Quadcopter Simulation Reference Manual

January 24, 2020

Introduction

Advancements in modern semiconductor devices have made quadcopters, helicopters with four independent motors, inexpensive, lightweight and controllable. In quadcopter control, the six degrees of freedom: three translational and three rotational, are coupled and controlled by four independent inputs (motor speeds). The dynamics of flight tend to be complex and nonlinear due to complicated aerodynamic effects, lack of friction for stopping and stability in air, so damping has to be actively produced by the control system. In this sequence of homework assignments, we will model these dynamics, provide means for simulation and apply control signals to stabilize the quadcopter.

Kinematics

To simplify the kinematic model of dynamics of flight it is common to have two reference frames: body and inertial frames. The body frame is attached to the rigid body of the quadcopter, whereas inertial frame is attached to ground. Over time these reference frames will be moving with respect to another (see Figure 1) and our simulation will track that progress.

Lets start by introducing some of the state variables that will be used to track the quadopter. For example, the position and velocity of the quadcopter in the inertial frame are denoted, respectively, as $x = (x, y, z)^T$ and $\dot{x} = (\dot{x}, \dot{y}, \dot{z})^T$. Similarly, the roll, pitch, and yaw angles in the inertial frame are denote as $\Theta = (\phi, \theta, \psi)^T$ (see Figure 3), and angular velocities $\dot{\Theta} = (\dot{\phi}, \dot{\theta}, \dot{\psi})^T$.

The position of the quadcopter is defined as a series of transformations/rotations with Euler angles whose order is important. In figure 2, the initial position is marked in black and the final position is marked in cyan. The rotations start with ψ, followed by θ and ϕ.

Transformations

At each successive step of our simulation, quadcopter's state variables track its position, velocity, acceleration, orientation, etc. Over time these variables will change and will get updated accordingly. In some situations computation is easier to be done in one frame than anotherm thus, to transition between them

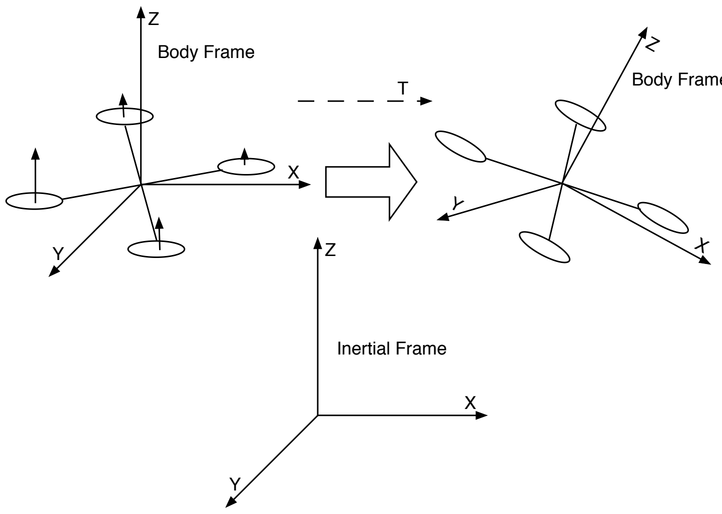

Figure 1: Body and Inertial Reference Frames

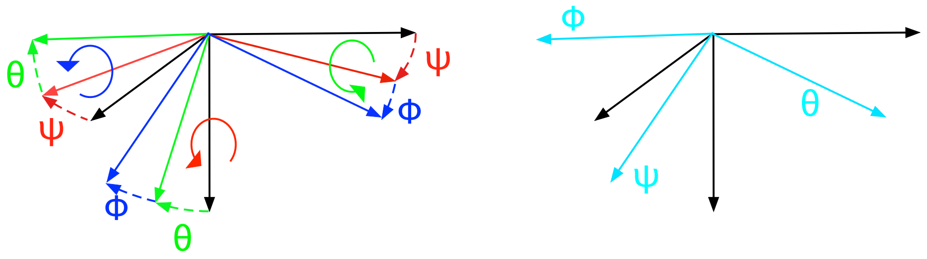

Figure 2: Definition of Θ

we will use transformations. To introduce the transformations in the 3D space let's cover them in 2D space first. We can simplify transformations into 3 types:

- Translate

$$\begin{Vmatrix} x_2 \\ y_2 \end{Vmatrix} = \begin{Vmatrix} x_1 \\ y_1 \end{Vmatrix} + \begin{Vmatrix} t_x \\ t_y \end{Vmatrix} \tag{1}$$

