Robust Trajectory Tracking for Quadrotor UAVs using Sliding Mode Control

RBE-502 Project

Bhaavin Kishore Jogeshwar, Vishrut Bohara

December 18, 2022

1 Trajectory Generation

Quintic Trajectories (position, velocity, and acceleration) for the translational coordinates of the Crazyflie quadrotor are generated, as shown in the figure below. The quadrotor visits five waypoints in sequence. The five waypoints are as follows:

- 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 velocity and acceleration at each waypoint are equal to zero.

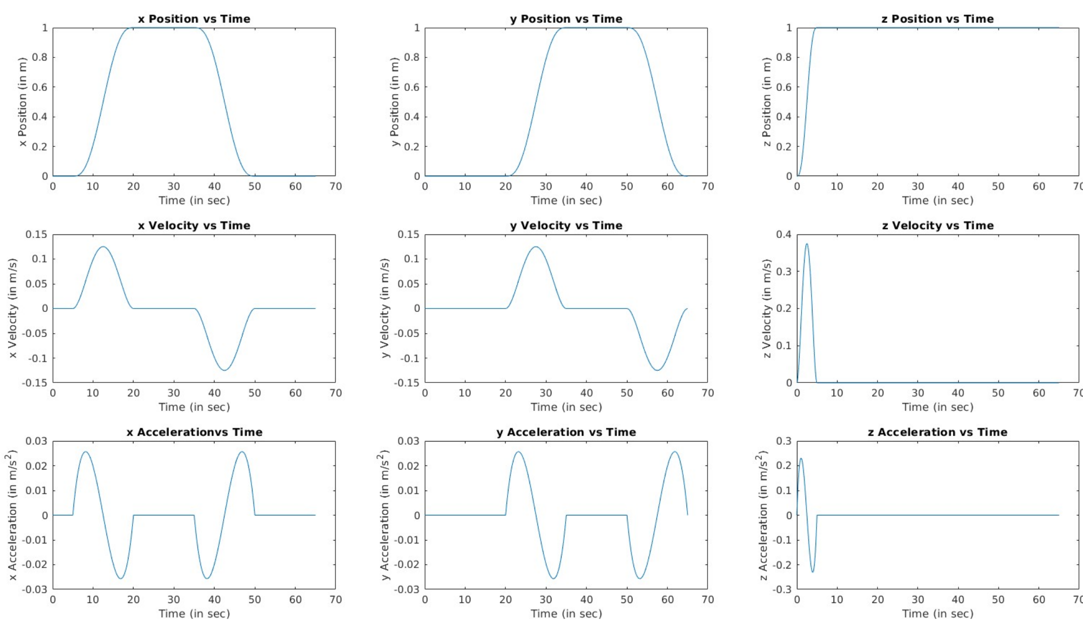

Figure 1: A plot of the desired trajectories.

2 Sliding Mode Control Laws

2.1 Controller Design

Equation of Motions are :

$$\ddot{x} = \frac{1}{m}(cos\phi\ sin\theta\ cos\psi + sin\phi\ sin\psi)\ u_1 \tag{2.1}$$

$$\ddot{y} = \frac{1}{m}(cos\phi\ sin\theta\ sin\psi - sin\phi\ cos\psi)\ u_1 \tag{2.2}$$

$$\ddot{z} = \frac{1}{m}(cos\phi\ cos\theta)\ u_1 - g \tag{2.3}$$

$$\ddot{\phi} = \dot{\theta}\dot{\psi}\ \frac{I_y - I_z}{I_x} - \frac{I_p}{I_x}\ \Omega\ \dot{\theta} + \frac{1}{I_x}u_2 \tag{2.4}$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\ \frac{I_z - I_x}{I_y} + \frac{I_p}{I_y}\ \Omega\ \dot{\phi} + \frac{1}{I_y}u_3 \tag{2.5}$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\ \frac{I_x - I_y}{I_z} + \frac{1}{I_z}u_4 \tag{2.6}$$

