ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CƠ KHÍ

---------∞❁∞---------

BÁO CÁO CUỐI KÌ HỌC PHẦN ĐIỀU KHIỂN LOGIC MỜ

ĐỀ TÀI: THIẾT KẾ BỘ ĐIỀU KHIỂN FUZZY PID CHO HỆ THỐNG QUADCOPTER DRONE

Giảng viên hướng dẫn: TS. Phạm Anh Đức

Nhóm sinh viên thực hiện:

Lê Ngọc Hân - 101210045

Nguyễn Đức Cường - 101210255

Lớp: 21CDT1

Da Nang, 2025

MỤC LỤC

PHẦN I: TÓM TẮT PHẦN XÂY DỰNG MÔ HÌNH TOÁN HỌC CHO DRONE QUADCOPTER 3

1. Hệ quy chiếu 3
2. Newton – Euler equations 4
3. Các hiệu ứng khí cản động học ảnh hưởng tới hệ 4
4. Xây dựng các công thức toán học cho các bộ điều khiển, thiết kế bộ điều khiển PID 5

PHẦN II: XÂY DỰNG MÔ HÌNH TRONG MATLAB SIMULINK VÀ MÔ PHỎNG ĐỘ ĐÁP ỨNG CỦA BỘ ĐIỀU KHIỂN PID 7

1. Xây dựng mô hình Simulink 7
2. Tiến hành mô phỏng, đánh giá đáp ứng khi sử dụng bộ điều khiển PID 10

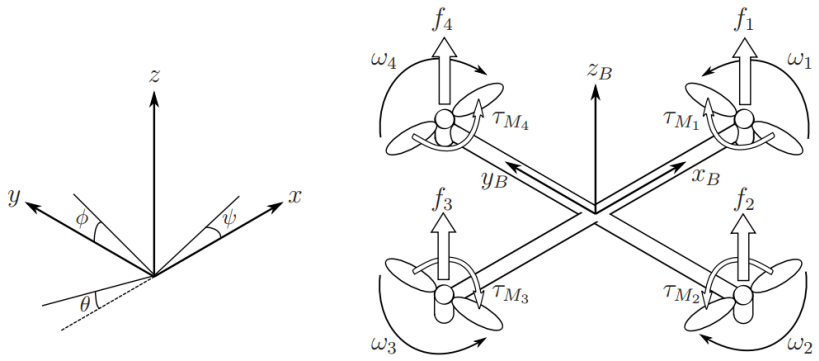
PHẦN III: XÂY DỰNG BỘ ĐIỀU KHIỂN FUZZY PID VÀ SO SÁNH VỚI PID 17

1. Tổng quan về Hệ mờ 17
2. Thiết kế hệ thống mờ điều chỉnh thông số PID 17
3. Xây dựng bộ điều khiển fuzzy PID cho hệ quadcopter drone 19
4. Mô hình và mô phỏng trong MATLAB Simulink 25

PHẦN IV: ĐÁNH GIÁ CHẤT LƯỢNG HỆ THỐNG VÀ SO SÁNH GIỮA PID VÀ FUZZY PID 30

PHẦN I: TÓM TẮT PHẦN XÂY DỰNG MÔ HÌNH TOÁN HỌC CHO DRONE QUADCOPTER

1. Hệ quy chiếu



Hình I.1. Hệ quy chiếu quán tính và hệ quy chiếu gắn với thân quadcopter

$$\xi = (x, y, z) \epsilon R^3$$

$$\eta = (\Phi, \theta, \psi) \epsilon R^3$$

Trong đó $\xi=(x;y;z)$ biểu diễn vectơ vị trí của khối tâm của máy bay bốn cánh so với hệ quy chiếu quán tính cố định. $\eta=(\phi;\theta;\psi)$ ký hiệu động lực học quay của máy bay bốn cánh được cho bởi các góc Euler ϕ, θ và ψ, hay còn gọi là vectơ vị trí góc.

Gốc của hệ quy chiếu gắn với thân nằm ở khối tâm của máy bay bốn cánh. Trong hệ quy chiếu gắn với thân, vận tốc tuyến tính được xác định bởi V_B và vận tốc góc được xác định bởi v.

$$V_B = \begin{bmatrix} v_{x,B} \\ v_{y,B} \\ v_{z,B} \end{bmatrix}, v = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Ma trận quán tính I:

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

Vận tốc góc của rô-tơ thứ i, ký hiệu là ωi, tạo ra lực fi theo hướng trục của rotor. Vận tốc góc và gia tốc của rotor cũng tạo ra mô-men xoắn τMi quanh trục của rô-tơ:

$$f_i = k\omega_i^2, \tau_{M_i} = b\omega_i^2 + I_M\dot{\omega}_\text{i}$$

trong đó k là hằng số lực nâng, b là hằng số lực cản và IM là mô-men quán tính của rô-tơ. Thông thường, ảnh hưởng của ω˙i được coi là nhỏ và do đó nó bị bỏ qua.

Tổng hợp lực của các rô-tơ tạo ra lực đẩy T theo hướng trục z của thân. Mô-men xoắn τB bao gồm các mô-men xoắn τϕ, τθ và τψ theo hướng của các góc tương ứng của hệ tọa độ gắn với thân.

$$T = \sum_{i=1}^{4} f_i = k\sum_{i=1}^{4}\omega_i^2, T^B = \begin{bmatrix}0\\0\\T\end{bmatrix}$$

$$\tau_B = \begin{bmatrix}\tau_\varphi\\\tau_\theta\\\tau_\psi\end{bmatrix} = \begin{bmatrix}lk(-\omega_2^2 + \omega_4^2)\\lk\left(-\omega_\text{1}^2 + \omega_3^2\right)\\\sum_{i=\text{1}}^{4}\tau_{M_i}\end{bmatrix}$$

trong đó l là khoảng cách giữa rô-tơ và tâm khối của quadcopter

2. Newton – Euler equations

Quadcopter drone được coi là một vật rắn, và do đó các phương trình Newton-Euler có thể được sử dụng để mô tả động lực học của nó.

Trong hệ tọa độ gắn với thân, gia tốc góc nhân với ma trận quán tính (Iν˙), cộng với các mô-men do lực quán tính ly tâm (ν×(Iν)) và các mô-men do hiệu ứng con quay hồi chuyển (Γ), bằng với mô-men ngoại lực tác dụng (τ).

$$I\dot{\nu} + \text{ν}/\times (I\nu) + \Gamma = \tau,$$

$$\begin{bmatrix}\dot{p}\\\dot{q}\\\dot{r}\end{bmatrix} = \begin{bmatrix}(I_{yy} - I_{zz})qr/I_{xx}\\(I_{zz} - I_{xx})pr/I_{yy}\\(I_{xx} - I_{yy})pq/I_{zz}\end{bmatrix} - I_r\begin{bmatrix}q/I_{xx}\\-p/I_{yy}\\0\end{bmatrix}\omega_\Gamma + \begin{bmatrix}\tau_\varphi/I_{xx}\\\tau_\theta/I_{yy}\\\tau_\psi/I_{zz}\end{bmatrix}$$

Trong đó: $\omega_\Gamma = \omega_1 - \omega_2 + \omega_3 - \omega_4$

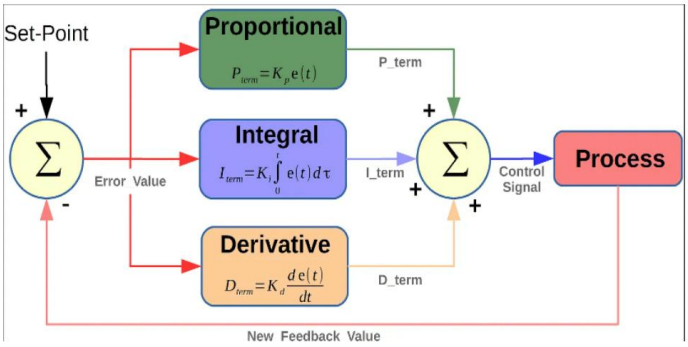
3. Các hiệu ứng khí cản động học ảnh hưởng tới hệ

Mô hình trên là một sự đơn giản hóa của các tương tác động lực học phức tạp. Để tăng cường tính thực tế cho hành vi của máy bay bốn cánh, lực cản do sức cản của không khí được đưa vào.

$$\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix} = -g\begin{bmatrix}0\\0\\1\end{bmatrix} + \frac{T}{m}\begin{bmatrix}C_\psi S_\theta C_\varphi + S_\psi S_\varphi\\ S_\psi S_\theta C_\varphi - C_\psi S_\varphi\\ C_\theta C_\varphi\end{bmatrix} - \frac{1}{m}\begin{bmatrix}A_x & 0 & 0\\ 0 & A_y & 0\\ 0 & 0 & A_z\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix}$$

trong đó Ax, Ay và Az là các hệ số lực cản cho vận tốc theo các hướng tương ứng của hệ tọa độ quán tính.

4. Xây dựng các công thức toán học cho các bộ điều khiển, thiết kế bộ điều khiển PID



Hình I.2. Sơ đồ mô tả bộ điều khiển PID

$$e(t) = x_d(t) - x(t)\,,$$

$$u(t) = K_P e(t) + K_I \int_0^t e(\tau)d\tau + K_D \frac{\mathrm{d}e(t)}{\mathrm{d}t}$$

Trong đó u là tín hiệu điều khiển, e là sai lệch giữa trạng thái mong muốn xd và trạng thái hiện tại x, còn Kp, Ki và Kd là các tham số cho thành phần tỉ lệ (Proportional), tích phân (Integral) và đạo hàm (Derivative) của bộ điều khiển PID.

Trong quadcopter drone, có sáu trạng thái, vị trí (x,y,z) và các góc (ϕ,θ,ψ), nhưng chỉ có bốn tín hiệu điều khiển, là vận tốc góc của bốn rô-tơ (ω1,ω2,ω3,ω4). Các tương tác giữa các trạng thái và lực đẩy tổng T và các mô-men xoắn τ=[τϕ,τθ,τψ]T được tạo ra bởi các rô-tơ có thể thấy được từ động lực học của quadcopter được định nghĩa bởi các phương trình dưới đây.

Lực đẩy tổng T ảnh hưởng đến gia tốc theo hướng trục z và giữ máy bay bốn cánh trên không. Mô-men xoắn τϕ có ảnh hưởng đến gia tốc của góc ϕ, mô-men xoắn τθ ảnh hưởng đến gia tốc của góc θ, và mô-men xoắn τψ ảnh hưởng đến gia tốc của góc ψ.

$$T = \left(g + K_{z,D}(z_d - z) + K_{z,P}(z_d - z)\right)\frac{m}{C_\varphi C_\theta}, (1)$$

$$\tau_\varphi = (K_{\varphi,D}\left(\dot{\varphi}_d - \dot{\varphi} + K_{\varphi,P}(\varphi_d - \varphi)\right)I_{xx}, \quad (2)$$

$$\tau_\theta = (K_{\theta,D}\left(\dot{\theta}_d - \dot{\theta} + K_{\theta,P}(\theta_d - \theta)\right)I_{yy}, \quad (3)$$

$$\tau_\psi = (K_{\psi,D}\left(\dot{\psi}_d - \dot{\psi} + K_{\psi,P}(\psi_d - \psi)\right)I_{zz}, \quad (4)$$

Vận tốc góc chính xác của các rô-tơ ω_i có thể được tính toán như sau:

$$\omega_1^2 = \frac{T}{4k} - \frac{\tau_\theta}{2kl} - \frac{\tau_\psi}{4b} \quad (5)$$

$$\omega_2^2 = \frac{T}{4k} - \frac{\tau_\varphi}{2kl} + \frac{\tau_\psi}{4b} \quad (6)$$

$$\omega_3^2 = \frac{T}{4k} + \frac{\tau_\theta}{2kl} - \frac{\tau_\psi}{4b} \quad (7)$$

