Project Statement Phase 1

RBE 502 - Robot Control

October 23, 2024

1 Problem Statement

The objective of the project is to design several controllers and a path planning system for a quadrotor that travels in a restricted airspace. Given an specific destination, the quadrotor should reach and hover at that destination without collision with any obstacles. Figure 1 illustrates a sketch of the concept.

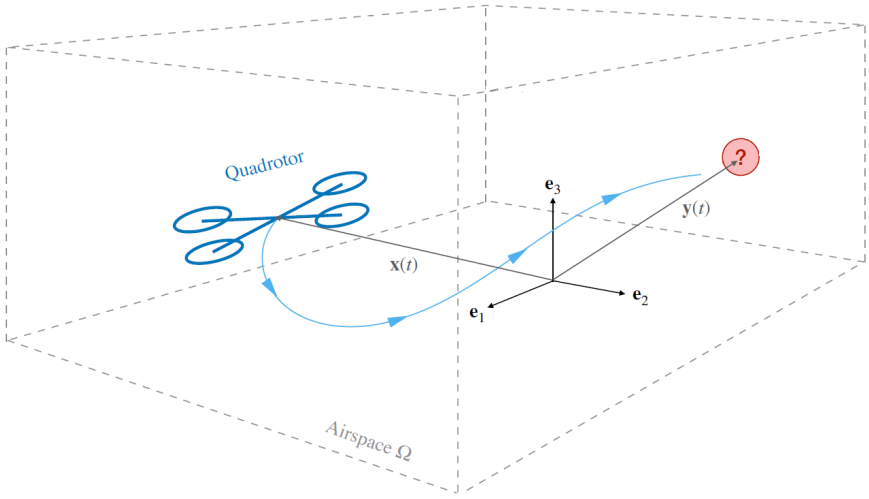

Figure 1: Conceptual illustration of the problem.

2 Problem Specifications

2.1 Coordinate Systems and Reference frames

The coordinate systems and free body diagram for the quadrotor are shown in Figure 2. The inertial frame, $\mathcal{A}$, is defined by the triad $\mathbf{a}_1, \mathbf{a}_2$, and $\mathbf{a}_3$ with $\mathbf{a}_3$ pointing upward. The body frame, $\mathcal{B}$, is attached to the center of mass of the quadrotor with $\mathbf{b}_1$ coinciding with the preferred forward direction and $\mathbf{b}_3$ perpendicular to the plane of the rotors pointing vertically up during perfect hover. These vectors are parallel to the principal axes. The center of mass is C. Rotor 1 is a distance L away along $\mathbf{b}_1$, 2 is L away along $\mathbf{b}_2$, while 3 and 4 are similarly L away along the negative $\mathbf{b}_1$ and $\mathbf{b}_2$ respectively.

2.2 Rigid Body Dynamics

Kinematics We will use $Z - X - Y$ Euler angles to model the rotation of the quadrotor in the world frame. To get from $\mathcal{A}$ to $\mathcal{B}$, we first rotate about $\mathbf{a}_3$ through the the yaw angle, ψ, to get the triad $\mathbf{e}_i$. A rotation about the $\mathbf{e}_1$ through the roll angle, ϕ gets us to the triad $\mathbf{f}_i$ (not shown in the figure). A third pitch rotation about $\mathbf{f}_2$ through θ results in the body-fixed triad $\mathbf{b}_i$. You will need the rotation matrix for transforming components of vectors in $\mathcal{B}$ to components of vectors in $\mathcal{A}$:

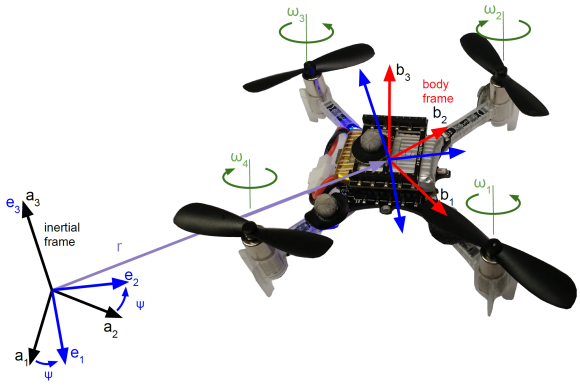

Figure 2: The CrazyFlie 2.0 robot. Note the reflective motion capture markers attached. A pair of motors spins counter clockwise while the other pair spins clockwise, such that when all propellers spin at the same speed, the net torque in the yaw direction is zero. The pitches on the corresponding propellers are reversed so that the thrust is always pointing in the $\mathbf{b}_3$ direction for all propellers. Shown also is the transformation from the inertial frame to the body-fixed frame. First a rotation by ψ around the $\mathbf{a_3}$ axis (leading to coordinate system $\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3}$) is performed, followed by a translation $\mathbf{r}$ to the center of mass C of the robot. Subsequent rotations by ϕ and θ generate the final body-fixed coordinate system $\mathcal{B}$, where the axes $\mathbf{b}_1$ and $\mathbf{b}_2$ are aligned with the arms, and $\mathbf{b}_3$ is perpendicular to them.

$$ {}^{\mathcal{A}}[R]_{\mathcal{B}} = \begin{bmatrix} c\psi c\theta - s\phi s\psi s\theta & -c\phi s\psi & c\psi s\theta + c\theta s\phi s\psi \\ c\theta s\psi + c\psi s\phi s\theta & c\phi c\psi & s\psi s\theta - c\psi c\theta s\phi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{bmatrix} $$

We will denote the components of angular velocity of the robot in the body frame by p, q, and r :

$$ {}^{\mathcal{A}}\omega_{\mathcal{B}} = p\mathbf{b}_1 + q\mathbf{b}_2 + r\mathbf{b}_3. $$

These values are related to the derivatives of the roll, pitch, and yaw angles according to

$$ \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} c\theta & 0 & -c\phi s\theta \\ 0 & 1 & s\phi \\ s\theta & 0 & c\phi c\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} $$

Newton's Equations of Motion Let $\mathbf{r}$ denote the position vector of C in $\mathcal{A}$. The forces on the system are gravity, in the $-\mathbf{a}_3$ direction, and the forces from each of the rotors, F_i, in the $\mathbf{b}_3$ direction. The equations governing the acceleration of the center of mass are

$$ m\ddot{\mathbf{r}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix} \tag{1} $$

We will define our first input u_1 to be

$$ u_1 = \Sigma_{i=1}^{4} F_i $$

Euler's Equations of Motion In addition to forces, each rotor produces a moment perpendicular to the plane of rotation of the blade, M_i. Rotors 1 and 3 rotate in the $-\mathbf{b}_3$ direction while 2 and 4 rotate in the $+\mathbf{b}_3$ direction. Since the moment produced on the quadrotor is opposite to the direction of rotation of the blades, M_1 and M_3 act in the $\mathbf{b}_3$ direction while M_2 and M_4 act in the $-\mathbf{b}_3$ direction. L is the distance from the axis of rotation of the rotors to the center of mass of the quadrotor.

The angular acceleration determined by the Euler equations is

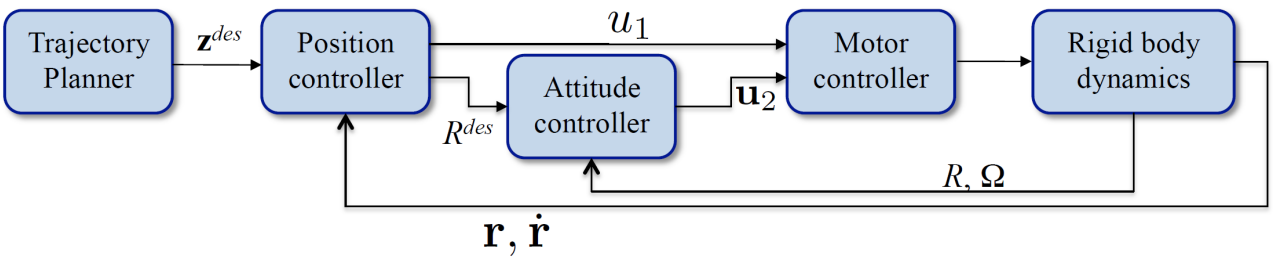

Figure 3: Block Diagram for a Quadrotor Control.

$$I \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} L(F_2 - F_4) \\ L(F_3 - F_1) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