$$F_x = m(-k_p(x - x_d) - k_d(\dot{x} - \dot{x}_d) + \ddot{x}_d) \tag{2.7}$$

$$F_y = m(-k_p(y - y_d) - k_d(\dot{y} - \dot{y}_d) + \ddot{y}_d) \tag{2.8}$$

$$\theta_d = sin^{-1}(\frac{F_x}{u_1}) \tag{2.9}$$

$$\phi_d = sin^{-1}(\frac{-F_y}{u_1}) \tag{2.10}$$

$$\dot{\theta}_d = 0, \dot{\phi}_d = 0, \psi_d = 0, \dot{\psi}_d = 0 \tag{2.11}$$

Considering equation (2.3):

$$\begin{aligned} e &= z_d - z \\ \dot{e} &= \dot{z}_d - \dot{z} \\ \ddot{e} &= \ddot{z}_d - \ddot{z} = \ddot{z}_d - \frac{1}{m}(cos\phi\ cos\theta)\ u_1 + g \end{aligned} \tag{2.12}$$

$$\begin{aligned} s &= \dot{e} + \lambda_1 e \\ \dot{s} &= \ddot{e} + \lambda_1 \dot{e} \\ &= \ddot{z}_d - \frac{1}{m}(cos\phi\ cos\theta)\ u_1 + g + \lambda_1 \dot{e} \\ s\dot{s} &\leq -K_1|s| \end{aligned} \tag{2.13}$$

Let $u_1 = \frac{m}{(cos\phi\ cos\theta)}(\ddot{z}_d + g + \lambda_1\dot{e} + u_{1r})$,

$$\begin{aligned} \therefore -s(u_{1r}) &\leq -K_1|s| \\ u_{1r} &= K_1 sign(s) \\ u_1 &= \frac{m}{(cos\phi\ cos\theta)}(\ddot{z}_d + g + \lambda_1\dot{e} + K_1 sign(s)) \end{aligned}$$

To avoid chattering, we switched the sign function with the saturation (sat) function. This gives control input as:

$$u_1 = \frac{m}{(cos\phi\ cos\theta)}(\ddot{z}_d + g + \lambda_1\dot{e} + K_1 sat(s, \rho_1)) \tag{\textbf{2.14}}$$

Considering equation (2.4):

$$\begin{aligned} e &= \phi_d - \phi \\ \dot{e} &= \dot{\phi}_d - \dot{\phi} \\ \ddot{e} &= \ddot{\phi}_d - \ddot{\phi} = \ddot{\phi}_d - \ \dot{\theta}\dot{\psi}\ \frac{I_y - I_z}{I_x} + \frac{I_p}{I_x}\ \Omega\ \dot{\theta} - \frac{1}{I_x}u_2 \end{aligned} \tag{2.15}$$

$$
\begin{aligned}
s &= \dot{e} + \lambda_2 e \\
\dot{s} &= \ddot{e} + \lambda_2 \dot{e} \\
&= \ddot{\phi}_d - \ \dot{\theta}\dot{\psi}\ \frac{I_y - I_z}{I_x} + \frac{I_p}{I_x}\ \Omega\ \dot{\theta} - \frac{1}{I_x} u_2 + \lambda_2 \dot{e} \\
s\dot{s} &\leq -K_2 |s|
\end{aligned}
\tag{2.16}
$$

Considering Ω to be an unknown parameter varying with control inputs, let $u_2 = I_x\ddot{\phi}_d - \ \dot{\theta}\dot{\psi}\ (I_y - I_z) + I_p\ \hat{\Omega}\ \dot{\theta} + I_x\lambda_2\dot{e} + I_x u_{2r}$,

$$
\begin{aligned}
\therefore -s(u_{2r} + \frac{I_p}{I_x}(\Omega - \hat{\Omega})) &\leq -K_2|s| \\
u_{2r} &= (K_2 + \frac{I_p}{I_x}\eta) sign(s)\ \text{, where } \eta \geq |\Omega - \hat{\Omega}| \\
u_2 &= I_x\ddot{\phi}_d - \ \dot{\theta}\dot{\psi}\ (I_y - I_z) + I_p\ \hat{\Omega}\ \dot{\theta} + I_x\lambda_2\dot{e} + (I_x K_2 + I_p\eta) sign(s)
\end{aligned}
$$