- Scale

$$\begin{Vmatrix} x_2 \\ y_2 \end{Vmatrix} = \begin{Vmatrix} s_x x_1 \\ s_y y_1 \end{Vmatrix} = \begin{Vmatrix} s_x & 0 \\ 0 & s_y \end{Vmatrix} \begin{Vmatrix} x_1 \\ y_1 \end{Vmatrix} \tag{2}$$

- Rotate

$$\begin{Vmatrix} s_x x_1 \\ s_y y_1 \end{Vmatrix} = \begin{Vmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{Vmatrix} \begin{Vmatrix} x_1 \\ y_1 \end{Vmatrix} \tag{3}$$

There are also other transformation such as reflections, shear, etc. However, we will mainly utilize rotational and translational transformations for simulation of the quadcopter, because certain phenomena are easier to compute in one

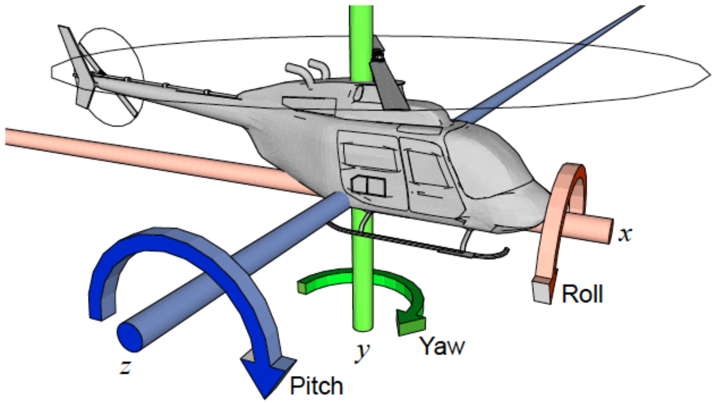

Figure 3: Roll, Pitch and Yaw

reference frame and not in another. For instance, it is easier to include all internal body physics computations in the body reference frame where the outside world has no influence but only thrusts generated by the quadcopter's motors do. The actuation of the motors is easier to observe in the body reference frame. This is because motors are fixed to the rigid frame, hence it is more convenient to transform their impact on the Center of Gravity (CoG) from body to the inertial reference frame than it is to consider the body physics in the inertial frame of reference, where we are only interested in the CoG's direction of motion and position. Thus, we will summarize these rotational transformations with 3 matrices $R(\psi), R(\phi)$ and $R(\theta)$ for each body frame rotation respectively. When computing motion of the CoG in the inertial reference frame, we will use the combined effort from each rotational plane. In contrast, we will use derivative transformation when taking into consideration the external forces, such as air drag resistance and gravitation, on the body.

$$R(\phi) = \begin{Vmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{Vmatrix} \tag{4}$$

$$R(\theta) = \begin{Vmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{Vmatrix} \tag{5}$$

$$R(\psi) = \begin{Vmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{Vmatrix} \tag{6}$$

Therefore, we can relate the body and inertial frame by a rotation matrix R, which allows transformation of a vector from the body frame to the inertial frame. This matrix is derived by successively undoing the yaw, pitch, and roll.

$$R = R(-\psi)R(-\theta)R(-\phi) \tag{7}$$

Thus, R can be expressed as: (Assignment 1)

$$R = \begin{Vmatrix} ---------- & ---------- & ---------- \\ ---------- & ---------- & ---------- \\ ---------- & ---------- & ---------- \end{Vmatrix} \tag{8}$$

$$R = \begin{Vmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{Vmatrix} \tag{9}$$

For a given vector $\vec{v}$ in the body frame, the corresponding vector is given by $R\vec{v}$ in the inertial frame.

On the other hand, the relationship between Euler(Inertial)-Angle rates and Body-Axis rates can be described with the following equation:

$$\omega = \begin{Vmatrix} \dot{\phi} \\ 0 \\ 0 \end{Vmatrix} + R(\phi) \begin{Vmatrix} 0 \\ \dot{\theta} \\ 0 \end{Vmatrix} + R(\phi)R(\theta) \begin{Vmatrix} 0 \\ 0 \\ \dot{\psi} \end{Vmatrix} \tag{10}$$

or

$$\omega = W\dot{\Theta} \tag{11}$$

if

$$\dot{\Theta} = \begin{Vmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Vmatrix} \tag{12}$$

then we can deduce that (Assignment 2)

$$W = \begin{Vmatrix} -\,-\,-\,-\,-\,-\,-\,-\,- & -\,-\,-\,-\,-\,-\,-\,-\,- & -\,-\,-\,-\,-\,-\,-\,-\,- \\ -\,-\,-\,-\,-\,-\,-\,-\,- & -\,-\,-\,-\,-\,-\,-\,-\,- & -\,-\,-\,-\,-\,-\,-\,-\,- \\ -\,-\,-\,-\,-\,-\,-\,-\,- & -\,-\,-\,-\,-\,-\,-\,-\,- & -\,-\,-\,-\,-\,-\,-\,-\,- \end{Vmatrix} \tag{13}$$

in other words:

$$\omega = \begin{Vmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{Vmatrix} \begin{Vmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Vmatrix} \tag{14}$$

where ω is the angular velocity vector in the body frame.