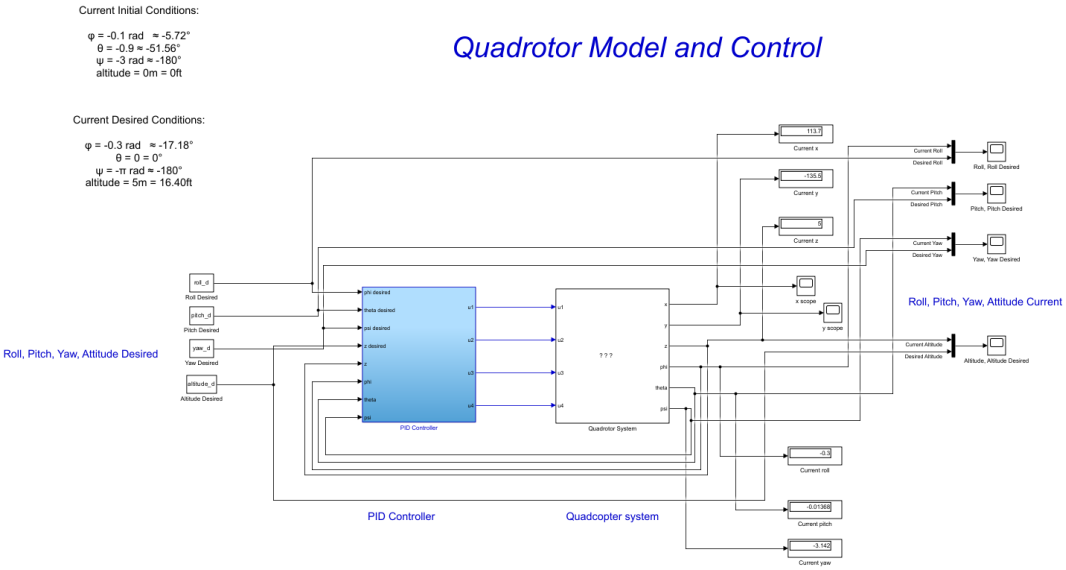
$$\omega_4^2 = \frac{T}{4k} + \frac{T_\beta}{2kl} + \frac{\tau_\psi}{4b} \quad (8)$$

PHẦN II: XÂY DỰNG MÔ HÌNH TRONG MATLAB SIMULINK VÀ MÔ PHỎNG ĐỘ ĐÁP ỨNG CỦA BỘ ĐIỀU KHIỂN PID

1. Xây dựng mô hình Simulink

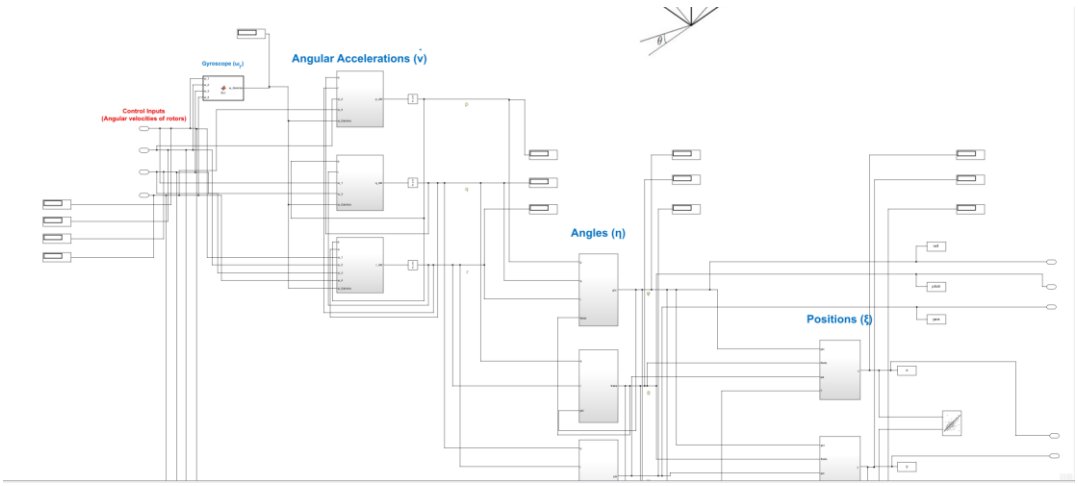
Dựa trên các công thức đã trình bày ở phần I, ta xây dựng mô hình điều khiển quadcopter trong MATLAB Simulink như sau:

Với 2 khối Subsystem là khối Controller và Quadcopter system

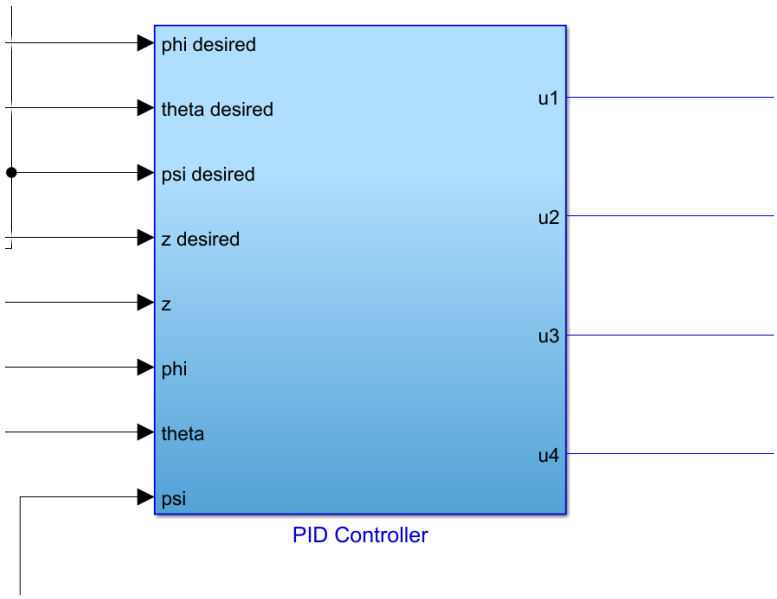


*Hình II.1.K*hối Controller và Quadcopter system

Dựa vào các công thức toán học ở Phần I, ta xây dựng khối Quadcopter system, đầu ra là *x, y, z*, và các góc *phi, theta, psi* và đầu vào là tốc độ của 4 động cơ *w1, w2, w3, w4*

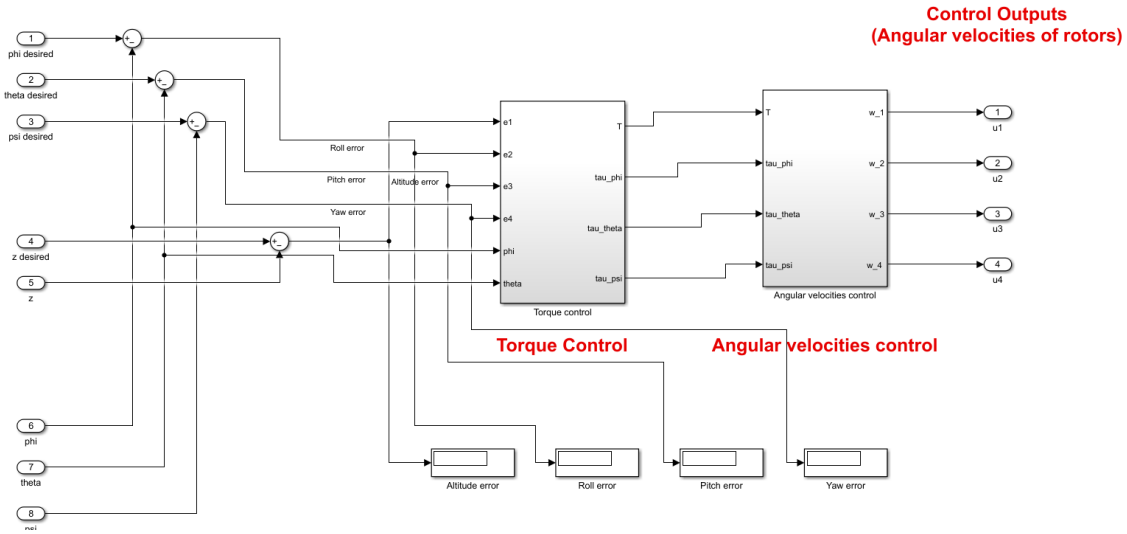


Hình II.2.Bên trong khối Quadcoptersystem



Hình II.3.Khối PID Controller

Bên trong khối Controller, khối này sẽ tính sai số giữa đầu vào và đầu ra của 4 thông số roll, pitch, yaw, attitude, sử dụng bộ điều khiển PID để tính toán các momen xoắn mong muốn, từ các giá trị momen xoắn đó ta tính ra tốc độ mà các động cơ cần đạt được, các phương trình đã được mô tả ở phần I bài báo cáo

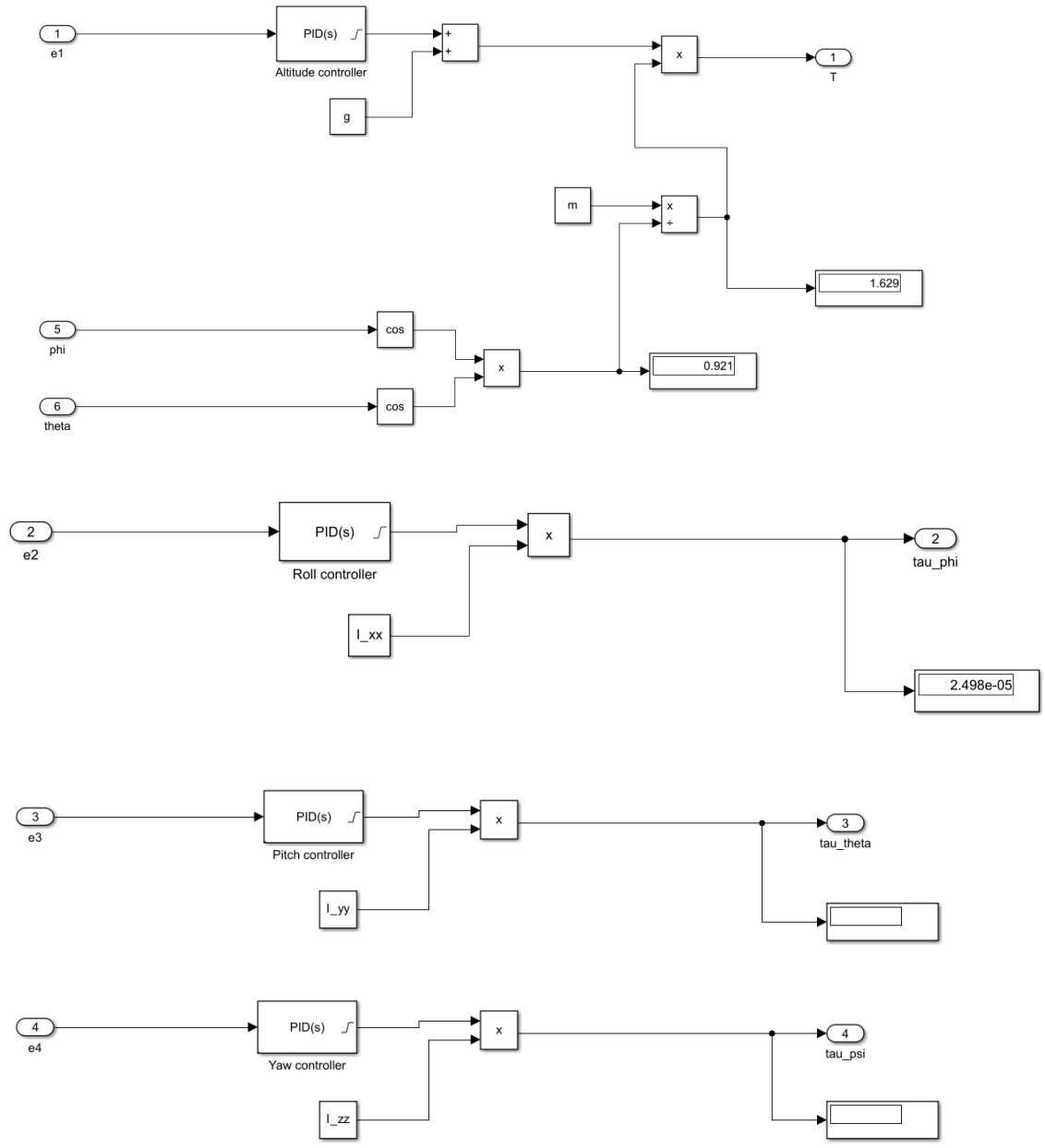


*Hình II.3.K*hối Controller

Bên trong khối Torque Control sử dụng bộ điều khiển PID, dựa vào sai số và công thức *(1), (2), (3), (4) ở trang 6* biểu diễn mối quan hệ giữa sai số và momen xoắn để điều khiển đầu ra là các giá trị momen xoắn tương ứng