To avoid chattering, we swap the sign function with the saturation (sat) function. This gives control input as:

$$
u_2 = I_x\ddot{\phi}_d - \ \dot{\theta}\dot{\psi}\ (I_y - I_z) + I_p\ \hat{\Omega}\ \dot{\theta} + I_x\lambda_2\dot{e} + (I_x K_2 + I_p\eta) sat(s, \rho_2) \tag{\textbf{2.17}}
$$

Considering equation (2.5):

$$
\begin{aligned}
e &= \theta_d - \theta \\
\dot{e} &= \dot{\theta}_d - \dot{\theta} \\
\ddot{e} &= \ddot{\theta}_d - \ddot{\theta} = \ddot{\theta}_d - \ \dot{\phi}\dot{\psi}\ \frac{I_z - I_x}{I_y} - \frac{I_p}{I_y}\ \Omega\ \dot{\phi} - \frac{1}{I_y} u_3
\end{aligned}
\tag{2.18}
$$

$$
\begin{aligned}
s &= \dot{e} + \lambda_3 e \\
\dot{s} &= \ddot{e} + \lambda_3 \dot{e} \\
&= \ddot{\theta}_d - \ \dot{\phi}\dot{\psi}\ \frac{I_z - I_x}{I_y} - \frac{I_p}{I_y}\ \Omega\ \dot{\phi} - \frac{1}{I_y} u_3 + \lambda_3 \dot{e} \\
s\dot{s} &\leq -K_3 |s|
\end{aligned}
\tag{2.19}
$$

Let $u_3 = I_y\ddot{\theta}_d - \ \dot{\phi}\dot{\psi}\ (I_z - I_x) - I_p\ \hat{\Omega}\ \dot{\phi} + I_y\lambda_3\dot{e} + I_y u_{3r}$

$$
\begin{aligned}
\therefore -s(u_{3r} + \frac{I_p}{I_y}(\hat{\Omega} - \Omega)) &\leq -K_3|s| \\
u_{3r} &= (K_3 + \frac{I_p}{I_y}\eta) sign(s)\ \text{, where } \eta \geq |\Omega - \hat{\Omega}| \\
u_3 &= I_y\ddot{\theta}_d - \ \dot{\phi}\dot{\psi}\ (I_z - I_x) - I_p\ \hat{\Omega}\ \dot{\phi} + I_y\lambda_3\dot{e} + (I_y K_3 + I_p\eta) sign(s)
\end{aligned}
$$

To avoid chattering, we switch the sign function with the saturation (sat) function. This gives control input as:

$$
u_3 = I_y\ddot{\theta}_d - \ \dot{\phi}\dot{\psi}\ (I_z - I_x) - I_p\ \hat{\Omega}\ \dot{\phi} + I_y\lambda_3\dot{e} + (I_y K_3 + I_p\eta) sat(s, \rho_3) \tag{\textbf{2.20}}
$$

Considering equation (2.6):

$$
\begin{aligned}
e &= \psi_d - \psi \\
\dot{e} &= \dot{\psi}_d - \dot{\psi} \\
\ddot{e} &= \ddot{\psi}_d - \ddot{\psi} = \ddot{\psi}_d - \ \dot{\phi}\dot{\theta}\ \frac{I_x - I_y}{I_z} - \frac{1}{I_z} u_4
\end{aligned}
\tag{2.21}
$$

$$
\begin{aligned}
s &= \dot{e} + \lambda_4 e \\
\dot{s} &= \ddot{e} + \lambda_4 \dot{e} \\
&= \ddot{\psi}_d - \ \dot{\phi}\dot{\theta}\ \frac{I_x - I_y}{I_z} - \frac{1}{I_z} u_4 + \lambda_4 \dot{e} \\
s\dot{s} &\leq -K_4 |s|
\end{aligned}
\tag{2.22}
$$

