RBE 502 Final Project

Sayan Das, Haoying Zhou

Date: May 2, 2022

RBE 502 — Robot Control

Instructor: Siavash Farzan

Spring 2022

Final Project:

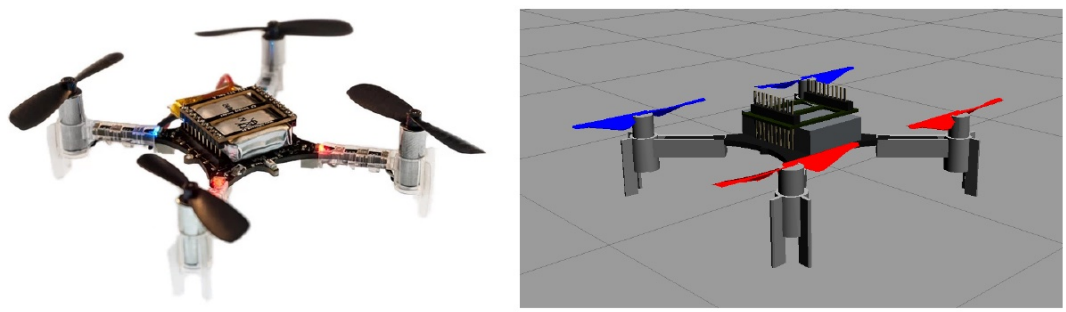
Robust Trajectory Tracking for Quadrotor UAVs using Sliding Mode Control

1.1 Overview

The objective of this project is to develop a robust control scheme to enable a quadrotor to track desired trajectories in the presence of external disturbances.

The control design under study will be tested on the Crazyflie 2.0 platform. Crazyflie is a quadrotor that is classified as a micro air vehicle (MAV), as it only weighs 27 grams and can fit in your hand. The size makes it ideal for flying inside a lab without trashing half the interior. Even though the propellers spin at high RPMs, they are soft and the torque in the motors is very low when compared to a brushless motor, making it relatively crash tolerant. The Crazyflie 2.0 features four 7mm coreless DC-motors that give the Crazyflie a maximum takeoff weight of 42g.

The Crazyflie 2.0 is an open source project, with source code and hardware design both documented and available. For more information, please see the link below:
`https://www.bitcraze.io/products/old-products/crazyflie-2-0/`



(a) (b)

Figure 1: Crazyflie 2.0 Quadrotor (a) hardware, (b) Gazebo physics based simulation

1.2 Crazyflie 2.0 Setup in Gazebo

For this project, we assume that you have already installed Ubuntu 20.04 and ROS Noetic by following the instructions provided in Programming Assignment 0.

To set up the Crazyflie 2.0 quadrotor in Gazebo, we need to install additional ROS dependencies for building packages as below:

```
sudo apt update
sudo apt install ros-noetic-joy ros-noetic-octomap-ros ros-noetic-mavlink
sudo apt install ros-noetic-octomap-mapping ros-noetic-control-toolbox
sudo apt install python3-vcstool python3-catkin-tools protobuf-compiler
    libgoogle-glog-dev
rosdep update
sudo apt-get install ros-noetic-ros libgoogle-glog-dev
```

We are now ready to create a new ROS workspace and download the ROS packages for the robot:

```
mkdir -p ~/rbe502_project/src
cd ~/rbe502_project/src
catkin_init_workspace  # initialize your catkin workspace
cd ~/rbe502_project
catkin init
cd ~/rbe502_project/src
git clone -b dev/ros-noetic https://github.com/gsilano/CrazyS.git
git clone -b med18_gazebo9 https://github.com/gsilano/mav_comm.git
```

Note: a new ROS workspace is needed for the project, because the `CrazyS` Gazebo package is built using the `catkin build` tool, instead of `catkin_make`.

