

# Análise e Síntese de Controladores: Simulação e Construção de um Quadricoptero

Mauro S. Quiroga Coronado \* Francisco J. Triveno Vargas \*\*

\* *Sur Aerospace Company, Calle Ramon Rivero 41, +591-4-4221408  
Cochabamba, Bolivia., (e-mail: msquirogac@gmail.com).*

\*\* *Engenharia Elétrica, Universidade de Araraquara, SP, Brasil  
(e-mail: fjt Vargas@uniara.edu.br)*

**Abstract:** This work presents the implementation of an experimental platform of unmanned aerial vehicles (UAVs) with rotary wing, using four-rotor vehicles (quadcopter) as a model, however, the development can be extended to more engines or configurations. The vehicle is studied in its main aspects, starting with the mathematical modelling that describes its behavior, as well as its internal components. Algorithms are also studied to condition and mix the sensors measurements that are soon processed by the cascade control algorithms composed of PID strategies that allow to maintain the flight stable. The prototype is implemented, not only at the physical level, but also at the *hardware in the loop* (HIL) simulation level using the X-plane software.

## Resumo:

Este trabalho apresenta a implementação de uma plataforma experimental de veículos aéreos não tripulados (VANTs) de asa rotativa, usando veículos de quatro rotores (quadcoptero) como modelo, no entanto, o desenvolvimento pode ser estendido a mais motores ou configurações geométricas. O veículo é estudado em seus principais aspectos, começando pela modelagem matemática que descreve seu comportamento, bem como seus componentes internos. Também são estudados algoritmos para condicionar e misturar as medições dos sensores que logo são processados pelos algoritmos de controle em cascata compostos por estratégias PID que permitem manter o voo estável. O protótipo é implementado, não apenas no nível físico, mas também em nível de simulação *hardware in the loop* (HIL) usando o software X-plane.

**Keywords:** UAVs, Quadcopter, PID, Cascade control, Simulation, X-Plane, HIL.

**Palavras-chaves:** VANTs, Quadcoptero, PID, Controle cascata, Simulação, X-Plane, HIL.

## 1. INTRODUÇÃO

Com a chegada e a evolução da tecnologia em componentes  $\mu$  eletromecânicos (MEMS), hoje é possível obter sensores como giroscópios, acelerômetros ou barômetros, entre outros, com tamanho reduzido e baixo custo, uma diferença muito grande em relação ao passado, onde apenas aplicações militares ou aplicações altamente especializadas como os aviões permitiam seu uso. Da mesma forma, devido à constante melhoria na capacidade de processamento dos  $\mu$ controladores, é também possível implementar algoritmos altamente complexos em tempo real.

Em grande medida, isso promoveu o surgimento de veículos aéreos não tripulados VANTs, ou drones, como são conhecidos em termos populares, como um produto disponível ao público em geral. Outro fator chave foi sua grande versatilidade, permitindo que novas e melhores soluções para vários tipos de problemas fossem desenvolvidas, tornando este veículo muito popular.

No entanto, a operação, suas capacidades e limitações, seus cuidados e principalmente os algoritmos nele processados, não foram divulgados na mesma extensão; portanto, não é incomum que ocorram acidentes, que incluem danos ao próprio veículo e acidentes graves envolvendo pessoas ou outros seres vivos. Isto, acrescentado das limitações da tecnologia atual, desencadeou no mundo inteiro uma série de regulamentos, restringindo seu uso em ambientes urbanos e, conseqüentemente, seu incrível potencial.

O protótipo de VANT deste trabalho corresponde a um quadcoptero, veículo aéreo não tripulado de asa rotativa e, como o nome sugere, de quatro rotores. O desenvolvimento começa com a determinação do modelo matemático que explica o comportamento dinâmico do sistema a ser controlado. Logo, é proposto um algoritmo de controle em cascata para a rotação e a translação do veículo inicialmente testado por simulação. A simulação fornece um ambiente virtual no qual é possível verificar não apenas os algoritmos de controle como tal, mas também sua implementação em uma placa de desenvolvimento eletrônica, categorizado como simulação de *hardware in the loop* (HIL). Por fim, feita a validação do firmware por meio do ambiente si-

\* Os autores agradecem à Universidade Simon e Patiño da Bolívia e à UNIARA pelo suporte financeiro.

mulado, o protótipo é construído e testes em campo são executados. Ressaltar que este artigo é um extrato do trabalho desenvolvido por Quiroga (2018).

## 2. MODELAGEM MATEMÁTICA

Em aeronáutica estão presentes minimamente dois sistemas de referencia, um fixo e outro móvel, o ultimo associado ao centro de massa do proprio veiculo como ilustrado na Fig. 1.

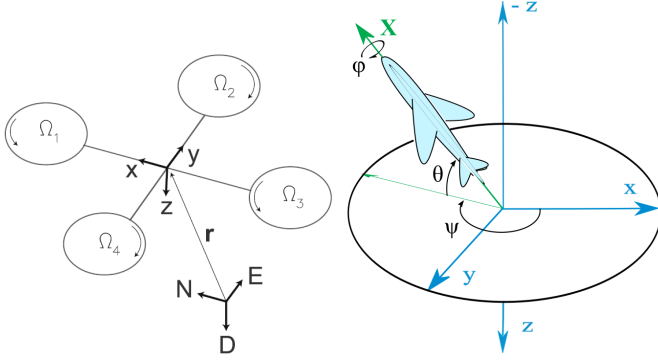


Figura 1. (Esquerda) Sistemas de referência, (Direita) Ângulos de Euler

### 2.1 Sistemas de coordenadas

A relação de orientação entre os dois sistemas de referência é obtida através dos ângulos de **Euler**  $\Phi = \{\phi, \theta, \psi\}$ , neste trabalho a ordem ZYX é adotada por ser a mais empregada em navegação de aeronaves Cai et al. (2011). A ordem de rotação mostra que a ida do sistema fixo ao sistema móvel é iniciada com uma rotação com ângulo  $\psi$  ao redor de  $Z_e$ , seguida de uma segunda rotação  $\theta$  ao redor de  $Y'_e$  e uma rotação final ao redor de  $X''_e$  com ângulo  $\phi$ <sup>1</sup>. Segundo Diebel (2006), estas rotações são expressas através da matriz de cossenos diretores e seu efeito total é representado pela multiplicação sequencial dessas matrizes obtendo-se:

$$R_{b/e} = \begin{bmatrix} (c\theta c\psi) & (c\theta s\psi) & (-s\theta) \\ (s\phi s\theta c\psi - c\phi s\psi) & (s\phi s\theta s\psi + c\phi c\psi) & (s\phi c\theta) \\ (c\phi s\theta c\psi + s\phi s\psi) & (c\phi s\theta s\psi - s\phi c\psi) & (c\phi c\theta) \end{bmatrix} \quad (1)$$

Em forma complementar, para expressar variáveis do sistema móvel no sistema fixo, a matriz inversa é empregada, que devido à sua natureza ortonormal, pode ser substituída pela transposta.

$$R_{e/b} = R_{b/e}^{-1} = R_{b/e}^T \quad (2)$$

Outra relação importante para o modelado é a velocidade angular medida no sistema de referencia móvel  $\omega_b$  através da derivada dos ângulos de **Euler**  $\dot{\Phi} = \{\dot{\phi}, \dot{\theta}, \dot{\psi}\}$ . Em uma primeira instância entende-se que ambos os parâmetros são iguais, i.e para valores específicos dos ângulos de **Euler**  $\omega_b = \dot{\Phi}$ , no entanto devido a que os sistemas de referência são criados implicitamente com cada rotação, esta igualdade não é cumprida para valores diferentes de 0 [rad].

Para estabelecer esta relação se deve aplicar as mesmas transformações intermédias a cada componente de  $\dot{\Phi}$  como explica Stevens et al. (2015). Esta sequencia de operações é resumida na matriz  $H(\Phi)$  dada por:

$$H(\Phi) = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)\sec(\theta) & \cos(\phi)\sec(\theta) \end{bmatrix} \quad (3)$$

que da lugar à igualdade:

$$\dot{\Phi} = H(\Phi)\omega_b \quad (4)$$

Com as bases bem estabelecidas, agora se procede a definir o modelo cinemático e dinâmico.

### 2.2 Modelo cinemático e dinâmico

A modelagem é feita considerado um veiculo com seis graus de liberdade, tal que o modelo observe seus parâmetros de rotação  $\Phi$  e de translação  $r_e$ , isto define o vetor de estados a seguir com suas respectivas derivadas:

$$x_q = [\Phi \ \omega_b \ r_e \ v_e]^T \quad (5)$$

A variação no tempo das componentes da Equação (5) permite obter o modelo *dinâmico* do veiculo Bouadi et al. (2007), Stingu and Lewis (2009).

$$\dot{x}_q = \begin{bmatrix} \dot{\Phi} \\ \dot{\omega}_b \\ \dot{r}_e \\ \dot{v}_e \end{bmatrix} = \begin{bmatrix} H(\Phi)\omega_b \\ J^{-1}(M_b - \omega_b \times (J\omega_b)) \\ v_e \\ \frac{1}{m}F_e \end{bmatrix} \quad (6)$$

A equação (6) é o resultado de aplicar a segunda lei de **Newton**

$$m\dot{v} = \sum_{i=1}^n F_i$$

para a aceleração linear expressado no sistema de referencia inercial e a equação

$$J\dot{\omega} + \omega \times J\omega = \sum_{i=1}^n M_i$$

para a aceleração angular expressado no sistema móvel.

Sendo a força  $F_e$  e momento  $M_b$  totais os parâmetros externos aplicados ao sistema.

$$F_e = R_{e/b}F_r + F_g \quad (7)$$

As contribuições importantes da força, a ação da gravidade  $F_g$  e a força gerada pelos rotores dos veículos  $F_r$  as quais interagem como apresentado em (7). Ressaltar que as ultimas estão originalmente expressadas no sistema móvel pelo que deve-se aplicar a transformação (1) na sua forma inversa (2) para igualar o sistema de referência ao usado por  $F_e$ .

$$F_g = [0 \ 0 \ mg]^T \quad (8)$$

Em quanto ao momento total temos a contribuição do momento  $M_r$  gerado por cada rotor homologa a  $F_r$  e o momento giroscópio  $M_g$  resultado da oposição à mudança de orientação que apresentara o proprio rotor devido a sua inércia intrínseca, assim como acontece na roda de uma bicicleta.

$$M_b = M_r - M_g \quad (9)$$

De Brandt and Selig (2011) e da teoria de hélices é conhecido que tanto a força quanto o torque que geram as

<sup>1</sup> Cada uma das rotações criara um sistema de referência intermédio

mesmas é proporcional ao quadrado da velocidade angular  $\Omega$ , pelo que inserindo os parâmetros  $K_F$  e  $K_M$  para a força e o torque, as contribuições descritas em função da velocidade angular são:

$$\mathbf{F}_r = \sum_{i=1}^4 \mathbf{F}_{r,i} = \sum_{i=1}^4 \begin{bmatrix} 0 \\ 0 \\ -K_{F,i} \Omega_i^2 \end{bmatrix} \quad (10)$$

