Target group: M2SAAS - Universite Evry Paris Saclay

Flight Planning Lab Report

Prepared by: Melkamu Amare ==> ID: 20245847

Abel Yehuala ==> ID: 20245853

M2SAAS - 2024

Submitted to: Mr. Amayas Benoudiba

Universite Evry Paris-Saclay

Table of Contents

Exercise 1 : TRIM trajectories

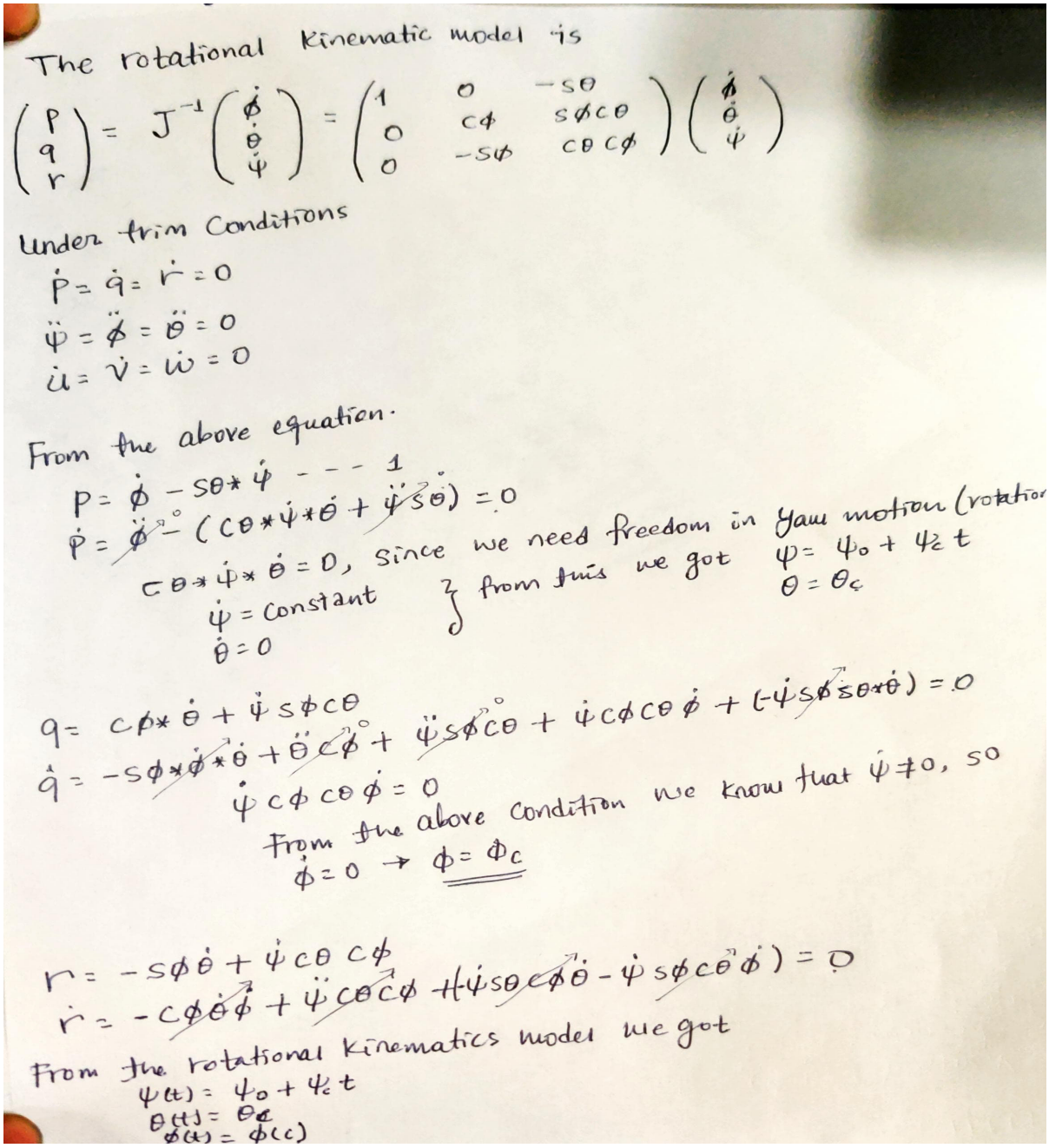
Prove using the kinematic model of translation and orientation that equilibrium paths or trim trajectories are represented by helices. The translational kinematic model is given by:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = R \begin{pmatrix} u \\ v \\ w \end{pmatrix} \text{ where } R = \begin{pmatrix} c\theta c\psi & s\theta c\psi s\phi - s\psi c\phi & s\theta c\psi c\phi + s\psi s\phi \\ c\theta s\psi & s\theta s\psi s\phi + c\psi c\phi & s\theta s\psi c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\psi c\phi \end{pmatrix}$$

