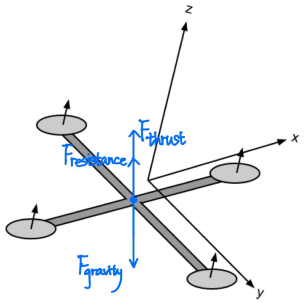

Figure 1: Force Balance

Individual Coursework 2

Huijuan Ma, zczqhm0@ucl.ac.uk, Dec. 14. 2023

1. Assume $\gamma_i = \omega_i^2$, $x_1 = \mathbf{p}$, $x_2 = \dot{\mathbf{p}}$, $x_3 = \theta$, $x_4 = \omega$, then get $\dot{x_1} = x_2$, $\dot{x_2} = \ddot{\mathbf{p}}$, $\dot{x_3} = \dot{\theta}$, $\dot{x_4} = \dot{\omega}$.
Code: set the variable Q in the 34^{th} row of $Sim\_Quadcopter\_q1.m$ to 1.1, 1.2 and 1.3 for the following three simulation scenarios respectively.
Analysis a. Equilibrium: In equilibrium with unchanged position and orientation, the quadcopter should be subjected to force balance when it is aloft with zero velocities and accelerations.

   Set $\gamma_i = \omega_i$ as 4 inputs, then iterate each state x with $\dot{\mathbf{x}}$. I set the equilibrium point with $\dot{x} = 0$, $x_1 = \mathbf{p} = [0, 0, 5]$, $x_2 = \dot{\mathbf{p}} = [0, 0, 0]$, $x_3 = \theta = [0, 0, 0]$ and $x_4 = \omega = [0, 0, 0]$. The quadcopter will be aloft with no rotating in any direction with zero vector sum of all external torques.
   To determine the input γ_i $(i = 1, 2, 3, 4)$ and according to the force balance analysis (Figure 1), we have

$$ma_z = k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) - mg - k_d v_z \tag{1}$$

   along z-axis based on $ma_z = F_{thrust} - mg + F_{resistance}$ with $F_{thrust} = k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2)$ and $F_{resistance} = -k_d v_z$. So, at the equilibrium, we have $mg = F_{thrust}$, leading to $\sum \gamma_i = \frac{mg}{k}$.
   According to the equation

$$\tau_B = \begin{bmatrix} Lk(\gamma_1 - \gamma_3) \\ Lk(\gamma_2 - \gamma_4) \\ b\left(\gamma_1 - \gamma_2 + \gamma_3 - \gamma_4\right) \end{bmatrix} \tag{2}$$

   γ_i should set as $\gamma_1 = \gamma_3$, $\gamma_2 = \gamma_4$ to ensure zero vector sum of torques so that the quadcopter will not rotate by any axis ($x_4 = \omega = [0, 0, 0]$). Therefore, set $\gamma_1 = \gamma_3 = \gamma_2 = \gamma_4 = \frac{mg}{4k}$ for the equilibrim.
   b. Free fall: If the circuit system of the quadcopter outputs no power, it will be in free fall. For z-axis, With the equation 1 and 2, we can set the inputs $\gamma_i = \omega_i^2 = 0$. In this case, there will be no thrust operated on the robot. For z-axis, the acceleration should be $a_z = -g - \frac{k_d v_z}{m}$ where $v_z < 0$.
   c. A constant altitude: Based on the setting of the previous equilibrium point, to make the quadcopter exhibit a change in rotation, I set a torque at z-axis/yaw-axis. According to the equation 2, the quadcopter should have 0 torque in Roll-axis and Pitch-axis with a non-zero torque in Yaw-axis. Therefore, I set $\gamma_1 = \gamma_3 = \frac{mg}{4k} + 0.5$, $\gamma_2 = \gamma_4 = \frac{mg}{4k} - 0.5$ which keeps $\sum \gamma_i = \frac{mg}{k}$.
   Test: For each scenario, the drone starts at the height of 5m, and then keep stationary, falling down and rotating respectively.