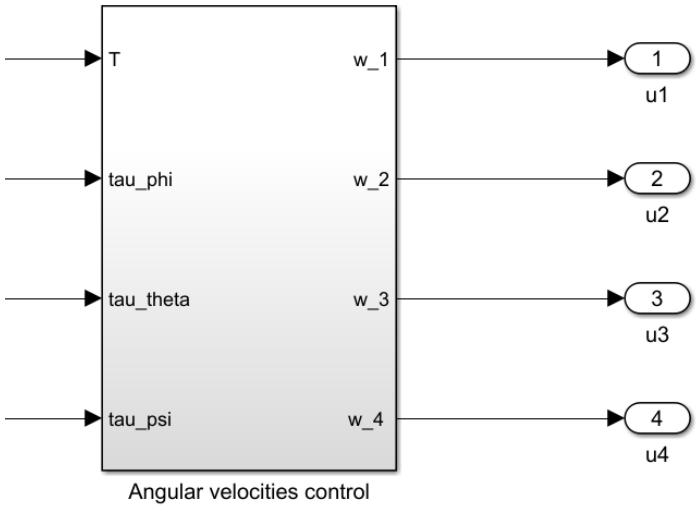
Input: Sai số $e_1, e_2, e_3, e_4, phi, theta$

Output: Momen xoắn $T, \tau_\varphi, \tau_\theta, \tau_\psi$



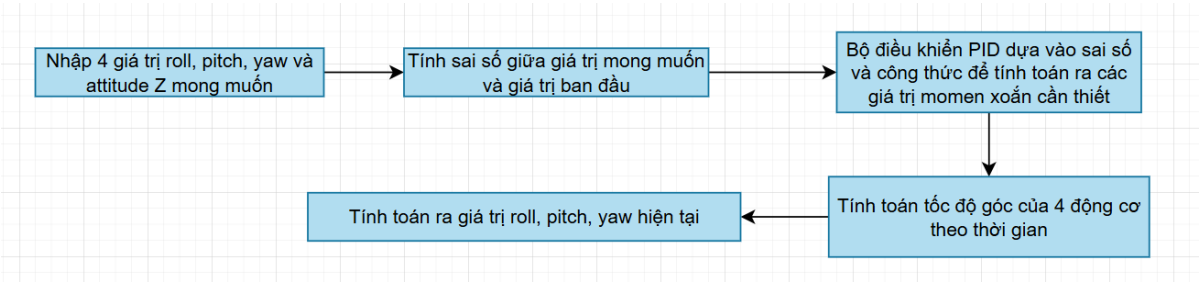
*Hình II.4.K*hối PID

Dựa vào công thức *(5), (6), (7), (8) ở trang 6*, từ các giá trị momen xoắn ta tính được tốc độ cần cấp cho 4 động cơ, tính toán ra các góc roll, pitch, yaw, attitude Z, phản hồi về để tính sai số tạo thành hệ vòng kín



*Hình II.5.K*hối Angular velocities

Tóm tắt lại thuật toán điều khiển:



Hình II.6.Sơ đồ bộ điều khiển cho hệ quadcopter drone

2. Tiến hành mô phỏng, đánh giá đáp ứng khi sử dụng bộ điều khiển PID

Giá trị các đại lượng ở bảng sau:

Tham số	Giá trị	Đơn vị
g	9.81	m/s^2
m	0.468	kg
l	0.225	m
k	$2.980.10^{-6}$	kg m^2
b	$1.140.10^{-7}$	
I_M	$3.357.10^{-5}$	
I_{xx}	$4.856.10^{-3}$	kg m^2
I_{yy}	$4.856.10^{-3}$	kg m^2
I_{zz}	$8.801.10^{-3}$	kg m^2
A_x	0.25	kg/s
A_y	0.25	kg/s
A_z	0.25	kg/s

Cho giá trị ban đầu và giá trị mong muốn của roll, pitch, yaw, attitude

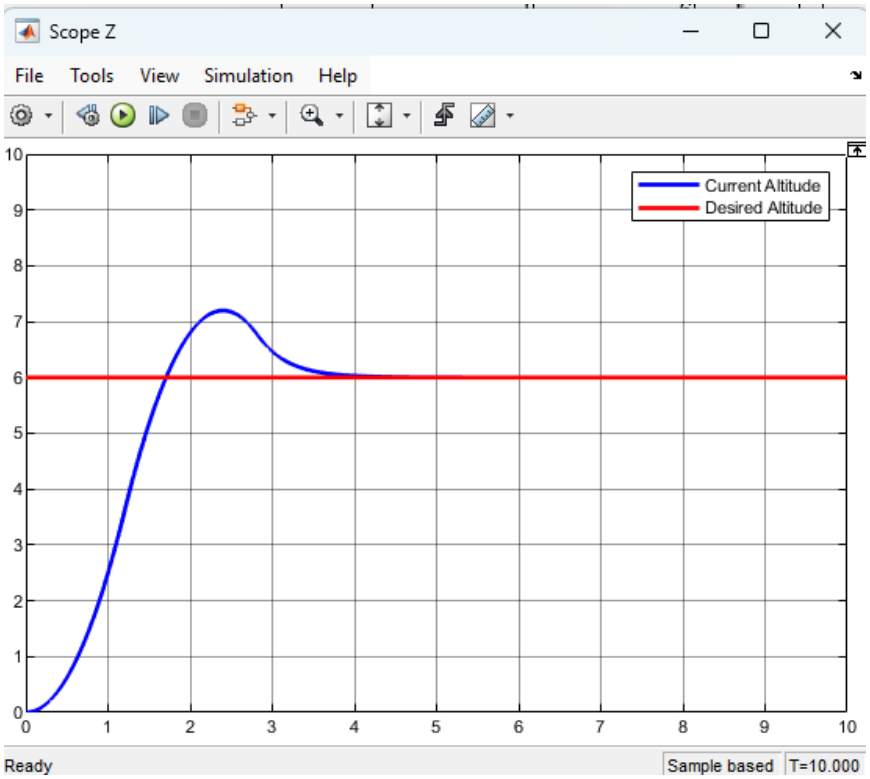
```
% Initial configuration
roll_i = -0.1;   % rad
pitch_i = -0.8;  % rad
yaw_i = -4;      % rad
altitude_i = 0;  % meters
% Desired configuration
roll_d = -0.4;   % rad
pitch_d = 0;     % rad
yaw_d = -pi;     % rad
altitude_d = 6;  % meters
```

Nhóm đã tiến hành tìm các hệ số PID sao cho hệ thống đáp ứng tốt bằng phương pháp thông thường (dò bằng tay)

2.1.Độ cao Z (Attitude)

Bảng II.1. Hệ số PID cho điều khiển độ cao Z

P	57
I	0.01
D	23



Hình II.7. Biểu đồ mô phỏng độ cao Z sử dụng bộ điều khiển PID

Đánh giá chất lượng của hệ thống sử dụng PID điều khiển độ cao Z:

Thời gian lên (Rise time): 1-2 giây

Độ vọt lố: 10% - 15%

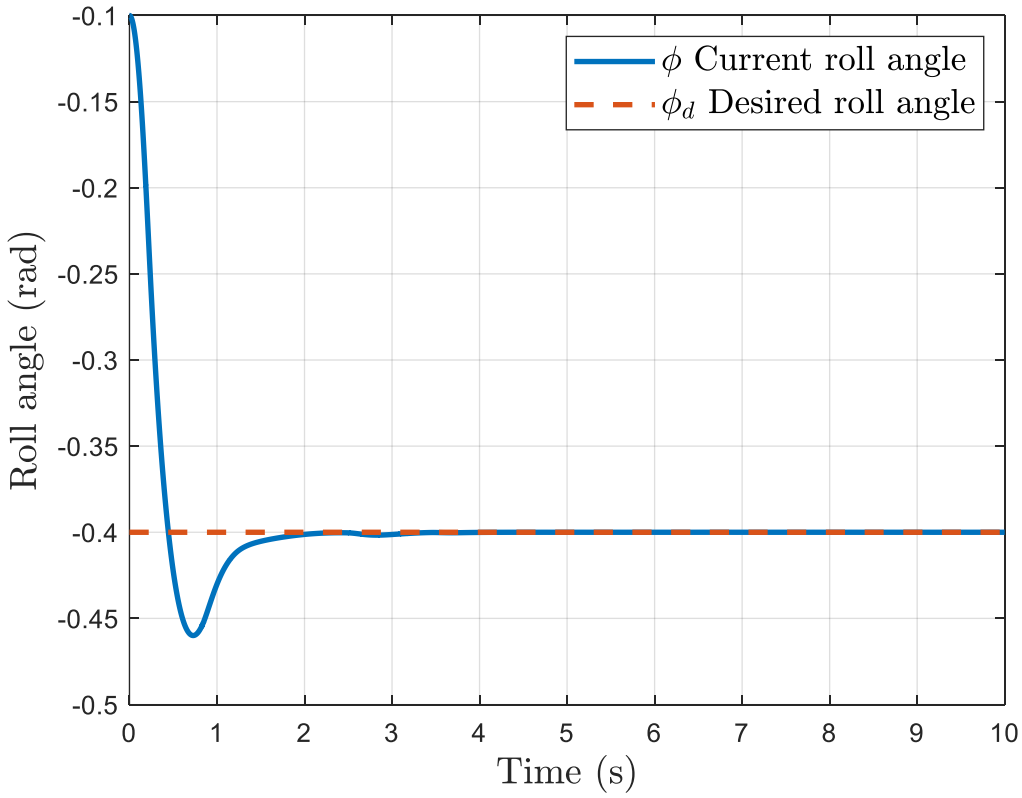
Thời gian để trạng thái đạt được trạng thái ổn định: 4 giây

Sai số xác lập: 0

2.2.Góc Roll

Bảng II.2. Hệ số PID cho điều khiển góc Roll

P	55
I	0.01
D	14.5



Hình II.8. Biểu đồ mô phỏng góc Roll sử dụng bộ điều khiển PID

Đánh giá chất lượng của hệ thống sử dụng PID điều khiển góc roll:

Thời gian lên (Rise time): 0.287 giây

Độ vọt lố: 19.97%

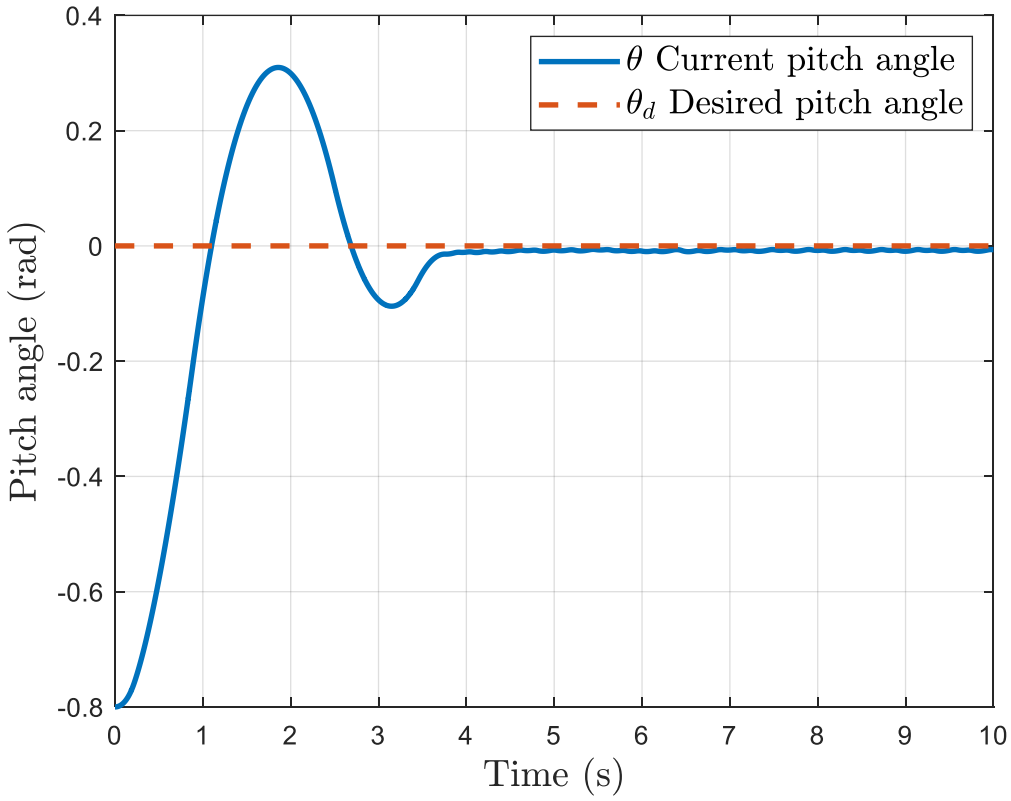
Thời gian để trạng thái đạt được trạng thái ổn định: 2 giây