and

$$W = \begin{Vmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{Vmatrix} \tag{15}$$

Physics

We assume that all motors are identical and that diagonally connected motors have identical propellers attached to them. Two adjacent motors are spinning in opposite directions and their propellers are respectively made to generate vertical thrust. This enables the quadcopter in ideal situation to maintain a stable position. We will also assume that motors with propellers well be properly spaced and that battery can supply enough current generate the desired angular velocities.

Solving for the thrust magnitude T, we obtain that thrust is proportional to the square of angular velocity of the motor:

$$T = k\omega^2 \tag{16}$$

where k is thrust constant of the motors. Summing over all the motors, we find that the total thrust on the quadcopter (in the body frame) is given by

$$T_B = \sum_{i=1}^{4} T_i = k \begin{Vmatrix} 0 \\ 0 \\ \sum {\omega_i}^2 \end{Vmatrix} \tag{17}$$

We will also simplify the air drag or resistance by introducing the k_d the drag coefficient. Thus, we can also model drag force as:

$$F_D = \begin{Vmatrix} -k_d\dot{x} \\ -k_d\dot{y} \\ -k_d\dot{z} \end{Vmatrix} \tag{18}$$

Motor Torques

Now that we have computed the forces on the quadcopter, we would also like to compute the torques. Each rotor contributes some torque about the body z axis. This torque is the torque required to keep the propeller spinning and providing thrust; it creates the instantaneous angular acceleration and overcomes the frictional drag forces.

We can then write the complete torque about the z axis for a motor as:

$$\tau_z = k_b \omega^2 + I\dot{\omega} \tag{19}$$

where I is the moment of inertia about the motor z axis, $\dot{\omega}$ is the angular acceleration of the propeller, and k_b is the drag coefficient of the propeller. Note that in steady state flight (i.e. not takeoff or landing), $\dot{\omega} \approx 0$, since most of the time the propellers will be maintaining a constant (or almost constant) thrust and won't be accelerating. Thus, we ignore this term, simplifying the entire expression to

$$\tau_z = k_b {\omega_i}^2 \tag{20}$$

However, the propeller's spinning direction will make this term either positive or negative, clockwise vs. counterclockwise. Thus, the total torque about the z axis will be an alternating sum of torques for each propeller:

$$\tau_\psi = k_b \left({\omega_1}^2 - {\omega_2}^2 + {\omega_3}^2 - {\omega_4}^2 \right) \tag{21}$$

The roll and pitch torques are derived from standard mechanics. We can arbitrarily choose the $i = 1$ and $i = 3$ motors to be on the roll axis, so

$$\tau_\phi = L(k{\omega_1}^2 - k{\omega_3}^2) = Lk({\omega_1}^2 - {\omega_3}^2) \tag{22}$$

Similarly, the pitch torque is given by a similar expression

$$\tau_\theta = Lk({\omega_2}^2 - {\omega_4}^2) \tag{23}$$

where L is the distance from the center of the quadcopter to any of the propellers. All together, we find that the torque vector can be expressed as:

$$\tau_B = \begin{Vmatrix} Lk({\omega_1}^2 - {\omega_3}^2) \\ Lk({\omega_2}^2 - {\omega_4}^2) \\ k_b \left({\omega_1}^2 - {\omega_2}^2 + {\omega_3}^2 - {\omega_4}^2 \right) \end{Vmatrix} \tag{24}$$

Equations of Motion

In the inertial frame, the acceleration of the quadcopter is due to thrust, gravity, and linear friction. We can obtain the thrust vector in the inertial frame by using our rotation matrix R to map the thrust vector from the body frame to the inertial frame. Thus, the linear motion can be summarized as:

$$T + F_D - F_G = m\ddot{x} \tag{25}$$

where $\ddot{x}$ is the acceleration vector, F_D is the drag force, T is the thrust vector in the inertial frame, and F_G $((0, 0, mg)^T)$ is the gravitational Force vector. Note that we can calculate T, using R transformation from the thrust computed in the body frame.