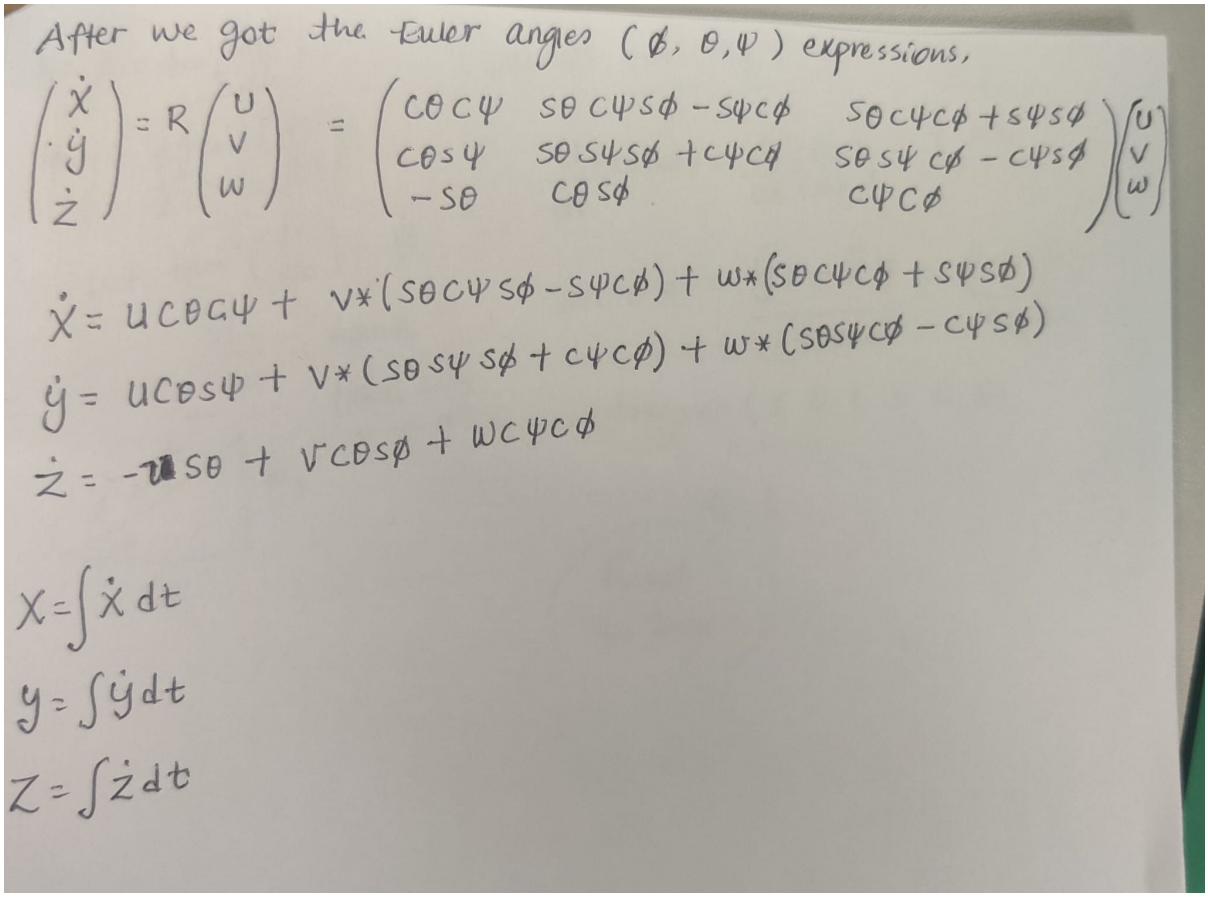
and the rotational kinematic model is:

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = J \begin{pmatrix} p \\ q \\ r \end{pmatrix} \text{ where } J = \begin{pmatrix} 1 & s\phi \tan\theta & c\phi \tan\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\theta/c\phi \end{pmatrix}$$

$$\text{or } \begin{pmatrix} p \\ q \\ r \end{pmatrix} = J^{-1} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -s\theta \\ 0 & c\phi & s\phi c\theta \\ 0 & -s\phi & c\theta c\phi \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix}$$



We have got the angular values at equilibrium from the above handmade calculations, now we can substitute it into R matrix and integrate the product of the matrix and the velocities to get the positions in inertial frame of reference.



