Drone Stabilization

Matthew & friends

April 2025

Setting up the model

We denote the state of a quadcopter drone in terms of its position and state as

$$\mathbf{s}(t) = \big(x(t), y(t), z(t), \phi(t), \theta(t), \psi(t)\big)$$

where (x, y, z) represents the position of the center of the drone, and (ϕ, θ, ψ) represents the orientation of the drone in terms of its Tait-Bryan angles, a set of three angles that describes the orientation where $(\phi, \theta, \psi) = \mathbf{0}$ means the drone is not tilted and points north.

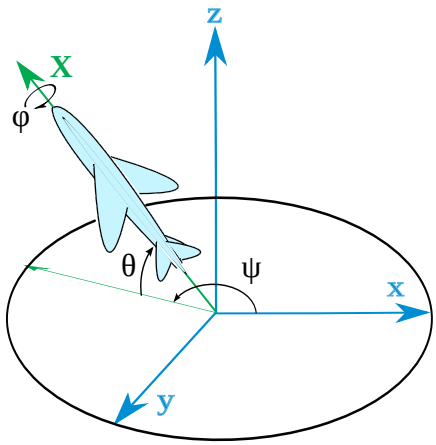

Figure 1: Tait-Bryan angles we're using (just imagine a drone instead of a plane!)

We denote our control over the speed rotors of the drone with

$$\mathbf{u} = \big(u_1(t), u_2(t), u_3(t), u_4(t)\big)$$

Explain u

We set a fixed final time t_f and seek to control the drone via the cost functional

$$J(\mathbf{u}) = \int_0^{t_f} \Big(\|\mathbf{u}\|_2^2 + \alpha\|\mathbf{s}\|_2^2\Big)dt$$

where $\alpha > 0$ weights our hopes to quickly stabilize the drone versus save energy.

We set $\mathbf{s}(0)$ to be some arbitrary $\mathbf{s}_0$ and allow $\mathbf{s}(t_f)$ to be free, allowing the cost functional to ensure that our state remains stable.

State evolution

We proceed to describe how the control interacts with the state of the drone.

Add Newton-Euler equations

We make the assumption that the angular velocity is always small, allowing us to ignore the last term.

The force on the drone from its rotors when $(\phi, \theta, \psi) = \mathbf{0}$ is simply

$$\begin{pmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} u_i \end{pmatrix}$$

since the rotors thrust upward relative to the drone.

In the same orientation, the torque on the drone is

$$\begin{pmatrix} u_2 - u_4 \\ u_3 - u_1 \\ \lambda(u_1 + u_3 - u_2 - u_4) \end{pmatrix}$$

with $\lambda > 0$, since two of the rotors spin clockwise and the other two spin counter-clockwise, causing a bit of torque rotating the drone.

Explain further

Now we need to be able to express this force no matter what the orientation of the drone is. We use the following rotation matrices described in this Wikipedia article on the subject:

$$R_x(\phi) = \text{Roll}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

$$R_y(\theta) = \text{Pitch}(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\psi) = \text{Yaw}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The matrix of the composed rotations is

$$R = \text{Yaw}(\psi)\,\text{Pitch}(\theta)\,\text{Roll}(\phi) = R_z(\psi)R_y(\theta)R_x(\phi).$$

We can express the force and torque on the drone at an arbitrary orientation with R and g as a gravitational constant:

$$\mathbf{F} = R(\phi, \theta, \psi) \begin{pmatrix} 0 \\ 0 \\ \sum_i u_i - g \end{pmatrix}$$
$$\boldsymbol{\tau} = R(\phi, \theta, \psi) \begin{pmatrix} u_2 - u_4 \\ u_3 - u_1 \\ \lambda(u_1 + u_3 - u_2 - u_4) \end{pmatrix}$$

Now we are ready to model the evolution of the state of the drone. The Newton-Euler equations extend the classic relationship $F = ma$ into three dimensions and add in torque and gyroscopic effects for a moving object.

$$\begin{pmatrix} \boldsymbol{F} \\ \boldsymbol{\tau} \end{pmatrix} = \begin{pmatrix} mI & 0 \\ 0 & \mathcal{I} \end{pmatrix} \ddot{\boldsymbol{s}} + \begin{pmatrix} 0 \\ \boldsymbol{\omega} \times \mathcal{I}\boldsymbol{\omega} \end{pmatrix}$$

where m is the mass of the object, I is a 3×3 identity matrix, $\mathcal{I}$ is a diagonal matrix representing the moment of inertia of the object, and $\boldsymbol{\omega} = (\dot{\phi}, \dot{\theta}, \dot{\psi})$.

We assume the angular velocity $\boldsymbol{\omega}$ (we hope the drone is not rapidly flipping end over end or spinning like a top), reorganize the equation, and plug in our model for $\boldsymbol{F}$ and $\boldsymbol{\tau}$ to yield the state evolution equation

$$\ddot{\mathbf{s}} = \begin{pmatrix} mI & \mathbf{0} \\ \mathbf{0} & \mathcal{I} \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{F} \\ \boldsymbol{\tau} \end{pmatrix} = \begin{pmatrix} mI & \mathbf{0} \\ \mathbf{0} & \mathcal{I} \end{pmatrix}^{-1} \begin{pmatrix} R(\phi, \theta, \psi) \begin{pmatrix} 0 \\ 0 \\ \sum_i u_i \end{pmatrix} \\ R(\phi, \theta, \psi) \begin{pmatrix} u_2 - u_4 \\ u_3 - u_1 \\ \lambda(u_1 + u_3 - u_2 - u_4) \end{pmatrix} \end{pmatrix} \tag{1}$$