Sai số xác lập: 0

2.3. Góc Pitch

Bảng II.3. Hệ số PID cho điều khiển góc Pitch

P	55
I	0.01
D	11



Hình II.9. Biểu đồ mô phỏng góc Pitch sử dụng bộ điều khiển PID

Đánh giá chất lượng của hệ thống sử dụng PID điều khiển góc Pitch:

Thời gian lên (Rise time): 0.715 giây

Độ vọt lố: 38.7%

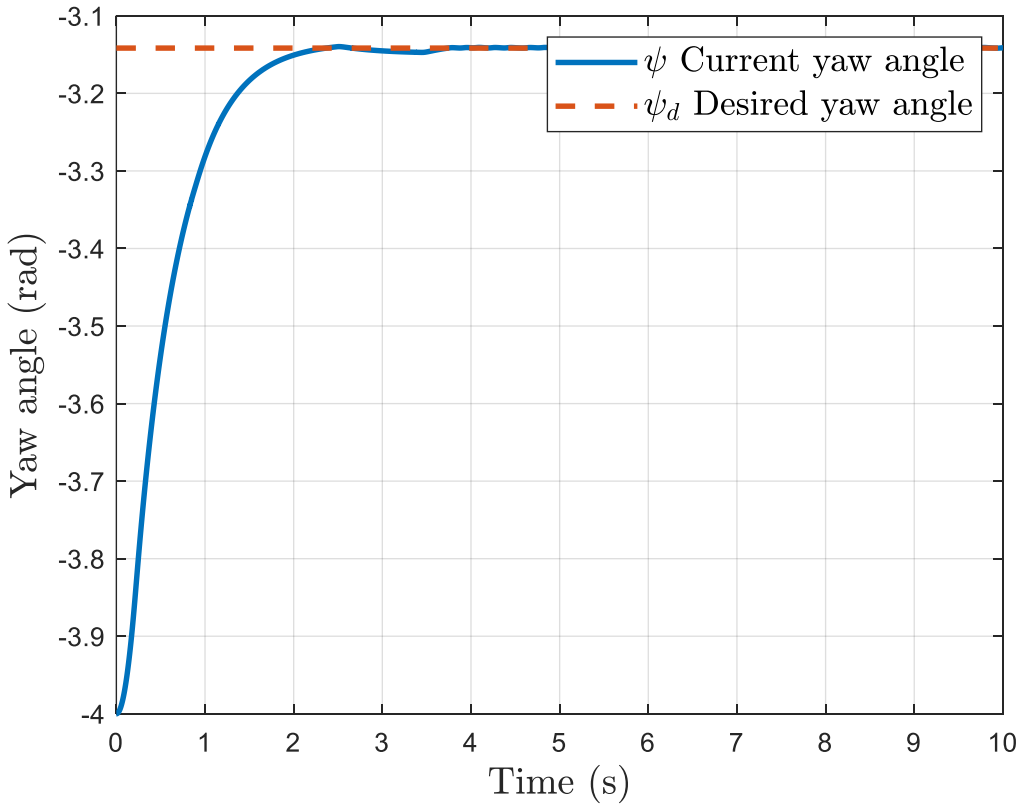
Thời gian để trạng thái đạt được trạng thái ổn định: 3.986 giây

Sai số xác lập: 0

2.4.Góc Yaw

Bảng II.4. Hệ số PID cho điều khiển góc Yaw

P	53
I	0.04
D	26.2



Hình II.10. Biểu đồ mô phỏng góc Yaw sử dụng bộ điều khiển PID

Đánh giá chất lượng của hệ thống sử dụng PID điều khiển góc Yaw:

Thời gian lên (Rise time): 1.05 giây

Độ vọt lố: 0 %

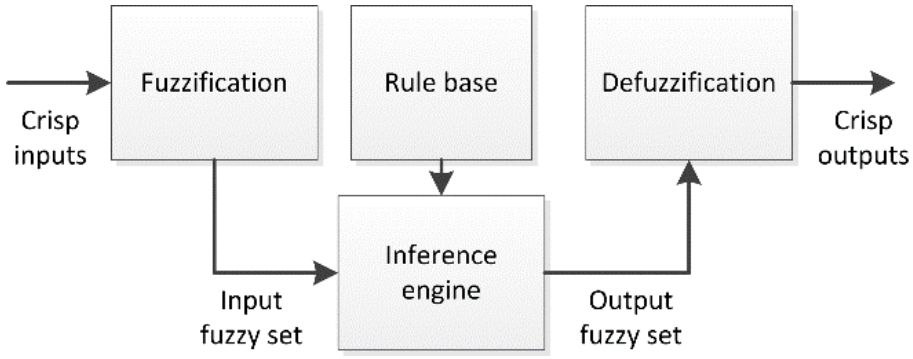
Thời gian để trạng thái đạt được trạng thái ổn định: 2.313 giây

Sai số xác lập: 0

PHẦN III: XÂY DỰNG BỘ ĐIỀU KHIỂN FUZZY PID VÀ SO SÁNH VỚI PID

1. Tổng quan về Hệ mờ

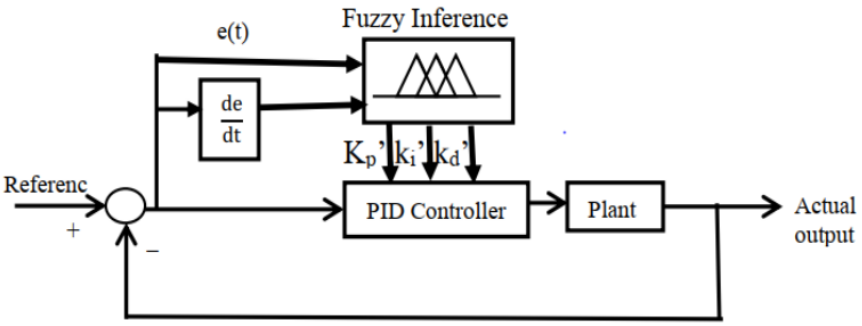
Kể từ khi Zadeh lần đầu tiên giới thiệu lý thuyết tập mờ và Mamdani đã áp dụng nó để thay thế các toán tử trong các ứng dụng điều khiển công nghiệp, nhiều ứng dụng Logic Mờ mới được phát triển đang xuất hiện hàng ngày. Tất cả các ứng dụng rộng rãi này có thể được phân loại thành hai lĩnh vực riêng biệt: điều khiển và hệ chuyên gia. Cả hai lĩnh vực đều áp dụng suy luận và lập luận gần đúng bằng cách sử dụng cơ sở luật với sự hỗ trợ của các hàm thuộc trên các tập logic mờ [0, 1], trái ngược với các tập logic cổ điển {0, 1}. Trong các ứng dụng điều khiển, hai công cụ suy luận ngôn ngữ mờ được chấp nhận rộng rãi là phương pháp Mamdani và Takagi-Sugeno. Công trình này xem xét phương pháp thiết kế bộ điều khiển Logic Mờ (FL) dựa trên Mamdani (Mamdani 1981, Sugeno 1985).



Hình III.1. Sơ đồ khối của hệ mờ

2. Thiết kế hệ thống mờ điều chỉnh thông số PID

Thiết kế hệ thống mờ để điều chỉnh độ lợi PID là phương pháp phổ biến để cải thiện hiệu suất điều khiển, đặc biệt là khi xử lý các hệ thống phi tuyến tính, thay đổi theo thời gian hoặc không chắc chắn. Ý tưởng là sử dụng hệ thống suy luận mờ (FIS) để điều chỉnh độ lớn các tỷ lệ (Kp), tích phân (Ki) và đạo hàm (Kd) của bộ điều khiển PID theo thời gian thực, dựa trên giá hiện tại của sai số và sự thay đổi của nó.

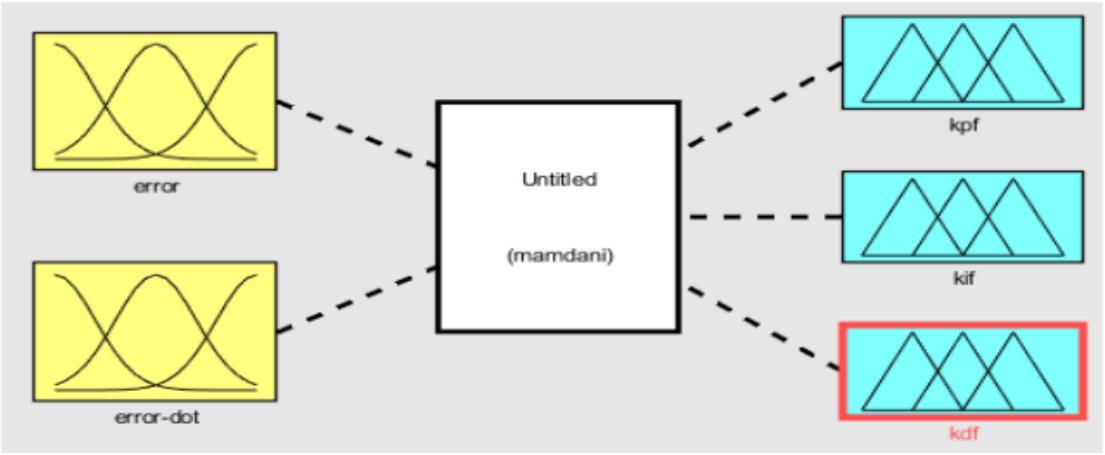


Hình III.2. Sơ đồ khối của hệ thống mờ dùng để hiệu chỉnh thông số của bộ PID

Hệ thống mờ có hai đầu vào (e và Δe) và cho ra ba đầu ra (k'_p, k'_i, k'_d). Sơ đồ khối này chỉ thể hiện điều khiển một bậc tự do của quadrotor. Bằng cách sử dụng cùng phương pháp này, hai bộ điều khiển khác cũng được thiết kế để điều khiển tư thế (phi và theta) và hướng (psi) của quadrotor.

Trong đó k_pf, k_if và k_df là các hệ số của bộ điều khiển PID mờ tương ứng với thành phần tỉ lệ, tích phân và vi phân. Ban đầu, các hệ số PID có thể được tính toán từ hệ tuyến tính hóa bằng cách sử dụng công cụ điều chỉnh trực tuyến (online tuner) trong Matlab/Simulink. Dải ổn định của các hệ số này cũng được xác định cho từng giá trị.

Sai số (error) và đạo hàm của sai số (tốc độ thay đổi của sai số) được cung cấp làm đầu vào cho bộ điều khiển logic mờ. Sau đó, các đầu vào này được phân tích trong hệ thống logic mờ bằng logic "nếu – thì" (if-then), và kết quả là các giá trị Kp, Ki và Kd được xuất ra từ các bộ điều khiển.



