Hover Stabilizing Drone

Overview

I apply principles taught in ENME605 Advanced Systems Control to design and simulate a full-state feedback control system for hover control of a quadcopter drone. I begin by deriving a state-space model, linearizing the equations of motion about the hover state, and verifying controllability. Using the Linear Quadratic Regulator (LQR) optimal control framework, I design the quadcopter's hover controller and improve upon it by refining the LQR weighting matrices. I then confirm observability and design an observer via pole placement to reconstruct unmeasured states. My focus with this project is on applying state-space modeling, LQR, and state estimation techniques. Finally, I evaluate the performance of the controller with Monte Carlo trials, determining a 3.5s 95th %tile settling time with 2cm/2deg under +-25% pertubations with <11% thrust overhead.

Deriving the Plant Dynamics

I derive the state-space model for the quadcopter drone. I define the physical parameters of the drone, I define the quadcopter's degrees of freedom, I define a suitable state vector, I define the quadcopter's control inputs, I derive the equations of motion, I linearize the equations of motion about the hover condition, and I cast the linearized equations of motion into a state-space form suitable for linear control design.

Defining the Quadcopter's Physical parameters

I define the quadcopter's physical parameters. I represent the quadcopter as a rigid body that has both rotational symmetry and line symmetry. Viewed from above, the quadcopter is X-shaped, having 90° rotational symmetry about its z-axis and line symmetry across its x and y axes. Its inertia tensor is a diagonal matrix. The force of gravity acts on the quadcopter at all times. These mass properties are consistent with those of consumer-grade micro-UAVs.

Defining the Quadcopter's Degrees of Freedom

I define the quadcopter's degrees of freedom. The quadcopter has six degrees of freedom, three translational and three rotational. I employ Euler angles to represent the three rotational degrees of freedom. The particular Euler angle sequence I employ for these derivations is the Z-Y-X Euler angle sequence.

Table: Quadcopter Degrees of Freedom

Category	Property	Symbol
Linear Position	X-coordinate	x
	Y-coordinate	y
	Z-coordinate	z

Category	Property	Symbol
Orientation	Roll (about X-axis)	φ (phi)
	Pitch (about Y-axis)	θ (theta)
	Yaw (about Z-axis)	ψ (psi)

Defining the Quadcopter's State Variables

I define the quadcopter's state variables. There are three state variables for linear position, three state variables for 3D orientation, three state variables for linear velocity, and three state variables for angular velocity. All state variables corresponding to linear position and linear velocity are world-frame coordinates. The angular displacement state variables are the Z-Y-X Euler angles. The angular velocity state variables are measured in the body frame.

$$
\mathbf{x} =
\begin{bmatrix}
x & y & z & \phi & \theta & \psi & \dot{x} & \dot{y} & \dot{z} & p & q & r
\end{bmatrix}^\top
$$

Defining the Quadcopter's Control inputs

I define the quadcopter's control inputs. There are four control inputs. The first is the total thrust force produced by the quadcopter. This force is always directed in the positive z-direction of the body frame. The second, third, and fourth control input are the roll moment, the pitch moment, and the yaw moment, in that order.

$$
\begin{equation}
\mathbf{u} =
\begin{bmatrix}
F_c & M_x & M_y & M_z
\end{bmatrix}^\top
\end{equation}
$$

Deriving the Quadcopter's Equations of Motion

I derive the quadcopter's nonlinear equations of motion. To do so, I employ the Newton-Euler equations from classical mechanics. The result is a nonlinear system of second-order differential equations that will serve the purpose of simulating the quadcopter's true dynamics, that is, how the quadcopter moves in response to control inputs.