Let $u_4 = I_z\ddot{\psi}_d - \dot{\phi}\dot{\psi}\ (I_x - I_y) + I_z\lambda_4\dot{e} + I_z u_{4r}$

$$\therefore -s(u_{4r}) \leq -K_4|s|$$
$$u_{4r} = K_4 sign(s)$$
$$u_4 = I_z\ddot{\psi}_d - \dot{\phi}\dot{\psi}\ (I_x - I_y) + I_z\lambda_4\dot{e} + I_z K_4 sign(s)$$

To avoid chattering, we switch the sign function with the saturation (sat) function. This gives control input as:

$$u_4 = I_z\ddot{\psi}_d - \dot{\phi}\dot{\psi}\ (I_x - I_y) + I_z\lambda_4\dot{e} + I_z K_4 sat(s, \rho_4) \quad \textbf{(2.23)}$$

2.2 Extension to Controller

Updated desired trajectory calculations:

$$m(\ddot{x}\ sin\psi - \ddot{y}\ cos\psi) = sin\phi\ u_1 \quad (2.24)$$
$$m(\ddot{x}\ cos\psi + \ddot{y}\ sin\psi) = sin\theta\ cos\phi\ u_1 \quad (2.25)$$
$$F_p = F_x sin\psi - F_y cos\psi \quad (2.26)$$
$$F_t = F_x cos\psi + F_y sin\psi \quad (2.27)$$
$$\phi_d = sin^{-1}(\frac{F_p}{u_1}) \quad (2.28)$$
$$\theta_d = sin^{-1}(\frac{F_t}{u_1 cos\phi}) \quad (2.29)$$
$$\dot{F}_x = m(-k_p(\dot{x} - \dot{x}_d) - k_d(\ddot{x} - \ddot{x}_d) + \dddot{x}_d) \text{ ,where we considered } \ddot{x} = 0 \quad (2.30)$$
$$\dot{F}_y = m(-k_p(\dot{y} - \dot{y}_d) - k_d(\ddot{y} - \ddot{y}_d) + \dddot{y}_d) \text{ ,where we considered } \ddot{y} = 0 \quad (2.31)$$
$$\dot{F}_p = \dot{F}_x sin\psi - \dot{F}_y cos\psi + F_t\dot{\psi} \quad (2.32)$$
$$\dot{F}_t = \dot{F}_x cos\psi + \dot{F}_y sin\psi - F_p\dot{\psi} \quad (2.33)$$
$$\dot{\phi}_d = \frac{\dot{F}_p}{u_1 cos\phi_d}, \quad (2.34)$$
$$\dot{\theta}_d = \frac{\dot{F}_t cos\phi + F_t sin\phi}{u_1\ cos^2\phi\ cos\theta_d} \quad (2.35)$$

2.3 Design and tuning parameters

Parameter	Description	Values Used
λ_i ; $i\epsilon 1,2,3,4$	It controls the rate of exponential convergence to the origin.	$4, 6, 6, 16$
K_i ; $i\epsilon 1,2,3,4$	It controls the convergence rate to the sliding surface.	$10, 400, 450, 8$
ρ_i ; $i\epsilon 1,2,3,4$	Boundary layer parameter reduces the chattering and makes the control inputs smoother, but it affects the convergence to the origin and adds a tracking error.	$0.1, 1, 1, 0.01$
η	This represents the error in Ω. Higher the error assumption, more aggressive the controller for ϕ and θ.	$0.2\ \Omega$
K_{px}, K_{dx}	These gains controls the error in x direction which is directly proportional to θ_d, therefore high gains can lead to aggressive pitch controller.	$30, 11$
K_{py}, K_{dy}	These gains control the error in the y direction, which is directly proportional to ϕ_d, therefore high gains can lead to aggressive roll controller.	$56, 15$