We need to build the project workspace using `python_catkin_tools`, therefore we need to configure it:

```
cd ~/rbe502_project
rosdep install --from-paths src -i
rosdep update
catkin config --cmake-args -DCMAKE_BUILD_TYPE=Release -DCATKIN_ENABLE_TESTING=
    False
catkin build
```

Do not forget to add sourcing to your `.bashrc` file:

```
echo "source ~/rbe502_project/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

With all dependencies ready, we can build the ROS package by the following commands:

```
cd ~/rbe502_project
catkin build
```

To spawn the quadrotor in Gazebo, we can run the following launch file:

```
roslaunch rotors_gazebo crazyflie2_without_controller.launch
```

1.3 Dynamic Model

The quadrotor model is shown in Figure 2. Considering two coordinate frames — specifically the world coordinate frame O_W and the body coordinate frame O_B – the generalized coordinate for a quadrotor model are defined as:

$$q = \begin{bmatrix} x & y & z & \phi & \theta & \psi \end{bmatrix}^T$$

with the translational coordinates x, y, z with respect to the world frame, and the roll ϕ, pitch θ and yaw ψ angles with respect to the body frame.

The control inputs on the system can be considered simply as

$$u = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 \end{bmatrix}$$

where u_1 is the force from all the propellers, and u_2, u_3, and u_4 are the moments applied about the body frame axes by the propellers.

For a set of desired control inputs, the desired rotor speeds (i.e. ω_i for $i = 1, 2, 3, 4$) are obtained by using the "allocation matrix":

$$\begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4k_F} & -\frac{\sqrt{2}}{4k_F l} & -\frac{\sqrt{2}}{4k_F l} & -\frac{1}{4k_M k_F} \\ \frac{1}{4k_F} & -\frac{\sqrt{2}}{4k_F l} & \frac{\sqrt{2}}{4k_F l} & \frac{1}{4k_M k_F} \\ \frac{1}{4k_F} & \frac{\sqrt{2}}{4k_F l} & \frac{\sqrt{2}}{4k_F l} & -\frac{1}{4k_M k_F} \\ \frac{1}{4k_F} & \frac{\sqrt{2}}{4k_F l} & -\frac{\sqrt{2}}{4k_F l} & \frac{1}{4k_M k_F} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}$$

where k_F and k_M denote the propeller thrust factor and moment factor, respectively.

Considering the generalized coordinates and the control inputs defined above, the simplified equations of motion (assuming small angles) for the translational accelerations and body frame angular accelerations are derived as:

$$\begin{aligned}
\ddot{x} &= \frac{1}{m}\big(\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi\big) u_1 \\
\ddot{y} &= \frac{1}{m}\big(\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi\big) u_1 \\
\ddot{z} &= \frac{1}{m}\big(\cos\phi \cos\theta\big)\, u_1 - g \\
\ddot{\phi} &= \dot{\theta}\,\dot{\psi}\,\frac{I_y - I_z}{I_x} - \frac{I_p}{I_x}\,\Omega\,\dot{\theta} + \frac{1}{I_x}\,u_2 \\
\ddot{\theta} &= \dot{\phi}\,\dot{\psi}\,\frac{I_z - I_x}{I_y} + \frac{I_p}{I_y}\,\Omega\,\dot{\phi} + \frac{1}{I_y}\,u_3 \\