$$
\begin{aligned}
\ddot{x} &= \frac{F_c (\sin(\phi) \sin(\psi) + \cos(\phi) \cos(\psi) \sin(\theta))}{m}, \

$$
\ddot{y} &= -\frac{F_c (\cos(\psi) \sin(\phi) - \cos(\phi) \sin(\psi) \sin(\theta))}{m}, \
\ddot{z} &= \frac{F_c \cos(\phi) \cos(\theta)}{m} - g, \
\dot{p} &= \frac{M_x}{I_x} - \frac{(I_z - I_y) r q}{I_x}, \
\dot{q} &= \frac{M_y}{I_y} - \frac{(I_x - I_z) p r}{I_y}, \
\dot{r} &= \frac{M_z}{I_z} - \frac{(I_y - I_x) q p}{I_z}.
\end{aligned}
$$

Linearizing the Equations of Motion about the Hover State

I linearize the equations of motion about the hover state. Since the quadcopter represents a nonlinear system and since LQR control requires a linear dynamics model of the plant rather than a nonlinear model, linearization is necessary. Since my control objective is hover stabilization, I linearize the dynamics about the hover state operating point. At the hover point, the quadcopter's state is effectively "zeroed", representative of having straight and level orientation, not translating, not rotating, and not accelerating angularly or linearly. It's weight is exactly balanced by the collective thrust force it generates.

$$
\mathbf{x}_0 =
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}^\top, \quad
\mathbf{u}_0 =
\begin{bmatrix}
mg & 0 & 0 & 0
\end{bmatrix}^\top
$$

Casting the Equations of Motion into State-Space Form

I cast the equations of motion into state-space form. This is the first step of my linearization process and effectively reformulates the system of second-order differential equations into an equivalent system of first-order differential equations, referred to as state equations. The system of state equations can be represented by a single vector-valued function dependent on only the states and inputs.

$$
\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})
$$

The following are the explicitly stated state equations for the quadcopter.

$$
\begin{equation}
\begin{aligned}
\dot{x} &= \dot{x}, \
\dot{y} &= \dot{y}, \
\dot{z} &= \dot{z}, \
\dot{\phi} &= p + r \cos(\phi) \tan(\theta) + q \sin(\phi) \tan(\theta), \

$$
\begin{equation}
\begin{aligned}
\dot{\theta} &= q \cos(\phi) - r \sin(\phi), \\
\dot{\psi} &= \frac{r \cos(\phi)}{\cos(\theta)} + \frac{q \sin(\phi)}{\cos(\theta)}, \\
\ddot{x} &= \frac{F_c}{m} (\sin(\phi)\sin(\psi) + \cos(\phi)\cos(\psi)\sin(\theta)), \\
\ddot{y} &= -\frac{F_c}{m} (\cos(\psi)\sin(\phi) - \cos(\phi)\sin(\psi)\sin(\theta)), \\
\ddot{z} &= \frac{F_c \cos(\phi) \cos(\theta)}{m} - g, \\
\dot{p} &= \frac{M_x}{I_x} + \frac{q r (I_y - I_z)}{I_x}, \\
\dot{q} &= \frac{M_y}{I_y} - \frac{p r (I_x - I_z)}{I_y}, \\
\dot{r} &= \frac{M_z}{I_z} + \frac{p q (I_x - I_y)}{I_z}.
\end{aligned}
\end{equation}
$$

Applying the Linear Approximation at the Hover State

I apply the notion of first-order Taylor series linear approximation to linearize the dynamics about the hover state. This process involves computing the Jacobian matrix of the state equations with respect to the states, computing the Jacobian matrix with respect to the input variables, evaluating both Jacobian matrices at the hover state, and combining them with the state and input vectors. The resulting linear model is valid for states near the hover state.

$$
\dot{\mathbf{x}} =
\left. \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right|_{\mathbf{x}_0, \mathbf{u}*0} *mathbf{x}* +
*left. *frac{\partial *mathbf{f}}{\partial *mathbf{u}} *right*|{\mathbf{x}_0, \mathbf{u}_0} \mathbf{u} =
\mathbf{A} \mathbf{x} + \mathbf{B} \mathbf{u}.
$$