Reducing to first-order system

The state evolution equation described above is second-order. We hope to work with a first order system, so we define

$$\boldsymbol{\sigma} = \begin{pmatrix} \boldsymbol{\sigma}_1 \\ \boldsymbol{\sigma}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{s} \\ \dot{\boldsymbol{s}} \end{pmatrix} = (x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi})$$

and observe that

$$\dot{\sigma} = \begin{pmatrix} \dot{\boldsymbol{\sigma}}_1 \\ \dot{\boldsymbol{\sigma}}_2 \end{pmatrix} = \begin{pmatrix} \dot{\boldsymbol{s}} \\ \ddot{\boldsymbol{s}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\sigma}_2 \\ \ddot{\boldsymbol{s}} \end{pmatrix}$$

with $\ddot{\boldsymbol{s}}$ as in Equation 1. This first order equation is the equation we will work with in what follows, thus thinking of the "state" as the 12-dimensional $\boldsymbol{\sigma}$ encoding both translational and angular position and velocity.

Deriving optimal control

We now set up the Hamiltonian H and solve for the optimal control. Let $\boldsymbol{p}(t)$ be the 12-dimensional costate evolving alongside $\boldsymbol{\sigma}(t)$.

$$H = \boldsymbol{p} \cdot \dot{\boldsymbol{\sigma}} - \Big(\|\mathbf{u}\|_2^2 + \alpha \|\mathbf{s}\|_2^2 \Big)$$

Since $\boldsymbol{u}$ maximizes H,

$$\begin{aligned} 0 &= \frac{\partial H}{\partial \boldsymbol{u}} \\ &= \boldsymbol{p} \cdot \frac{\partial}{\partial \boldsymbol{u}} \dot{\boldsymbol{\sigma}} - \frac{\partial}{\partial \boldsymbol{u}} \|\boldsymbol{u}\|_2^2 - \alpha \frac{\partial}{\partial \boldsymbol{u}} \|\boldsymbol{s}\|_2^2 \begin{pmatrix} mI & \mathbf{0} \\ \mathbf{0} & \mathcal{I} \end{pmatrix}^{-1} \begin{pmatrix} R(\phi,\theta,\psi) \begin{pmatrix} 0 \\ 0 \\ \sum_i u_i \end{pmatrix} \\ R(\phi,\theta,\psi) \begin{pmatrix} u_2 - u_4 \\ u_3 - u_1 \\ \lambda(u_1 + u_3 - u_2 - u_4) \end{pmatrix} \end{pmatrix} \\ &= \boldsymbol{p} \cdot \frac{\partial}{\partial \boldsymbol{u}} \dot{\boldsymbol{\sigma}} - 2\boldsymbol{u} - 0 \\ \implies \boldsymbol{u} &= \frac{1}{2} \Big(\boldsymbol{p} \cdot \frac{\partial}{\partial \boldsymbol{u}} \dot{\boldsymbol{\sigma}} \Big) \end{aligned} \tag{2}$$

We need to calculate $\frac{\partial}{\partial \boldsymbol{u}}\dot{\boldsymbol{\sigma}}$. Let $i \in \{1, \ldots, 12\}$.

$$
\begin{aligned}
\frac{\partial}{\partial u_i}\dot{\boldsymbol{\sigma}} &= \frac{\partial}{\partial u_i}\begin{pmatrix}\boldsymbol{\sigma}_2 \\ \ddot{\boldsymbol{s}}\end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{\partial}{\partial u_i}\begin{pmatrix} mI & \mathbf{0} \\ \mathbf{0} & \mathcal{I}\end{pmatrix}^{-1}\begin{pmatrix} R(\phi,\theta,\psi)\begin{pmatrix} 0 \\ 0 \\ \sum_i u_i\end{pmatrix} \\ R(\phi,\theta,\psi)\begin{pmatrix} u_2 - u_4 \\ u_3 - u_1 \\ \lambda(u_1+u_3-u_2-u_4)\end{pmatrix}\end{pmatrix}\end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \begin{pmatrix} mI & \mathbf{0} \\ \mathbf{0} & \mathcal{I}\end{pmatrix}^{-1}\begin{pmatrix} R(\phi,\theta,\psi)\begin{pmatrix} 0 \\ 0 \\ \frac{\partial}{\partial u_i}\sum_i u_i\end{pmatrix} \\ R(\phi,\theta,\psi)\frac{\partial}{\partial u_i}\begin{pmatrix} u_2 - u_4 \\ u_3 - u_1 \\ \lambda(u_1+u_3-u_2-u_4)\end{pmatrix}\end{pmatrix}\end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \begin{pmatrix} mI & \mathbf{0} \\ \mathbf{0} & \mathcal{I}\end{pmatrix}^{-1}\begin{pmatrix} R(\phi,\theta,\psi)\begin{pmatrix} 0 \\ 0 \\ 1\end{pmatrix} \\ R(\phi,\theta,\psi)\frac{\partial}{\partial u_i}\begin{pmatrix} u_2 - u_4 \\ u_3 - u_1 \\ \gamma_i\end{pmatrix}\end{pmatrix}\end{pmatrix}
\end{aligned}
\tag{3}
$$

where

$$
\gamma_i = \Big\{(0,-1,\lambda) \quad \text{if } i = 1 \tag{4}
$$