Hình III.3. Cấu trúc cơ bản của 1 bộ điều khiển mờ

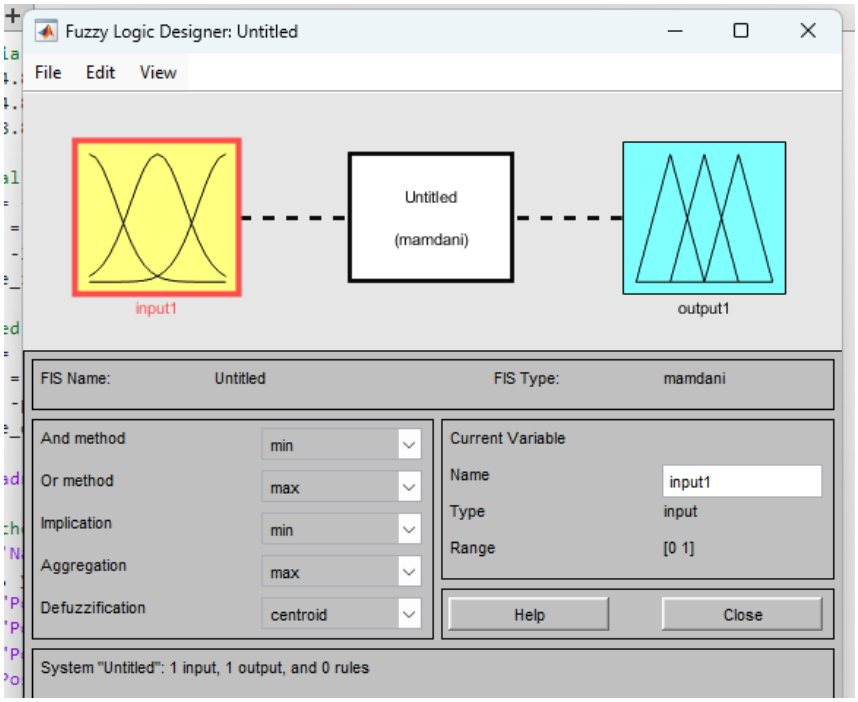
3. Xây dựng bộ điều khiển fuzzy PID cho hệ quadcopter drone

Xây dựng bộ điều khiển fuzzy PID cho hệ quadcopter drone (3 góc roll pitch yaw) có đầu ra là 3 hệ số Kp, Ki, Kd để điều khiển và tính toán được các giá trị momen xoắn cần để drone đạt giá trị roll, pitch, yaw & attitude mong muốn

Các biến ngôn ngữ được chọn cho 2 giá trị đầu vào "e" (sai số) và "Δe" (đạo hàm của sai số) gồm bảy giá trị mờ: NB, NM, NS, Z, PS, PM, PB, lần lượt biểu thị cho Rất Âm (Negative Big), Âm Vừa (Negative Medium), Âm Nhỏ (Negative Small), Không (Zero), Dương Nhỏ (Positive Small), Dương Vừa (Positive Medium) và Dương Lớn (Positive Big).

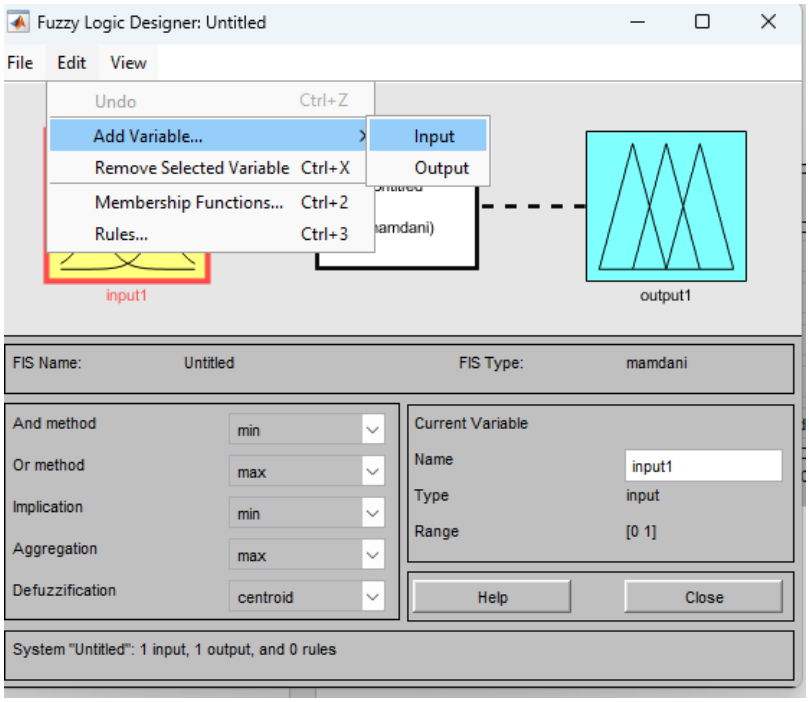
Đối với các giá trị đầu ra, ba giá trị mờ được chọn là: N, Z, P, lần lượt biểu thị cho Nhỏ (Small), Trung bình (Medium) và Lớn (Big).

Bước 1: Đầu tiên, gõ *fuzzy* ở Command Window của MATLAB, màn hình sẽ hiện ra như *Hình III.4:*



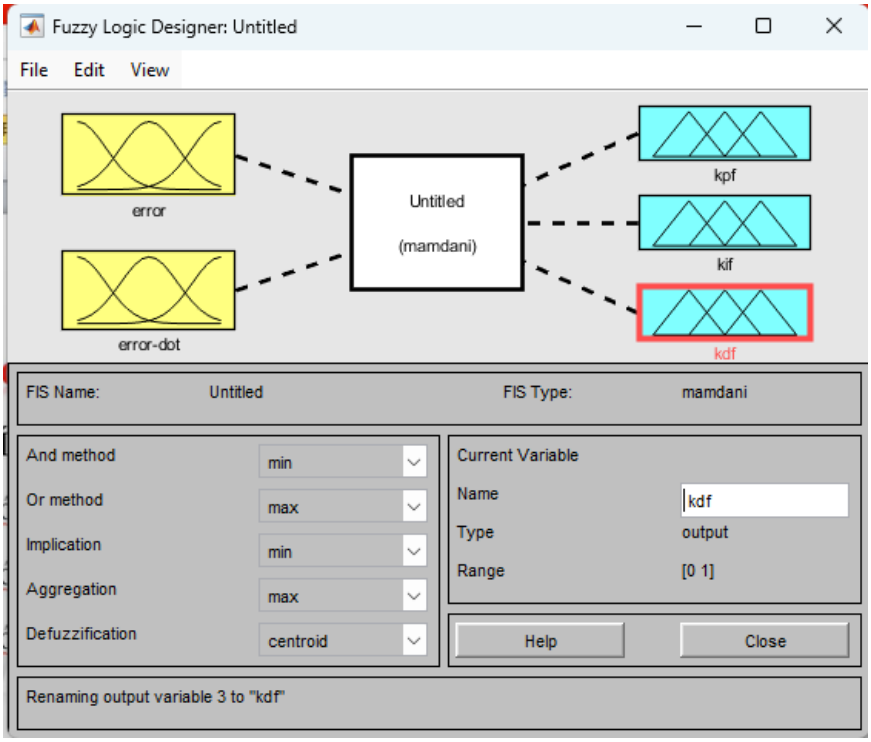
Hình III.4. Thiết lập bộ điều khiển mờ trong MATLAB

Bước 2: Thêm đầu vào và đầu ra, đây là hệ MIMO, 2 vào (e & de) 3 ra (K_{pf}, K_{if}, K_{df}), hệ thống sử dụng phương pháp suy luận mờ mamdani



Hình III.5. Thêm các biến bộ điều khiển mờ trong MATLAB

Sau khi thêm các biến:



Hình III.6. Thêm các biến bộ điều khiển mờ trong MATLAB

Bước 3:

Vì ở đây ta thiết kế bộ điều khiển Fuzzy PID cho 3 góc roll, pitch, yaw nên dải giá trị của 2 tập mờ đầu vào (e & de) được chọn trong khoảng [-1 đến 1] và thêm các biến ngôn ngữ NB, NM, NS, Z, PS, PM, PB. Hai đầu vào sử dụng các hàm thành viên và biến ngôn ngữ như nhau

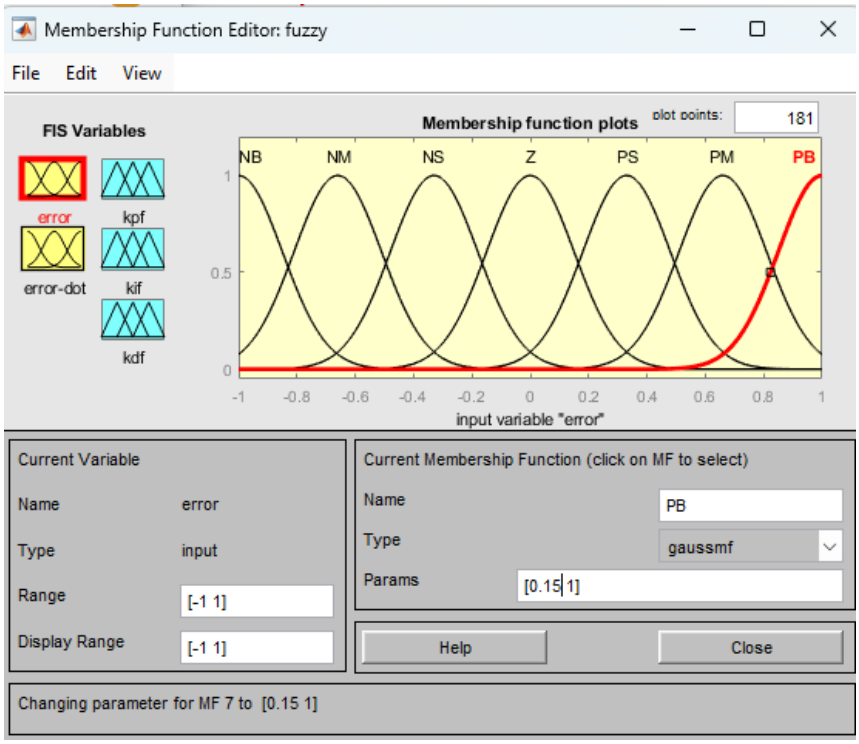
Sử dụng hàm guassian cho các hàm liên thuộc:

$$\mu(x) = exp\left(-\frac{(x - c)^2}{2\sigma^2}\right)$$

Trong đó:

c: điểm mà tại đó giá trị của hàm $\mu(x)$ đạt cực đại

σ: độ dốc, độ rộng của đường cong



Hình III.7. Thêm các biến ngôn ngữ và các hàm liên thuộc cho 2 đầu vào error và error - dot

Đối với 3 đầu ra K_{pf}, K_{if}, K_{df} của bộ điều khiển. Các biến ngôn ngữ N, Z, P được sử dụng cho cả 3 đầu ra. Các thông số cài đặt được thể hiện trong 3 hình *Hình III.8, Hình III.9, Hình III.10*

