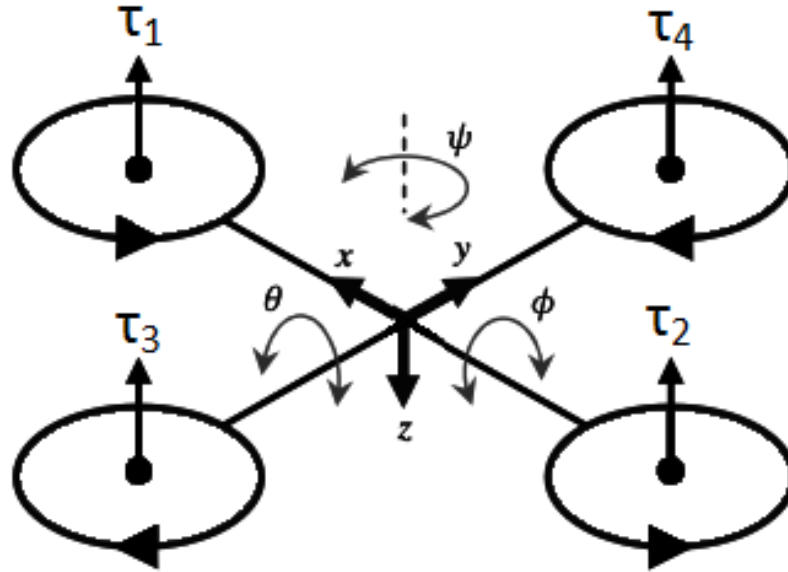


# 1 Modelagem de um Quadricóptero

Como definido no Capítulo ??, um quadricóptero é uma aeronave cuja propulsão é obtida a partir do uso de quatro rotores e sua representação é mostrada na Figura 1.

Figura 1 – Representação de um quadricóptero



Fonte: Adaptado de [Basri et al. \(2014\)](#)

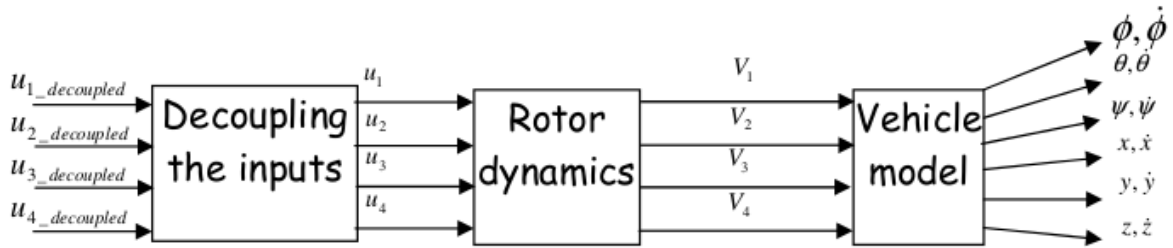
Neste diagrama,  $x$ ,  $y$  e  $z$  indicam a orientação dos eixos,  $\tau_1$ ,  $\tau_2$ ,  $\tau_3$  e  $\tau_4$  são as forças geradas pela propulsão de cada um dos motores. As variáveis  $\phi$ ,  $\theta$ , e  $\psi$  são os ângulos de rotação do quadricóptero em relação aos eixos  $x$ ,  $y$  e  $z$  respectivamente. Estes ângulos são também chamados ângulos de *roll* (rolamento), *pitch* (arfagem) e *yaw* (guinada), respectivamente. Além disto, o diagrama mostra ainda o sentido de rotação de cada um dos quatro rotores. Como se pode ver, os rotores dispostos sobre um mesmo eixo possuem um mesmo sentido de rotação. Desta forma, os rotores 1 e 2 (dispostos sobre o eixo  $x$ ), giram no sentido anti-horário. Já os rotores 3 e 4 (dispostos sobre o eixo  $y$ ) giram no sentido horário. Esta disposição dos rotores permite que os empuxos horizontais se anulem, possibilitando a estabilidade do quadricóptero quando submetido a uma ação de controle devida ([BASRI et al., 2014](#), p. 1).

A estabilidade de um quadricóptero, entretanto, assume mais de uma forma, sendo dividida em dois aspectos principais: altitude e atitude. O primeiro diz respeito ao posicionamento do quadricóptero sobre o eixo  $z$  e seu controle é necessário tanto para possibilitar que ele se mantenha num estado estacionário quanto para fazer movimentações no plano XY numa altitude fixa, o que pode ser crucial se ele estiver sendo operado

num ambiente limitado inferior e/ou superiormente por alguma espécie de barreira. Já o segundo aspecto, a atitude, se refere à estabilidade angular do quadricóptero, permitindo que sua orientação, representada pelos ângulos  $\phi$ ,  $\theta$ , e  $\psi$  seja controlada.