$$T = RT_B \tag{26}$$

We can also deduce the acceleration of center of gravity:

$$\ddot{x} = \frac{1}{m}(RT_B + F_D - F_G) \tag{27}$$

While it is convenient to have the linear equations of motion in the inertial frame, the rotational equations of motion are useful to us in the body frame, so that we can express rotations about the center of the quadcopter instead of about our inertial center. We derive the rotational equations of motion from Euler's equations for rigid body dynamics. Expressed in vector form, Euler's equations are written as

$$I\dot{\omega} + \omega \times (I\omega) = \tau \tag{28}$$

where ω is the angular velocity vector, I is the inertia matrix, τ is a vector of external torques, and $\times$ is the cross product between two vectors. We can rewrite this as:

$$\dot{\omega} = I^{-1}\left(\tau - \omega \times (I\omega)\right) \tag{29}$$

If we model our quadcopter as two thin uniform rods crossed at the origin with a point mass (motor) at the end of each, then we can deduce that

$$I = \begin{Vmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{Vmatrix} \tag{30}$$

Therefore, we obtain our final result for the body frame rotational equations of motion

$$\dot{\omega} = \begin{Vmatrix} (\tau_\phi - \omega_y\omega_z(I_{yy} - I_{zz}))I_{xx}{}^{-1} \\ (\tau_\theta - \omega_x\omega_z(I_{zz} - I_{xx}))I_{yy}{}^{-1} \\ (\tau_\psi - \omega_x\omega_y(I_{xx} - I_{yy}))I_{zz}{}^{-1} \end{Vmatrix} \tag{31}$$

Controller Design

The purpose of deriving a mathematical model of a quadcopter is to assist in developing controllers for physical quadcopters. The inputs to our system consist of the angular velocities of each rotor, since all we can control is the voltages across the motors. Note that in our simplified model, we only use the square of the angular velocities, ${\omega_i}^2$, and never the angular velocity itself, ω_i. For notational simplicity, let us introduce the inputs:

$$in_i = {\omega_i}^2 \tag{32}$$

As we can set ω_i, we can clearly set in_i as well. With this, we can write our system as a first order differential equation in state space. Let x_1 be the position in space of the quadcopter, x_2 be the quadcopter linear velocity, x_3 be the roll, pitch, and yaw angles, and x_4 be the angular velocity vector. (Note that all of these are 3-vectors.) With these being our state, we can write the state space equations for the evolution of our state.

$$\dot{x_1} = x_2 \tag{33}$$

$$\dot{x_2} = \left\|\begin{matrix} 0 \\ 0 \\ -g \end{matrix}\right\| + \frac{1}{m} R T_B + \frac{1}{m} F_D \tag{34}$$

$$\dot{x_3} = \left\|\begin{matrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{matrix}\right\|^{-1} x_4 \tag{35}$$

$$\dot{x_4} = \left\|\begin{matrix} (\tau_\phi - \omega_y \omega_z (I_{yy} - I_{zz})) {I_{xx}}^{-1} \\ (\tau_\theta - \omega_x \omega_z (I_{zz} - I_{xx})) {I_{yy}}^{-1} \\ (\tau_\psi - \omega_x \omega_y (I_{xx} - I_{yy})) {I_{zz}}^{-1} \end{matrix}\right\| \tag{36}$$

Note that our inputs are not used in these equations directly. However, as we will see, we can choose values for τ and T, and then solve for values of in_i.

Control Mechanism

In order to control the quadcopter, we will use a PID control, with a component proportional to the error between our desired trajectory and the observed trajectory, a component proportional to the integral of the error and a component proportional to the derivative of the error. Our quadcopter will only have a gyro, so we will only be able to use the angle derivatives $\dot{\phi}$, $\dot{\theta}$, and $\dot{\psi}$ in our controller; these measured values will give us the derivative of our error, and their integral will provide us with the actual error. We would like to stabilize the quadcopter in a horizontal position, so our desired velocities and angles will all be zero. Torques are related to our angular velocities by

$$\tau = I\ddot{\theta} \tag{37}$$

so we would like to set the torques proportional to the output of our controller, with

$$\tau = I u(t) \tag{38}$$

Putting these together we get that the torques and inputs have the following relationship:

$$\tau_B = \left\|\begin{matrix} Lk(in_1 - in_3) \\ Lk(in_2 - in_4) \\ k_b\,(in_1 - in_2 + in_3 - in_4) \end{matrix}\right\| = \left\|\begin{matrix} -I_{xx} e_\phi \\ -I_{yy} e_\theta \\ -I_{zz} e_\psi \end{matrix}\right\| \tag{39}$$

This gives us a set of three equations with four unknowns. We can constrain this by enforcing that our inputs (individual motor thrusts) keep the quadcopter aloft if their z component in the inertial frame is equal to weight of the quadcopter:

$$T_z = mg \tag{40}$$

If we enforce the magnitude of total thrust vector to be

$$T_B = \frac{mg}{\cos\theta\cos\phi} \tag{41}$$

then the z axis component of the thrust in the inertial frame will be equal to mg.

