CONCEPÇÃO DE UMA PLATAFORMA EXPERIMENTAL PARA ESTUDO DE CONTROLE DE UM MODELO REDUZIDO DE HELICÓPTERO

Antônio Padilha Lanari Bo
Hélio Henrique Fonseca Miranda

Monografia apresentada ao Curso de Engenharia Mecatrônica da Universidade de Brasília como requisito parcial para a obtenção do título de Engenheiro de Controle e Automação.

Orientador: Prof. Geovany Araújo Borges (ENE)

Brasília, Distrito Federal, Brasil
Junho, 2004

Dedicatória

A minha mãe, Maria Beatriz Oswald Padilha, e todos da minha família, dedico este trabalho.

Antônio Padilha Lanari Bo

A Heleno Vieira Miranda e Marlucy Vieira Miranda, dedico este trabalho e a conclusão do curso de engenharia.

Hélio Henrique Fonseca Miranda

Agradecimentos

Gostaríamos de expressar nossa mais profunda gratidão ao Prof. Geovany Araújo Borges, orientador do projeto, pela dedicação, pela disponibilidade em horários não muito convencionais, por escutar nossas propostas, pela paciência com nossas dúvidas e pelas soluções corretas aos mais variados problemas encontrados. Além disso, somos gratos por ter renovado nossa crença de que a universidade ainda pode ser o ambiente ideal para produção e difusão de conhecimento.

Agradecemos aos mestrandos e graduandos do Laboratório do Controle e Visão por Computador, em especial ao Flávio Vidal de Barros, pelos vídeos demonstrativos do helimodelo e pelos momentos de descontração.

Agradecemos também aos técnicos do SG-9, em especial ao Artur, Marcos e Xavier, que foram indispensáveis para a construção do suporte mecânico.

Gostaríamos de agradecer também à Júlia Campos Clímaco pelo auxílio na redação deste documento.

Pelo apoio e pela infra-estrutura disponível no laboratório, somos gratos ao Prof. Victor Hugo Casanova Alcalde.

Sumário

Lista de ilustrações

Lista de símbolos e abreviações

UAV	Aeronave não-tripulada, do inglês Unmaned Aerial Vehicle
VTOL	Veículo de aterrissagem e pouso vertical, do ingês Vertical Take Off and Landing
θ	Ângulo de arfagem [rad]
ϕ	Ângulo de rolagem [rad]
ψ	Ângulo de guinada [rad]
δ_a	Ângulo de incidência proporcionado pelo servo de arfagem [rad]
δ_r	Ângulo de incidência proporcionado pelo servo de rolagem [rad]
I_{Ma}	Momento de inércia referente à arfagem [Kg.m^2]
T_P	Torque devido ao empuxo gerado pelo rotor principal [N.m]
T_A	Torque gerado pelo atrito [N.m]
T_M	Torque gerado pela massa do corpo do helicóptero [N.m]
T_G	Torque gerado pelo efeito giroscópico [N.m]
T_C	Torque gerado pelo cíclico [N.m]
T_T	Torque devido ao empuxo gerado pelo rotor traseiro [N.m]
l_M	Distância do centro de gravidade ao centro de rotação [m]
m	Massa do sistema [kg]
g	Aceleração da gravidade [m/s^2]
ω_P	Velocidade de rotação do rotor principal [rad/s]

l_P	Distância paralela ao eixo do helicóptero entre o rotor principal e ao centro da junta [m]
r_P	Raio do rotor principal [m]
μ_{Aa}	Coeficiente de atrito viscoso da arfagem [Kg.m/s.rad]
K_{Aa}	Componente de atrito estático da arfagem, que depende da força normal total aplicada [N.m]
ω_T	Velocidade de rotação do rotor de cauda [rad/s]
l_T	Distância paralela ao eixo do helicóptero entre o rotor de cauda e ao centro da junta [m]
r_T	Raio do rotor de cauda [m]
I_{Mr}	Momento de inércia referente à rolagem [Kg.m^2]
μ_{Ar}	Coeficiente de atrito viscoso da rolagem [Kg.m/s.rad]
K_{Ar}	Componente de atrito estático da rolagem, que depende da força normal total aplicada [N.m]
I_{Mg}	Momento de inércia referente à guinada [Kg.m^2]
T_U	Torque gerado pelo rotor de cauda [N.m]
T_R	Torque gerado pela reação ao rotor principal [N.m^2]
I_{Mr}	Momento de inércia do rotor principal [Kg.m]
K_R	Constante determinada pela viscosidade do ar, pelo formato da pá e pelo ângulo de incidência.
S_D	Área frontal da pá, perpendicular ao plano de rotação [m^2]
T	Período de amostragem [s]
K_P	Ganho proporcional
T_I	Termo integral
T_D	Termo derivativo
m	Número de “atrasos” de entrada
n	Número de “atrasos” de saída

