Reminder of our goal: Develop techniques for making a quadrotor hover in place.

Approach we are pursuing:

(1) Figure out how the quadrotor will behave when you apply different propeller commands [**Dynamics**].
(2) Figure out a mechanism for taking corrective actions when the quadrotor moves away from the desired hover configuration [**Feedback Control**].

Last lecture:

(1) Discussed dynamics of planar quadrotor.
(2) Started discussing dynamics of 3D quadrotor.

Plan for today: Complete discussing dynamics of 3D quadrotor and start discussion of feedback control.

1. 3D QUADROTOR DYNAMICS

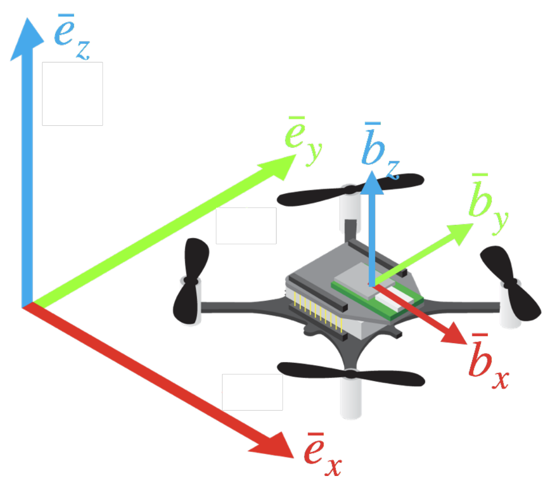


FIGURE 1. Axes for (Crazyflie) quadrotor; source: Bitcraze.io.

States: $\bar{x} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, p, q, r]^T$.

Here, $[\phi, \theta, \psi]$ are Euler angles (in space 1-2-3 convention). Also, $[p, q, r]$ corresponds to the angular velocity vector (expressed in the body frame); the direction of this vector corresponds to the instantaneous axis of rotation and the magnitude corresponds to the instantaneous rate of rotation. The vector $[p, q, r]$ is directly related to and can be computed from $\dot{\phi}, \dot{\theta}, \dot{\psi}$.

Control inputs: $\bar{u} = [F_{tot}, M_1, M_2, M_3]^T$.

Here, F_{tot} is the total thrust from the propellers, and M_1, M_2, M_3 are the moments (torques) about the body frame axes $\bar{b}_x, \bar{b}_y, \bar{b}_z$ (see Figure **??**).

2. Motor model

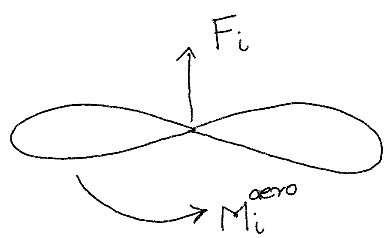
For the planar quadrotor, we discussed the relationship between thrust produced by a rotor and the rotor's angular speed:

$$F_i = k_f \omega_i^2, \quad i = 1, 2, 3, 4, \tag{1}$$

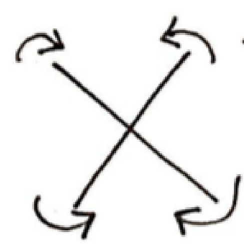
i.e., the thrust F_i produced by rotor i is equal to the thrust coefficient k_f times the rotational speed of rotor i squared.

For the 3D setting, the spinning of the propellers (rotors) results in both a thrust *and* a moment.

The moment is due to **aerodynamic drag**, and acts about the $\bar{b}_z$ direction. This is actually what allows the quadrotor to turn in the yaw direction.



(a) Directions of thrust force and aerodynamic drag moment.



(b) Top view: Adjacent rotors spin in *opposite* directions. Thus, the moments they produce are in opposite directions.

Figure 2

The aerodynamic moment produced by propeller i is given by:

$$M_i^{\text{aero}} = k_m \omega_i^2, \quad i = 1, 2, 3, 4, \tag{2}$$

where k_m is the drag/moment coefficient.

3. Full 3D quadrotor dynamics (sketch)

Remember that dynamics will be of the form:

$$\dot{\bar{x}} = f(\bar{x}, \bar{u}). \tag{3}$$

In Assignment 1, you will have a chance to go through and type in the full 3D dynamics. I'll just sketch the equations here.