2. **Code**: set the variable Q in the 107^{th} row of $Sim\_Quadcopter\_q2.m$ to 1, 2 or 3 for three simulation scenarios respectively. As soon as the code starts, it will generate 5 plots for linear nonerror (figure 5), nonlinear small error (figure 1), linear small error(figure 2), nonlinear large error (figure 3) and linear large error (figure 4), and then start simulating in order, you can check these figures to see the movement. At the end, the code will plot all trajectories for system with errors.
   linearised model: The system is non-linear. We can linearlize it use Jacobians.

$$\dot{x} = \begin{bmatrix} x_{2,x} & x_{2,y} & x_{2,z} \\ x_{4,x} & x_{4,y} & \sigma_1 \\ x_{4,x} & x_{4,y} & \sigma_1 \\ \frac{\gamma_1}{4} - \frac{\gamma_3}{4} + \frac{3x_{4,y}x_{4,z}}{5} & \frac{\gamma_2}{4} - \frac{\gamma_4}{4} - \frac{3x_{4,x}x_{4,z}}{5} & \frac{\gamma_1}{2} - \frac{\gamma_2}{2} + \frac{\gamma_3}{2} - \frac{\gamma_4}{2} \end{bmatrix} \tag{3}$$

   Set $\mathbf{u} = [\gamma_1, \gamma_2, \gamma_3, \gamma_4]$, and a non-linear continuous model for the system $f(x, u) = \dot{\mathbf{x}}, y = h(x)$. To linearize the model at our operating point (equilibrium: $\delta\dot{x}_0 = f(x_0, u_0) = 0$), use Jacobians: $\dot{\delta_x} = f(x + \delta_x, u + \delta_u) \approx \frac{\partial f}{\partial x}(x_0, u_0)\,\delta_x + \frac{\partial f}{\partial u}(x_0, u_0)\,\delta_u$. At the equilibrium point, $\gamma_i = \omega_i^2 = 0$, $x_1 = \mathbf{p} = [0, 0, 0]$, $x_2 = \dot{\mathbf{p}} = [0, 0, 0]$, $x_3 = \theta = [0, 0, 0]$, $x_4 = \omega = [0, 0, 0]$. $f_1 = \dot{x_1} = x_2$: $\frac{\partial f_1}{\partial x_1} = 0$,

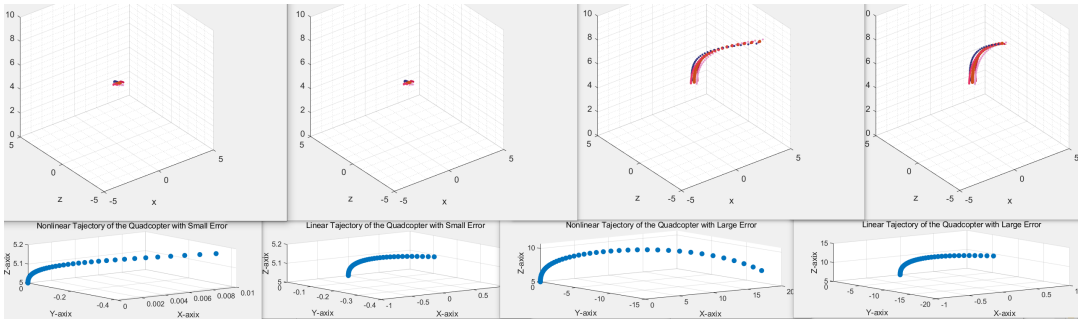

Figure 2: All Zero States Except Height As The Equilibrium Point

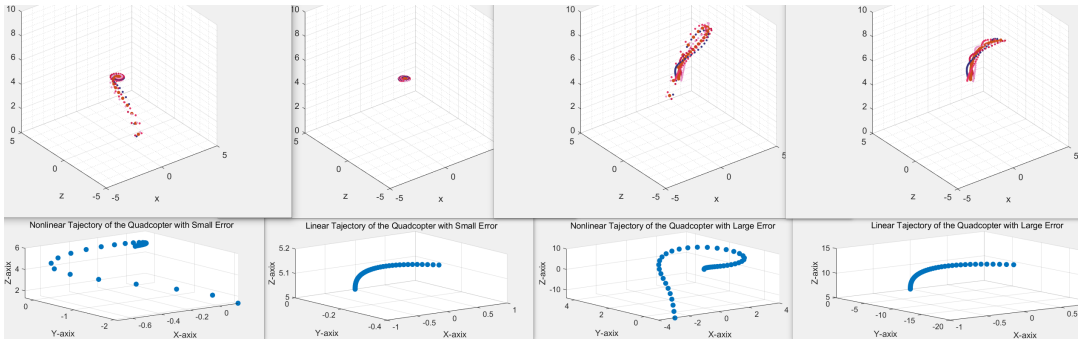

Figure 3: A Change in Rotation at a Constant Altitude

$\frac{\partial f_1}{\partial x_2} = I$, $\frac{\partial f_1}{\partial x_3} = 0$, $\frac{\partial f_1}{\partial x_4} = 0$