Resumos

Resumo: No desenvolvimento de um helicóptero autônomo, um dos pontos-chave e principais problemas é o controle de atitude, que diz respeito ao controle dos movimentos de arfagem, rolagem e guinada. Esse trabalho descreve o projeto de uma plataforma experimental construída para prover um maior conhecimento sobre a dinâmica de atitude de um modelo reduzido de helicóptero. O projeto dessa plataforma abarca a concepção de um suporte que possibilite o movimento do helicóptero nos três graus de liberdade, bem como o projeto da instrumentação necessária para a aquisição de dados, acionamento dos atuadores e interface com o PC, que foi a plataforma de controle digital escolhida. Ela deve possibilitar o estudo de estratégias de controle de atitude para um modelo em escala reduzida de helicóptero.

Abstract: *In the development of a fully autonomous helicopter, one of the key points and major problems is the attitude control, which holds for the control of the pitch, roll and yaw motions. This work describes the project of a experimental workbench built to provide a better understanding of the attitude dynamics of a scale model helicopter. The workbench's design includes the mechanical structure which allows the movement in the three degrees of freedom, and also the instrumentation needed for data aquisition, actuators driving and PC interfacing, which was the chosen platform for digital control. The workbench must provide the possibility to a comprehensive study of control strategies to a model scale helicopter.*

1. Introdução

1.1. Definição do problema

As primeiras aeronaves não-tripuladas ou UAVs (*Unmanned Aerial Vehicle*, em inglês) surgiram na década de 60 em missões militares de reconhecimento em áreas de inimigas. Desde então, a importância desse tipo de veículo tem sido cada vez maior, em especial para o caso das aeronaves autônomas.

Os veículos aéreos de decolagem e aterrissagem vertical ou VTOL (*Vertical Take Off and Landing*, em inglês) também têm merecido muita atenção desde o início do século 20, devido às diversas aplicações em que seu uso é mais indicado em relação às aeronaves de asa fixa.

Para o projeto de um sistema VTOL não-tripulado é necessário o desenvolvimento de um sistema complexo de controle. Entretanto, uma das primeiras etapas a ser realizada diz respeito ao estudo da atitude da aeronave. A atitude corresponde à posição nos três eixos de orientação. Esse estudo é importante pois é a partir de um controle preciso da atitude da aeronave que se torna possível o controle da trajetória e a navegação da mesma.

Este projeto propõe a concepção de uma plataforma experimental para o estudo de estratégias de controle digital da atitude de um modelo de helicóptero em escala reduzida, a plataforma VTOL escolhida.

1.2. Contextualização

Um projeto que envolve diversas áreas dentro da engenharia vem ao encontro dos objetivos principais de se realizar um projeto final de graduação em engenharia.

No contexto da engenharia mecatrônica, o nosso projeto lida com múltiplos assuntos abordados durante o curso, tais como o projeto mecânico do suporte e da junta universal, o projeto eletrônico da instrumentação de controle e da parte de alta potência e o controle propriamente dito. Nesse sentido, o projeto cumpre os objetivos pedagógicos que justificam sua existência.

Além disso, o projeto no Brasil de concepção de um VTOL autônomo é pioneiro no Brasil, de acordo com a pesquisa bibliográfica, e traz novas perspectivas de pesquisa para a Universidade de Brasília, em especial para o Laboratório de Controle e Visão por Computador, onde a robótica aérea pode vir a se tornar um proeminente campo de estudo.

Para a sociedade brasileira como um todo, para a qual a pesquisa universitária se pretende, o projeto pode trazer grandes benefícios. Em especial, pode-se citar as aplicações deste sistema no sensoriamento remoto aplicado à agricultura e à preservação do meio ambiente, a inspeção de linhas de transmissão e parques industriais, a vigilância em zonas urbanas, entre outras. Em todas essas aplicações, um modelo reduzido e autônomo de helicóptero pode realizar tais tarefas com maior eficiência e menor custo.