$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{pmatrix} \frac{2v\cos(\phi)\cos(\sigma_1)^2 - 2w\cos(\sigma_1)^2\sin(\phi) + 2u\cos(\sigma_1)\cos(\theta)\sin(\sigma_1) + 2w\cos(\phi)\cos(\sigma_1)\sin(\sigma_1)\sin(\theta) + 2v\cos(\sigma_1)\sin(\phi)\sin(\sigma_1)\sin(\theta)}{\psi} \\ -\frac{2u\cos(\sigma_1)^2\cos(\theta) - 2v\cos(\phi)\cos(\sigma_1)\sin(\sigma_1) + 2w\cos(\sigma_1)\sin(\phi)\sin(\sigma_1) + 2w\cos(\phi)\cos(\sigma_1)^2\sin(\theta) + 2v\cos(\sigma_1)^2\sin(\phi)\sin(\theta)}{\psi} \\ t\,(w\cos(\phi)\cos(\theta) - u\sin(\theta) + v\cos(\theta)\sin(\phi)) \end{pmatrix}$$

where

$$\sigma_1 = \frac{psic}{2} + \frac{\psi\, t}{2}$$

The integration is done in matlab as follows using symbolic toolbox.

```
syms t psi1(t)
syms theta phi u v w psi psic real

psi1(t) = psi*t + psic;

% Define rotation matrix R
```

```
R = [cos(theta)*cos(psi1(t)), sin(phi)*sin(theta)*cos(psi1(t))
- cos(phi)*sin(psi1(t)), cos(phi)*sin(theta)*cos(psi1(t)) +
sin(phi)*sin(psi1(t));
     cos(theta)*sin(psi1(t)), sin(phi)*sin(theta)*sin(psi1(t))
+ cos(phi)*cos(psi1(t)), cos(phi)*sin(theta)*sin(psi1(t)) -
sin(phi)*cos(psi1(t));
     -sin(theta),    cos(theta)*sin(phi),
cos(theta)*cos(phi)];

% Translational model
dpos = R * [u; v; w];
pos = int(dpos, t);
pos = simplify(pos)
```

```
% substitute variables with numbers
% u = 2; v = 2;  w = 1; phi = pi/6; theta = pi/6; psic = pi/6; psi = pi/3;
xx = subs(pos, [u, v, w, theta, phi, psi, psic], [2, 2, 1, pi/6, pi/6,
pi/3, pi/6])
```

xx =

$$\begin{pmatrix} \dfrac{6\,\sigma_1\left(\frac{\sigma_2}{2}-\frac{\sigma_1}{2}+\sqrt{3}\,\sigma_1+\frac{5\,\sqrt{3}\,\sigma_2}{4}\right)}{\pi} \\ -\dfrac{6\,\sigma_1\left(\frac{\sigma_1}{2}+\frac{\sigma_2}{2}+\frac{5\,\sqrt{3}\,\sigma_1}{4}-\sqrt{3}\,\sigma_2\right)}{\pi} \\ t\left(\frac{\sqrt{3}}{2}-\frac{1}{4}\right) \end{pmatrix}$$

where

$$\sigma_1 = \cos\left(\frac{\pi}{12}+\frac{\pi\,t}{6}\right)$$

$$\sigma_2 = \sin\left(\frac{\pi}{12}+\frac{\pi\,t}{6}\right)$$

```
t = 0:0.1:100;
length(t)
```

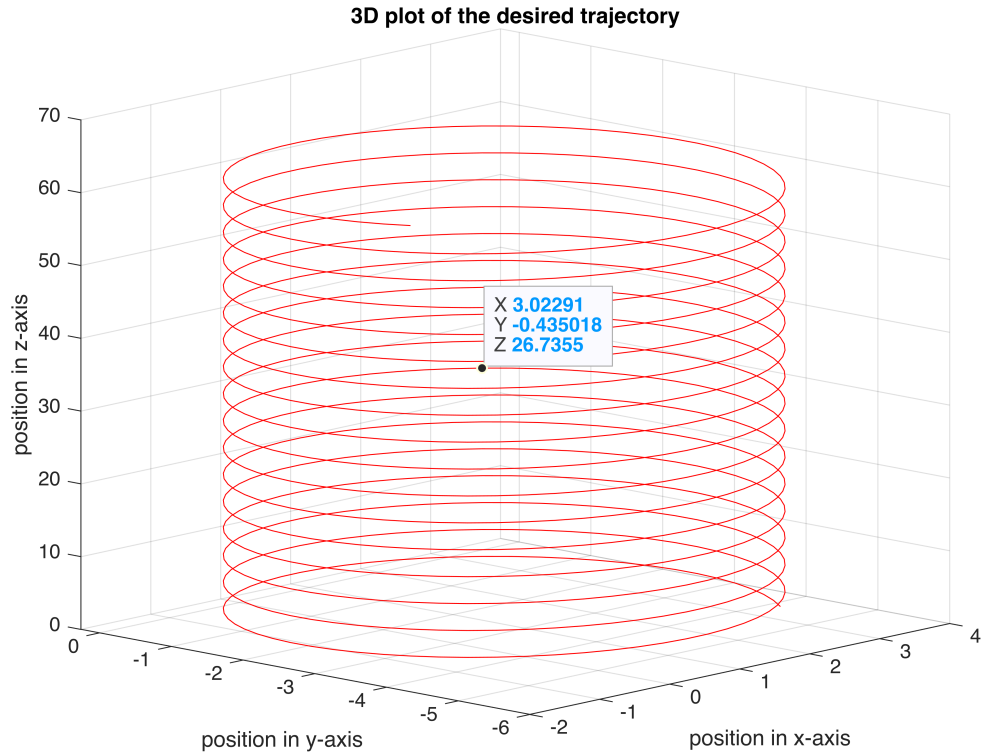
```
ans =
1001
```

```
pos1 = eval(subs(xx, t))
```

```
pos1 = 3×1001
    3.4679    3.6242    3.7538    3.8550    3.9270    3.9688    3.9800    3.9605 ···
```

```
   -4.1607   -3.9123   -3.6490   -3.3735   -3.0890   -2.7985   -2.5053   -2.2124
         0    0.0616    0.1232    0.1848    0.2464    0.3080    0.3696    0.4312
```

```
%pos1 = eval(xx)
plot3(pos1(1,:), pos1(2,:), pos1(3,:), 'r-');
grid on;
title('3D plot of the desired trajectory');
xlabel('position in x-axis');
ylabel('position in y-axis');
zlabel('position in z-axis');
```