$f_2 = \dot{x_2} = \ddot{p}$: $\frac{\partial f_2}{\partial x_1} = 0$, $\frac{\partial f_2}{\partial x_2} = -\frac{k_d}{m} I$, $\frac{\partial f_2}{\partial x_3} = \begin{bmatrix} 0 & g & 0 \\ -g & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$, $\frac{\partial f_2}{\partial x_4} = 0$

$f_3 = \dot{x_3} = \dot{\theta}$: $\frac{\partial f_3}{\partial x_1} = \frac{\partial \dot{\theta}}{\partial p} = 0$, $\frac{\partial f_3}{\partial x_3} = \frac{\partial \dot{\theta}}{\partial \theta} = 0$, $\frac{\partial f_3}{\partial x_4} = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$f_4 = \dot{x_4} = \dot{\omega}$: $\frac{\partial f_4}{\partial x_1} = \frac{\partial \dot{\theta}}{\partial p} = 0$, $\frac{\partial f_4}{\partial x_3} = \frac{\partial \dot{\theta}}{\partial \theta} = 0$, $\frac{\partial f_4}{\partial x_4} = 0$.

So the linearized model has matrixs of state space representation in continous system with the equilibrium point $x = 0$:

$A = \begin{bmatrix} 0 & I & 0 & 0 \\ 0 & -\frac{k_d}{m} I & \begin{bmatrix} 0 & g & 0 \\ -g & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & 0 \\ 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 \end{bmatrix}$. Matrix B can also be generated in this form. Assume all states are observable (y is a 12-by-1 matrix), we can set C as a 12-by-12 matrix.

Test&Result: Set small error as 0.01 and large error as 0.5. Change an input γ_1 with these error and compare the performance of linear system and nonlinear system. If the error is small, the drone will generally leave the equilibrium point more slowly in linear system than that in nonlinear system, because linear system focus on states near equilibrium points which is less complex than that in nonlinear system, leading to less sensetive about disturbance. If the error is large, the states leave away from the equilibrium point, the drone will not fit that system any more and its physical behaviour is not correct. The performances are shown in figures 2, 4 and 3.

3. **Code**: run $Sim\_Quadcopter\_q3.m$.
Implement (Build a full-state feedback controller):
$\mathbf{x}[k+1] = (A_d - BK)\mathbf{x}[k] + (B_d K)\mathbf{r_x}$, where $\mathbf{r_x}$ is set to the desired equilibrium point.
a. Linearlization: From previous questions **Q2**, the continuous system can be linearized ($sysa$) using **Jacobian Matrix**.
Discrete the system: the LTI system should be discreted first $code : sysd = c2d(sysa, Ts, 'zoh')$ to get the discrete state space representation

$$\mathbf{x}[k+1] = A_d \mathbf{x}[k] + B_d \mathbf{u}[k] \tag{4}$$

b. Reachability: Check the reachability with $W_r = [B_d \quad A_d B_d \quad A_d A_d B_d \quad A_d A_d A_d B_d]]$. If W_r is full