\ddot{\psi} &= \dot{\phi}\,\dot{\theta}\,\frac{I_x - I_y}{I_z} + \frac{1}{I_z}\,u_4
\end{aligned}$$

where m is the quadrotor mass, l denotes the quadrotor arm length (i.e. the distance from the axis of rotation of the rotors to the center of the quadrotor), g is the gravitational acceleration, I_p is the propeller moment of inertia, and I_x, I_y, I_z indicate the quadrotor moment of inertia along the x, y and z axes, respectively. Moreover, the term Ω is expressed as: $\Omega = \omega_1 - \omega_2 + \omega_3 - \omega_4$.

The physical parameters for the Crazyflie 2.0 hardware are listed in Table 1.

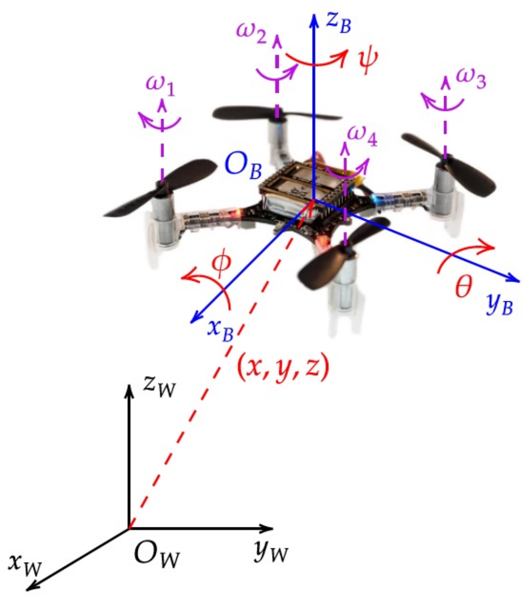


Figure 2: Crazyflie in the body (O_B) and the world (O_W) coordinate frames. The angular velocity ω_i of each rotor/propeller and the generalized coordinates x, y z ϕ, θ, ψ are depicted.

Table 1: Physical parameters of the Crazyflie 2.0 hardware platform.

Parameter	*Symbol*	*Value*
Quadrotor mass	m	27 g
Quadrotor arm length	l	46 mm
Quadrotor inertia along x-axis	I_x	16.571710×10^{-6} kg $\cdot$ m^2
Quadrotor inertia along y-axis	I_y	16.571710×10^{-6} kg $\cdot$ m^2
Quadrotor inertia along z-axis	I_z	29.261652×10^{-6} kg $\cdot$ m^2
Propeller moment of inertia	I_p	12.65625×10^{-6} kg $\cdot$ m^2
Propeller thrust factor	k_F	1.28192×10^{-8} N $\cdot$ s^2
Propeller moment factor	k_M	5.964552×10^{-3} m
Rotor maximum speed	ω_{max}	2618 rad/s
Rotor minimum speed	ω_{min}	0 rad/s

Remark 1: As shown in the equations of motion above, the quadrotor system has six D0F, with only four control inputs. As a result, the control of quadrotors is typically done by controlling only the altitude z and the roll-pitch-yaw angles ϕ, θ and ψ.

1.4 Problem Statement

Design a sliding mode controller for *altitude* and *attitude* control of the Crazyflie 2.0 to enable the quadrotor to track desired trajectories and visit a set of desired waypoints.

The main components of the project are described below.

1 Problem Part 1

Part 1. Write a MATLAB or Python script to generate quintic (fifth-order) trajectories (position, velocity and acceleration) for the translational coordinates (x, y, z) of Crazyflie. The quadrotor is supposed to start from the origin $p_0 = (0, 0, 0)$ and visit five waypoints in sequence. The waypoints to visit are

- $p_0 = (0, 0, 0)$ to $p_1 = (0, 0, 1)$ in 5 seconds
- $p_1 = (0, 0, 1)$ to $p_2 = (1, 0, 1)$ in 15 seconds
- $p_2 = (1, 0, 1)$ to $p_3 = (1, 1, 1)$ in 15 seconds
- $p_3 = (1, 1, 1)$ to $p_4 = (0, 1, 1)$ in 15 seconds
- $p_4 = (0, 1, 1)$ to $p_5 = (0, 0, 1)$ in 15 seconds

The sequence of visiting the waypoints does matter.

Include a plot of the desired trajectories in your final report.

Generally speaking, to generate a quintic trajectory, we need to solve Equation 1. Given start time t_0, end time t_f, initial position q_0, initial velocity $\dot{q}_0$, initial acceleration $\ddot{q}_0$, goal position q_f, goal velocity $\dot{q}_f$ and goal acceleration $\ddot{q}_f$, we can obtain that:

$$\begin{bmatrix} 1 & t_0 & {t_0}^2 & {t_0}^3 & {t_0}^4 & {t_0}^5 \\ 0 & 1 & 2t_0 & 3{t_0}^2 & 4{t_0}^3 & 5{t_0}^4 \\ 0 & 0 & 2 & 6t_0 & 12{t_0}^2 & 20{t_0}^3 \\ 1 & t_f & {t_f}^2 & {t_f}^3 & {t_f}^4 & {t_f}^5 \\ 0 & 1 & 2t_f & 3{t_f}^2 & 4{t_f}^3 & 5{t_f}^4 \\ 0 & 0 & 2 & 6t_f & 12{t_f}^2 & 20{t_f}^3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} q_0 \\ \dot{q}_0 \\ \ddot{q}_0 \\ q_f \\ \dot{q}_f \\ \ddot{q}_f \end{bmatrix} \tag{1}$$

Substitute the given time and positions into Equation 1, set the velocities and accelerations to be zero, we can solve the desired trajectory:

$$x_d = \begin{cases} 0 & 0 \le t \le 5 \\ 0.00296(t-5)^3 - 0.000296(t-5)^4 + 0.0000079(t-5)^5 & 5 < t \le 20 \\ 1 & 20 < t \le 35 \\ 1 - 0.00296(t-35)^3 + 0.000296(t-35)^4 - 0.0000079(t-35)^5 & 35 < t \le 50 \\ 0 & 50 < t \le 65 \end{cases}$$

$$y_d = \begin{cases} 0 & 0 \le t \le 5 \\ 0 & 5 < t \le 20 \\ 0.00296(t-20)^3 - 0.000296(t-20)^4 + 0.0000079(t-20)^5 & 20 < t \le 35 \\ 1 & 35 < t \le 50 \\ 1 - 0.00296(t-50)^3 + 0.000296(t-50)^4 - 0.0000079(t-50)^5 & 50 < t \le 65 \end{cases}$$

$$z_d = \begin{cases} 0.08t^3 - 0.024t^4 + 0.00192t^5 & 0 \le t \le 5 \\ 1 & 5 < t \le 20 \\ 1 & 20 < t \le 35 \\ 1 & 35 < t \le 50 \\ 1 & 50 < t \le 65 \end{cases}$$

Correspondingly, we can solve the velocities and accelerations by taking the time derivatives

$$\dot{x}_d = \begin{cases} 0 & 0 \leq t \leq 5 \\ 0.00888(t-5)^2 - 0.001184(t-5)^3 + 0.0000395(t-5)^4 & 5 < t \leq 20 \\ 0 & 20 < t \leq 35 \\ -0.00888(t-35)^2 + 0.001184(t-35)^3 - 0.0000395(t-35)^4 & 35 < t \leq 50 \\ 0 & 50 < t \leq 65 \end{cases}$$

$$\dot{y}_d = \begin{cases} 0 & 0 \leq t \leq 5 \\ 0 & 5 < t \leq 20 \\ 0.00888(t-20)^2 - 0.001184(t-20)^3 + 0.0000395(t-20)^4 & 20 < t \leq 35 \\ 0 & 35 < t \leq 50 \\ -0.00888(t-50)^2 + 0.001184(t-50)^3 - 0.0000395(t-50)^4 & 50 < t \leq 65 \end{cases}$$

$$\dot{z}_d = \begin{cases} 0.24t^2 - 0.096t^3 + 0.0096t^4 & 0 \leq t \leq 5 \\ 0 & 5 < t \leq 20 \\ 0 & 20 < t \leq 35 \\ 0 & 35 < t \leq 50 \\ 0 & 50 < t \leq 65 \end{cases}$$

$$\ddot{x}_d = \begin{cases} 0 & 0 \leq t \leq 5 \\ 0.01776(t-5) - 0.0035525(t-5)^2 + 0.000158(t-5)^3 & 5 < t \leq 20 \\ 0 & 20 < t \leq 35 \\ -0.01776(t-35) + 0.0035525(t-35)^2 - 0.000158(t-35)^3 & 35 < t \leq 50 \\ 0 & 50 < t \leq 65 \end{cases}$$

$$\ddot{y}_d = \begin{cases} 0 & 0 \leq t \leq 5 \\ 0 & 5 < t \leq 20 \\ 19.358 - 3.042t + 0.174t^2 - 0.004t^3 & 20 < t \leq 35 \\ 0 & 35 < t \leq 50 \\ -417.284 + 29.53t - 0.78t^2 + 0.008t^3 & 50 < t \leq 65 \end{cases}$$

$$\ddot{z}_d = \begin{cases} 0.48t - 0.288t^2 + 0.0384t^3 & 0 \leq t \leq 5 \\ 0 & 5 < t \leq 20 \\ 0 & 20 < t \leq 35 \\ 0 & 35 < t \leq 50 \\ 0 & 50 < t \leq 65 \end{cases}$$

2 Problem Part 2