Considerando o eixo Z para abaixo positivo, a contribuição da gravidade resultará em uma força positiva, enquanto os rotores apresentarão uma contribuição oposta ou negativa, razão da direção da sinal  $K_F$  na equação (10).

$$\begin{aligned} \mathbf{M}_r &= \sum_{i=1}^4 \mathbf{l}_i \times \mathbf{F}_{r,i} + \mathbf{M}_h, i \\ &= \sum_{i=1}^4 \begin{bmatrix} l_{x,i} \\ l_{y,i} \\ l_{z,i} \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ -K_{F,i} \Omega_i^2 \end{bmatrix} + \dots \quad (11) \\ &\begin{bmatrix} 0 \\ 0 \\ (-1)^{(i+1)} K_{M,i} \Omega_i^2 \end{bmatrix} \end{aligned}$$

Por outro lado, o momento  $\mathbf{M}_r$  é definido por duas contribuições importantes, a *primeira* o par gerado pela força de cada rotor, que pela geometria do veículo são aplicadas a uma distancia  $\mathbf{l}$  em relação ao centro de gravidade. A *segunda*  $\mathbf{M}_h$  surge do arrasto gerado pelas hélices, ressaltando que o sentido de giro é alternado como ilustrado na Fig. 1.

$$\mathbf{M}_g = \sum_{i=1}^4 \boldsymbol{\omega}_b \times \begin{bmatrix} 0 \\ 0 \\ (-1)^i J_{r,i} \Omega_i \end{bmatrix} \quad (12)$$

Como complemento, o momento giroscópico é apresentado, nesta equação (12)  $J_r$  é a inércia em Z do próprio rotor. Observa-se que essa contribuição é muito menor que as outras, portanto, para fins práticos e analíticos, ela pode ser desprezada.

### 3. ALGORITMOS DE CONTROLE

O quadcoptero é um sistema naturalmente instável e, portanto, para operar corretamente precisa de um sistema de controle que compense essa situação e, assim, alcance um certo grau de estabilidade e segurança. A seguir são apresentadas as estruturas dos controladores implementados, bem como a interação entre eles.

#### 3.1 Controle em cascata

Quando vários sensores estão disponíveis para medir os estados de um processo, a técnica de controle em cascata geralmente tem um desempenho melhor do que um controlador de malha única Blevins and Nixon (2011), Cooper (2015).

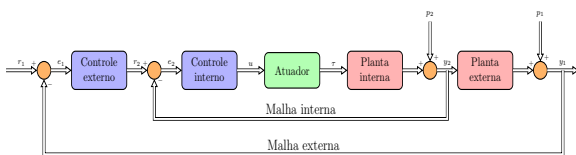


Figura 2. Controle em cascata, malha interna e externa

Esta estrutura tem dois (ou mais) controladores dos quais a saída de um corresponde à referência do outro assim como ilustrado na Fig. 2.

### 4. CONTROLE DO VANT

A arquitetura de controle utilizada é uma adaptação do trabalho de Nagaty et al. (2012), que divide o problema de controle em vários subsistemas conforme ilustrado na Fig. 3.

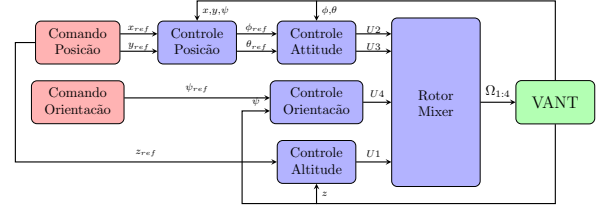


Figura 3. Arquitetura de controle

Nesta figura observa-se que os subsistemas de rotação e translação são dependentes um do outro.

#### 4.1 Controle de rotação

A arquitetura consiste em dos controladores PID Szafranski and Czyba (2011), Cho et al. (2014), Beauregard (2011) configurados para operar em cascata como ilustrado na Fig. 4.

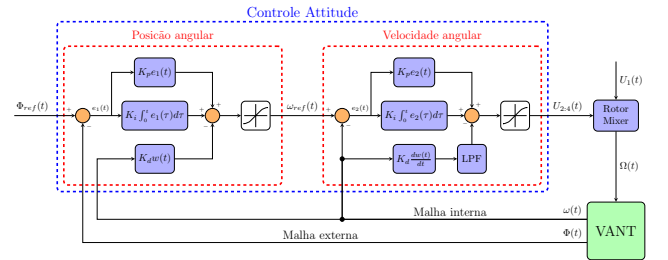


Figura 4. Controle angular malha interna e externa

As equações do erro, do controle PID modificado<sup>2</sup> e da saída são:

$$\begin{aligned} e_1(t) &= \Phi_{ref} - \Phi(t) \\ u(t) &= K_p e_1(t) + K_i \int_0^t e_1(\tau) d\tau - K_d \omega(t) \\ \omega_{ref}(t) &= \text{SAT}(u(t)) \end{aligned} \quad (13)$$

Por outro lado, as equações da malha interna encarregada de controlar a velocidade angular são:

$$\begin{aligned} e_2(t) &= \omega_{ref} - \omega(t) \\ u(t) &= K_p e_2(t) + K_i \int_0^t e_2(\tau) d\tau - K_d \text{LPF} \left( \frac{d\omega(t)}{dt} \right) \\ U_{2:4}(t) &= \text{SAT}(u(t)) \end{aligned} \quad (14)$$

onde *LPF* corresponde ao filtro de baixa passagem necessário para atenuar o ruído de alta frequência introduzido ao estimar a derivada em forma numérica.