We can rewrite this as:

$$I \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & L & 0 & -L \\ -L & 0 & L & 0 \\ \gamma & -\gamma & \gamma & -\gamma \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2}$$

where $\gamma = \frac{k_M}{k_F}$ is the relationship between lift and drag given by Equations (1-2). Accordingly, we will define our second set of inputs to be the vector of moments $\mathbf{u}_2$ given by:

$$\mathbf{u}_2 = \begin{bmatrix} 0 & L & 0 & -L \\ -L & 0 & L & 0 \\ \gamma & -\gamma & \gamma & -\gamma \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix}$$

3 Robot Controllers

3.1 The Nominal State

Our controllers are derived by linearizing the equations of motion and motor models at an operating point that corresponds to the nominal hover state, $\mathbf{r} = \mathbf{r}_0, \theta = \phi = 0, \psi = \psi_0, \dot{\mathbf{r}} = 0$, and $\dot{\phi} = \dot{\theta} = \dot{\psi} = 0$, where the roll and pitch angles are small ($c\phi \approx 1, c\theta \approx 1, s\phi \approx \phi$, and $s\theta \approx \theta$). At this state the lift from the propellers is given by:

$$F_{i,0} = \frac{mg}{4}$$

The nominal values for the inputs at hover are $u_{1,0} = mg, \mathbf{u}_{2,0} = \mathbf{0}$. Linearizing eq. (1), we get:

$$\begin{aligned} \ddot{r}_1 &= g(\Delta\theta \cos\psi_0 + \Delta\phi \sin\psi_0) \\ \ddot{r}_2 &= g(\Delta\theta \sin\psi_0 - \Delta\phi \cos\psi_0) \\ \ddot{r}_3 &= \frac{1}{m} u_1 - g \end{aligned} \tag{3}$$

Linearizing eq. (2), we get:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = I^{-1} \begin{bmatrix} 0 & L & 0 & -L \\ -L & 0 & L & 0 \\ \gamma & -\gamma & \gamma & -\gamma \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix}$$

If we assume the rotor craft is symmetric so $I_{xx} = I_{yy}$, we get:

$$\begin{aligned}
\dot{p} &= \frac{u_{2,x}}{I_{xx}} = \frac{L}{I_{xx}}\left(F_2 - F_4\right) \\
\dot{q} &= \frac{u_{2,y}}{I_{yy}} = \frac{L}{I_{yy}}\left(F_3 - F_1\right) \\
\dot{r} &= \frac{u_{2,z}}{I_{zz}} = \frac{\gamma}{I_{zz}}\left(F_1 - F_2 + F_3 - F_4\right)
\end{aligned} \tag{4}$$

In other words, the equations of motion are decoupled in terms of angular accelerations. Each component of angular acceleration depends only on the appropriate component of $\mathbf{u}_2$.

3.2 Position and Attitude Control

The control problem is to determine the four inputs, $\{u_1, \mathbf{u}_2\}$ required to hover or to follow a desired trajectory, $\mathbf{z}^{\text{des}}$. As shown in Figure 2, we will use errors in the robot's position to drive a position controller from eq. (3) which directly determines u_1. The model in eq. (3) also allows us to derive a desired orientation. The attitude controller for this orientation is derived from the model in eq. (4).

The transformation of the inputs into $\{u_1, \mathbf{u}_2\}$ is straightforward and is described in [1]. The inner attitude control loop uses onboard accelerometers and gyros to control the roll, pitch, and yaw and runs at approximately 500 Hz , while the outer position control loop uses estimates of position and velocity of the center of mass to control the trajectory in three dimensions at $100 - 200$ Hz.

3.2.1 Attitude Control

We now present a proportional plus derivative (PD) attitude controller to track a trajectory in $SO(3)$ specified in terms of a desired roll, pitch and yaw angle. Since our development of the controller will be based on linearized equations of motion, the attitude must be close to the nominal hover state where the roll and pitch angles are small.