Part 2. Considering the equations of motion provided in Section 1.3, design boundary layer-based sliding mode control laws for the z, ϕ, θ, ψ coordinates of the quadrotor to track desired trajectories z_d, ϕ_d, θ_d, and ψ_d. Include the control formulations (as symbolic expressions) in your final report.

Remark 2: To convert the desired position trajectories (x_d, y_d, z_d) to desired roll and pitch angles (ϕ_d, θ_d), the desired forces in x and y direction can be calculated using PD control (according to Eq. (1) and (2)), and the resulting desired forces can be then converted to desired ϕ and θ according to Eq. (3) and Eq. (4):

$$F_x = m\left(-k_p(x - x_d) - k_d(\dot{x} - \dot{x}_d) + \ddot{x}_d\right), \tag{1}$$

$$F_y = m\left(-k_p(y - y_d) - k_d(\dot{y} - \dot{y}_d) + \ddot{y}_d\right), \tag{2}$$

$$\theta_d = \sin^{-1}\left(\frac{F_x}{u_1}\right) \tag{3}$$

$$\phi_d = \sin^{-1}\left(\frac{-F_y}{u_1}\right) \tag{4}$$

Remark 3: For the purpose of this assignment, the desired yaw angle ψ, and also the desired angular velocities $\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$ and the desired angular accelerations $\ddot{\phi}$, $\ddot{\theta}$, $\ddot{\psi}$ can be considered zero during the motion, i.e:

$$\psi_d = 0 \qquad \text{and} \qquad \dot{\phi}_d = \dot{\theta}_d = \dot{\psi}_d = 0 \qquad \text{and} \qquad \ddot{\phi}_d = \ddot{\theta}_d = \ddot{\psi}_d = 0$$

The resulting discrepancy can be considered an external disturbance that is handled through the robust control design in this assignment.

Remark 4: When designing the sliding mode control laws, assume that all the model parameters are known. In fact, the objective of this assignment is to design a sliding mode controller to be robust under reasonable external disturbances, as discussed in class.

The sign function is defined as following:

$$sgn(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases} \tag{2}$$

The saturation function is defined as:

$$sat(x, BW) = \begin{cases} 1 & \frac{x}{BW} > 1 \\ \frac{x}{BW} & -1 \leq x \leq 1 \\ -1 & \frac{x}{BW} < -1 \end{cases} \tag{3}$$

where BW is band width of the saturation.

The gains of PD controller for calculating ϕ_d and θ_d from x and y are:

$$K_p = 33$$
$$K_d = 9$$

2.1 Control Law for z

Firstly, rewrite the dynamic equation for z:

$$\begin{aligned} \ddot{z} &= \frac{1}{m}(\cos\phi\cos\theta)u_1 - g \\ &= -g + g_z u_1 \end{aligned} \tag{4}$$

where

$$g_z = \frac{1}{m}(\cos\phi\cos\theta)$$

Define the sliding surface s_z as following:

$$\begin{aligned} s_z &= \dot{e}_z + \lambda_z e_z \\ \dot{e}_z &= \dot{z} - \dot{z}_d \\ e_z &= z - z_d \end{aligned} \tag{5}$$

Then, based on the above equations, we can derive the sliding condition equation as following:

$$\begin{aligned} s_z\dot{s}_z &\leq -k_z|s_z| \\ \dot{s}_z &\leq -k_z sgn(s_z) \\ \ddot{e}_z + \lambda_z\dot{e}_z &\leq -k_z sgn(s_z) \\ \ddot{z} - \ddot{z}_d + \lambda_z\dot{e}_z &\leq -k_z sgn(s_z) \\ -g + g_z u_1 - \ddot{z}_d + \lambda_z\dot{e}_z &\leq -k_z sgn(s_z) \\ u_1 &\leq \frac{-(-g - \ddot{z}_d + \lambda_z\dot{e}_z) - k_z sgn(s_z)}{g_z} \end{aligned} \tag{6}$$

To reduce the chattering effect, replace the sign function $sgn(\cdot)$ with saturation function $sat(\cdot)$, then we can write our the control law:

$$\begin{aligned} u_1 &= \frac{-(-g - \ddot{z}_d + \lambda_z\dot{e}_z) + u_{1R}}{g_z} \\ u_{1R} &= -k_z sat(s_z, b_z) \end{aligned} \tag{7}$$

where z_d and $\dot{z}_d$ can be calculated via Part 1 solution. s_z can be obtained from Equation 5, and

$$\begin{aligned} g_z &= \frac{1}{m}(\cos\phi\cos\theta) \\ \dot{e}_z &= \dot{z} - \dot{z}_d \\ k_z &= 50 \\ \lambda_z &= 10 \\ b_z &= 0.95 \end{aligned}$$

2.2 Control Law for ϕ

Firstly, rewrite the dynamic equation for ϕ:

$$
\begin{aligned}
\ddot{\phi} &= \dot{\theta}\dot{\psi}\frac{I_y - I_z}{I_x} - \frac{I_p}{I_x}\Omega\dot{\theta} + \frac{1}{I_x}u_2 \\
&= f_\phi + g_\phi u_2
\end{aligned} \tag{8}
$$

where

$$
\begin{aligned}
f_\phi &= \dot{\theta}\dot{\psi}\frac{I_y - I_z}{I_x} - \frac{I_p}{I_x}\Omega\dot{\theta} \\
g_\phi &= \frac{1}{I_x}
\end{aligned}
$$

Define the sliding surface s_z as following:

$$
\begin{aligned}
s_\phi &= \dot{e}_\phi + \lambda_\phi e_\phi \\
\dot{e}_\phi &= \dot{\phi} - \dot{\phi}_d \\
e_\phi &= \phi - \phi_d
\end{aligned} \tag{9}
$$

where the desired positions and velocities can be obtained following:

$$
\begin{aligned}
& F_y = m[-K_p(y - y_d) - K_d(\dot{y} - \dot{y}_d) + \ddot{y}_d] \\
& \phi_d = \arcsin\left(\frac{-F_y}{u_1}\right) \\
& \dot{\phi}_d = 0
\end{aligned} \tag{10}
$$

Then, based on the above equations, we can derive the sliding condition equation as following:

$$
\begin{aligned}
s_\phi \dot{s}_\phi &\leq -k_\phi |s_\phi| \\
\dot{s}_\phi &\leq -k_\phi sgn(s_\phi) \\
\ddot{e}_\phi + \lambda_\phi \dot{e}_\phi &\leq -k_\phi sgn(s_\phi) \\
\ddot{\phi} - \ddot{\phi}_d + \lambda_\phi \dot{e}_\phi &\leq -k_\phi sgn(s_\phi) \\
f_\phi + g_\phi u_2 - \ddot{\phi}_d + \lambda_\phi \dot{e}_\phi &\leq -k_\phi sgn(s_\phi) \\
u_2 &\leq \frac{-(f_\phi - \ddot{\phi}_d + \lambda_\phi \dot{e}_\phi) - k_\phi sgn(s_\phi)}{g_\phi}
\end{aligned} \tag{11}
$$

To reduce the chattering effect, replace the sign function $sgn(\cdot)$ with saturation function $sat(\cdot)$, then we can write our the control law:

$$
\begin{aligned}
u_2 &= \frac{-(f_\phi - \ddot{\phi}_d + \lambda_\phi \dot{e}_\phi) + u_{2R}}{g_\phi} \\
u_{2R} &= -k_\phi sat(s_\phi, b_\phi)
\end{aligned} \tag{12}
$$

where ϕ_d and $\dot{\phi}_d$ can be calculated via Equation 10. s_ϕ can be obtained from Equation 9, and

$$
\begin{aligned}
f_\phi &= \dot{\theta}\dot{\psi}\frac{I_y - I_z}{I_x} - \frac{I_p}{I_x}\Omega\dot{\theta} \\
g_\phi &= \frac{1}{I_x} \\
\dot{e}_\phi &= \dot{\phi} - \dot{\phi}_d \\