<sup>2</sup> considera como entrada apenas o valor medido pelo giroscópio e os limites de saturação.

Agora, adotando quatro rotores iguais com velocidades de rotação  $\Omega = \{\Omega_1, \Omega_2, \Omega_3, \Omega_4\}$  fixados a uma distancia  $l$ , como ilustrado na Fig. 1, é possível fazer operações e simplificar as equações (10) e (11) como:

$$\mathbf{M}_r = \begin{bmatrix} 0 & -l \cdot K_F & 0 & l \cdot K_F \\ l \cdot K_F & 0 & -l \cdot K_F & 0 \\ K_M & -K_M & K_M & -K_M \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \quad (15)$$

$$\mathbf{F}_r = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -K_F & -K_F & -K_F & -K_F \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \quad (16)$$

onde fica em evidencia que para a força  $\mathbf{F}_r$  somente a componente Z é relevante, pelo que todos os efeitos de  $\Omega$  agrupados são:

$$\begin{bmatrix} F_{r,z} \\ M_{r,x} \\ M_{r,y} \\ M_{r,z} \end{bmatrix} = \begin{bmatrix} -K_F & -K_F & -K_F & -K_F \\ 0 & -l \cdot K_F & 0 & l \cdot K_F \\ l \cdot K_F & 0 & -l \cdot K_F & 0 \\ K_M & -K_M & K_M & -K_M \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \quad (17)$$

No entanto, a saída final do controle em cascata é equivalente aos momentos necessários em cada eixo e, portanto, é necessaria a inversa da equação (17), de modo que o oposto seja verdadeiro. Isso resulta na equação (18), que é o *Rotor Mixer* do veículo.

$$\begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} = \begin{bmatrix} -\frac{1}{4K_F} & 0 & \frac{1}{2l \cdot K_F} & -\frac{1}{4K_M} \\ -\frac{1}{4K_F} & -\frac{1}{2l \cdot K_F} & 0 & -\frac{1}{4K_M} \\ -\frac{1}{4K_F} & 0 & -\frac{1}{2l \cdot K_F} & \frac{1}{4K_M} \\ -\frac{1}{4K_F} & \frac{1}{2l \cdot K_F} & 0 & -\frac{1}{4K_M} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \quad (18)$$

#### 4.2 Controle de translação

A arquitetura de controle é ilustrada na Fig. 5.

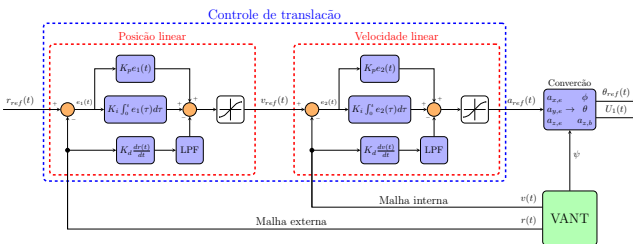


Figura 5. Controle linear malha interna e externa

Os dados de entrada deste controlador são a posição observada a partir das medições e a posição desejada como referência, que pode ser enviada um piloto ou por um algoritmo de navegação. Finalmente, a saída desse bloco é equivalente à velocidade desejada e é limitada pela saturação de acordo com a velocidade máxima que o veículo pode atingir.

Esta descrição se traduz nas seguintes equações para o erro, controlador e sua saída:

$$\begin{aligned} e_1(t) &= r_{ref} - r(t) \\ u(t) &= K_p e_1(t) + K_i \int_0^t e_1(\tau) d\tau - K_d \text{LPF} \left( \frac{dr(t)}{dt} \right) \\ v_{ref}(t) &= \text{SAT}(u(t)) \end{aligned} \quad (19)$$

A malha interna segue uma estrutura semelhante, no lugar das posições as velocidades são os dados de entrada:

$$\begin{aligned} e_2(t) &= v_{ref} - v(t) \\ u(t) &= K_p e_2(t) + K_i \int_0^t e_2(\tau) d\tau - K_d \text{LPF} \left( \frac{dv(t)}{dt} \right) \\ a_{ref}(t) &= \text{SAT}(u(t)) \end{aligned} \quad (20)$$

Deve-se converter os comandos de traslação gerados a angulos, por tanto é necessario encontrar uma função que relacione  $\mathbf{a}$  com  $\phi$  e  $\theta$ . As forças geradas pelos rotores no sistema inercial são considerados como ponto de partida.

$$\mathbf{a} = \frac{1}{m} R_{e/b} \mathbf{F}_r = \frac{F_{r,z}}{m} \begin{bmatrix} \cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi \\ \cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi \\ \cos \phi \cos \theta \end{bmatrix} \quad (21)$$

Logo, se lineariza para  $\phi$  e  $\theta$  proximos ao zero, onde é obtido:

$$\begin{bmatrix} a_x \\ a_y \end{bmatrix} = \frac{F_{r,z}}{m} \begin{bmatrix} \theta \cos \psi + \phi \sin \psi \\ \theta \sin \psi - \phi \cos \psi \end{bmatrix} \quad (22)$$

Finalmente, se colocam em evidencia  $\phi$  e  $\theta$ , obtendo assim os angulos de referencia para o controle de orientação em função da aceleração de referencia.

$$\begin{bmatrix} \phi_{ref} \\ \theta_{ref} \end{bmatrix} = \frac{m}{F_{r,z}} \begin{bmatrix} \sin \psi & -\cos \psi \\ \cos \psi & \sin \psi \end{bmatrix} \begin{bmatrix} a_{x,ref} \\ a_{y,ref} \end{bmatrix} \quad (23)$$

Para o controle de translação vertical consideando a componente Z da Equação (21) é obtido:

$$F_{r,z} = \frac{m}{\cos \phi \cos \theta} a_{z,ref} = U_1 \quad (24)$$

O resultado é usado como força de propulsão na equações do *Rotor Mixer*, o que permite a implementação do controle de altura.