Uma vez descrito o comportamento geral do sistema, pode-se partir para a sua modelagem matemática. A modelagem retratada é a que foi desenvolvida por Balas (2007) e é representada na Figura 2

Figura 2 – Diagrama do sistema representando um quadrotor modelado por Balas (2007)



Fonte: Balas (2007, p. 49)

## 1.1 Modelagem Matemática

O quadricóptero é controlado a partir da variação da velocidade dos seus quatro rotores, que são completamente independentes entre si. Desta forma, sendo  $l$  o comprimento de cada haste a partir do centro geométrico do quadricóptero e considerando  $v_i$  e  $\tau_i$  respectivamente como o torque e o impulso do  $i$ -ésimo rotor, podem-se considerar as entradas do sistema como sendo (BALAS, 2007, p. 4):

$$u_1 = \tau_1 + \tau_2 + \tau_3 + \tau_4 \quad (1)$$

$$u_2 = l(\tau_3 - \tau_4) \quad (2)$$

$$u_3 = l(\tau_1 - \tau_2) \quad (3)$$

$$u_4 = v_1 + v_2 + v_3 + v_4 \quad (4)$$

em que  $u_1$  é o impulso total,  $u_2$  é o momento de rolamento,  $u_3$  é o momento de arfagem e  $u_4$  é o momento de guinada.

Além disso, aplicando as leis de Newton, a aceleração em cada eixo pode ser representada por (BALAS, 2007, p. 5):

$$\ddot{x} = -\frac{\sin(\theta)\cos(\phi)}{m}u_1 \quad (5)$$

$$\ddot{y} = \frac{\sin(\phi)}{m}u_1 \quad (6)$$

$$\ddot{z} = -\frac{\cos(\theta)\cos(\phi)}{m}u_1 + g \quad (7)$$

em que  $\phi$  e  $\theta$  são os ângulos em torno dos eixos  $x$  e  $y$  respectivamente,  $m$  é a massa do quadricóptero e  $g$  é a gravidade à qual ele é submetido.

Além das acelerações em cada eixo, é também necessário se definir a aceleração em torno de cada eixo, ou seja, as acelerações angulares em torno de  $x$ ,  $y$  e  $z$ . Para tanto, assumindo que a estrutura do quadricóptero seja rígida, sendo  $I_{xx}$ ,  $I_{yy}$  e  $I_{zz}$  o momento de inércia do quadricóptero ao longo dos eixos  $x$ ,  $y$  e  $z$  respectivamente e considerando que os momentos de inércia ao longo de  $x$  e  $y$  se equivalem, as acelerações angulares podem ser representadas a partir das seguintes equações (BALAS, 2007, p. 6):

$$\ddot{\phi} = -\dot{\psi}\dot{\theta}\cos(\phi) + \frac{\cos(\psi)}{I_{xx}}u_2 - \frac{\sin(\psi)}{I_{yy}}u_3 + \frac{I_{yy} - I_{zz}}{I_{xx}}(\dot{\psi} - \dot{\theta}\sin(\phi))\dot{\theta}\cos(\phi) \quad (8)$$

$$\ddot{\theta} = \frac{\dot{\psi}\dot{\phi}}{\cos(\phi)} + \dot{\phi}\dot{\theta}\tan(\phi) + \frac{\sin(\psi)}{\cos(\phi)I_{xx}}u_2 + \frac{\cos(\psi)}{\cos(\phi)I_{yy}}u_3 - \frac{I_{yy} - I_{zz}}{I_{xx}}(\dot{\psi} - \dot{\theta}\sin(\phi))\frac{\dot{\phi}}{\cos(\phi)} \quad (9)$$

$$\ddot{\psi} = \dot{\phi}\dot{\psi}\tan(\phi) + \frac{\dot{\phi}\dot{\theta}}{\cos(\phi)} + \frac{\sin(\psi)\tan(\phi)}{I_{xx}}u_2 + \frac{\cos(\phi)\tan(\psi)}{I_{yy}}u_3 + \frac{1}{I_{zz}}u_4 - \frac{I_{yy}I_{zz}}{I_{xx}}(\dot{\psi} - \dot{\theta}\sin(\phi))\dot{\phi}\tan(\phi) \quad (10)$$