1.3. Objetivos do projeto

O projeto dessa plataforma experimental envolve a fabricação de um suporte mecânico que possibilite o movimento do helicóptero nos três graus de liberdade, o projeto dos circuitos de acionamento dos atuadores e interface com o PC e a concepção de um software de monitoramento e controle.

Um dos requisitos principais deste projeto é que o suporte restrinja o mínimo possível o movimento do helicóptero, ao mesmo tempo que deve possibilitar a medição dos ângulos correspondentes.

O projeto dos circuitos de acionamento dos atuadores e interface com o PC, que foi a plataforma de controle digital escolhida, compreende a concepção de toda instrumentação necessária para atingir os requisitos da plataforma experimental.

A conclusão desses objetivos principais deve prover um ambiente de estudo das diferentes estratégias de controle de atitude do helicóptero. Assim, o futuro usuário do sistema não terá necessidade de se preocupar com questões relativas ao acionamento e aquisição de dados do processo. Caberá a ele apenas a computação da lei de controle a ser utilizada.

1.4. Apresentação do manuscrito

No capítulo 2 é apresentada uma revisão bibliográfica sobre o helicóptero. São descritos também pesqui Em seguida, no capítulo 3 descreve o projeto da plataforma experimental, abordando todas os problemas encontrados e as devidas soluções propostas. No capítulo seguinte é discutido o modelamento matemático do modelo utilizado. Os resultados experimentais são apresentados no capítulo 5, que é seguido que é seguido das conclusões na seção 5. Os anexos contém material complementar.

2. O helicóptero

Neste capítulo encontra-se a maior parte da pesquisa bibliográfica efetuada para a realização do projeto. A primeira parte do capítulo refere-se ao estudo do próprio helicóptero, enquanto que uma segunda seção trata de artigos sobre sistemas semelhantes ao nosso.

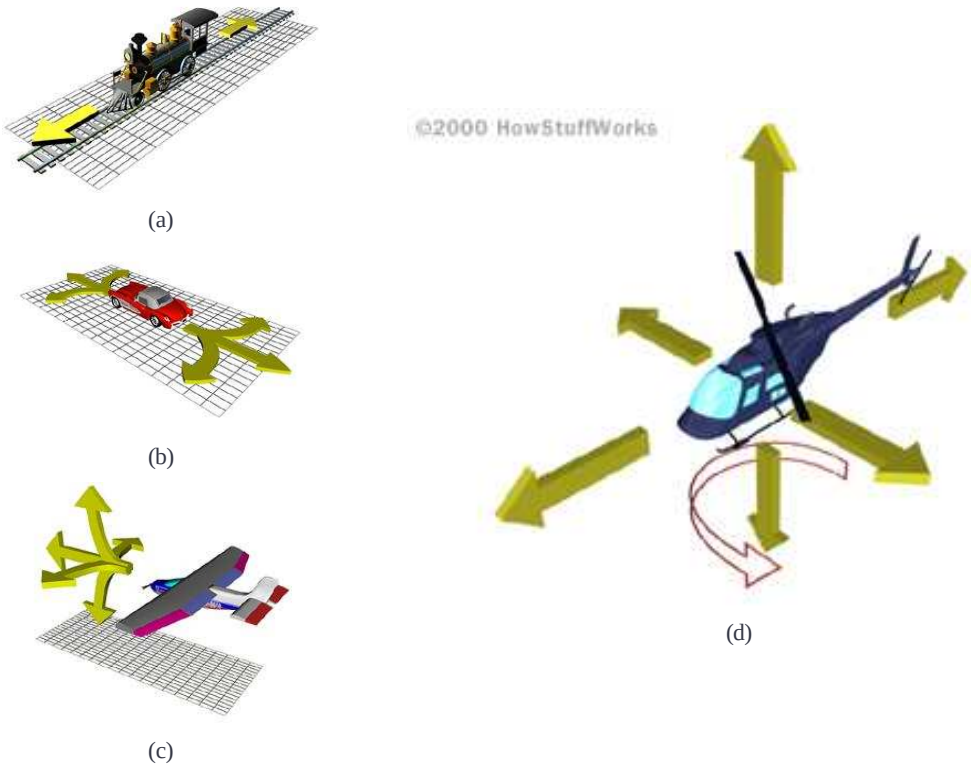
2.1. Introdução

Ao compararmos alguns dos meios de transporte atualmente disponíveis, podemos identificar a altíssima versatilidade apresentada pelo helicóptero. Como pode ser observado na Figura 2.1, o helicóptero proporciona total liberdade de movimento.