Table 1: Description of Design and Tuning parameters used

3 Code Description

Below is the pseudo python code:

```
def get_velocities(trajectories):
    global Omega
    # Calculate u1
    if(u1 != 0):
        #Calculate desired phi and theta
    else:
        phi_d=0
        theta_d=0
    #Calculate u2 and u3 based on calculated phi_d and theta_d
    #Calculate u4 considering shi_d and dshi_d = 0
    Omega = u1-u2+u3-u4
    motor_vel = Allocation_matrix*[u1;u2;u3;u4]

    return motor_vel

def smc_control():
    trajectories = trajectory_evaluate()
    motor_vel = get_velocities(trajectories)
    motor_speed_pub.publish(motor_vel)
```

In the *smc_controller*, we are first calculating trajectories using the *trajectory_evaluate* function. This function returns desired values for $x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}, z, \dot{z}, \ddot{z}$. Then we call *get_velocities* function, which uses desired trajectories, tuning parameters (from section 2.3), and the equations derived in section 2.1, to calculate u_1, u_2, u_3, u_4. We then use the allocation matrix to calculate rotor speeds which are then published to Crazyflie using the *motor_speed_pub.publish*() command.

3.1 Submission

Bohara_Jogeshwar_Codes:

- Main.m The main MATLAB script.
- GetTraj.m: A function that returns a trajectory for a given timestamp.
- sat.m: Saturation function.
- crazy_controller.m: An ODE function replicating Crazyflie dynamics in MATLAB.

- Part1.m: Contains the code for Part 1 of the final project. It uses the $function\_traj\_var.m$ file.
- function_traj_var.m: A function to calculate the a,b,c,d,e,f values of the quintic trajectories.
- follow.py: A python script for Part 3 of the final project.

4 Results

The motion in x induces error in ψ, which affects the trajectory tracking, but after switching to the extended version of the controller, our trajectory tracking became independent of error in ψ, which resulted in better tracking.

A slight deviation in trajectory is seen due to the error in ψ that was expected to be 0.

Our SMC is robust enough to track trajectories efficiently.