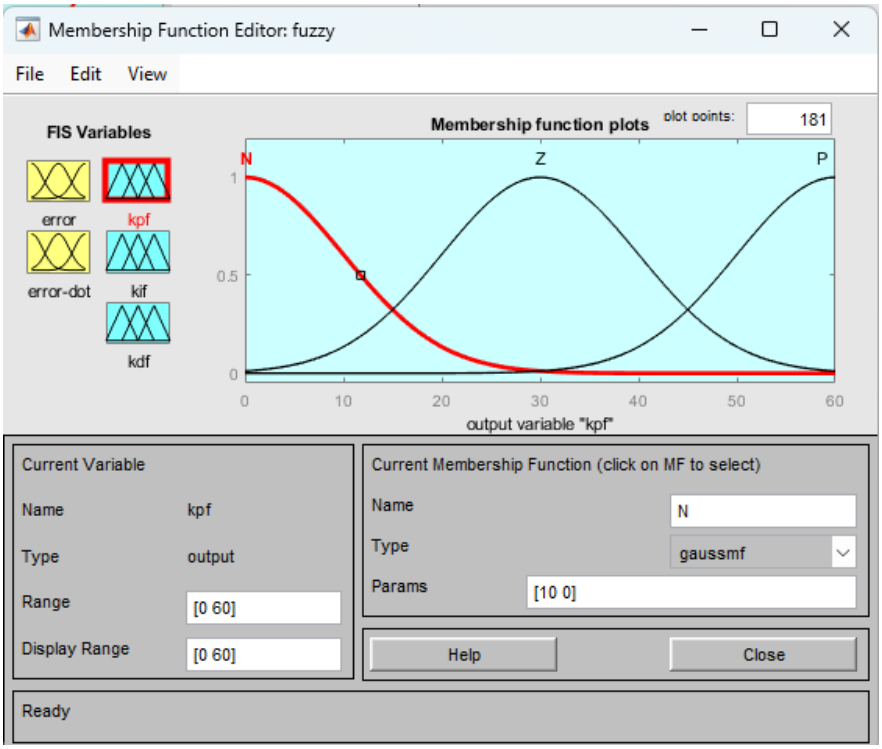
Ở phần II, bộ điều khiển PID đã được chạy thành công với các giá trị Ki, Kp, Kd được hiệu chỉnh bằng tay sao cho ra được đáp ứng tốt nhất ở đầu ra. Với các giá trị đó, ta ước tính được miền của giá trị Kp, Ki, Kd ở bộ điều khiển mờ

K_{pf} được chọn trong khoảng [0 60]

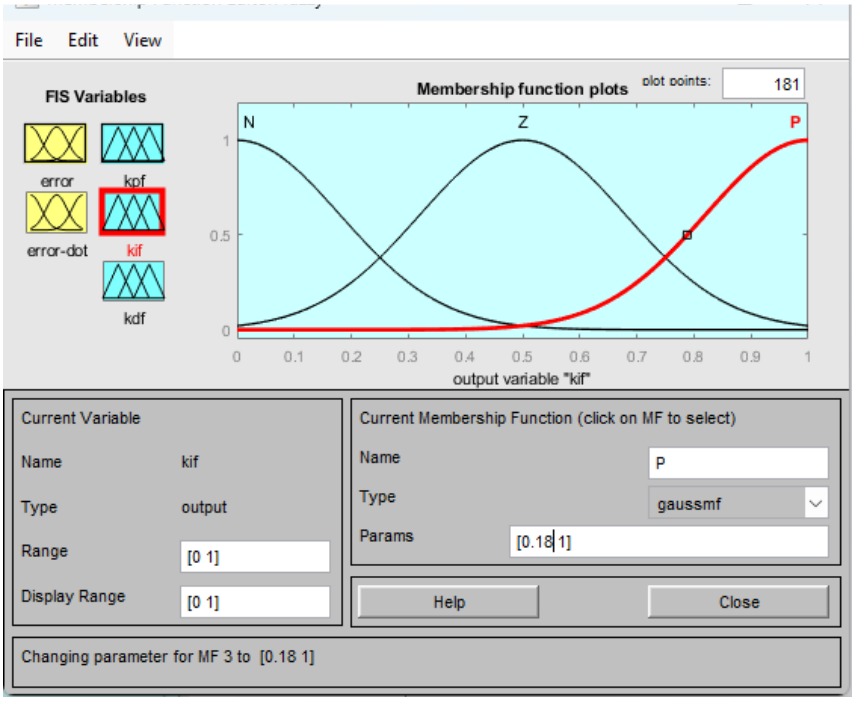
K_{if} được chọn trong khoảng [0 1]

K_{df} được chọn trong khoảng [0 20]

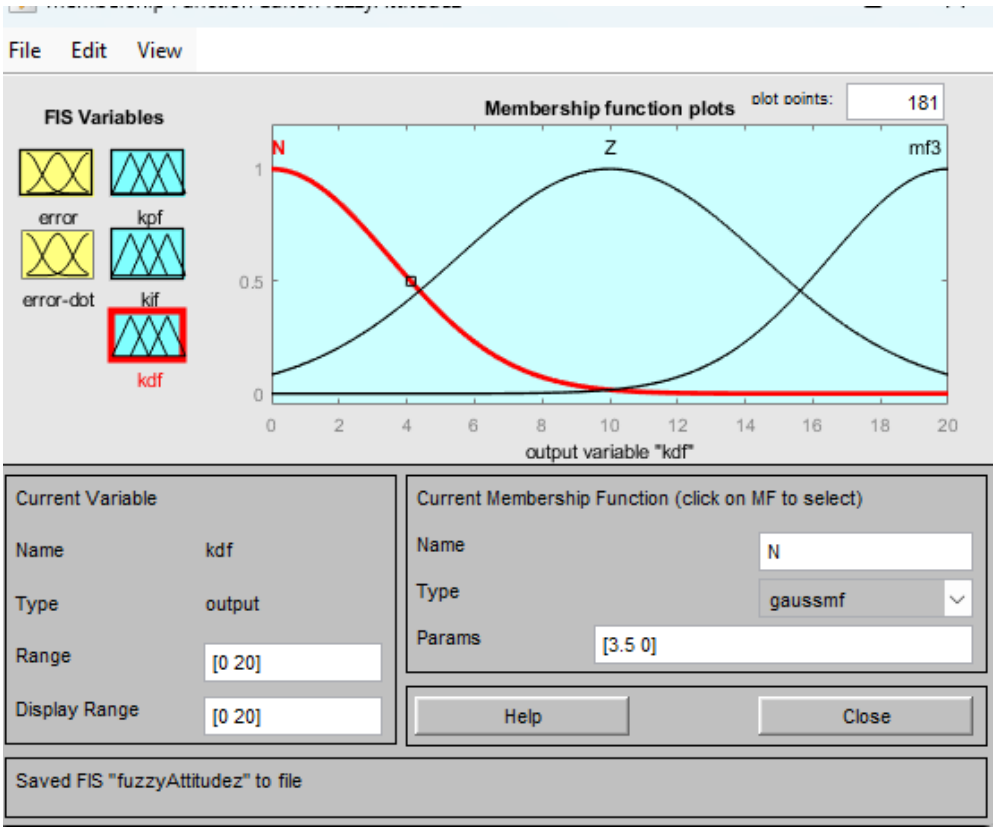
Sử dụng hàm guassian cho các hàm liên thuộc



Hình III.8. Thêm các biến ngôn ngữ và các hàm liên thuộc cho đầu ra Kpf



Hình III.9. Thêm các biến ngôn ngữ và các hàm liên thuộc cho đầu ra Kif



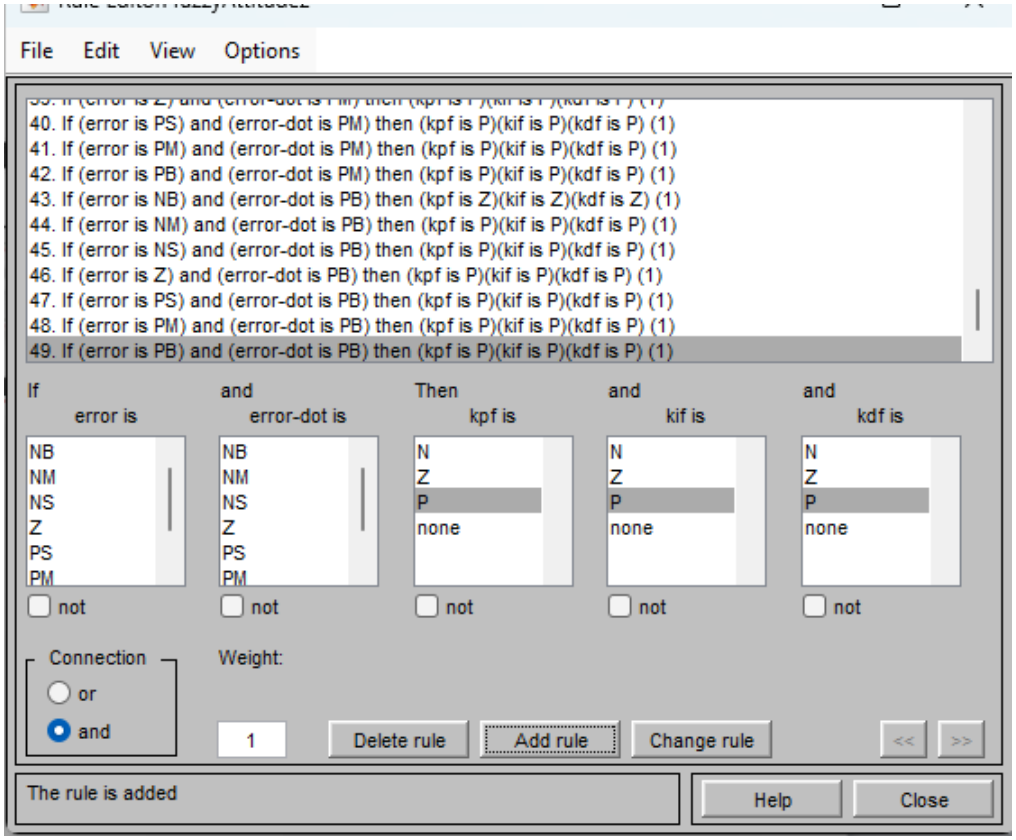
Hình III.10. Thêm các biến ngôn ngữ và các hàm liên thuộc cho đầu ra Kdf

Bước 4: Xây dựng các luật hợp thành, ta có 49 luật hợp thành được thể hiện ở *Hình III.11*

e(t)/Δe(t)	NB	NM	NS	Z	PS	PM	PB
NB	N	N	N	N	N	N	Z
NM	N	N	N	N	N	Z	P
NS	N	N	N	N	Z	P	P
Z	N	N	N	Z	P	P	P
PS	N	N	Z	P	P	P	P
PM	N	Z	P	P	P	P	P
PB	Z	P	P	P	P	P	P

Hình III.11. Bảng luật hợp thành

Thêm 49 luật hợp thành vào hệ thống fuzzy PID



Hình III.12. Thiết lập các luật hợp thành

Từ đây ta có đầu ra của bộ fuzzy PID được mô tả như sau:

$$u(t) = K_{Pf}e(t) + K_{If}\int_0^t e(\tau)d\tau + K_{Df}\frac{\mathrm{d}e(t)}{\mathrm{d}t}$$

4. Mô hình và mô phỏng trong MATLAB Simulink

Sau khi đã tạo được bộ điều khiển fuzzy PID, ta thêm vào mô hình để tiến hành mô phỏng

Mô hình được tạo với mục đích điều khiển momen xoắn $\tau_\varphi, \tau_\theta, \tau_\psi$, T , từ đó tính ra tốc độ động cơ cần cấp vào để đạt được đầu ra roll, pitch, yaw mong muốn

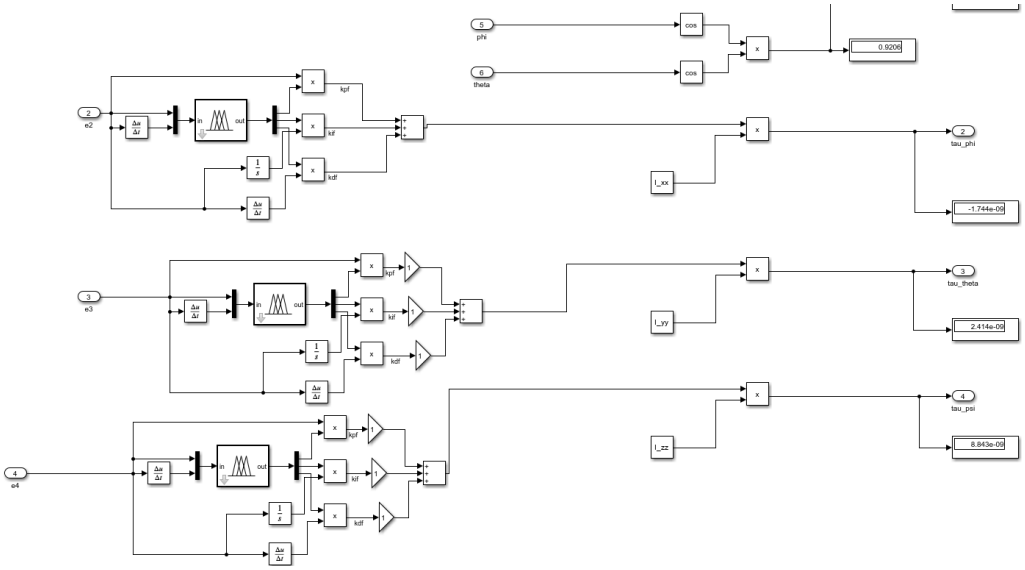
Các giá trị đầu ra K_p, K_i, K_d , được sử dụng để tính toán ra momen xoắn như các công thức dưới đây, đã được mô tả ở phần I

$$T = \left(g + K_{z,D}(z_d - z) + K_{z,P}(z_d - z)\right)\frac{m}{C_\varphi C_\theta},$$