Summary

In this section, we have proved the helical trajectory generation using the rotational and translational kinematics. We have derived the angular values from the given relationship and necessary conditions for helical trajectory. After that, we substituted the angular values in the translational kinematics and integrated the dx = R*(u; v; w) in time domain to get the position in 3D that create the helix. The integration was done by Matlab and the result is take as a latex code. After the integration, we substituted the symbolic variables by numbers to see the results and plot the trajectory. From the result of the 3D plot, we have observed that a smooth helical trajectory is generated.

Exercise 2 : Quad-rotor trim trajectories characterization

The dynamic model of a quad-rotor is given by the following equations:

$$\dot{X} = f(X, Y) = \begin{pmatrix} \dot{\phi} \\ a_1\dot{\theta}\dot{\psi} + a_2\dot{\theta}\Omega_r + b_1 U_1 \\ \dot{\theta} \\ a_3\dot{\phi}\dot{\psi} + a_4\dot{\phi}\Omega_r + b_2 U_2 \\ \dot{\psi} \\ a_5\dot{\theta}\dot{\phi} + b_3 U_3 \\ \dot{x} \\ \frac{\sin\psi\sin\phi + \cos\psi\sin\theta\cos\phi}{m} U_4 \\ \dot{y} \\ -\frac{-\cos\psi\sin\phi + \sin\psi\sin\theta\cos\phi}{m} U_4 \\ \dot{z} \\ g - \frac{\cos\psi\cos\phi}{m} U_4 \end{pmatrix}$$

with $X = \left(\phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}, x, \dot{x}, y, \dot{y}, z, \dot{z}\right)^T$ and

$$\begin{aligned} U_1 &= b\left(-\Omega_2^2 + \Omega_4^2\right) \\ U_2 &= b\left(\Omega_1^2 - \Omega_3^2\right) \\ U_3 &= d\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) \\ U_4 &= b\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \\ \Omega_r &= -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{aligned}$$

$$\begin{aligned} a_1 &= \frac{I_{yy} - I_{zz}}{I_{xx}}, a_2 = \frac{J_r}{I_{xx}}, a_3 = \frac{I_{zz} - I_{xx}}{I_{yy}}, a_4 = \frac{J_r}{I_{yy},}, a_5 = \frac{I_{xx} - I_{yy}}{I_{zz}} \\ b_1 &= \frac{l}{I_{xx}}, b_2 = \frac{l}{I_{yy}}, b_3 = \frac{l}{I_{zz}} \end{aligned}$$

Characterize the trim trajectories taking into account effective limitations on thrust and angles.

Lets start by taking some assumptions:

To satisfy trim conditions:

- Angular rates $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ are constant.
- Linear velocities $(\dot{x}, \dot{y}, \dot{z})$ are constant.

- Linear and angular accelerations are zero.

A helical trajectory implies:

- **Linear upward motion**: The velocity along the z-axis ($\dot{z}$) is constant and different from zero.
- **Rotational motion**: The yaw rate ($\dot{\psi}$) is constantt while roll (ϕ) and pitch (θ) angles remain constant.
- **No lateral motion**: The velocities along the x- and y-axes ($\dot{x}$ and $\dot{y}$) are determined by the rotational motion due to the yaw rate.