## 5. SIMULADOR IMPLEMENTADO

Trabalhos como o de Ribeiro and Oliveira (2010), Figueiredo and Saotome (2012), Bittar et al. (2014) e Bouallègue and Fessi (2016) abordam esta temática através da difusão de simuladores de voo empregando o software X-Plane que fornece um ambiente virtual, seguro e muito mais realista para testar algoritmos de controle usando técnicas de simulação *hardware in the loop* cuja principal característica é o uso da placa de controle real, com todas as limitações de cálculo y memoria associadas. O Ambiente virtual implementado em X-plane é ilustrado na Fig. 6.

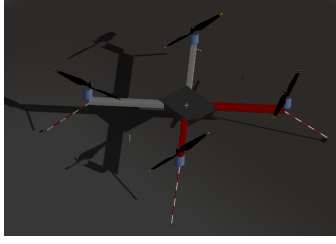


Figura 6. Quadcopter básico em X-Plane

A Figura 7 apresenta a implementação da técnica, onde observa-se a relação entre a placa de desenvolvimento (verde), o algoritmo de controle (azul), computador e periféricos (vermelho), software de simulação (laranja) e elementos de software complementar (preto).

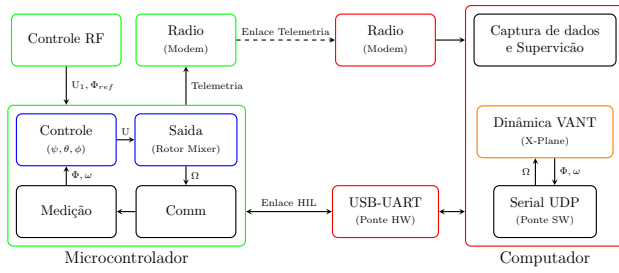


Figura 7. Arquitetura de simulação HIL com X-plane

A placa de desenvolvimento possui três meios de comunicação com o exterior, o enlace de controle de rádio que recebe os comandos (referências e empuxo total) do piloto, o modem para estabelecer o enlace de telemetria que permite monitorar e gravar as informações do desempenho do veículo e, finalmente, o enlace da simulação HIL necessário para estabelecer comunicação com o simulador.

O enlace HIL requer dois tipos de gateway, sendo o primeiro o hardware para converter a porta UART da placa de desenvolvimento à porta USB do computador e o segundo o software para converter esta porta serial na porta de rede UDP do simulador. É baseado no trabalho de Lewis (2016) que implementamos nosso software gateway no Python 3. A tabela 1 apresenta um resumo da estrutura de conteúdo que o pacote UDP deve ter.

Field	Offset	Lenght (bytes)	Type
SOF	0	4	uint8
Private	4	1	uint8
Identifier	5	4	uint32
Payload (1:8)	9	32	float

Tabela 1. Estrutura do protocolo UDP de X-Plane

Este protocolo possui três campos modificáveis, em que SOF indica o início de um novo pacote e sempre possui o valor hexadecimal [0x44, 0x41, 0x54, 0x41] ou DATA, se for traduzido para código ASCII. O campo de identificação indica o grupo da variável que contém o payload e sempre tem o mesmo comprimento, no entanto, um pacote UDP

contém tantos desses blocos quanto os grupos de variáveis ativadas.

O simulador X-Plane precisa de alguns ajustes na sua configuração para funcionar corretamente, sendo o mais importante ativar as variáveis que serão transmitidas pelo protocolo UDP. O conteúdo de cada pacote depende dessa configuração, como ilustrado na Figura 8

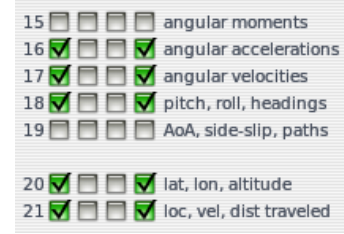


Figura 8. Configuração do software X-Plane

Outro parâmetro relevante é o aumento no número de iterações por quadro calculado pelo simulador, como ilustrado na Fig. 9<sup>3</sup>.

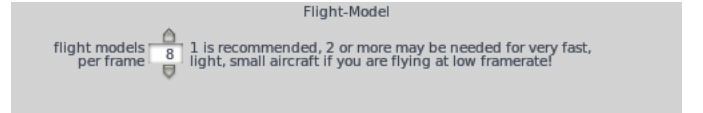


Figura 9. Configuração de otimização do software X-Plane

Finalmente, a Fig. 10. ilustra o resultado da adaptação, na que o veículo segue o caminho pré-programado.