Por fim, deve-se relacionar a velocidade de cada rotor  $i$  ao impulso e torque sobre ele. Esta relação é dada por (BALAS, 2007, p. 7):

$$v_i = \tau_i(V_c + v_i) + 0.125\rho bc R_p^4 \omega_{m_i}^2 C_d \quad (11)$$

sendo  $v_i$  o torque sobre o rotor,  $\tau_i$  o impulso atuando sobre ele,  $V_c$  a velocidade vertical<sup>1</sup>,  $v_i$  a velocidade induzida no motor,  $\rho$  a densidade do ar,  $b$  o número de pás,  $c$  o comprimento delas,  $R_p$  o raio da hélice,  $C_d$  o coeficiente de arrasto e  $\omega_{m_i}$  a velocidade angular do rotor.

Após a modelagem de um sistema complexo como este, é de se desejar que se possa isolar a resposta de determinadas variáveis às entradas. Para tanto, pode-se utilizar o processo de desacoplamento de entradas.

## 1.2 Desacoplamento das Entradas

Para obter uma resposta isolada das variáveis do sistema a partir dos sinais de entrada dele, um bloco de desacoplamento foi modelado em (BALAS, 2007, p. 62).

<sup>1</sup> A velocidade vertical do quadricóptero também é indicada por  $\dot{z}$

Neste caso, foram considerados os acoplamentos entre  $\phi, \theta, \psi$  e  $u_2, u_3, u_4$ , relacionados da seguinte maneira:

$$u_{2\_decoupled} = \cos(\psi_0)u_2 - \frac{I_{xx}\sin(\psi_0)}{I_{yy}}u_3 = I_{xx}\ddot{\phi} \quad (12)$$

$$u_{3\_decoupled} = \frac{\sin(\psi_0)I_{yy}}{\cos(\phi_0)I_{xx}}u_2 + \frac{\cos(\psi_0)}{\cos(\phi_0)}u_3 = I_{yy}\ddot{\theta} \quad (13)$$

$$u_{4\_decoupled} = \frac{I_{zz}\sin(\psi_0)\tan(\phi_0)}{I_{xx}}u_2 + \frac{I_{zz}\cos(\psi_0)\tan(\phi_0)}{I_{zz}}u_3 + u_4 = I_{zz}\ddot{\psi} \quad (14)$$

em que  $\phi_0, \theta_0$  e  $\psi_0$  são os ângulos no iniciais de rolamento, arfagem e guinada respectivamente;  $\dot{\phi}, \dot{\theta}$  e  $\dot{\psi}$  são os ângulos atuais de rolamento, arfagem e guinada; e  $I_{xx}, I_{yy}$  e  $I_{zz}$  são o momento de inércia em torno dos eixos  $x, y$  e  $z$  respectivamente.

Como se pode ver pelas equações, aplicando esse desacoplamento obtêm-se as variáveis  $u_{2\_decoupled}, u_{3\_decoupled}$  e  $u_{4\_decoupled}$  relacionadas aos ângulos  $\phi, \theta$  e  $\psi$  respectivamente e considerando o momento de inércia sobre os respectivos eixos, isolando assim a resposta das diferentes variáveis do sistema.

Além disso, a partir destas equações, podem-se representar essas transformações utilizando o formato matricial da seguinte maneira (BALAS, 2007, p. 62):

$$\begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} \cos(\psi_0) & \sin(\psi_0)\cos(\phi_0)\frac{I_{xx}}{I_{yy}} & 0 \\ -\sin(\psi_0)\frac{I_{yy}}{I_{xx}} & \cos(\psi_0)\cos(\phi_0) & 0 \\ 0 & -\sin(\phi_0)\frac{I_{zz}}{I_{yy}} & 1 \end{bmatrix} \begin{bmatrix} u_{2\_decoupled} \\ u_{3\_decoupled} \\ u_{4\_decoupled} \end{bmatrix}$$

Após os processos de modelagem matemática do sistema e o devido desacoplamento de suas entradas, pode-se representar seu comportamento geral a partir dos espaços de estados, prática comum para descrever sistemas dinâmicos.

### 1.2.1 Representação no Espaço de Estados

Como mostrado em (BALAS, 2007, p. 63), o sistema modelado, já incluindo o desacoplamento de entradas, pode ser representado da seguinte forma no espaço de estados.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \quad (15)$$

Com o vetor de estados  $X$  sendo dado por:

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

O vetor de entrada  $U$  sendo:

$$U = \begin{bmatrix} u_1 & u_{2\_decoupled} & u_{3\_decoupled} & u_{4\_decoupled} \end{bmatrix}^T$$



$$D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

em que  $g$  é a gravidade,  $m$ , a massa do quadricóptero e os ângulos  $\phi_0$ ,  $\theta_0$  e  $\psi_0$  representam os ângulos iniciais em torno dos eixos  $x$ ,  $y$  e  $z$  respectivamente. Além disto,  $I_{xx}$ ,  $I_{yy}$  e  $I_{zz}$  representam o momento de inércia do quadricóptero ao longo destes mesmos eixos.