Now, lets split the angular and linear dynamics:

$$\begin{bmatrix} \mathrm{dx} \\ \mathrm{ddx} \\ \mathrm{dy} \\ \mathrm{ddy} \\ \mathrm{dz} \\ \mathrm{ddz} \end{bmatrix} = \begin{bmatrix} \mathrm{dx} \\ (c\psi s\phi + c\psi s\theta c\phi)\frac{u_4}{m} \\ \mathrm{dy} \\ (-c\psi s\phi + s\psi s\theta c\phi)\frac{u_4}{m} \\ \mathrm{dz} \\ g - (c\psi c\phi)\frac{u_4}{m} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} d\phi \\ \mathrm{dd}\phi \\ d\theta \\ \mathrm{dd}\theta \\ d\psi \\ \mathrm{dd}\psi \end{bmatrix} = \begin{bmatrix} d\phi \\ (a_1 d\theta d\psi + a_2 d\theta \Omega_r) + b_1 u_1 \\ d\theta \\ (a_3 d\theta d\psi + a_4 d\phi \Omega_r) + b_2 u_2 \\ \mathrm{dz} \\ a_5 d\theta d\phi + b_3 u_3 \end{bmatrix}$$

according to the helical trajectory assumption written above :

$$\begin{bmatrix} \mathrm{dx} \\ \mathrm{ddx} \\ \mathrm{dy} \\ \mathrm{ddy} \\ \mathrm{dz} \\ \mathrm{ddz} \end{bmatrix} = \begin{bmatrix} x_c \\ 0 \\ y_c \\ 0 \\ \mathrm{dz} \\ 0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} d\phi \\ \mathrm{dd}\phi \\ d\theta \\ \mathrm{dd}\theta \\ d\psi \\ \mathrm{dd}\psi \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \psi_c \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} \phi \\ d\phi \\ \theta \\ d\theta \\ \psi \\ \psi \end{bmatrix} = \int \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \psi_c \\ 0 \end{bmatrix} \mathrm{dt} = \begin{bmatrix} \phi_c \\ 0 \\ \theta_c \\ 0 \\ \psi_c * t \\ 0 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2 * d\left({\Omega_2}^2 - {\Omega_1}^2\right) \\ \frac{\mathrm{mg}}{c\psi c\phi} + m * \mathrm{dz} \end{bmatrix}$$ ==> m*dz is added to trigger a vertical motion

$$\text{Then, x} = \int \text{dxdt} \implies \begin{bmatrix} x \\ \text{dx} \\ y \\ \text{dy} \\ z \\ \text{dz} \end{bmatrix} = \int \begin{bmatrix} \text{dx} \\ \text{ddx} \\ \text{dy} \\ \text{ddy} \\ \text{dz} \\ \text{ddz} \end{bmatrix} \text{dt} = \begin{bmatrix} x_c \\ 0 \\ y_c \\ 0 \\ z \\ 0 \end{bmatrix} = \begin{bmatrix} r * \cos(\psi) \\ 0 \\ r * \sin(\psi) \\ 0 \\ z + z_c \\ 0 \end{bmatrix} = \begin{bmatrix} r * \cos(\psi_c * t) \\ 0 \\ r * \sin(\psi_c * t) \\ 0 \\ k * t + z_c \\ 0 \end{bmatrix};$$

$$\text{where: r} = \sqrt{(x_c^2 + y_c^2)}$$

Here, we can draw our helical trajectory using x, y, and z in the plot3() function.

The MATLAB code provides a simplified way to simulate the helical trajectory.

Trajectory Calculation:

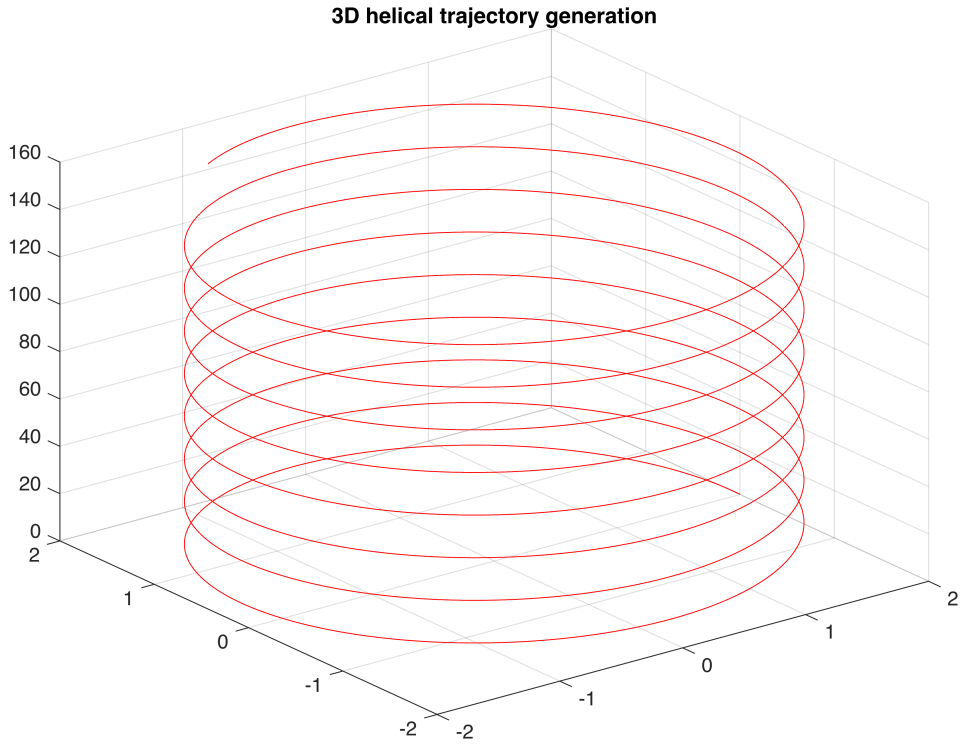
Compute x(t), y(t), and z(t) for each time step using the parametric equations.