k_\phi &= 92.5 \\
\lambda_\phi &= 12 \\
b_\phi &= 0.95
\end{aligned}
$$

2.3 Control Law for θ

Firstly, rewrite the dynamic equation for θ:

$$
\begin{aligned}
\ddot{\theta} &= \dot{\phi}\dot{\psi}\frac{I_z - I_x}{I_y} + \frac{I_p}{I_x}\Omega\dot{\phi} + \frac{1}{I_y}u_3 \\
&= f_\theta + g_\theta u_3
\end{aligned}
\tag{13}
$$

where

$$
\begin{aligned}
f_\theta &= \dot{\phi}\dot{\psi}\frac{I_z - I_x}{I_y} + \frac{I_p}{I_x}\Omega\dot{\phi} \\
g_\theta &= \frac{1}{I_y}
\end{aligned}
$$

Define the sliding surface s_z as following:

$$
\begin{aligned}
s_\theta &= \dot{e}_\theta + \lambda_\theta e_\theta \\
\dot{e}_\theta &= \dot{\theta} - \dot{\theta}_d \\
e_\theta &= \theta - \theta_d
\end{aligned}
\tag{14}
$$

where the desired positions and velocities can be obtained following:

$$
\begin{aligned}
F_x &= m[-K_p(x - x_d) - K_d(\dot{x} - \dot{x}_d) + \ddot{x}_d] \\
\theta_d &= \arcsin\left(\frac{F_x}{u_1}\right) \\
\dot{\theta}_d &= 0
\end{aligned}
\tag{15}
$$

Then, based on the above equations, we can derive the sliding condition equation as following:

$$
\begin{aligned}
s_\theta \dot{s}_\theta &\le -k_\theta |s_\theta| \\
\dot{s}_\theta &\le -k_\theta sgn(s_\theta) \\
\ddot{e}_\theta + \lambda_\theta \dot{e}_\theta &\le -k_\theta sgn(s_\theta) \\
\ddot{\theta} - \ddot{\theta}_d + \lambda_\theta \dot{e}_\theta &\le -k_\theta sgn(s_\theta) \\
f_\theta + g_\theta u_3 - \ddot{\theta}_d + \lambda_\theta \dot{e}_\theta &\le -k_\theta sgn(s_\theta) \\
u_3 &\le \frac{-(f_\theta - \ddot{\theta}_d + \lambda_\theta \dot{e}_\theta) - k_\theta sgn(s_\theta)}{g_\theta}
\end{aligned}
\tag{16}
$$

To reduce the chattering effect, replace the sign function $sgn(\cdot)$ with saturation function $sat(\cdot)$, then we can write our the control law:

$$\begin{aligned} u_3 &= \frac{-(f_\theta - \ddot{\theta}_d + \lambda_\theta \dot{e}_\theta) + u_{3R}}{g_\theta} \\ u_{3R} &= -k_\theta sat(s_\theta, b_\theta) \end{aligned} \tag{17}$$

where θ_d and $\dot{\theta}_d$ can be calculated via Equation 15. s_θ can be obtained from Equation 14, and

$$\begin{aligned} f_\theta &= \dot{\phi}\dot{\psi}\frac{I_z - I_x}{I_y} + \frac{I_p}{I_x}\Omega\dot{\phi} \\ g_\theta &= \frac{1}{I_y} \\ \dot{e}_\theta &= \dot{\theta} - \dot{\theta}_d \\ k_\theta &= 92.5 \\ \lambda_\theta &= 12 \\ b_\theta &= 0.95 \end{aligned}$$

2.4 Control Law for ψ

Firstly, rewrite the dynamic equation for ψ:

$$\begin{aligned} \ddot{\psi} &= \dot{\phi}\dot{\theta}\frac{I_x - I_y}{I_z} + \frac{1}{I_z}u_3 \\ &= f_\psi + g_\psi u_4 \end{aligned} \tag{18}$$

where

$$\begin{aligned} f_\psi &= \dot{\phi}\dot{\theta}\frac{I_x - I_y}{I_z} \\ g_\psi &= \frac{1}{I_z} \end{aligned}$$

Define the sliding surface s_z as following:

$$\begin{aligned} s_\psi &= \dot{e}_\psi + \lambda_\psi e_\psi \\ \dot{e}_\psi &= \dot{\psi} - \dot{\psi}_d \\ e_\psi &= \psi - \psi_d \end{aligned} \tag{19}$$