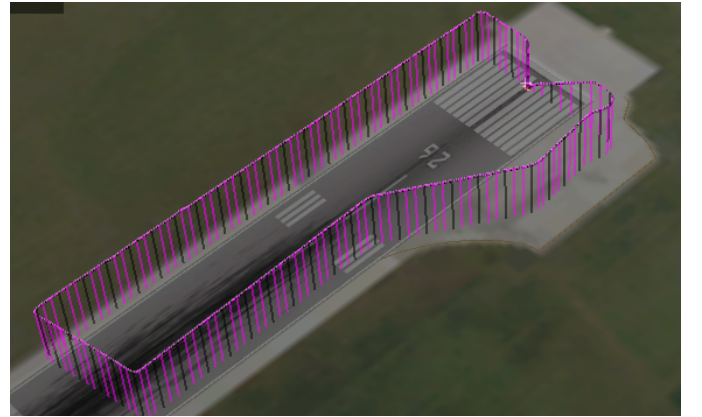


Figura 10. Resultado da simulação com X-Plane

## 6. DETALHES DO PROCESO DE SINTONIA DOS CONTROLADORES

Considerando três parâmetros por cada PID, resulta em um total de pelo menos 18 ganhos de rotação e o mesmo número de translação, isso sem considerar os limites da função de saturação. Consequentemente, ajustar os ganhos para um desempenho global ideal do sistema é um problema complexo de fato, neste sentido passamos a explicar como foi feito o ajuste.

<sup>3</sup> De outra forma poderia ser instável.



Optou-se por utilizar o método Ziegler-Nichols para determinar os parâmetros de forma experimental, começando pelos controladores que atuam diretamente nos atuadores (malha interna) e terminando em aqueles de nível mais alto (malha externa). O processo de ajuste é resumido como:

- Desconecte a referência do controlador do sistema em cascata e defina para a mesma um valor de memória
- Ajuste o ganho proporcional
- Ajuste o ganho derivado e ajuste o ganho proporcional para otimizar o tempo de resposta
- Ajuste o ganho integral
- Ajuste os limites de saturação
- Reconecte a referência

Este procedimento é feito com a inclusão de duas peças, sendo a primeira o hardware ST-Link presente na placa de desenvolvimento, que permite a leitura e modificação dos valores na RAM sem interromper o processador e também controlar o fluxo do programa usando pontos de interrupção. A segunda peça é o elo de telemetria que permite observar em tempo real, qualquer variável ou estado na memória RAM. O conjunto abre a possibilidade de modificar os ganhos e observar seus efeitos durante o funcionamento do sistema, simplificando drasticamente a tarefa de ajustá-los.

## 7. RESULTADOS EXPERIMENTAIS

Se procedeu à construção de uma placa eletrônica para ser usada nas provas com o simulador HIL, seguidamente esta placa foi montada no protótipo. O protótipo e o ambiente de teste são ilustrados na Fig. 11.



Figura 11. Protótipo (superior), placa de controle (inferior)

As diretrizes expostas por Lim et al. (2012) foram a base para a pesquisa e seleção dos componentes e, no final, optou-se pela placa **STM32F3Discovery** como o núcleo do sistema de controle, essa placa já possui sensores básicos,  $\mu$ controlador e programador, portanto apresentou-se adequado para um primeiro protótipo. Além disso, um controle de rádio de 6 canais **FS-CT6B** foi usado para transmissão de comando, o módulo **HC-05** para o enlace de telemetria e uma placa com o chip **FT232R** para o gateway **USB-UART** com o simulador.

A Tabela 2 apresenta a configuração dos componentes utilizados ressaltando que a estrutura, motores e ESC

são parte do kit básico a partir das quais foram feitas as modificações para acoplar a placa de controle.

Dada a diferença de frequências entre todos os componentes, foi escolhido o sistema operacional em tempo real FreeRTOS, atribuindo a maior prioridade às rotinas de aquisição e estimativa, seguido pelas tarefas de controle que operam em frequências de 500 [Hz] para controle de orientação e 100 [Hz] para controle de posição. Finalmente, a prioridade mais baixa é atribuída às rotinas de manipulação do link de telemetria, sendo 50 [Hz] a frequência de execução.

Tabela 2. Configuração componentes

Componente	Frequência operativa	Configuração restante	Interface
Giroscópio	760Hz	2000DPS	SPI
Acelerometro	1344Hz	4G	I2C
Magnetometro	220Hz	1.3GA	I2C
Barômetro	100Hz (4096OSR)		SPI
GNSS	10Hz	4G	UART
Ultrason	25Hz	2M/2us	TIMER
Bateria	50Hz	6800mV/V	ANALOG
ESCs	400Hz	500-2500us	TIMER
Receptor	50Hz	500-2500us	TIMER
Bluetooth	100Hz	1382400bps	UART

Após validar os algoritmos de controle e o *firmware* da placa eletrônica, passou-se à execução de provas reais. As mesmas que foram executadas em campo aberto a 2575 metros acima do nível do mar com céu nublado e vento leve de ( $\approx 15[\text{km/h}]$ ).

O tempo da prova foi de 15 minutos com uma bateria de 5000 mAh instalada, durante a qual o protótipo foi elevado a uma altura de 3 [m] acima do nível do solo e, em seguida, realizou uma série de manobras nos três eixos, neste tempo foi feita a captura de dados para avaliação do sistema de controle do protótipo fazendo uso do enlace de telemetria.

Os resultados das provas para o controle de altura  $z$  são ilustrados na Fig. 12.

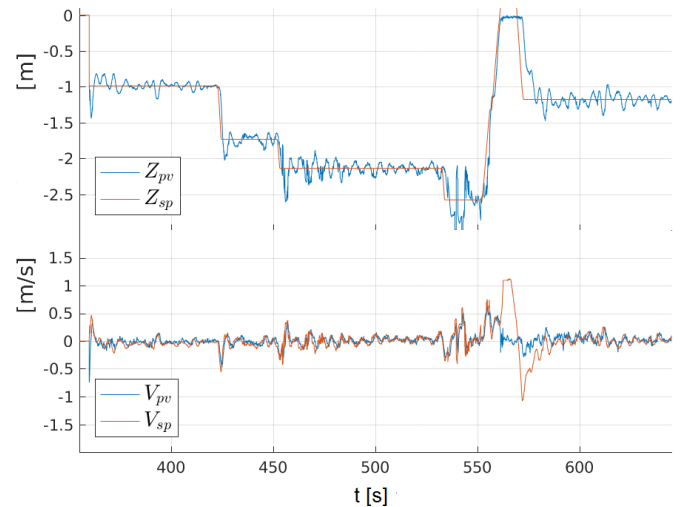


Figura 12. Altura e velocidade linear real e desejada

Os resultados dos testes para as rotações  $\phi$ ,  $\theta$  e  $\psi$  são ilustrados nas Figuras 13 a 15.