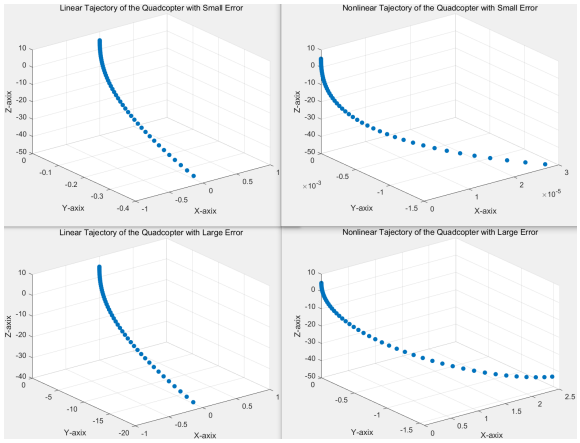

Figure 4: Free Falling

rank, the system can reach any arbitrary final state in finite time.
c. Pole Placement: Define the parameters of the controller K with **Pole Placement** to define K. Define eigenvalue/pole placement λ_i for $A - BK$ with $code : place(A_d, B_d, p)$ where p is a vector of eigenvalues/poles λ_i. λ_i shows the stability of the system. If $|\lambda_i| < 1$, the system is stable. An eigenvalue that is too small causes slow reaction, but an eigenvalue which is too close to 1 leads to unstability with disturbances. Because it is too massy to find suitable poles, I set the state/input weight matrices and use Linear Quadratic Regulator ($dlqr$ in Matlab) to generate the controller gain K.
d. Operate on the non-linear system in Q1: the full-state feedback controller is a linear combination of the errors between state variables and the desired reference $\mathbf{r_x}$, we have $\mathbf{u}[k] = -K(\mathbf{x}[k] - \mathbf{r_x})$. The input γ_i, which is related to the thrust (equation 1 and 2), can be modified based on the controller as $\Gamma = [\frac{mg}{4k} + 1; \frac{mg}{4k} - 1; \frac{mg}{4k} + 1; \frac{mg}{4k} - 1]$.
Test&Result: Discrete system sampled by $T_s = 0.1$.Collect all position points and plot it to check the circle trajactory in the figure 5.

4. **Code**: Run $Sim\_Quadcopter\_q4.m$.
(i) The quadcopter measurements come from noisy sensors that do not measure the entire state.
In this case, the matrix C is not diagnal. Depanding on existing sensors and techniques and considering $4 * 3 = 12$ states we have (position in 3D, line velocities, orientation with 3 axises and angular velocities), the next step is to define measurable and unmeasurable states.
Line velocities, Orientation and Angle velocities: **Inertial Measurement Unit (IMU)**
An IMU can be any combination of accelerometers, gyroscopes, and magnetmeters.
Line velocities can not measured directly, but can be calculated by integrating the acceleration generated by accelerometers. **Orientation** can be measured by magnetmeters. **Angle velocities** can be generated by gyroscopes [2]. In this system, the height, orientation and angle velocities are measurable, while horizontal position and are unmeasuable.

Position: GPS/LiDAR/camera/**ultrasonic sensor** [2]
GPS can meet the positioning needs of most outdoor scenes, and camera and LiDAR has the ability of simultaneous localization and mapping, which can provide more detailed and accurate distance information about location indoors and outdoors with obstacles [1]. Ultrasonic sensor is cheap and suitable for relatively close range measurements, but most of them cannot detect small distance near 0. Ultrasonic sensor and LiDAR are under consideration. However, installation of single line LiDAR is hard to make the laser perpendicular to the ground and multi-line LiDAR is too expensive and heavy. Assuming the ground is at $z = 0$ and the working field has no other obstacle, I suggest to mount the **GPS** which will not have any limitations caused by using GPS without installation method. The system also can combine the **GPS** with **IMU** to have a more accurate measurement.
In this case, I set the absolute velocity is unmeasurable, and the position, orientation and angular velocities are measurable.
(ii) Noise and Wind Disturbances: For **noise**, I use a set of measurement noise with a mean of 0 and a variance of 1. For **wind disturbances**, assume the average windward area $S = 0.1m^2$ and velocity of wind $v_{wind} = [v_{wx}, v_{wy}, v_{wz}]^T m/s$. We can have: $v_{wdrone} = [v_{dronex} + v_{wx}, v_{droney} + v_{wy}, v_{dronez} + v_{wz}]$. Air density $\rho = 1.225kgm^2$, Coefficient of air resistance $c = 0.08$, the rotation matrix from body frame to inetria frame ${}^{\mathbf{E}}\mathbf{R}_{\mathbf{B}}$. To simulate drone performance in a more general way, assume it is in a nature winds, which have the characteristics of being sudden, persistent, periodic, and uncertain. So I combine **Constant wind speed model**, **Gradient wind speed model** (a linear piecewise function) and **Random speed model** to model them. $v_{wind} = v_{w,constant} + v_{w,gradient} + v_{w,random}$.