Then, based on the above equations, we can derive the sliding condition equation as following:

$$\begin{aligned} s_\psi \dot{s}_\psi &\leq -k_\psi |s_\psi| \\ \dot{s}_\psi &\leq -k_\psi sgn(s_\psi) \\ \ddot{e}_\psi + \lambda_\psi \dot{e}_\psi &\leq -k_\psi sgn(s_\psi) \\ \ddot{\psi} - \ddot{\psi}_d + \lambda_\psi \dot{e}_\psi &\leq -k_\psi sgn(s_\psi) \\ f_\psi + g_\psi u_4 - \ddot{\psi}_d + \lambda_\psi \dot{e}_\psi &\leq -k_\psi sgn(s_\psi) \\ u_4 &\leq \frac{-(f_\psi - \ddot{\psi}_d + \lambda_\psi \dot{e}_\psi) - k_\psi sgn(s_\psi)}{g_\psi} \end{aligned} \tag{20}$$

To reduce the chattering effect, replace the sign function $sgn(\cdot)$ with saturation function $sat(\cdot)$, then we can write our the control law:

$$\begin{aligned}
u_4 &= \frac{-(f_\psi - \ddot{\psi}_d + \lambda_\psi \dot{e}_\psi) + u_{4R}}{g_\psi} \\
u_{4R} &= -k_\psi sat(s_\psi, b_\psi)
\end{aligned} \tag{21}$$

where $\psi_d = 0$ and $\dot{\psi}_d = 0$. s_ψ can be obtained from Equation 19 and

$$\begin{aligned}
f_\psi &= \dot{\phi}\dot{\theta}\frac{I_x - I_y}{I_z} \\
g_\psi &= \frac{1}{I_z} \\
\dot{e}_\psi &= \dot{\psi} - \dot{\psi}_d \\
k_\psi &= 20 \\
\lambda_\psi &= 20 \\
b_\psi &= 0.95
\end{aligned}$$

2.5 Motor speed

The motor speed can be calculated based on the "allocation matrix" shown following:

$$\begin{bmatrix} {\omega_1}^2 \\ {\omega_2}^2 \\ {\omega_3}^2 \\ {\omega_4}^2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4k_F} & -\frac{\sqrt{2}}{4k_F l} & -\frac{\sqrt{2}}{4k_F l} & -\frac{1}{4k_M k_F} \\ \frac{1}{4k_F} & -\frac{\sqrt{2}}{4k_F l} & \frac{\sqrt{2}}{4k_F l} & \frac{1}{4k_M k_F} \\ \frac{1}{4k_F} & \frac{\sqrt{2}}{4k_F l} & \frac{\sqrt{2}}{4k_F l} & -\frac{1}{4k_M k_F} \\ \frac{1}{4k_F} & \frac{\sqrt{2}}{4k_F l} & -\frac{\sqrt{2}}{4k_F l} & \frac{1}{4k_M k_F} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \tag{22}$$

Then, enforce the range of all ω_i to be $[\omega_{min}, \omega_{max}]$, where $\omega_{min} = 0\ rad/s$ and $\omega_{max} = 2618\ rad/s$

3 Problem Part 3

Part 3. Implement a ROS node in Python or MATLAB to evaluate the performance of the control design on the Crazyflie 2.0 quadrotor in Gazebo. You can create a new ROS package named `project` under the project workspace for this purpose. The script must implement the trajectories generated in Part 1 and the sliding mode control laws formulated in Part 2.

A Python sample code is provided as the starting point for your implementation. Please read carefully the provided code, and understand its functionality. Feel free to define new functions and variables in your program if needed.
If using MATLAB, you are required to re-write the provided Python codes in MATLAB.

Please refer to the code submission

4 Problem Part 4

The figure for desired and real trajectory is shown:

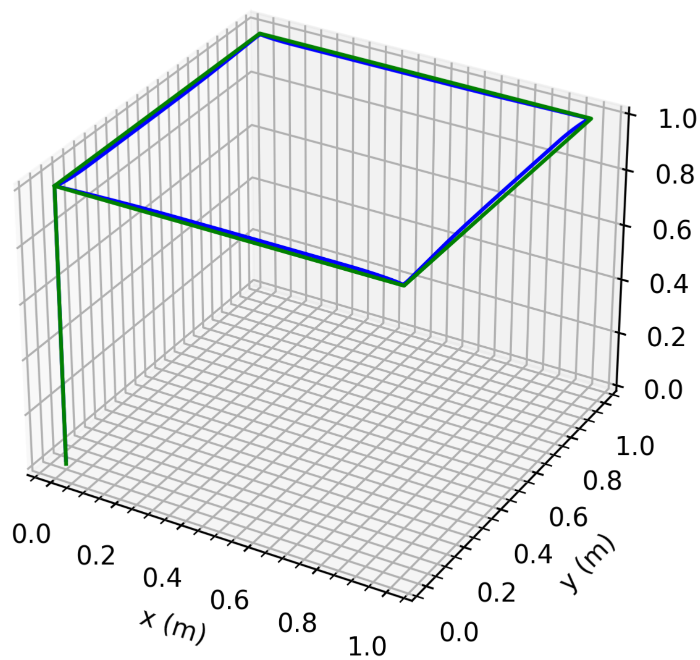


Figure 1: Desired and real trajectory