Near the nominal hover state the proportional plus derivative control laws take the form:

$$\mathbf{u}_2 = I \begin{bmatrix} k_{p,\phi}\left(\phi^{\text{des}} - \phi\right) + k_{d,\phi}\left(p^{\text{des}} - p\right) \\ k_{p,\theta}\left(\theta^{\text{des}} - \theta\right) + k_{d,\theta}\left(q^{\text{des}} - q\right) \\ k_{p,\psi}\left(\psi^{\text{des}} - \psi\right) + k_{d,\psi}\left(r^{\text{des}} - r\right) \end{bmatrix} \tag{5}$$

3.2.2 Position Control

In this subsection, we will present two position control methods that use the roll and pitch angles as inputs to drive the position of the quadrotor. In both methods, the position control algorithm will determine the desired roll and pitch angles, θ^{des} and ϕ^{des} , which can be used to compute the desired speeds from eq. (4).

The first, a hover controller, is used for station-keeping or maintaining the position at a desired position vector, $\mathbf{r}_0$. The second tracks a specified trajectory, $\mathbf{r}_T(t)$, in three dimensions. In both cases, the desired yaw angle, is specified independently. It can either be a constant, ψ_0 or a time varying quantity, $\psi_T(t)$. We will assume that the desired trajectory:

$$\mathbf{z}^{des} = \begin{bmatrix} \mathbf{r}_T(t) \\ \psi_T(t) \end{bmatrix}$$

will be provided by a trajectory planner as an input to specify the trajectory of the position vector and the yaw angle we are trying to track.

Hover Controller For hovering, $\mathbf{r}_T(t) = \mathbf{r}_0$ and $\psi_T(t) = \psi_0$. The command accelerations, $\ddot{r}_i^{\text{des}}$, are calculated from a proportional plus derivative (PD) controller. Define the position error in terms of components using the standard reference triad $\mathbf{a}_i$ by:

$$e_i = \left(r_{i,T} - r_i\right)$$

In order to guarantee that this error goes exponentially to zero, we require

$$\left(\ddot{r}_{i,T} - \ddot{r}_i^{\text{des}}\right) + k_{d,i}\left(\dot{r}_{i,T} - \dot{r}_i\right) + k_{p,i}\left(r_{i,T} - r_i\right) = 0$$

where $\dot{r}_{i,T} = \ddot{r}_{i,T} = 0$ for hover.

From eq. (3) we can obtain the relationship between the desired accelerations and roll and pitch angles. Given that $\Delta\theta = \theta - \theta_0 = \theta$ and $\Delta\phi = \phi - \phi_0 = \phi$, we can write

$$\begin{aligned} \ddot{r}_1^{\text{des}} &= g\left(\theta^{\text{des}} \cos\psi_T + \phi^{\text{des}} \sin\psi_T\right) \\ \ddot{r}_2^{\text{des}} &= g\left(\theta^{\text{des}} \sin\psi_T - \phi^{\text{des}} \cos\psi_T\right) \\ \ddot{r}_3^{\text{des}} &= \frac{1}{m} u_1 - g \end{aligned} \tag{6}$$

For hover, the last equation yields:

$$u_1 = mg + m\dot{r}_3^{\text{des}} \ = mg - m\left(k_{d,3}\dot{r}_3 + k_{p,3}\left(r_3 - r_{3,0}\right)\right)$$

The other two equations can be used to compute the desired roll and pitch angles for the attitude controller:

$$\begin{aligned} \phi^{\text{des}} &= \frac{1}{g}\left(\ddot{r}_1^{\text{des}} \sin\psi_T - \ddot{r}_2^{\text{des}} \cos\psi_T\right) \\ \theta^{\text{des}} &= \frac{1}{g}\left(\ddot{r}_1^{\text{des}} \cos\psi_T + \ddot{r}_2^{\text{des}} \sin\psi_T\right) \end{aligned} \tag{7}$$

The desired roll and pitch velocities are taken to be zero.

$$\begin{aligned} p^{\text{des}} &= 0 \\ q^{\text{des}} &= 0 \end{aligned} \tag{8}$$

Since the yaw, $\psi_T(t)$ is prescribed independently by the trajectory generator, we get:

$$\begin{aligned} \psi^{\text{des}} &= \psi_T(t) \\ r^{\text{des}} &= \dot{\psi}_T(t) \end{aligned} \tag{9}$$

These equations provide the setpoints for the attitude controller in eq.(5). Note that the attitude controller must run an order of magnitude faster than the position control loop in order for this to work. In practice as discussed in , the position controller runs at 100 Hz , while the inner attitude control loop runs at 500 Hz .