As principais aplicações de um veículo com essa versatilidade são no transporte aéreo em que as condições de pouso excluem o avião e nas aplicações derivadas do vôo pairado do helicóptero. Entre essas, pode-se citar o resgate de pessoas em situações de emergência, as filmagens e fotografias aéreas, transporte em locais de difícil acesso, etc.

Essa isenção de restrições traz determinados inconvenientes. Por exemplo, o helicóptero é um sistema inerentemente instável, que demanda um ajuste constante das variáveis de controle pelo piloto ou operador. Um outro aspecto é que a dinâmica que rege o seu comportamento é bastante complexa e varia enormemente entre os diferentes modos de vôo, tornando mais difícil o emprego a utilização de técnicas convencionais de controle.

Além disso, não se pode deixar de citar algumas desvantagens que o helicóptero possui em comparação com o avião, como reduzidas velocidade, altitude e alcance.



(a)

(b)

(c)

(d)

Figura 2.1 - Comparativo das possibilidades de movimento do (a) trem, (b) automóvel, (c) avião e (d) helicóptero.

2.2. Um breve histórico

A história da aeronave com a asa rotativa é muito antiga e, como em vários outros meios de transporte, foi construída por meio de muito trabalho, dedicação, insucessos e conquistas. A primeira referência encontrada é a de Leonardo da Vinci, que projetou uma aeronave com uma asa em espiral em torno de um eixo central[1] (Figura 2.2(a)). A próxima menção histórica relevante se dá somente no início do século XX, quando o francês Breguet[2], construiu um helicóptero formado por dois rotores coaxiais girando em sentido contrário, de modo a anular o efeito do torque no corpo do helicóptero devido à reação da

[1] Surge, então, o nome helicóptero, cuja etimologia é formada pelas palavras gregas *helix* (helicóide) e *pteron* (asa).
[2] Louis Breguet foi uma dos maiores personalidades da aviação mundial. Entre outros feitos, formulou uma relação que prevê o alcance das aeronaves por meio da taxa de consumo de combustível.

transmissão de força ao rotor. Apesar de ter sido responsável pelo primeiro vôo de sucesso da história do helicóptero, Breguet teve sérios problemas com o controle e estabilidade dele.

A partir de então, foram construídos vários protótipos que utilizavam dois rotores, porém com os mesmos problemas já citados. Somente no final da década de 30, que o russo Igor Sikorsky apresentou o primeiro modelo prático de helicóptero com apenas um rotor, o VS-300 (Figura 2.2(b)). Os anos seguintes foram marcados pela consolidação do helicóptero que conhecemos hoje, bem como por inúmeras pesquisas sobre modelos alternativos, como quadri-rotores, autogiros e aeronaves híbridas que buscam conciliar características tanto do helicóptero, quanto do avião, como o V-22 Osprey (Figura 2.2(c)).

(a) (b) (c)

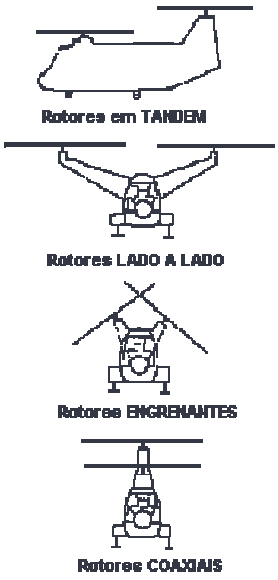
Figura 2.2 – (a) Esboço do helicóptero de Leonardo da Vinci, (b) o VS – 300 de Sikorsky , (c) o V-22 Osprey

O modelo idealizado por Sikorsky trouxe diversas inovações que libertaram o desenvolvimento do helicóptero após anos de estagnação. De uma forma geral, pode-se dizer que há muito tempo já estava difundida pelo mundo a idéia de um aparelho que possibilitasse o vôo vertical por meio de um sistema de asas rotativas, que forneceriam sustentação. Porém, além da necessidade de utilizar-se um motor de peso reduzido e potência considerável, havia a necessidade de encontrar soluções viáveis para o torque contrário à rotação exercida pelo rotor principal e para a geração de propulsão para os movimentos de arfagem e rolagem (descrição destes termos na seção 2.1.2.1). Essas questões foram solucionadas, respectivamente, através da instalação do rotor de cauda e do