Define:

$$\bar{r} \triangleq \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \text{(Position vector)} \tag{4}$$

Then,

$$\dot{r} \triangleq \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \qquad \text{(Velocity vector)} \tag{5}$$

From Newton's 2nd law:

$$\ddot{r} \triangleq \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \bar{R} \begin{bmatrix} 0 \\ 0 \\ F_{tot}/m \end{bmatrix}, \tag{6}$$

where $\bar{R}$ is a rotation matrix that takes vectors in the body frame and rotates them to the inertial frame.

Define:

$$\bar{\omega}_{BW} \triangleq \begin{bmatrix} p \\ q \\ r \end{bmatrix} \qquad \text{(Angular velocity vector)} \tag{7}$$

Then,

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ \dots & \dots & \dots \\ \dots & \dots & \dots \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{8}$$

$$\dot{\bar{\omega}}_{BW} = \mathbb{I}^{-1} \left[-\bar{\omega}_{BW} \times (\mathbb{I}\bar{\omega}_{BW}) + \begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \right], \tag{9}$$

where $\times$ is the cross product and $\mathbb{I}$ is known as the *inertia matrix*:

$$\mathbb{I} \triangleq \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{10}$$

Note: Again, I don't expect you to derive these equations. I just want you to understand that the equations have the form $\dot{\bar{x}} = f(\bar{x}, \bar{u})$. If you are interested, you can look at this reference: C. Powers, D. Mellinger, and V. Kumar, "Quadrotor Kinematics and Dynamics", Handbook of Unmanned Aerial Vehicles.

4. Linearizing the dynamics

The dynamics of both the 2D and 3D quadrotor are **nonlinear**. This makes things much more complicated when we do control. We will approximate the dynamics using **linear** approximations. Such an approximation will be good enough to make the drone hover.

Planar quadrotor dynamics (from previous lecture):

$$\dot{\bar{x}} = \frac{d}{dt} \begin{bmatrix} x \\ y \\ \theta \\ \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = f(\bar{x}, \bar{u}) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ -\frac{u_1}{m}\sin\theta \\ \frac{u_1}{m}\cos\theta - g \\ \frac{u_2}{I} \end{bmatrix} \tag{11}$$

We need some "nominal" (i.e., reference) point to linearize dynamics around. For hover, this will be:

$$\bar{x}_0 = \begin{bmatrix} x_0 \\ y_0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \bar{u}_0 = \begin{bmatrix} mg \\ 0 \end{bmatrix}. \tag{12}$$

Note: we can just take x_0, y_0 to be 0 if we want (by shifting the origin of our coordinate system). The nominal control input u_0 corresponds to the total thrust (mg) and moment (0) required to remain in the hover configuration if the robot starts off there.

Note:

$$\dot{\bar{x}} = f(\bar{x}_0, \bar{u}_0) = \bar{0}. \tag{13}$$

Intuitively, if the quadrotor starts off at $\bar{x}_0$ and applies $\bar{u}_0$, it will stay at $\bar{x}_0$.

We can linearize $f(\bar{x}, \bar{u})$ about $\bar{x}_0, \bar{u}_0$ to get:

$$\dot{\bar{x}} = A(\bar{x} - \bar{x}_0) + B(\bar{u} - \bar{u}_0), \tag{14}$$

where A and B are constant matrices (of size 6×6 and 6×2 respectively for the planar quadrotor).

In particular, we can use a Taylor expansion (from multivariable calculus):

$$\dot{\bar{x}} = f(\bar{x}, \bar{u}) \approx \underbrace{f(\bar{x}_0, \bar{u}_0)}_{=0} + \underbrace{\left[\frac{\partial f}{\partial \bar{x}}\right]\Bigg|_{\substack{\bar{x}=\bar{x}_0 \\ \bar{u}=\bar{u}_0}}}_{:=A} (\bar{x} - \bar{x}_0) + \underbrace{\left[\frac{\partial f}{\partial \bar{u}}\right]\Bigg|_{\substack{\bar{x}=\bar{x}_0 \\ \bar{u}=\bar{u}_0}}}_{:=B} (\bar{u} - \bar{u}_0). \tag{15}$$

