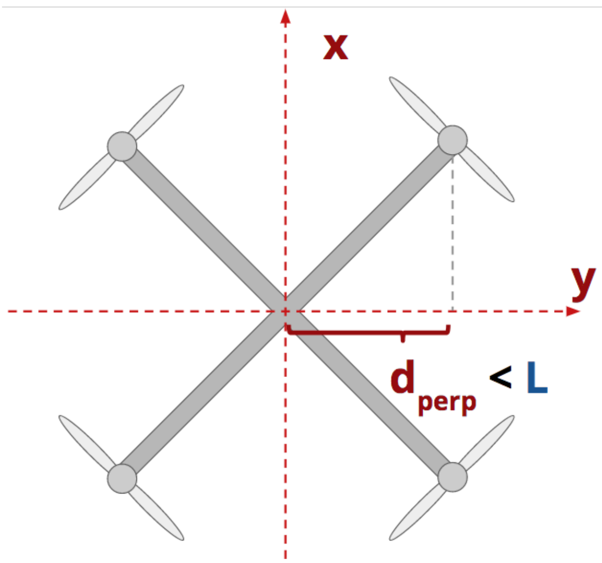
FCND Controller Notes

Boyan Zhang

November 14, 2018

1 Controller



For convention used in simulator and across the course, axis z is pointing into the page defined with right-hand rule.

1.1 Setting the propeller angular velocities (Drone)

Solve below equations for obtaining desired thrust for each propeller

$$F_1 - F_2 - F_3 + F_4 = \frac{M_x}{l} \tag{1}$$

$$F_1 + F_2 - F_3 - F_4 = \frac{M_y}{l} \tag{2}$$

$$F_1 - F_2 + F_3 - F_4 = -\frac{M_z}{\kappa} \tag{3}$$

$$F_1 + F_2 + F_3 + F_4 = F_{total} \tag{4}$$

where κ is drag thrust ratio. $l = \frac{L}{\sqrt{2}}$ is equivalent length of leverage

1.2 Body rate control

Implement P controller for deciding desired *moment* along each axis in previous step.

1.3 Roll pitch control

First order system, P controller.

$$\begin{bmatrix} p_c \\ q_c \end{bmatrix} = \frac{1}{R_{33}} \begin{bmatrix} R_{21} & -R_{11} \\ R_{22} & -R_{12} \end{bmatrix} \begin{bmatrix} \dot{b}_c^x \\ \dot{b}_c^y \end{bmatrix} \tag{5}$$

The velocities (p_c, q_c) using the estimated attitude R

Velocity yaw control can be considered separately, since rotations around the body z-axis do not affect the above dynamics

For this project, estimation of R is provided.

$$\dot{b}_c^x = k_p(b_c^x - b_a^x) \tag{6}$$

$$\dot{b}_c^y = k_p(b_c^y - b_a^y) \tag{7}$$

where $b_a^x = R_{13}$ and $b_a^y = R_{23}$. Commanded roll pitch angle (tilt angle) b_a^x and b_a^y can be obtained from

$$b_a^x = \frac{a_c^x}{a_c} \tag{8}$$

$$b_a^y = \frac{a_c^y}{a_c} \tag{9}$$

where $a_c = \frac{c_{total}}{m_{total}}$, c_{total} represents collective thrust

Hence, the given values can be converted into the angular velocities into the body frame by the next matrix multiplication.

$$\begin{pmatrix} p_c \\ q_c \end{pmatrix} = \frac{1}{R_{33}} \begin{pmatrix} R_{21} & -R_{11} \\ R_{22} & -R_{12} \end{pmatrix} \times \begin{pmatrix} \dot{b}_c^x \\ \dot{b}_c^y \end{pmatrix} \tag{10}$$

1.4 Altitude Control

Second order system, PID controller. Linear acceleration can be expressed by the next linear equation (Translational motion of a quadrocopter in the inertial frame)

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} + R \begin{pmatrix} 0 \\ 0 \\ c \end{pmatrix} \tag{11}$$

where $R = R(\psi) \times R(\theta) \times R(\phi)$.

The individual linear acceleration has the form of

$$\ddot{x} = cb^x \tag{12}$$

$$\ddot{y} = cb^y \tag{13}$$

$$\ddot{z} = cb^z + g \tag{14}$$

where $b^x = R_{13}$, $b^y = R_{23}$ and $b^z = R_{33}$ are the elements of the last column of the rotation matrix.

We are controlling the vertical acceleration:

$$\bar{u}_1 = \ddot{z} = cb^z + g \tag{15}$$

Therefore

$$c = (\bar{u}_1 - g)/b^z \tag{16}$$

A PID controller used for the altitude results in:

$$\bar{u}_1 = k_{p-z}(z_t - z_a) + k_{d-z}(\dot{z}_t - \dot{z}_a) + k_{i-z} \int_0^t e(t^‘)dt^‘ + \ddot{z}_t \tag{17}$$

1.5 Lateral Control

Second order system, PD controller

$$\ddot{x}_{\text{command}} = cb_c^x \tag{18}$$
$$\ddot{x}_{\text{command}} = k_p^x(x_t - x_a) + k_d^x(\dot{x}_t - \dot{x}_a) + \ddot{x}_t \tag{19}$$
$$b_c^x = \ddot{x}_{\text{command}}/c \tag{20}$$

1.6 Yaw Control

First order system. P controller. Control over yaw is decoupled from the other directions. A P controller is used to control the drone's yaw.

$$r_c = k_p(\psi_t - \psi_a) \tag{21}$$

Proper radius normalization is required