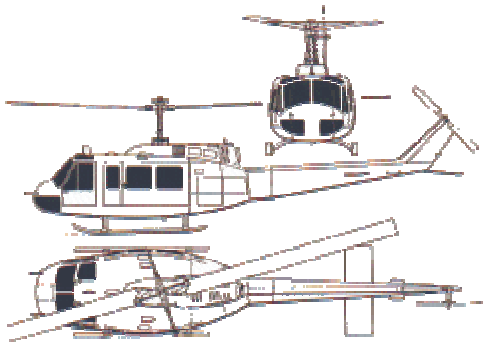
mecanismo de posicionamento dos ângulos de incidência, explicados mais adiante. Essas soluções deram forma a maior parte dos helicópteros de hoje.

2.3. Princípios de funcionamento

A maioria dos helicópteros que vemos hoje em dia possui um grande rotor principal para sustentação e um rotor de cauda para estabilidade. Sabemos que existem vários outros modelos de aeronaves de asas rotativas como mostrado na Figura 2.3(a). Mas aqui vamos abordar os princípios básicos de funcionamento do helicóptero mais comum. O helicóptero com rotor principal e rotor de cauda, ilustrado na Figura 2.3(b).



(a)



(b)

Figura 2.3 – (a) Modelos de aeronaves com asas rotativas. (b) Modelo com rotor principal e rotor de cauda.

Essencialmente, o helicóptero é sustentado por um rotor principal (Figura 2.4), uma grande hélice que gira em torno de um eixo vertical. As lâminas do rotor são semelhantes a asas de avião (Figura 2.5) e, à medida que giram, deslocam uma grande massa de ar gerando empuxo vertical para cima.

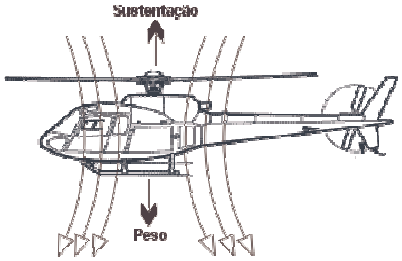


Figura 2.4 – Princípio de sustentação de um helicóptero.

Figura 2.5- Seção transversal da hélice de um helicóptero (Fonte: www.jhelicopteros.com.br).

Um problema encontrado é que, quando o rotor gira, o corpo do helicóptero recebe um torque de reação que o faz girar no sentido contrário. O rotor de cauda tem, portanto, o papel de se opor a esse efeito. Os modelos que não possuem rotor de cauda compensam o torque de reação de outras formas, como dois rotores principais girando em sentidos opostos, por exemplo.

O rotor de cauda também é chamado de rotor antitorque, justamente porque impede que o helicóptero gire em torno de si mesmo devido ao torque de reação ao giro do rotor principal.

O grande desafio dos idealizadores do helicóptero era, além de estabilizá-lo, controlar sua direção e trajetória. Para um helicóptero voar plenamente é necessário poder pilotá-lo de forma segura. Como já foi visto, a grande versatilidade do helicóptero vem, justamente, da sua grande possibilidade de movimentos. Enquanto um avião tem de mover-se para frente, um helicóptero pode deslocar-se tanto horizontalmente (para trás ou para frente), quanto verticalmente (para cima ou para baixo), e ainda pode deslocar-se

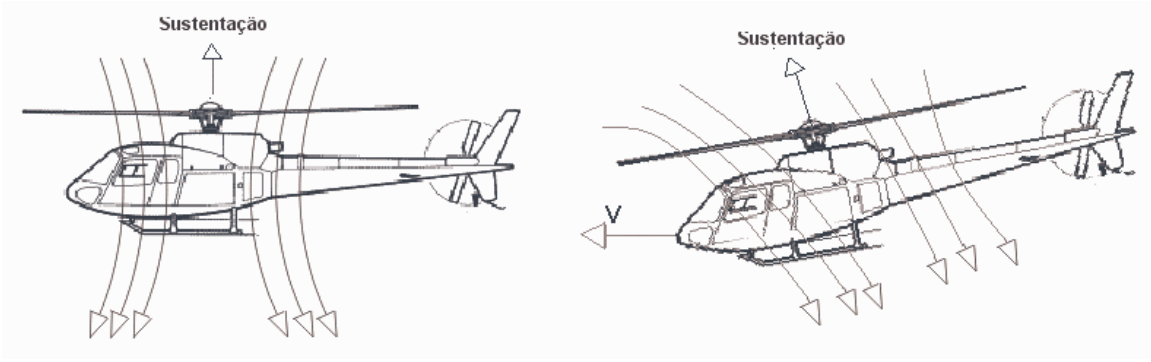
lateralmente (para a esquerda ou para a direita) ou fazer uma combinação de todos esse movimentos (Figura 2.6).