For the planar quadrotor:

$$A = \left.\begin{bmatrix}
\frac{\partial \dot{x}}{\partial x} & \frac{\partial \dot{x}}{\partial y} & \frac{\partial \dot{x}}{\partial \theta} & \frac{\partial \dot{x}}{\partial \dot{x}} & \frac{\partial \dot{x}}{\partial \dot{y}} & \frac{\partial \dot{x}}{\partial \dot{\theta}} \\
\frac{\partial \dot{y}}{\partial x} & \frac{\partial \dot{y}}{\partial y} & \frac{\partial \dot{y}}{\partial \theta} & \frac{\partial \dot{y}}{\partial \dot{x}} & \frac{\partial \dot{y}}{\partial \dot{y}} & \frac{\partial \dot{y}}{\partial \dot{\theta}} \\
\frac{\partial \dot{\theta}}{\partial x} & \frac{\partial \dot{\theta}}{\partial y} & \frac{\partial \dot{\theta}}{\partial \theta} & \frac{\partial \dot{\theta}}{\partial \dot{x}} & \frac{\partial \dot{\theta}}{\partial \dot{y}} & \frac{\partial \dot{\theta}}{\partial \dot{\theta}} \\
\frac{\partial(-\frac{u_1}{m}\sin\theta)}{\partial x} & \frac{\partial(-\frac{u_1}{m}\sin\theta)}{\partial y} & \frac{\partial(-\frac{u_1}{m}\sin\theta)}{\partial \theta} & \frac{\partial(-\frac{u_1}{m}\sin\theta)}{\partial \dot{x}} & \frac{\partial(-\frac{u_1}{m}\sin\theta)}{\partial \dot{y}} & \frac{\partial(-\frac{u_1}{m}\sin\theta)}{\partial \dot{\theta}} \\
\frac{\partial(\frac{u_1}{m}\cos\theta - g)}{\partial x} & \frac{\partial(\frac{u_1}{m}\cos\theta - g)}{\partial y} & \frac{\partial(\frac{u_1}{m}\cos\theta - g)}{\partial \theta} & \frac{\partial(\frac{u_1}{m}\cos\theta - g)}{\partial \dot{x}} & \frac{\partial(\frac{u_1}{m}\cos\theta - g)}{\partial \dot{y}} & \frac{\partial(\frac{u_1}{m}\cos\theta - g)}{\partial \dot{\theta}} \\
\frac{\partial(\frac{u_2}{I})}{\partial x} & \frac{\partial(\frac{u_2}{I})}{\partial y} & \frac{\partial(\frac{u_2}{I})}{\partial \theta} & \frac{\partial(\frac{u_2}{I})}{\partial \dot{x}} & \frac{\partial(\frac{u_2}{I})}{\partial \dot{y}} & \frac{\partial(\frac{u_2}{I})}{\partial \dot{\theta}}
\end{bmatrix}\right|_{\substack{\bar{x}=\bar{x}_0 \\ \bar{u}=\bar{u}_0}} \tag{16}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -g & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{17}$$

Similarly,

$$B = \left.\begin{bmatrix} \frac{\partial \dot{x}}{\partial u_1} & \frac{\partial \dot{x}}{\partial u_2} \\ \frac{\partial \dot{y}}{\partial u_1} & \frac{\partial \dot{y}}{\partial u_2} \\ \frac{\partial \dot{\theta}}{\partial u_1} & \frac{\partial \dot{\theta}}{\partial u_2} \\ \frac{\partial(-\frac{u_1}{m}\sin\theta)}{\partial u_1} & \frac{\partial(-\frac{u_1}{m}\sin\theta)}{\partial u_2} \\ \frac{\partial(\frac{u_1}{m}\cos\theta - g)}{\partial u_1} & \frac{\partial(\frac{u_1}{m}\cos\theta - g)}{\partial u_2} \\ \frac{\partial(\frac{u_2}{I})}{\partial u_1} & \frac{\partial(\frac{u_2}{I})}{\partial u_2} \end{bmatrix}\right|_{\substack{\bar{x}=\bar{x}_0 \\ \bar{u}=\bar{u}_0}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{m} & 0 \\ 0 & \frac{1}{I} \end{bmatrix}. \tag{18}$$

Note: In practice, these matrices can be annoying to calculate by hand. In Assignment 1, you will use Python to compute these matrices automatically for the 3D quadrotor. But it's good to understand how it works and go through the exercise of computing these by hand on your own for simpler systems like the planar quadrotor above.

5. Feedback Control

Question: What is feedback control and why do we need it?

Recall our discussion of the "sense-think-act" cycle from Lecture 1. Feedback control is an instantiation of this.