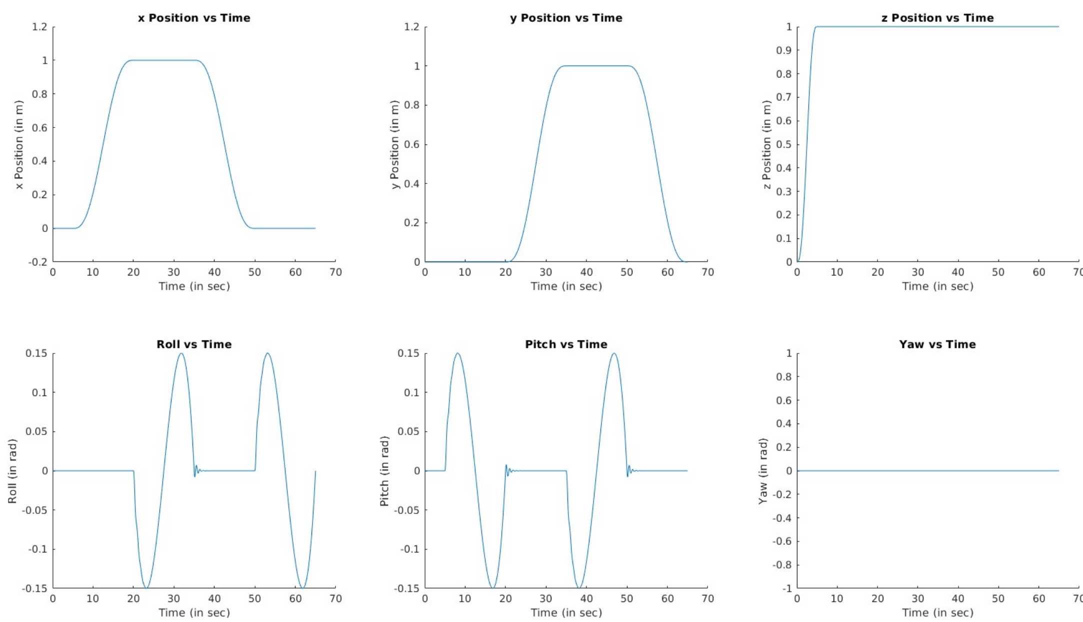

Figure 2: State transition plot using MATLAB

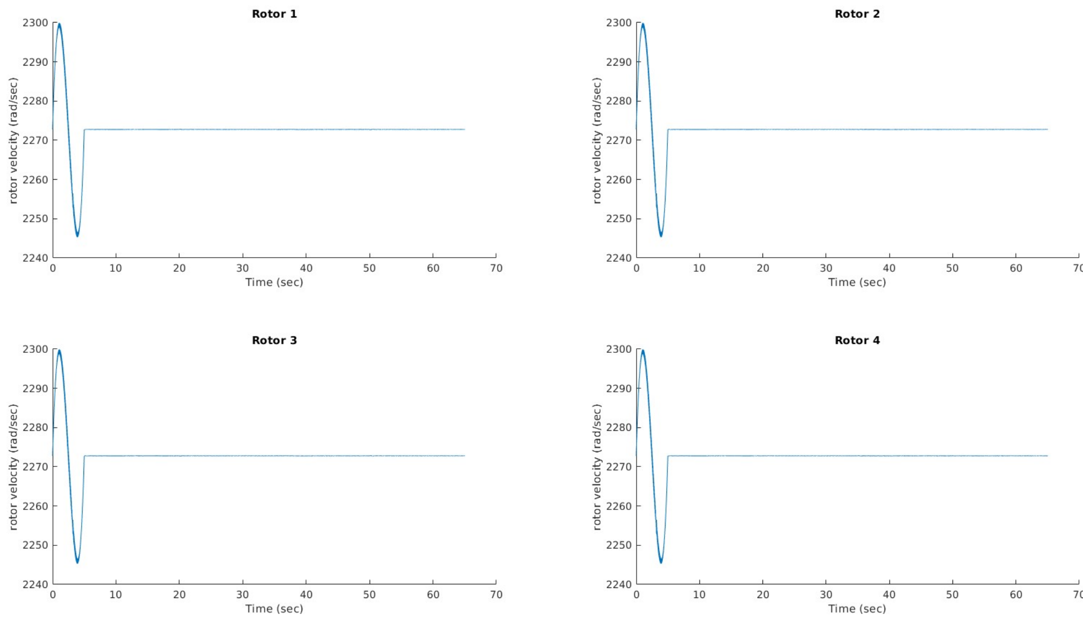

Figure 3: Rotor velocities plot using MATLAB

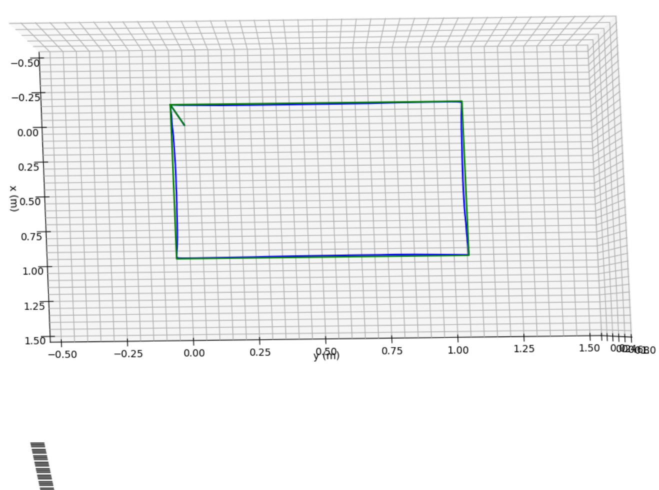

Figure 4: Top view of Actual Trajectory for 15-second steps