Como se pode perceber, a matriz  $C$  é uma matriz diagonal, o que implica no fato de a saída do sistema ser representada pelo vetor de estados.

Ao longo deste trabalho, essa foi a modelagem adotada para simular o sistema de um quadricóptero.

## Referências

BALAS, C. **Modeling and Linear Control of a Quadrotor**. Dissertação (Mestrado) — Cranfield University, Reino Unido, 2007. Citado 3 vezes nas páginas [2](#), [3](#) e [4](#).

BASRI, M.; HUSAIN, A.; DANAPALASINGAM, K. Fuzzy supervisory backstepping controller for stabilization of quadrotor unmanned aerial vehicle. In: **Intelligent and Advanced Systems (ICIAS), 2014 5th International Conference on**. [S.l.: s.n.], 2014. p. 1–5. Citado na página [1](#).

## Apêndices



## APÊNDICE A – Código para Criação de Modelo Neuro-Fuzzy para Altitude e Definição de Dados para Treinamento

```

1      % le arquivo fis referente ao controle de altitude
2      fismat = readfis('fis_altitude.fis');
3
4      % define numero de casos a serem avaliados (treinamento + teste)
5      n = 300;
6      % define conjunto de n entradas aleatorias para o sistema fuzzy
7      % respeitando o range de cada entrada
8      input = zeros(n,2);
9      for i=1:n
10         z_value = rand * 2 - 1;
11         z_dot_value = rand * 10 - 5;
12         input(i,:) = [ z_value z_dot_value ];
13     end
14
15     % avalia resposta fuzzy para cada entrada
16     output= evalfis(input,fismat);
17
18     % define data como vetor relacionando cada conjunto de entradas ...
19     % a saida
20     % - obtida pelo sistema fuzzy
21     data = [];
22     for i=1:n
23         data(i,:) = [ input(i,:) output(i) ];
24     end
25
26     % define que 2/3 dos dados obtidos serao usados para treinamento
27     % e 1/3 sera usado para teste da rede
28     train = data(1:2*n/3,:);    % dados para treinamento
29     test = data(2*n/3+1:n,:);  % dados para validacao do sistema ...
30     treinado
31
32     % gera modelo fuzzy Sugeno a partir do Mamdani modelado
33     sugFIS = mam2sug(fismat);
34     % salva modelo Sugeno em disco com o nome fis_altitude_neuro.fis
35     writefis(sugFIS, 'fis_altitude_neuro.fis');

```

## APÊNDICE B – Código para Criação de Modelo Neuro-Fuzzy para Atitude e Definição de Dados para Treinamento

```

1      % le arquivo fis referente ao controle de atitude
2      fismat = readfis('fis_atitude.fis');
3
4      % define numero de casos a serem avaliados (treinamento + teste)
5      n = 300;
6      % define conjunto de n entradas aleatorias para o sistema fuzzy
7      % respeitando o range de cada entrada
8      input = zeros(n,2);
9      for i=1:n
10         phi_value = rand * 4 - 2;
11         phi_dot_value = rand * 3 - 1.5;
12         input(i,:) = [ phi_value phi_dot_value ];
13     end
14
15     % avalia resposta fuzzy para cada entrada
16     output= evalfis(input,fismat);
17
18     % define data como vetor relacionando cada conjunto de entradas ...
19     % a saída
20     % obtida pelo sistema fuzzy
21     data = [];
22     for i=1:n
23         data(i,:) = [ input(i,:) output(i) ];
24     end
25
26     % define que 2/3 dos dados obtidos serao usados para treinamento
27     % e 1/3 sera usado para teste da rede
28     train = data(1:2*n/3,:);    % dados para treinamento
29     test = data(2*n/3+1:n,:);  % dados para validação do sistema ...
30     % treinado
31
32     % gera modelo fuzzy Sugeno a partir do Mamdani modelado
33     sugFIS = mam2sug(fismat);
34     % salva modelo Sugeno em disco com o nome fis_atitude_neuro.fis
35     writefis(sugFIS, 'fis_atitude_neuro.fis');

```