A seguir serão explicados, de maneira simplificada, os princípios de funcionamento do helicóptero e como é possível fazê-lo voar de forma segura.

2.3.1. Tipos de vôos

Existem dois tipos básicos de vôos de helicópteros. O *vôo pairado* e o v*ôo em translação*.

No vôo pairado, o helicóptero é capaz de parar no ar. A velocidade vertical e a velocidade horizontal são nulas em relação ao solo. Esse tipo de vôo exige muita concentração e habilidade do piloto.



(a) (b)

Figura 2.6 - Vôo pairado (a) e vôo em translação (b).

No vôo em translação o helicóptero se desloca em relação ao solo. A partir de uma determinada velocidade, a dinâmica do vôo do helicóptero é alterada. O rotor principal passa a funcionar tanto como propulsor quanto como asa. Como propulsor, porque fornece o empuxo necessário para deslocar o helicóptero para frente; como asa, porque o rotor deflete para baixo o fluxo frontal de o ar, como a asa fixa de um avião. Esse efeito é conhecido como *lift de translação* (Figura 2.7). A sustentação gerada pelo lift de translação aumenta a eficiência do vôo . De fato, para a maioria dos helicópteros comerciais, a velocidade ótima de cruzeiro é em torno de 80 km/h.

Para voar para frente, o helicóptero precisa inclinar-se ligeiramente para frente. Para voar para a esquerda, o helicóptero precisa inclinar-se ligeiramente para esquerda e assim por diante. Pode-se concluir que a inclinação do helicóptero em relação ao plano horizontal é fundamental para definir sua direção, sentido e velocidade.