Figure **??** shows the anatomy of a control system. [Think about how general this framework is. Even though I am introducing this using quadrotors as an example, the framework is broadly applicable, e.g., making a humanoid robot balance, manipulating an object, etc.]

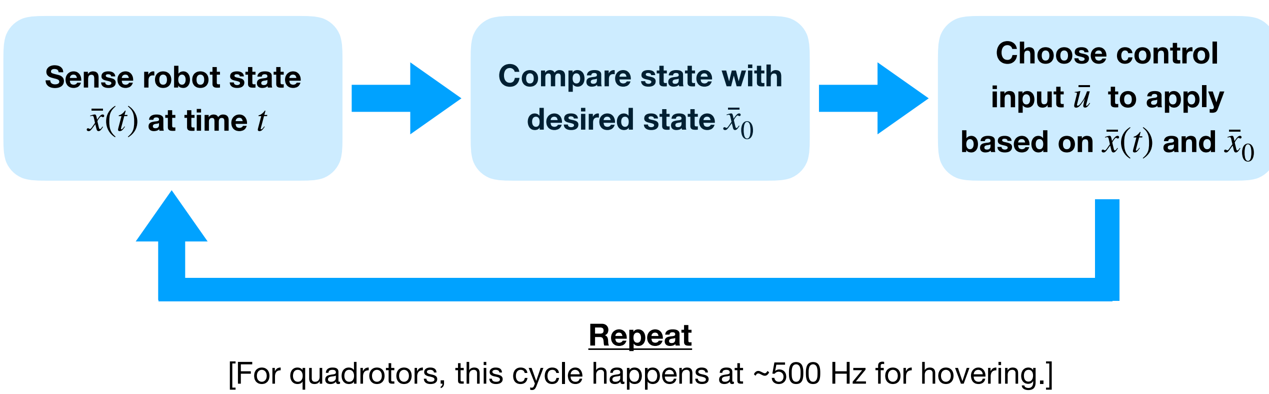
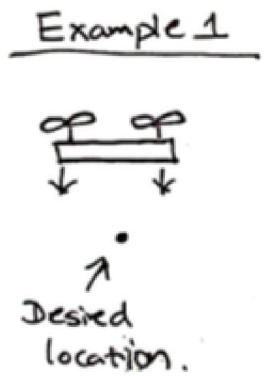
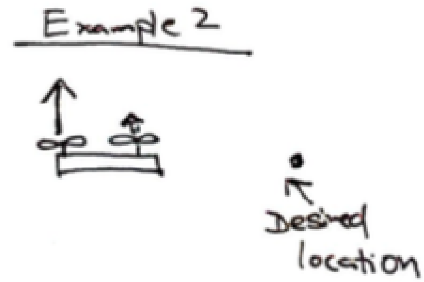


Figure 3. Anatomy of a control system.

Concrete example: Consider the planar quadrotor again.





In general, we want some strategy for taking actions (i.e., applying control inputs) that will drive the state to the desired state.

Feedback controller (or "control law"): Some function $\bar{u}(\bar{x})$ that achieves this.

For example, we could use:

$$\bar{u}(\bar{x}) \triangleq \bar{u}_0 + \bar{K}(\bar{x} - \bar{x}_0).$$

Think about what this is doing. The term $\bar{u}_0$ is the control input that is required for keeping the system in the hovering configuration, $\bar{K}$ is a matrix that we choose (we will say much more about how to choose it), and $\bar{x}_0$ is the desired state (e.g., corresponding to hovering). If the robot is already at the desired state (i.e., $\bar{x} = \bar{x}_0$), then the feedback controller just applied $\bar{u}_0$; however, if $\bar{x} \neq \bar{x}_0$, then we apply a control input that depends on the deviation of the state $\bar{x}$ from the desired state $\bar{x}_0$.

The whole game in control theory is to come up with these functions (i.e., feedback controllers) in a principled way. We will say much more about this in the next two lectures.

6. Why do we need feedback control?

To deal with **uncertainty**! As I mentioned in the first lecture, uncertainty is going to be a recurring theme in this course.

What kind of uncertainty does feedback control address?

- Uncertainty in initial conditions (this is the most direct application).
- Uncertainty about external disturbances (e.g., wind gusts).
- Uncertainty about model parameters (e.g., k_f, k_m, I).
- Uncertainty regarding the true state of the system (we will say more about this in the latter part of the course).