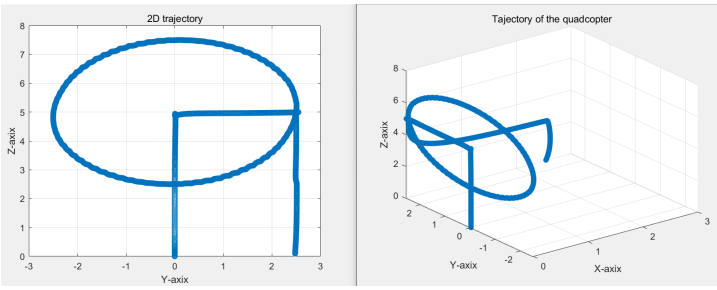

Figure 5: Trajactory (system with a full state feedback controller)

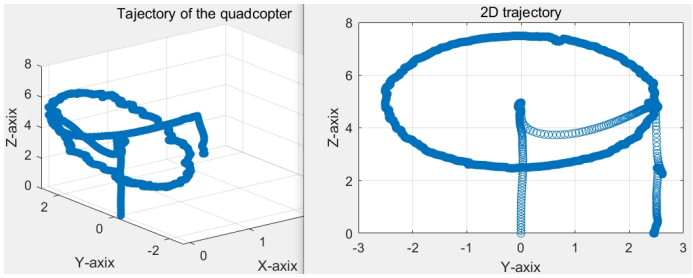

Figure 6: Trajectory

Assume $v_{w,constant} = 0.02m/s$, $v_{w,gradient} = \begin{cases} 0 & 0 \leqslant t \leqslant t_0 \text{ or } t > t_2 \\ \dfrac{v_{\max 1}(t-t_0)}{t_1-t_0} & t_0 < t < t_1 \\ \dfrac{v_{\max 1}(t-t_1)}{t_1-t_2} & t_1 \leqslant t \leqslant t_2 \end{cases}$ and $v_{w,random} = v_a[Rand(1) - 0.5]\sin(2\pi t + \omega)$. The drone's velocities $\mathbf{v_{drone}} = \mathbf{v} + \mathbf{v_{wind}}$. Baed on kinetic equation $ma = F_{thrust} + F_{gravity} + F_{drag} + F_{wind}$, $F_{thrust} = sum(inputs)$, $F_{gravity} = mg$, $F_{drag} = -k\mathbf{v}$ ($\mathbf{v}$ is absolute velocities vector), $F_{wind} = \frac{1}{2}\rho(v + v_{wind})^2 AC_D$. Therefore, $a = [0 \;\; 0 \;\; \frac{sum(inputs)}{m} + \frac{-k\mathbf{v}}{m} + \frac{1}{2m}\rho(v + v_{wind})^2 AC_D]^T$ at the equilibrium point. Assume $C_D = 0.08$ This wind model will directly change the force balance, leading to a change in acceleration, then effect other states.
(iii) State Observer: Assume $x[n+1] = Ax[n] + Bu[n] + w_k[n]$, $y[n] = Cx[n] + v_k[n]$. Use Kalman Filter update the state observer gain for each measurement with following functions:

$$\begin{aligned} &P_{k-1,k} = AP_{k-1}A^T + Q \quad Update\ filter. \\ &L_k = P_{k-1,k}C^T(HP_{k-1,k}H^T + R)^{-1} \quad Calculate\ Kalman\ Gain. \\ &\hat{x}[k+1] = x[k+1] + L_k(y[k] - Cx[k+1] \quad Update\ states. \\ &P_k = (I - L_kC)P_{k-1,k} \quad Update\ convarience\ matrix. \end{aligned} \tag{5}$$

Use L_k to continuously predict and correct through prediction and observation updates to derive a group of optimal drone state variables with the state feedback $\mathbf{u}[k] = -K(\mathbf{x}[k] - \mathbf{r}_x$ and a 9-by-12 matrix as C.
Test&Result: For each measurement, add a random noise to y as real y and used it to compensate for the current state estimation $L(y - y_{measure})$. Update next $y_{measure}$ with current state estimation $y_{newmeasure} = C_d x_{newmeasure}$ with C_d from the discrete system sampled by $T_s = 0.1$. The resulting path exhibits significant disturbances, but reach the target point and basically complete the task, showing in the figure 6.

References

[1] Ahmad Riyad Firdaus et al. "Indoor Localization Using Positional Tracking Feature of Stereo Camera on Quadcopter". eng. In: *Electronics (Basel)* 12.2 (2023), pp. 406–. ISSN: 2079-9292.

[2] Qilong Yuan and I-Ming Chen. "Localization and velocity tracking of human via 3 IMU sensors". In: *Sensors and Actuators A: Physical* 212 (2014), pp. 25–33.

END OF COURSEWORK