity about the body-fixed $y$ axis (rad/s) |
| $w_z$ | = | Angular Velocity about the body-fixed $z$ axis (rad/s) |
| $\tau_x$ | = | Net Torque About the body-fixed $x$ axis (N*m) |
| $\tau_y$ | = | Net Torque About the body-fixed $y$ axis (N*m) |
| $\tau_z$ | = | Net Torque About the body-fixed $z$ axis (N*m) |
| $f_z$ | = | Net Force along the body-fixed $z$ axis (N) |

## II. Introduction

The end goal of this project is to develop a reliable controller, observer, and tracking system to allow a quadrotor to travel through the orange rings as quickly as possible and land at the finish line. In order for this project to be considered a success, the control system must meet the following requirements: First, the quadrotor must finish the course in a timely manner (within 30 seconds) at least 50 percent of the time. This will be verified by running 100 trials of the PyBullet simulation and recording if the quadrotor finished the course within the aforementioned time requirement. Second, the average time of completion for the quadrotor should be less than 20 seconds. This will also be verified by running 100 trials of the PyBullet simulation and recording the time each trial took to complete and averaging the time elapsed for the finished trials at the end.

## III. Model

Per the project requirements, the quadrotor must implement a controller, observer, and tracker that satisfy the defined design requirements. The first step is to derive the equations of motion for the rotor, which were provided by Professor Ornik. The Equations of Motion described is shown in equation 1 below.

*Code, Report

$$\begin{bmatrix} \dot{p_x} \\ \dot{p_y} \\ \dot{p_z} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{v_x} \\ \dot{v_y} \\ \dot{v_z} \\ \dot{\omega_x} \\ \dot{\omega_y} \\ \dot{\omega_z} \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \frac{\omega_x \cos\psi - \omega_y \sin\psi}{\cos\theta} \\ \omega_x \cos\psi + \omega_y \cos\psi \\ -\omega_x \cos\psi \tan\theta + \omega_y \sin\psi \tan\theta + \omega_z \\ 2f_z \sin\phi \\ -2f_z \sin\phi \cos\phi \\ 2f_z \cos\phi \cos\theta - \frac{981}{100} \\ \frac{10000\tau_x}{23} - \frac{17\omega_y\omega_z}{23} \\ \frac{10000\tau_y}{23} - \frac{17\omega_x\omega_z}{23} \\ 250\tau_z \end{bmatrix} \tag{1}$$

In order to proceed to building a controller, both input and state need to be derived. The state and input were defined from the equations of motion and are shown in equation 2 below.

$$x = \begin{bmatrix} p_x - p_{xe} \\ p_y - p_{ye} \\ p_z - p_{ze} \\ \phi - \phi_e \\ \theta - \theta_e \\ \psi - \psi_e \\ v_x - v_{xe} \\ v_y - v_{ye} \\ v_z - v_{ze} \\ \omega_x - \omega_{xe} \\ \omega_y - \omega_{ye} \\ \omega_z - \omega_{ze} \end{bmatrix} u = \begin{bmatrix} \tau_x - \tau_{xe} \\ \tau_y - \tau_{ye} \\ \tau_z - \tau_{ze} \\ f_z - f_{ze} \end{bmatrix} \tag{2}$$

Similar to previous design problems, equilibrium points need to be chosen for each state parameter in order to linearize the system. into a new matrix. This matrix should represent the desired final state of the quadrotor, which should be filled with mostly zeroes except for the net force along body-fixed the z-axis $f_{ze}$.

$$\begin{bmatrix} p_{xe} \\ p_{ye} \\ p_{ze} \\ \phi_e \\ \theta_e \\ \psi_e \\ v_{xe} \\ v_{ye} \\ v_{ze} \\ \omega_{xe} \\ \omega_{ye} \\ \omega_{ze} \\ \tau_{xe} \\ \tau_{ye} \\ \tau_{ze} \\ f_{ze} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 4.905 \end{bmatrix} \tag{3}$$