Plotting and Visualization:

Simulate the path of quadrotor at its current position.

Use MATLAB's `plot3()` function to visualize the 3D trajectory

```
r = 2; % radius of helix
psi_c = pi/6;
z_c = 0.1;
k = 1.5;
t = 0:0.01:100;
x = r*cos(psi_c*t);
y = r*sin(psi_c*t);
z = k*t+z_c;
plot3(x, y, z, 'r-')
grid on;
title('3D helical trajectory generation');
```



Modeling in Simulink

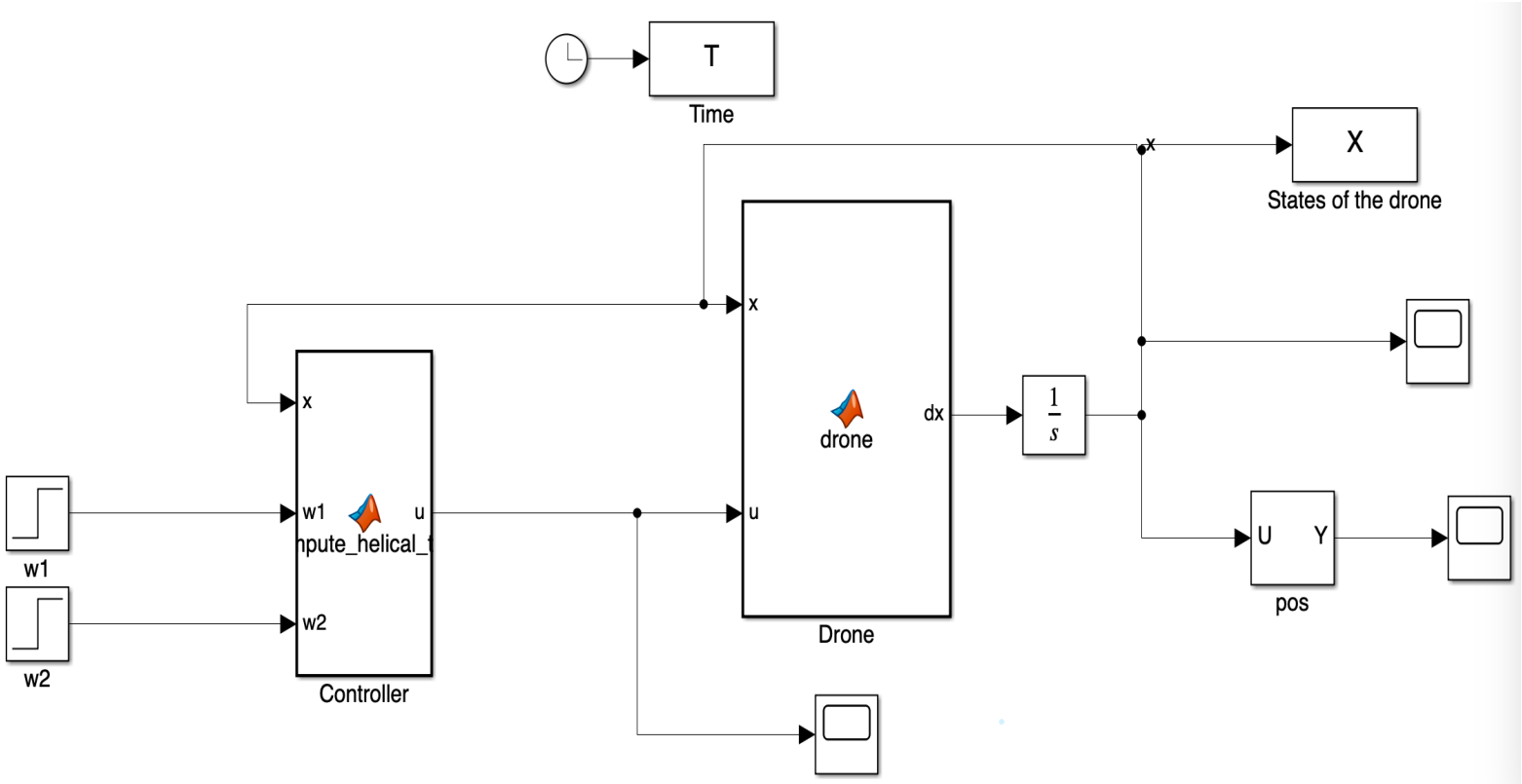


Fig 1: Designed Simulink model of the given drone

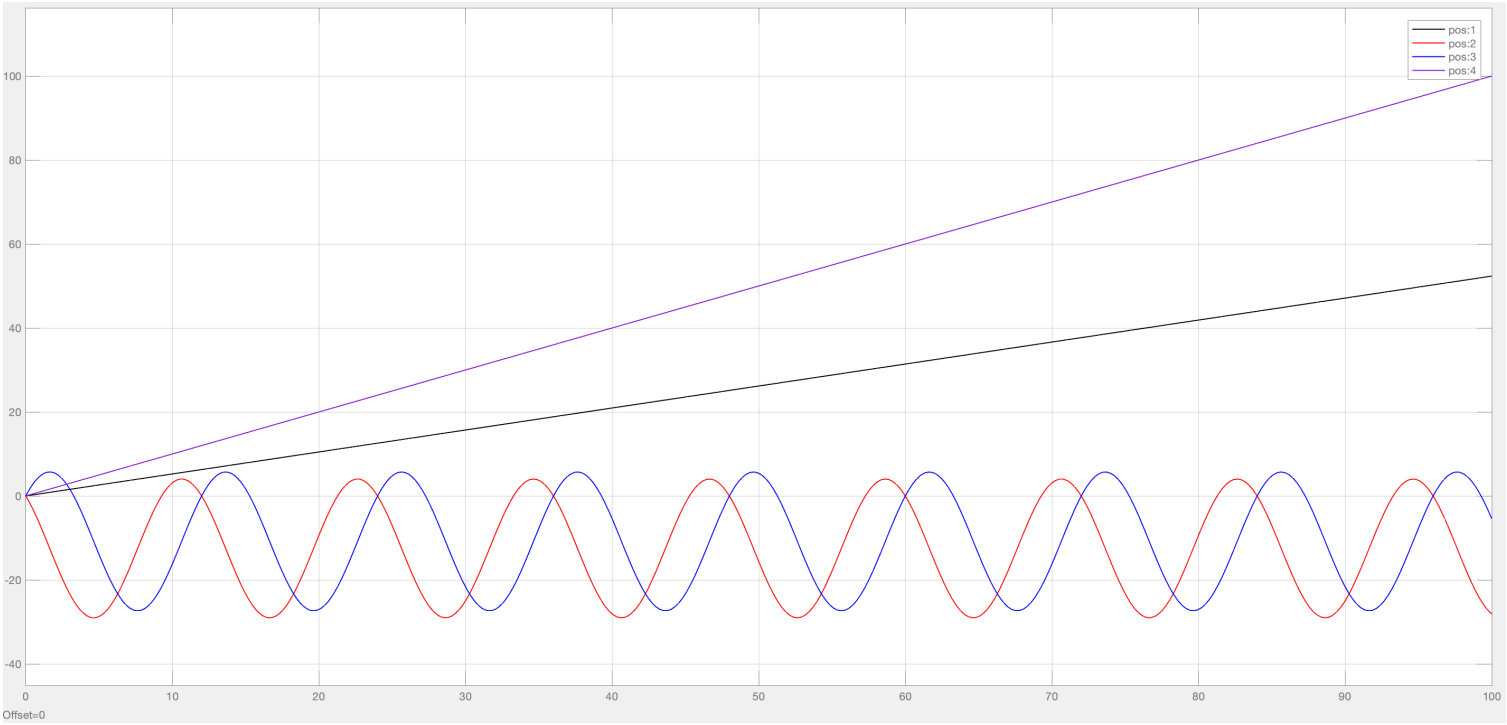


Fig 2: Simulink result in scope for yaw(black), x(red), y(blue) and z(pink)