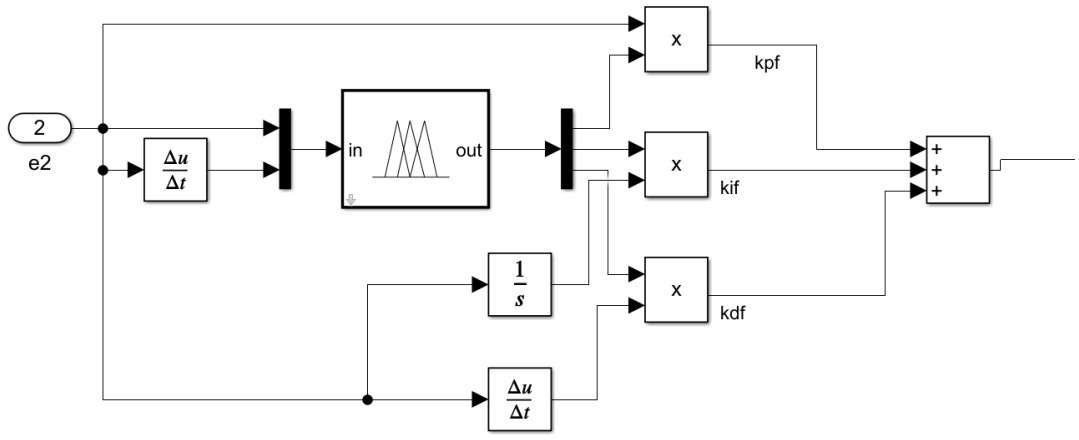
$$\tau_\varphi = (K_{\varphi,D}\left(\dot{\varphi}_d - \dot{\varphi} + K_{\varphi,P}(\varphi_d - \varphi)\right)I_{xx},$$

$$\tau_\theta = (K_{\theta,D}\left(\dot{\theta}_d - \dot{\theta} + K_{\theta,P}(\theta_d - \theta)\right) I_{yy},$$

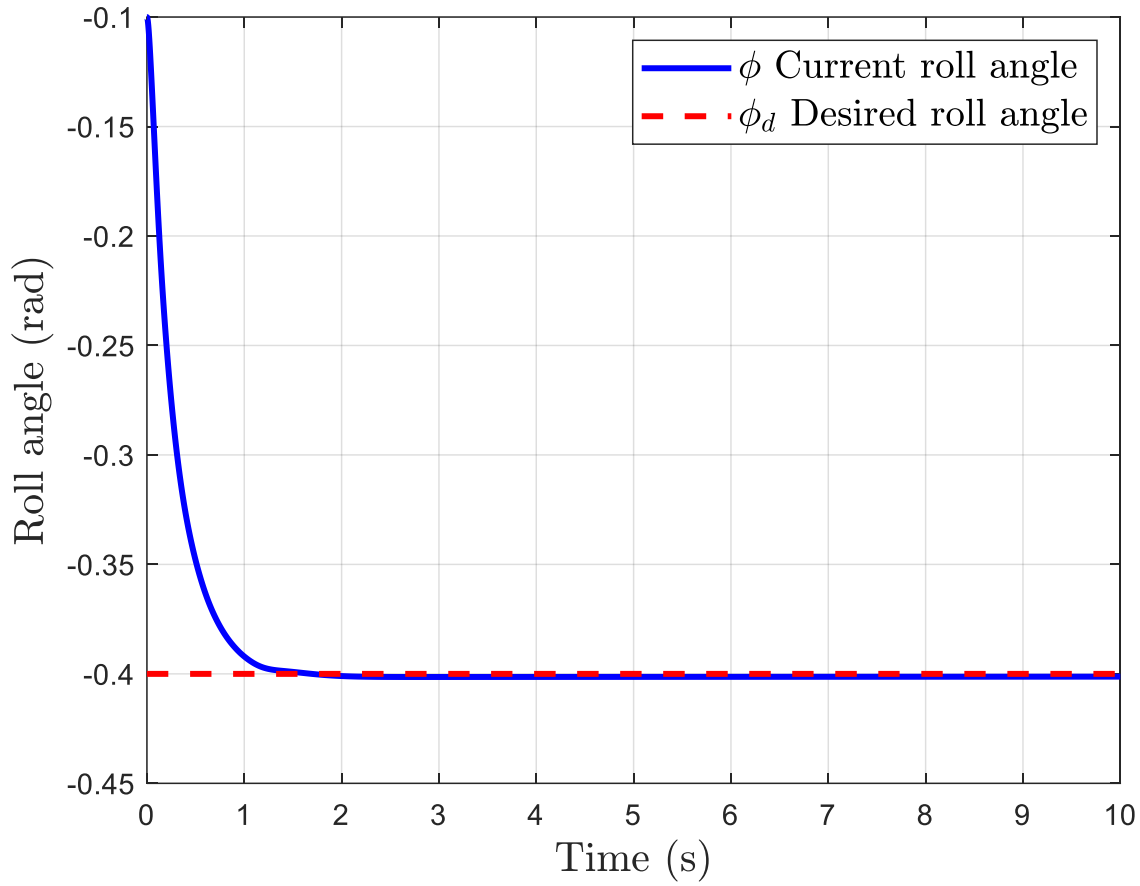
$$\tau_\psi = (K_{\psi,D}\left(\dot{\psi}_d - \dot{\psi} + K_{\psi,P}(\psi_d - \psi)\right) I_{zz},$$



Hình III.13. Mô hình điều khiển fuzzy PID roll pitch yaw cho quadcopter drone



Hình III.14. Chi tiết về mô hình fuzzy PID điều khiển góc roll



Hình III.15.Điều khiển góc roll mong muốn sử dụng fuzzy PID

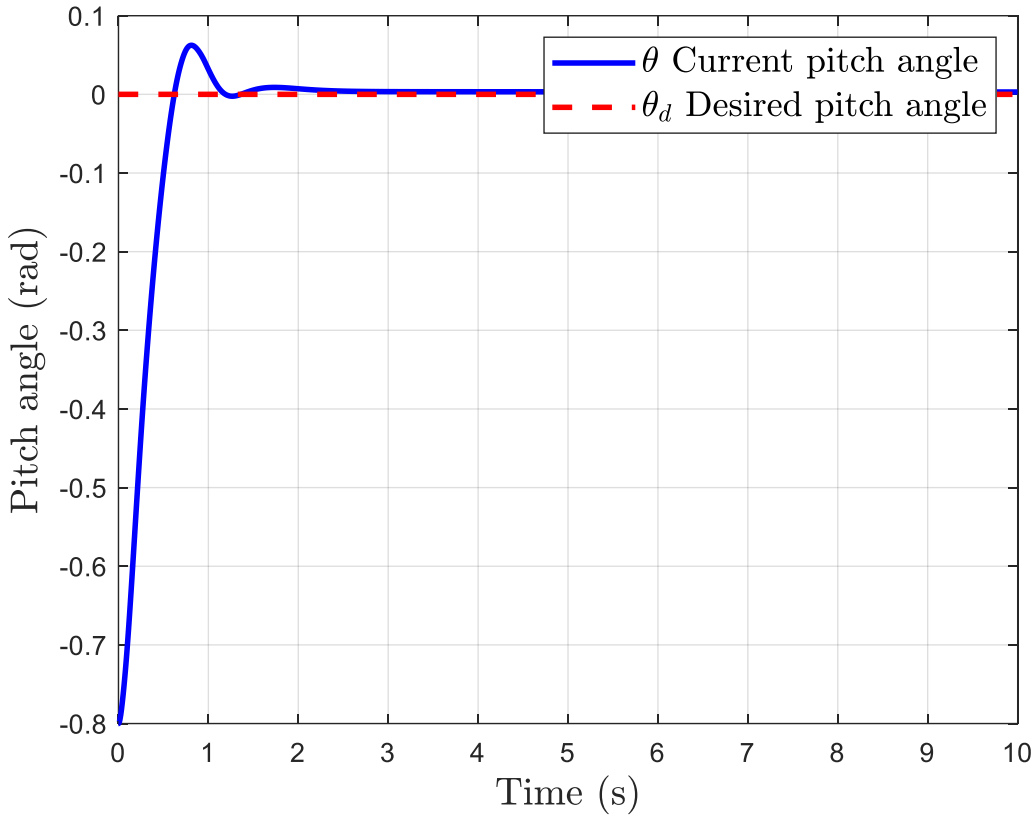
Đánh giá chất lượng của hệ thống sử dụng fuzzy PID điều khiển góc Roll:

Thời gian lên (Rise time): 0.621 giây

Độ vọt lố: 0%

Thời gian để trạng thái đạt được trạng thái ổn định: 1.487 giây

Sai số xác lập: 0



Hình III.16.Điều khiển góc pitch mong muốn sử dụng fuzzy PID

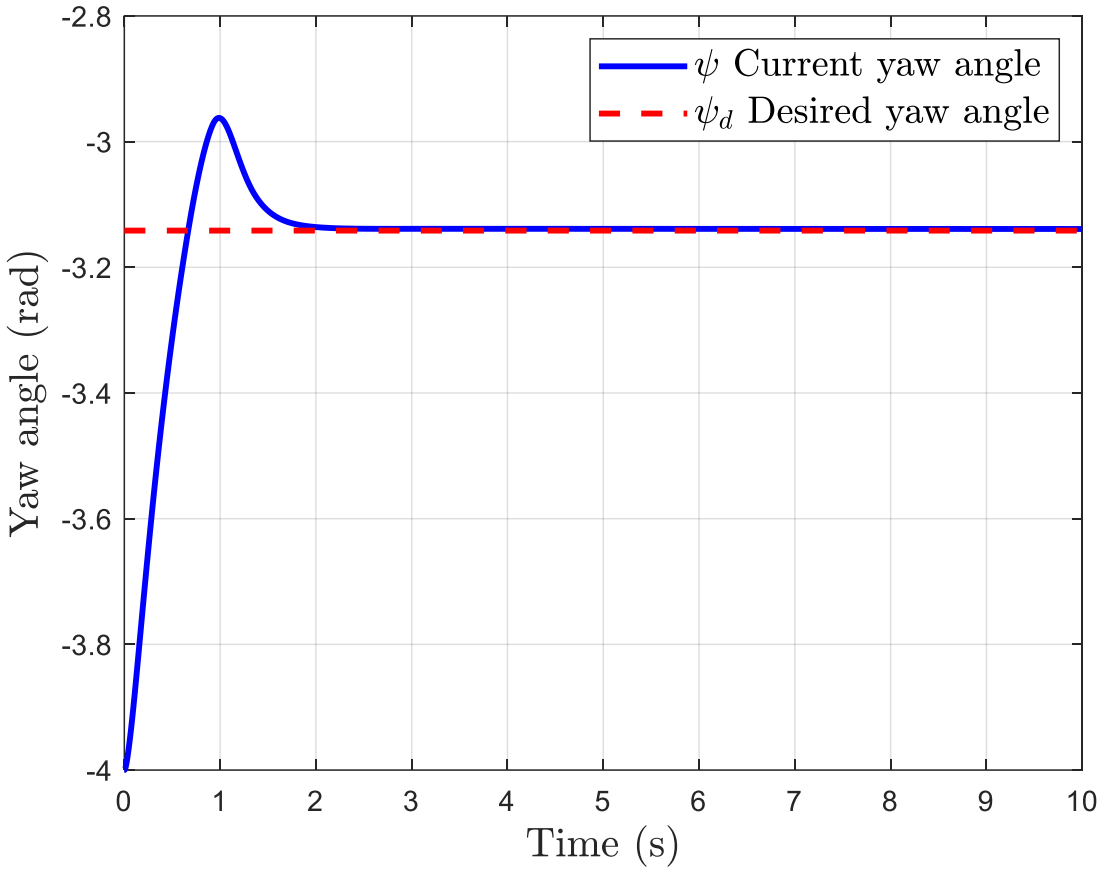
Đánh giá chất lượng của hệ thống sử dụng fuzzy PID điều khiển góc Pitch:

Thời gian lên (Rise time): 0.44 giây

Độ vọt lố: 7.8%

Thời gian để trạng thái đạt được trạng thái ổn định: 2.241 giây

Sai số xác lập: 0



Hình III.17.Điều khiển góc yaw mong muốn sử dụng fuzzy PID

Đánh giá chất lượng của hệ thống sử dụng fuzzy PID điều khiển góc Yaw:

Thời gian lên (Rise time): 0.492 giây

Độ vọt lố: 20.84%

Thời gian để trạng thái đạt được trạng thái ổn định: 2.031 giây

Sai số xác lập: 0

PHẦN IV: ĐÁNH GIÁ CHẤT LƯỢNG HỆ THỐNG VÀ SO SÁNH GIỮA PID VÀ FUZZY PID

Dựa trên kết quả mô phỏng ở phần trước, để có cái nhìn trực quan hơn nhóm sẽ lập bảng so sánh giữa PID và fuzzy PID dựa trên các chỉ tiêu đánh giá chất lượng của hệ thống.