## IV. Design

The subsequent step in linearizing the system was to derive the A and B matrices by taking the Jacobian of the state time derivative with respect to the state and input variables. The A and B matrices are then evaluated at the chosen equilibrium point and shown in equation 4 below:

$$A = \begin{bmatrix} 0. & 0. & 0. & 0. & 0. & 0. & 1 & 0. & 0. & 0. & 0 & 0 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0 & 1. & 0. & 0. & 0 & 0 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0 & 0. & 1. & 0. & 0 & 0 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0 & 0. & 0. & 1. & 0 & 0 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0 & 0. & 0. & 0. & 1 & 0 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0 & 0. & 0. & 0. & 0 & 1 \\ 0. & 0. & 0. & 0. & 9.81 & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & -9.81. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \end{bmatrix}, B = \begin{bmatrix} 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 2. \\ 434.78 & 0. & 0. & 0. \\ 0. & 434.78 & 0. & 0. \\ 0. & 0. & 250. & 0. \end{bmatrix} \tag{4}$$

It is good practice to make sure the equilibrium points were chosen correctly. When evaluated at the equilibrium point, the A matrix dimensions were 12 x 12, which assured me that I was heading in the right direction. However, both A and B matrices needed to be evaluated as controllable, which means all states must be independent of each other. Therefore, the rank of the controllability matrix must be the same as the number of states defined in our A and B matrices. As shown in code block 5 in *GenerateResults.ipynb*, the rank of W calculated to equal 12 using the method shown below, where n denotes the number of states:

$$W = \begin{bmatrix} B, AB, ..., A^{n-1}B \end{bmatrix}$$

(5)

Now that I have expressed the linearized model in state space form, the next step was to design a closed-loop, asymptotically stable, linear state feedback controller. To accomplish this, I needed to derive our $K$ matrix. This was achieved using Linear Quadratic Regulator(LQR), wherein Q and R matrices are assigned to be gains for the cost function. I wanted to prioritize the position over the velocities of the drone when designing the controller. Through trial and error, I saw the y position $p_y$ needed to be prioritized higher than x and z positions. The final Q and R matrices for the linear state feedback controller are shown in code block 6 of *GenerateResults.ipynb*. While there is no definitive solution for $K$, the real components of the A-BK matrix eigenvalues need to be negative, indicating $K$ is asymptotically stable. This process is shown in code block 7 of*GenerateResults.ipynb*. The controller eigenvalues were as follows:

$$S = \begin{bmatrix} -97.2 & -97.2 & -55.9 & -2.1 + 3.03j & -2.1 - 3.03j & -3.2 & -2.4 + 2.98j & -2.4 - 2.9j & -4.4 & -1. & -1. + 0.97j & -1. - 0.97j \end{bmatrix} \tag{6}$$

It is clear from these results that all the eigenvalues have a negative real part, so we could theoretically ensure that our controller results in an asymptotically stable system. However, since an exact state is not given to the controller directly, an observer must also be implemented. The new controller becomes:

$$u = -K\hat{x} \tag{7}$$

Similar to the procedure for A and B matrices, the Jacobian of the observer function can be taken with respect to the state variables and evaluated at the equilibrium point to obtain the C matrix as shown in *GenerateResults.ipynb* code block 8:

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{8}$$

Similar to the A and B matrices, observability of the C matrix can be verified by checking if the columns are independent of each other. As shown in code blocks 8 and 9 of *GenerateResults.ipynb*, the number of states is equal to the rank of the new controllability matrix O.

Now the LQR method must be used to obtain new Q and R matrices for the observer gains matrix (L). The real eigenvalues of A - LC must be negative, or else the observer cannot be stable. This process is shown in further detail in code block 10. The observer eigenvalues are shown below:

$$S = \begin{bmatrix} -23.87 & -23.45 & -1.08 & -1.00 & -23.66 & -23.87 & -23.45 & -23.66 & -1.08 & -1.00 & -1.00 & -1.00 \end{bmatrix} \tag{9}$$

Lastly, a tracker must be implemented into the controller. The tracker is derived from the desired state, which depends on the position of the orange rings the drone must fly through. Hence, the desired state was the position of the next ring center with all other state variables set to zero.

$$\hat{x}_{desired} = \begin{bmatrix} p_{xdes} - p_{xe} & p_{ydes} - p_{ye} & p_{zdes} - p_{ze} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{10}$$

After applying the state feedback controller, observer, and tracker, the quadrotor can be implemented in simulation.