3D Trajectory Control The 3-D Trajectory Controller is used to follow three-dimensional trajectories with modest accelerations so the nearhover assumptions hold. To derive this controller we follow the same steps as in eq.(6) except that $\dot{r}_{i,T}$ and $\ddot{r}_{i,T}$ are no longer zero but are obtained from the specification of the trajectory. If the near-hover assumption holds and the dynamics are linear with no saturation on the inputs, a controller that generates the desired acceleration $\ddot{r}_i^{\text{des}}$ according to eq.(6) is guaranteed to drive the error exponentially to zero. However, it is possible that the commanded trajectory has twists and turns that are too hard to follow perfectly because of errors in the model or limitations on the input thrusts. We suggest a modification that allows the robot to follow the path even if it may not be able to follow the time-parameterized trajectory.

Let $\mathbf{r}_T$ denote the closest point on the desired trajectory to the the current position $\mathbf{r}$, and let the desired velocity and acceleration obtained by differentiating the specified trajectory be given by $\dot{\mathbf{r}}_T$ and $\ddot{\mathbf{r}}_T$ respectively. Let the unit tangent vector of the trajectory (unit vector along $\dot{\mathbf{r}}_T$) be $\hat{\mathbf{t}}$. The unit normal to the trajectory, $\hat{\mathbf{n}}$, is derived by differentiating the tangent vector with respect to time or arc length, and finally the unit binormal vector is the cross product $\hat{\mathbf{b}} = \hat{\mathbf{t}} \times \hat{\mathbf{n}}$. We define the position and velocity errors according to the following equations:

$$\mathbf{e}_p = \left(\left(\mathbf{r}_T - \mathbf{r}\right) \cdot \hat{\mathbf{n}}\right) \hat{\mathbf{n}} + \left(\left(\mathbf{r}_T - \mathbf{r}\right) \cdot \hat{\mathbf{b}}\right) \hat{\mathbf{b}}$$

and

$$\mathbf{e}_v = \dot{\mathbf{r}}_T - \dot{\mathbf{r}}$$

Note that here we ignore position error in the tangential direction and only considering position error in the plane that is normal to the curve at the closest point. Once again we calculate the commanded acceleration, $\ddot{r}_i^{\text{des}}$, from the PD feedback as shown in eq.(6). In vector notation, this is:

$$\left(\ddot{\mathbf{r}}_T - \ddot{\mathbf{r}}^{\text{des}}\right) + \mathbf{k}_d \mathbf{e}_v + \mathbf{k}_p \mathbf{e}_p = 0$$

Finally we use (7 - 9) to compute the desired roll and pitch angles. Again we refer the readers to [1] for more details including experimental results obtained from these controllers.

4 Project Work Phase 1: PD/PID Controller

(Due November 5th, 2024, 11:59 PM)

You will then implement a PID/PD controller in file controller.m that enables the quadrotor to travel and hover at a destination. This will make sure that your quadrotor follows the desired trajectory. This controller will be used again in the following phases of the project. The controller takes in a cell struct qd, current time t, current quadrotor number qn and quadrotor parameters params and outputs thrust u_1, moments $\mathbf{u}_2$, desired thrust, roll, pitch and yaw trpy and derivatives of desired roll, pitch and yaw drpy.

In this phase, since we are doing simulation, you will only need to output thrust u_1 and moments $\mathbf{u}_2$. qd is a cell array contains structs for each quadrotor. This means qd{qn} contains all the information (position, velocity, euler angles and etc.) about quadrotor number qn. Those information can be accessed via qd $\{qn\}$. pos for example.

Result Expectation:

1. You are expected to submit a controller.m file that implements a PD/PID controller that outputs the correct commands. If your controller cannot output the correct commands and the simulation died, you will be graded 0 for this phase.

2. In the grading, we will initialize the quadrotor at 3 random hovering positions and headings. We will use your controller to travel to the destination. With your controller, the quadrotor should be able to travel to the destination $(0, 0, 0)$ and hover there within an error range of 0.05 m.

5 Submission

The students are expected to submit the following files:

1. pid_controller.m

References

[1] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The grasp multiple micro-uav testbed," *IEEE Robotics and Automation Magazine*, vol. 17, no. 3, pp. 56–65, 2010.