Thông số	**Bộ điều khiển**	**Roll (Φ)**	**Pitch (θ)**	**Yaw (ψ)**
Góc mong muốn (rad)	Cả hai	-0.4	0	-3.141
Vọt lố (%)	PID	19.97%	38.7%	0%
	Fuzzy-PID	0%	7.8%	20.84%
Rising time (s)	PID	0.287	0.715	1.05
	Fuzzy-PID	0.621	0.44	0.492
Thời gian đáp ứng (s)	PID	2	3.986	2.313
	Fuzzy-PID	1.487	2.241	2.031

Dựa trên kết quả so sánh hiệu năng điều khiển của hai bộ điều khiển PID và Fuzzy-PID đối với hệ thống quadrotor, có thể thấy rằng Fuzzy-PID cho hiệu quả điều khiển vượt trội ở hai trục quan trọng là **Roll (lật ngang)** và **Pitch (nghiêng về phía trước)**. Cụ thể, Fuzzy-PID hoàn toàn loại bỏ hiện tượng vọt lố ở trục Roll và giảm đáng kể vọt lố ở trục Pitch từ 38.7% xuống còn 7.8%. Đồng thời, thời gian đáp ứng ở cả hai trục đều được cải thiện, giúp hệ thống đạt đến trạng thái ổn định nhanh hơn.

Mặc dù ở trục **Yaw (xoay quanh trục thẳng đứng)**, Fuzzy-PID cho thời gian đáp ứng và rising time nhanh hơn PID, nhưng lại xuất hiện vọt lố tương đối lớn (20.84%), trong khi PID giữ được độ ổn định tốt hơn với vọt lố bằng 0%. Điều này cho thấy Fuzzy-PID chưa được tối ưu hoàn toàn cho trục Yaw, và cần được hiệu chỉnh thêm về luật mờ hoặc tham số để đảm bảo độ chính xác khi điều hướng.

Tổng thể, **bộ điều khiển Fuzzy-PID thể hiện khả năng nâng cao độ ổn định và hiệu suất động học của hệ thống quadrotor**, đặc biệt ở các trục ảnh hưởng trực tiếp đến tư thế bay (Roll và Pitch).

Với một số tinh chỉnh bổ sung cho trục Yaw, Fuzzy-PID hứa hẹn sẽ là một giải pháp điều khiển toàn diện, đáp ứng tốt yêu cầu về độ chính xác và tốc độ trong các ứng dụng bay tự hành.

PHỤ LỤC

Code của file parameterfuzzyPID.m

```
clc, clear, close all;

% Constants
g = 9.81;
m = 1.5;
k = 2.980e-6; % Thrust factor of rotor (depends on density
              % geometry, etc)
             % to the centre of gravity
l = 0.225; % Linear distance from the centre of the rotor
b = 1.140e-7; % Drag constant
Im = 3.357e-5; % Inertia moment of the rotor

% Inertia
I_xx = 4.856e-3; % kg*m^2
I_yy = 4.856e-3; % kg*m^2
I_zz = 8.801e-3; % kg*m^2

% Initial configuration
roll_i = -0.1;   % rad
pitch_i = -0.8;  % rad
yaw_i = -4;      % rad
altitude_i = 0;  % meters

% Desired configuration
roll_d = -0.4;   % rad
pitch_d = 0;     % rad
yaw_d = -pi;     % rad
altitude_d = 6;  % meters

sim("quadrotor_model") % Initialize Simulink
```

```
% 1. Load hệ thống fuzzy từ file .fis
fis = readfis('fuzzyAttitudez.fis');

% 2. Tạo biến vào Workspace
assignin('base', 'fis', fis);  % để dùng trong Simulink

% 3. Các thông số min/max PID để scale lại
kp_min = 0;  kp_max = 60;
ki_min = 0;  ki_max = 1.0;
kd_min = 0;  kd_max = 20;

assignin('base', 'kp_min', kp_min);
assignin('base', 'kp_max', kp_max);
assignin('base', 'ki_min', ki_min);
assignin('base', 'ki_max', ki_max);
assignin('base', 'kd_min', kd_min);
assignin('base', 'kd_max', kd_max);



% Plot the movements
figure('Name', 'Positions', 'NumberTitle','off')
plot3(x, y, z, 'LineWidth', 2)
xlabel("Position x")
ylabel("Position y")
zlabel("Position z")
title("Positions")
grid on

% Roll angle
figure('Name', 'Roll angle', 'NumberTitle','off')
plot(tout, roll, 'b', 'LineWidth', 2); hold on
```

```
plot(tout, ones(size(tout))*roll_d, '--r', 'LineWidth', 2);
xlabel('Time (s)', 'interpreter', 'latex')
ylabel('Roll angle (rad)', 'interpreter', 'latex')
l = legend('$\phi$ Current roll angle', '$\phi_d$ Desired roll angle');
set(l, 'interpreter', 'latex')

% Pitch angle
figure('Name', 'Pitch angle', 'NumberTitle','off')
plot(tout, pitch, 'b', 'LineWidth', 2); hold on
plot(tout, ones(size(tout))*pitch_d, '--r', 'LineWidth', 2);
xlabel('Time (s)', 'interpreter', 'latex')
ylabel('Pitch angle (rad)', 'interpreter', 'latex')
l = legend('$\theta$ Current pitch angle', '$\theta_d$ Desired pitch angle');
set(l, 'interpreter', 'latex')

% Yaw angle
figure('Name', 'Yaw angle', 'NumberTitle','off')
plot(tout, yaw, 'b', 'LineWidth', 2); hold on
plot(tout, ones(size(tout))*yaw_d, '--r', 'LineWidth', 2);
xlabel('Time (s)', 'interpreter', 'latex')
ylabel('Yaw angle (rad)', 'interpreter', 'latex')
l = legend('$\psi$ Current yaw angle', '$\psi_d$ Desired yaw angle');
set(l, 'interpreter', 'latex')
```

Code file fuzzy, được tạo ra khi đã thực hiện cấu hình

```
[System]
Name='fuzzyAttitudez'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=3
NumRules=49
AndMethod='min'
```

OrMethod='max'

ImpMethod='min'

AggMethod='max'

DefuzzMethod='centroid'

[Input1]

Name='error'

Range=[-1 1]

NumMFs=7

MF1='NB':'gaussmf',[0.15 -1]

MF2='NM':'gaussmf',[0.15 -0.66]

MF3='NS':'gaussmf',[0.15 -0.33]

MF4='Z':'gaussmf',[0.15 0]

MF5='PS':'gaussmf',[0.15 0.33]

MF6='PM':'gaussmf',[0.15 0.66]

MF7='PB':'gaussmf',[0.15 1]

[Input2]

Name='error-dot'

Range=[-1 1]

NumMFs=7

MF1='NB':'gaussmf',[0.15 -1]

MF2='NM':'gaussmf',[0.15 -0.66]

MF3='NS':'gaussmf',[0.15 -0.33]

MF4='Z':'gaussmf',[0.15 0]

MF5='PS':'gaussmf',[0.15 0.33]

MF6='PM':'gaussmf',[0.15 0.66]

MF7='PB':'gaussmf',[0.15 1]

[Output1]

Name='kpf'

Range=[0 60]

NumMFs=3

MF1='N':'gaussmf',[**10 0**]

MF2='Z':'gaussmf',[**10 30**]

MF3='P':'gaussmf',[**10 60**]

[Output2]

Name='kif'

Range=[**0 1**]

NumMFs=**3**

MF1='N':'gaussmf',[**0.18 0**]

MF2='Z':'gaussmf',[**0.18 0.5**]

MF3='P':'gaussmf',[**0.18 1**]

[Output3]

Name='kdf'

Range=[**0 20**]

NumMFs=**3**

MF1='N':'gaussmf',[**3.5 0**]

MF2='Z':'gaussmf',[**4.5 10**]

MF3='P':'gaussmf',[**3.5 20**]

[Rules]

1 1, **1 1 1** (**1**) : **1**

2 1, **1 1 1** (**1**) : **1**

3 1, **1 1 1** (**1**) : **1**

4 1, **1 1 1** (**1**) : **1**

5 1, **1 1 1** (**1**) : **1**

6 1, **1 1 1** (**1**) : **1**

7 1, **2 2 2** (**1**) : **1**

1 2, **1 1 1** (**1**) : **1**

2 2, **1 1 1** (**1**) : **1**

3 2, **1 1 1** (**1**) : **1**

4 2, **1 1 1** (**1**) : **1**

5 2, **1 1 1** (**1**) : **1**

6 2, **2 2 2** (**1**) : **1**

7 2, **3 3 3** (**1**) : **1**

1 3, **1 1 1** (**1**) : **1**

2 3, **1 1 1** (**1**) : **1**

3 3, **1 1 1** (**1**) : **1**

4 3, **1 1 1** (**1**) : **1**

5 3, **2 2 2** (**1**) : **1**

6 3, **3 3 3** (**1**) : **1**

7 3, **3 3 3** (**1**) : **1**

1 4, **1 1 1** (**1**) : **1**

2 4, **1 1 1** (**1**) : **1**

3 4, **1 1 1** (**1**) : **1**

4 4, **2 2 2** (**1**) : **1**

5 4, **3 3 3** (**1**) : **1**

6 4, **3 3 3** (**1**) : **1**

7 4, **3 3 3** (**1**) : **1**

1 5, **1 1 1** (**1**) : **1**

2 5, **1 1 1** (**1**) : **1**

3 5, **2 2 2** (**1**) : **1**

4 5, **3 3 3** (**1**) : **1**

5 5, **3 3 3** (**1**) : **1**

6 5, **3 3 3** (**1**) : **1**

7 5, **3 3 3** (**1**) : **1**

1 6, **1 1 1** (**1**) : **1**

2 6, **2 2 2** (**1**) : **1**

3 6, **3 3 3** (**1**) : **1**

4 6, **3 3 3** (**1**) : **1**

5 6, **3 3 3** (**1**) : **1**

6 6, **3 3 3** (**1**) : **1**

7 6, **3 3 3** (**1**) : **1**

1 7, **2 2 2** (**1**) : **1**

2 7, **3 3 3** (**1**) : **1**

3 7, **3 3 3** (**1**) : **1**

4 7, **3 3 3** (**1**) : **1**

5 7, **3 3 3** (**1**) : **1**

6 7, **3 3 3** (**1**) : **1**

7 7, **3 3 3** (**1**) : **1**