## V. Results

As aforementioned, this is verified through simulating 100 races with randomized ring and starting positions. The relevant data to be collected include the number of successful trials, the completion time for each trial, and average completion time. Three 100 trial simulations were run using the PyBullet and Matplotlib libraries, and the following data was collected from a set of 100 trials.

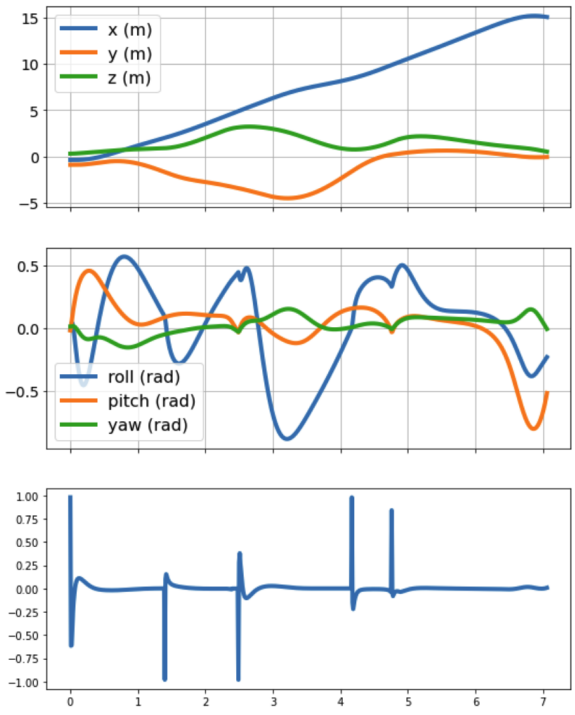


**Fig. 1 Trajectory of a Successful Drone Simulation**

| Trials | Successes | Average Time (s) |
|---|---|---|
| 100 | 54 | 14.9 |
| 100 | 57 | 15.4 |
| 100 | 51 | 14.7 |

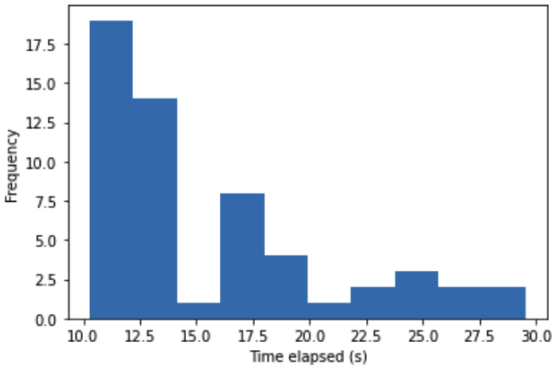


**Fig. 2 Completion Time of 100 Trials**

After comparing results from multiple PyBullet simulations, it is clear that the quadrotor completes the race in a timely manner (around 15 seconds) and is successful more than 50 percent of the trials. However, there are some trials that fall outside of the 20 second range. This was caused by including more rings into the flight path. Another cause is collisions and recovery time. When running individual trials, one of the major issues was the quadrotor bumping into the finish ring due to the low position of the last orange ring.

## VI. Conclusion

Overall, I was satisfied with the controller I designed since it met the requirement outlined at the beginning of the project. I gained more confidence when I was able to obtain a stable controller on the track. This project certainly taught me a significant amount of information regarding control systems and how even the smallest of changes to the equilibrium conditions caused the system to rapidly lose stability. Furthermore, this project helped me gain a new appreciation for the work that goes into drone controls. However, the optimal success rate in a real-life scenario would be 100 percent. It was already difficult for me to design a drone that is successful more than half the time, yet engineers have been able to design stable and extremely reliable multi-axis control systems that can must satisfy many more requirements than this project. There is a big difference between a working controller and an optimal controller. Ultimately, I feel I have learned a lot more about control systems and their practical applications from this project and am eager to apply these concepts in my future projects and career.

## Acknowledgments

I would like to thank Professor Melkior Ornik and TA Pranay Thangeda for their wonderful instruction and guidance. I would also like to acknowledge that the materials provided for this project, the help received from the Canvas discussion board, as well as information learned in the homework assignments assisted in building the foundation on which this project was created.