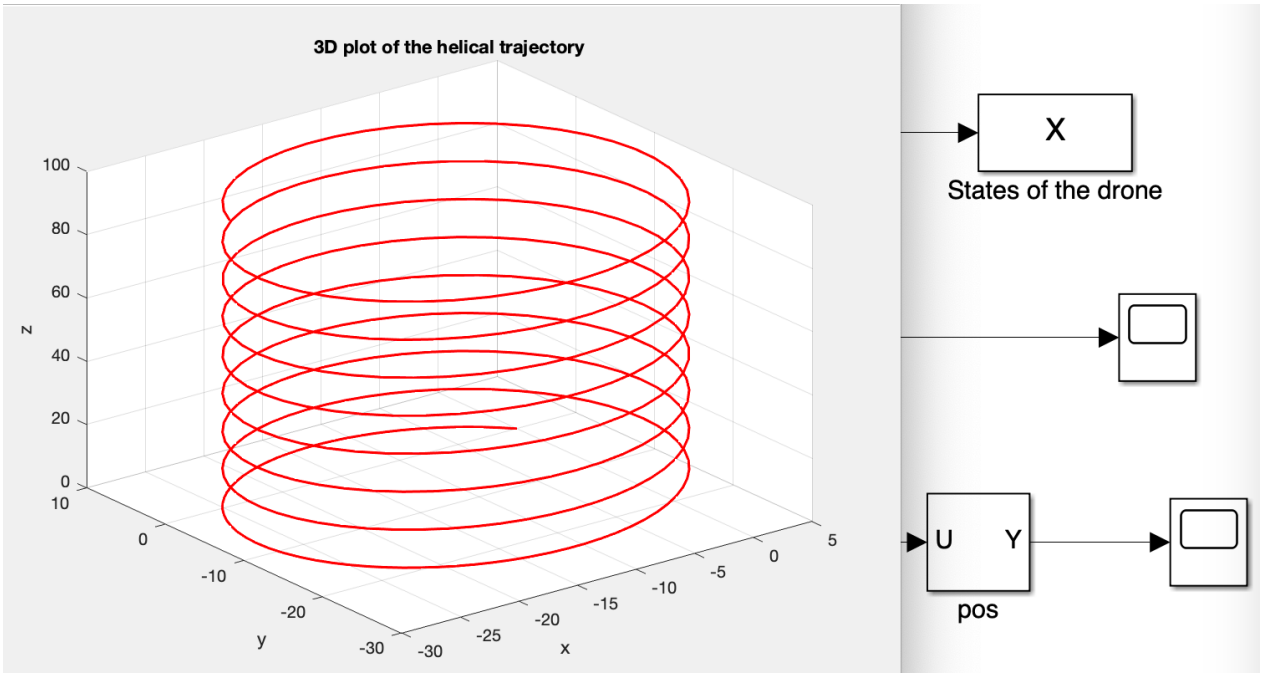


Fig 3: 3D plot of the helical trajectory for the given model

Functions used inside the block

```
function dx = drone(x, u)
    % Parameters
    g = -9.81;
    m = 0.10;
```

```
    l = 0.25;
    Ixx = 0.001;
    Iyy = 0.001;
    Izz = 0.002;
    Jr = 0.01;        % Rotor inertia

    % Coefficients from the image
    a1 = (Iyy - Izz) / Ixx;
    a2 = Jr / Ixx;
    a3 = (Izz - Ixx) / Iyy;
    a4 = Jr / Iyy;
    a5 = (Ixx - Iyy) / Izz;

    b1 = l / Ixx;
    b2 = l / Iyy;
    b3 = l / Izz;

    % State variables
    % x(1) = phi, x(2) = phi_dot, x(3) = theta, x(4) = theta_dot,
    % x(5) = psi, x(6) = psi_dot, x(7) = x, x(8) = x_dot,
    % x(9) = y, x(10) = y_dot, x(11) = z, x(12) = z_dot

    % Control inputs
    U1 = u(1); % Roll moment
    U2 = u(2); % Pitch moment
    U3 = u(3); % Yaw moment
    U4 = u(4); % Collective thrust
    Ur = u(5); % Omega_r


    % State derivatives and conditions
    dx = zeros(12,1);
    %x(1)=0; x(3)=0;
    x(4)=x(2)== 0; %x(8)=x(10)==0;
    x(6) = pi/5; x(12) = 1;

    % Angular dynamics
    dx(1) = x(2);
    dx(2) = a1 * x(4) * x(6) + a2 * x(4) * Ur + b1 * U1;
    dx(3) = x(4);
    dx(4) = a3 * x(2) * x(6) - a4 * x(2) * Ur + b2 * U2;
    dx(5) = pi/6;
    dx(6) = 0;

    % Translational dynamics
    dx(7) = x(8);
    dx(8) = (U4 / m) * (cos(x(1)) * sin(x(3)) * cos(x(5)) + sin(x(1)) *
sin(x(5)));
    dx(9) = x(10);
```

```
    dx(10) = (U4 / m) * (cos(x(1)) * sin(x(3)) * sin(x(5)) - sin(x(1)) *
cos(x(5)));
    dx(11) = x(12);
    dx(12) = -g + (U4 / m) * (cos(x(1)) * cos(x(3)));

end
```

```
function u = compute_helical_trim(x, w1, w2)

    u = zeros(5,1);
    m=0.1; g=-9.81; b=1; d = 1;

    % control inputs for the model at helix conditions
    u(5) = 2*(w2 - w1);
    u(4) = m*g/(cos(x(1))*cos(x(3))) + m*x(12);
    u(3) = 2*d*(w2^2 - w1^2);
    u(2) = 0;
    u(1) = 0;


end
```

Summary

The helical trajectory for the quadrotor was successfully simulated in **Simulink** and **MATLAB**. Both simulations showed smooth, continuous path for the helix. The quadrotor effectively followed the helical path in 3D space, by applying the conditions for helical trajectory trimming. This trajectory design is essential for real-world tasks like inspection and 3D mapping of vertical structures.