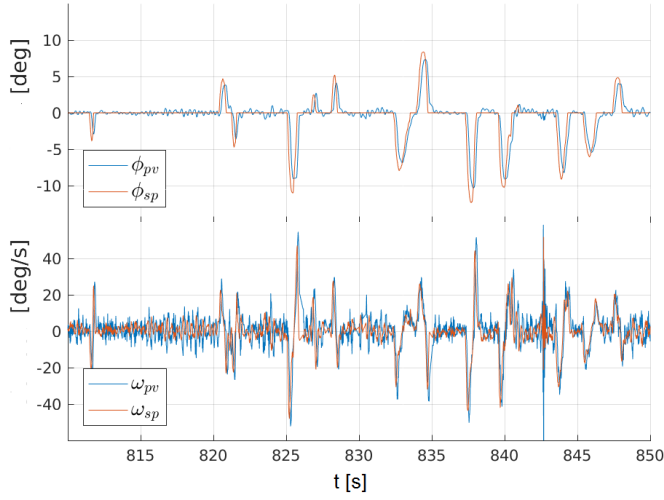


Figura 13. Ângulo  $\phi$

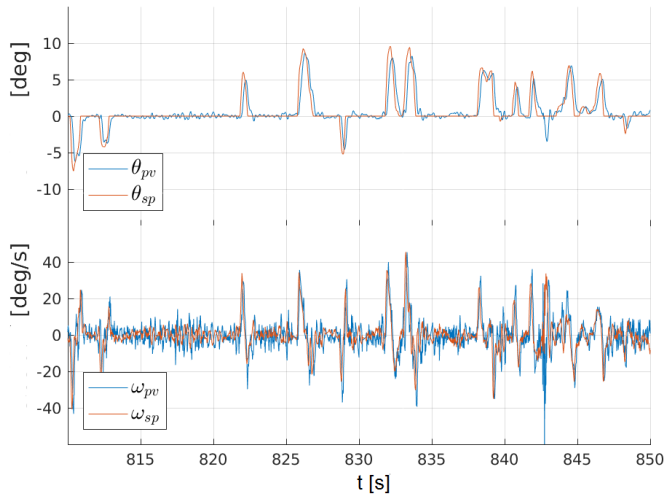


Figura 14. Ângulo  $\theta$

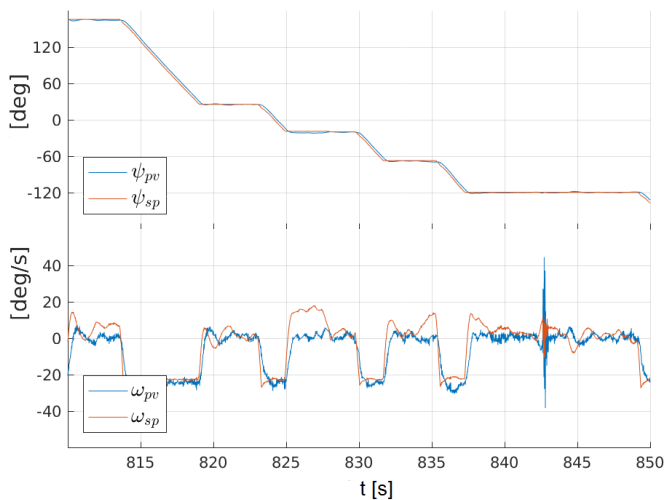


Figura 15. Proa  $\psi$

Os resultados apresentam o desempenho do sistema de controle muito satisfatório, tanto na malha externa (ângulos de navegação) quanto da malha interna (velocidade angular), para os três eixos de rotação. O desempenho também é satisfatório para o controle de altura.

Foi observado o monitoramento correto do sistema de controle, mesmo na presença de perturbações, como no instante 842 [s], onde um vento consideravelmente forte foi percebido o mesmo que foi compensado com sucesso.

## 8. CONCLUSÕES

O tempo entre a revisão bibliográfica, a aquisição de conhecimento em aeronáutica num país como Bolívia sem nenhuma tradição Vargas and Pozo (2014), a implementação do simulador, a construção do protótipo e a realização das provas de campo foi de aproximadamente três anos. Neste tempo diferentes problemas foram gradualmente superados, entre eles a falta de verba para a compra de componentes, em alguns casos o excessivo tempo para a chegada dos componentes adquiridos, etc. No entanto a etapa mais marcante foi aquela na que foi adquirido o conhecimento sob sensores, filtragem e estimação de sinais, os conceitos de simulação *software in the loop* e/ou *hardware in the loop* ate a finalização do protótipo foram extenuantes com a dificuldade inerente deste tipo de trabalho. Desta imensurável experiência algumas conclusões são:

- A simulação do tipo HIL tem as mesmas vantagens que o tipo SIL, além da capacidade de detectar e corrigir, desde cedo, os problemas que podem surgir devido às limitações técnicas do hardware. No entanto, a simulação SIL é especialmente útil em situações em que o hardware para montar uma simulação HIL ainda não está disponível menos ainda um protótipo.
- A simulação HIL, é muito útil para detectar erros na implementação antes de testar os sistemas reais, reduzindo o tempo e os custos associados a testes. Também vemos o uso potencial da técnica para ajudar no processo de seleção, teste e otimização de componentes, ou mesmo no *benchmarking* entre variantes de implementação de sistema.
- O sistema de controle em cascata apresentou-se adequado para o controle das rotações e as velocidades angulares do veículo assim como para o controle de altitude e velocidade linear do movimento de translação.
- Para chegar nos resultados apresentados neste trabalho a bagagem matemática (sistemas de referencia e modelagem), a composição de sinais a partir de diferentes sensores, filtragem, assim como a implementação de controladores e/ou algoritmos em tempo real teve que ser bem reforçada. Neste sentido outros artigos vêm sendo escritos para tocar os pontos que por falta de espaço não foram incluídos neste trabalho.

## REFERÊNCIAS

- Beauregard, B. (2011). Improving the beginner's pid. <http://brettbeauregard.com>.
- Bittar, A., Figueredo, H.V., Guimaraes, P.A., and Mendes, A.C. (2014). Guidance software-in-the-loop simu-

- lation using x-plane and simulink for uavs. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 993–1002. doi:10.1109/ICUAS.2014.6842350.
- Blevins, T. and Nixon, M. (2011). *Control Loop Foundation: Batch and Continuous Processes*. International Society of Automation.
- Bouadi, H., Bouchoucha, M., and Tadjine, M. (2007). Sliding mode control based on backstepping approach for an uav type-quadrotor. In *International Journal of Applied Mathematics and Computer Sciences, Vol.4, No.1*, 12–17.
- Bouallègue, S. and Fessi, R. (2016). Rapid control prototyping and pil co-simulation of a quadrotor uav based on ni myrio-1900 board. *International Journal of Advanced Computer Science and Applications*, 7(6). doi:10.14569/IJACSA.2016.070604.
- Brandt, J. and Selig (2011). Propeller performance data at low reynolds numbers. In *49th AIAA Aerospace Sciences Meeting*, 2011–1255.
- Cai, G., Chen, B.M., and Lee, T.H. (2011). *Unmanned Rotorcraft Systems*. Springer London. doi:10.1007/978-0-85729-635-1.
- Cho, Y., Kim, H.S., and Kim, D.Y. (2014). A QuadCopter posture stabilization control using noise attenuated sensor fusion and modified PID control. *Science and Engineering Research Support Society*. doi:10.14257/astl.2014.66.12.
- Cooper, D.J. (2015). The cascade control architecture. <http://controlguru.com/the-cascade-control-architecture/>.
- Diebel, J. (2006). Representing attitude: Euler angles, unit quaternions, and rotation vectors.
- Figueiredo, H.V. and Saotome, O. (2012). Simulation platform for quadricopter: Using matlab/simulink and x-plane. In *Proceedings of the 2012 Brazilian Robotics Symposium and Latin American Robotics Symposium*, SBR-LARS '12, 51–55. IEEE Computer Society, Washington, DC, USA. doi:10.1109/SBR-LARS.2012.15.
- Lewis, J. (2016). X-plane, udp, and visual basic, for x-plane version 9. [http://www.jefflewis.net/XPlaneUDP\\_9.html](http://www.jefflewis.net/XPlaneUDP_9.html).
- Lim, H., Park, J., Lee, D., and Kim, H.J. (2012). Build your own quadrotor: Open-source projects on unmanned aerial vehicles. *IEEE Robot. Automat. Mag.*, 19(3), 33–45. doi:10.1109/MRA.2012.2205629.
- Nagaty, A., Saeedi, S., Thibault, C., Seto, M., and Li, H. (2012). Control and navigation framework for quadrotor helicopters. *Journal of Intelligent & Robotic Systems*, 70(1-4), 1–12. doi:10.1007/s10846-012-9789-z.
- Quiroga, M. (2018). Plataforma para la experimentación con vehículos aéreos no tripulados de ala rotativa TCC. Universidad Simón I. Patiño - Bolivia.
- Ribeiro, L.R. and Oliveira, N.M.F. (2010). Uav autopilot controllers test platform using matlab/simulink and x-plane. In *2010 IEEE Frontiers in Education Conference (FIE)*, S2H–1–S2H–6. doi:10.1109/FIE.2010.5673378.
- Stevens, B., Lewis, F., and Johnson, E. (2015). *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. Wiley.
- Stingu, E. and Lewis, F. (2009). Design and implementation of a structured flight controller for a 6dof quadrotor using quaternions. In *Control and Automation, 2009. MED '09. 17th Mediterranean Conference on*, 1233–1238. doi:10.1109/MED.2009.5164715.
- Szafranski, G. and Czyba, R. (2011). Different approaches of pid control uav type quadrotor. In G. de Croon and M. Amelink (eds.), *Proceedings of the International Micro Air Vehicle Conference and Flight Competition 2011 Summer Edition*, 73–78. 't Harde, the Netherlands. doi:10.4233/uuid:eadf2fe7-7e5a-4cf8-88e5-6c247f5b6fa9.
- VanDoren, V. (2014). Fundamentals of cascade control. <http://www.controleng.com/single-article/fundamentals-of-cascade-control/bcedad6518aec409f583ba6bc9b72854.html>.
- Vargas, F.J.T. and Pozo, E.O. (2014). *Modelado, Simulación y Control de Sistemas Mecatronicos y Aeroespaciales con Matlab-Simulink y Mathematica*. Sociedad de Ingenieros de Bolivia - CBBA.