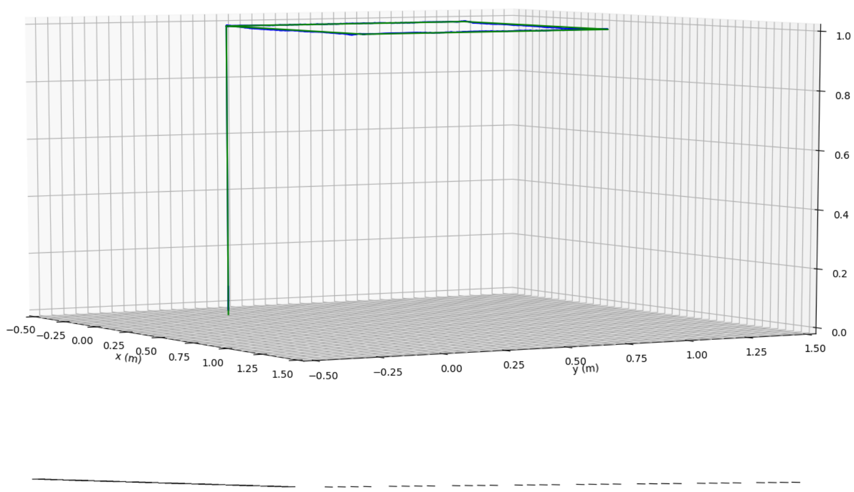

Figure 5: Side view of Actual Trajectory for 15-second steps

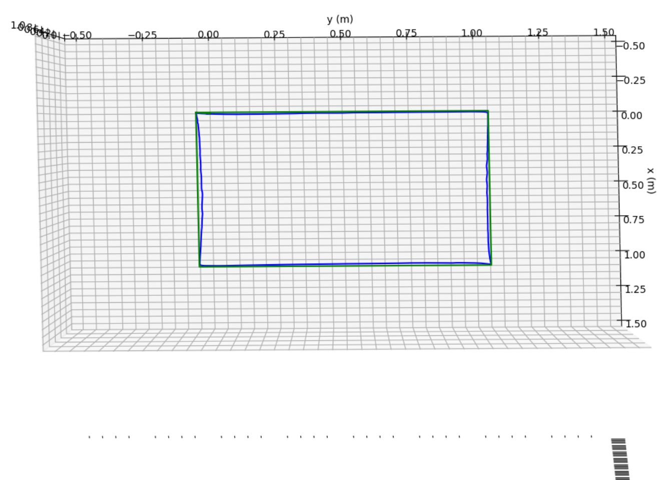

Figure 6: Top view of Actual Trajectory for 10-second steps

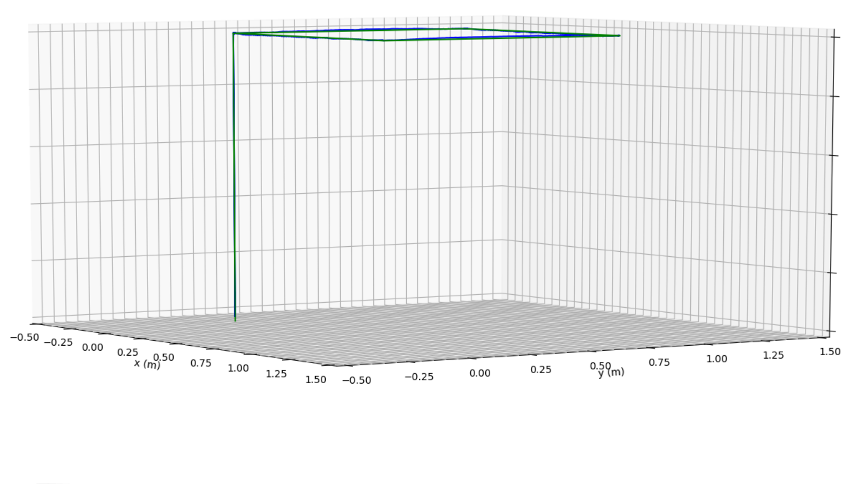

Figure 7: Side view of Actual Trajectory for 10-second steps