We know that the thrust is proportional to a weighted sum of the inputs:

$$T_B = \frac{mg}{\cos\theta\cos\phi} = k\sum in_i \implies \sum in_i = \frac{mg}{k\cos\theta\cos\phi} \tag{42}$$

With this extra constraint, we have a set of four linear equations with four unknowns in_i. We can then solve for each in_i, and obtain the following input values:

$$in_1 = \frac{mg}{4k\cos\theta\cos\phi} - \frac{e_\phi I_{xx}}{2kL} - \frac{e_\psi I_{zz}}{4k_b} \tag{43}$$

$$in_2 = \frac{mg}{4k\cos\theta\cos\phi} + \frac{e_\psi I_{zz}}{4k_b} - \frac{e_\theta I_{yy}}{2kL} \tag{44}$$

$$in_3 = \frac{mg}{4k\cos\theta\cos\phi} + \frac{e_\phi I_{xx}}{2kL} - \frac{e_\psi I_{zz}}{4k_b} \tag{45}$$

$$in_4 = \frac{mg}{4k\cos\theta\cos\phi} + \frac{e_\psi I_{zz}}{4k_b} + \frac{e_\theta I_{yy}}{2kL} \tag{46}$$

This is a complete specification for our controller. We can simulate this controller using our simulation environment. The controller drives the angular velocities and angles to zero. However, note that the angles are not completely driven to zero. This is even more so common problem with using PD (proportional - derivative) controllers for mechanical systems, and can be partially alleviated with a PID (proportional - integral - derivative) controller.

We are only controlling angular velocities, that's why the position and linear velocities do not converge to zero and are not consistent between the simulations. However, the z position will remain constant as we constrained the total vertical thrust to match the gravity. In this simulation, we restricted ourselves to maintaining quadcopter horizontal, however, we could potentially aim for another angle and this would change one element quiet significantly, and that our velocity and position. In fact, that's exactly human operators control the position of the quadcopter. It is also important to note that while we may compute the linear velocities and positions from the angular velocities, in practice the values can be very noisy. Thus, stabilizing the quadcopter angle via measuring angular velocity is a great beginning.

The equations for the error are:

$$e_\phi = K_p\phi + K_i\int_0^T \phi dt + K_d\dot{\phi} \tag{47}$$

$$e_\theta = K_p\theta + K_i\int_0^T \theta dt + K_d\dot{\theta} \tag{48}$$

$$e_\psi = K_p\psi + K_i\int_0^T \psi dt + K_d\dot{\psi} \tag{49}$$

or

$$e_\phi = K_p\int_0^T \dot{\phi} dt + K_i\int_0^T\int_0^T \dot{\phi} dtdt + K_d\dot{\phi} \tag{50}$$

$$e_\theta = K_p\int_0^T \dot{\theta} dt + K_i\int_0^T\int_0^T \dot{\theta} dtdt + K_d\dot{\theta} \tag{51}$$

$$e_\psi = K_p\int_0^T \dot{\psi} dt + K_i\int_0^T\int_0^T \dot{\psi} dtdt + K_d\dot{\psi} \tag{52}$$

However, PID controls come with their own shortcomings. One problem that commonly occurs with a PID control is known as integral windup. Large disturbances get integrated over time and cause large control signals, which do not go away unless system is driven to oscillation. In order to avoid this, we can reset the integral component.

Simulation (Assignment 3 - Part 2)

Now that we have complete equations of motion describing the dynamics of the system, we can complete the simulation environment to test and view the results of randomized inputs and controller parameters. The following list of steps describe the sequential steps to be performed in order to obtain the desired simulation system.

- Setup - setup state variables and simulation environment. (Assignment 1 - Part 1)
- Apply Control - See Section Control Mechanism (Assignment 2 - Part 1)
- Update Acceleration - Use the thrust from control to compute $\ddot{x}$ (Assignment 2 - Part 2)
- Update Ω - Use the knowledge of $\dot{\Theta}$ to derive Ω (Assignment 2 - Part 2)
- Update $\dot{\Omega}$ - See Section Control Design (Assignment 2 - Part 2)
- Advance - advance the simulation by incrementally integrating state parameters such as x and Θ, respectively, with a and $\dot{\Theta}$ over the dt step. (Assignment 3 - Part 1)
- Paint - will be updating the animation using the new orientation Θ and position x variables. (Given in the Matlab file.)