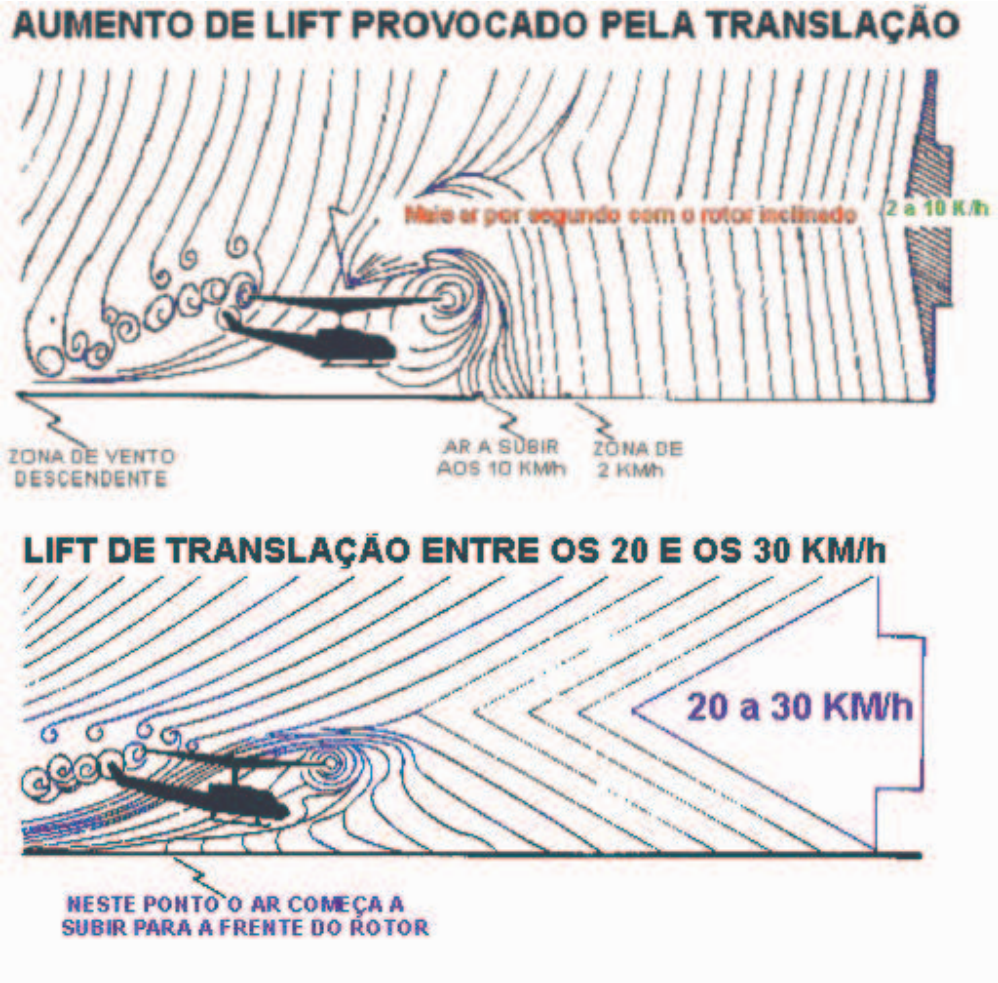


Figura 2.7 - Lift de translação. À medida que o fluxo frontal de ar aumenta, ocorre um aumento da sustentação gerada pelo rotor principal (Fonte: www.jhelicopteros.com.br).

Existem três movimentos principais que devem ser abordados: arfagem, rolagem e guinada. A arfagem constitui o movimento de rotação em torno do eixo horizontal, a

rolagem é o movimento de rotação em torno do eixo longitudinal e a guinada é o movimento de rotação em torno do eixo vertical (Figura 2.8).

2.3.2. Atitude

Arfagem, rolagem e guinada são variáveis importantes que modificam o que se chama de atitude do helicóptero. A atitude é a postura do helicóptero em relação ao plano horizontal e é medida por três ângulos: o ângulo de arfagem (θ), o ângulo de rolagem (ϕ) e o ângulo de guinada (ψ).

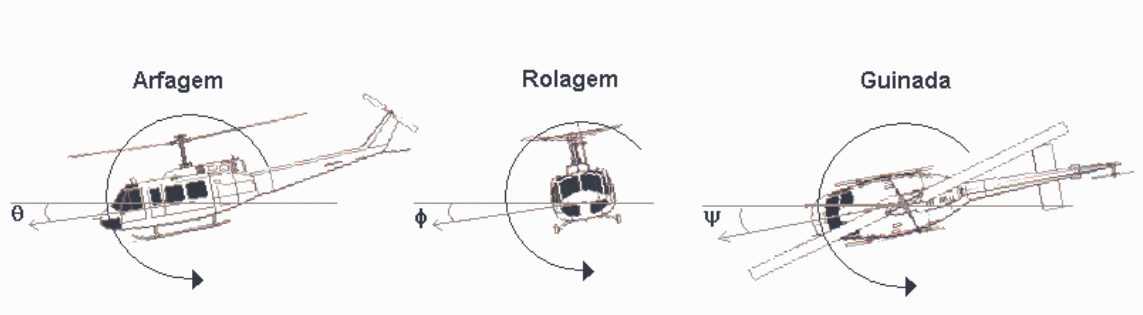


Figura 2.8 – Os três eixos de rotação do helicóptero.

Apesar de ter apenas duas hélices, o helicóptero é capaz de atuar nos três movimentos (arfagem, guinada e rolagem). É fácil entender que variando a força aplicada pelo rotor de cauda é possível controlar a guinada. Entretanto, como fazer o helicóptero inclinar-se para frente ou para os lados? Para responder a essa pergunta, é preciso primeiro entender como é constituído o rotor principal.

2.3.3. Rotor principal

O rotor principal é constituído de lâminas com passo variável. Isto significa que o ângulo de incidência das lâminas (que são, na verdade, asas) pode mudar de acordo com o comando do piloto. As lâminas são presas a hastes que controlam sua inclinação. As hastes, por sua vez, são presas a uma importante estrutura do helicóptero: a estrela rotativa (Figura 2.9).

A estrela rotativa pode subir e descer fazendo o passo das lâminas diminuir ou aumentar. Além disso, ela pode inclinar fazendo o passo de uma lâmina aumentar e o da outra diminuir na mesma proporção. A estrela rotativa está em contato com a estrela estacionária, que tem esse nome porque não gira em relação à fuselagem do helicóptero. Ela também sobe, desce ou se inclina, pela ação de barras de comando. É a estrela estacionária que transfere o movimento das barras de comando para a estrela rotativa que, por sua vez, o transfere para as lâminas do rotor.

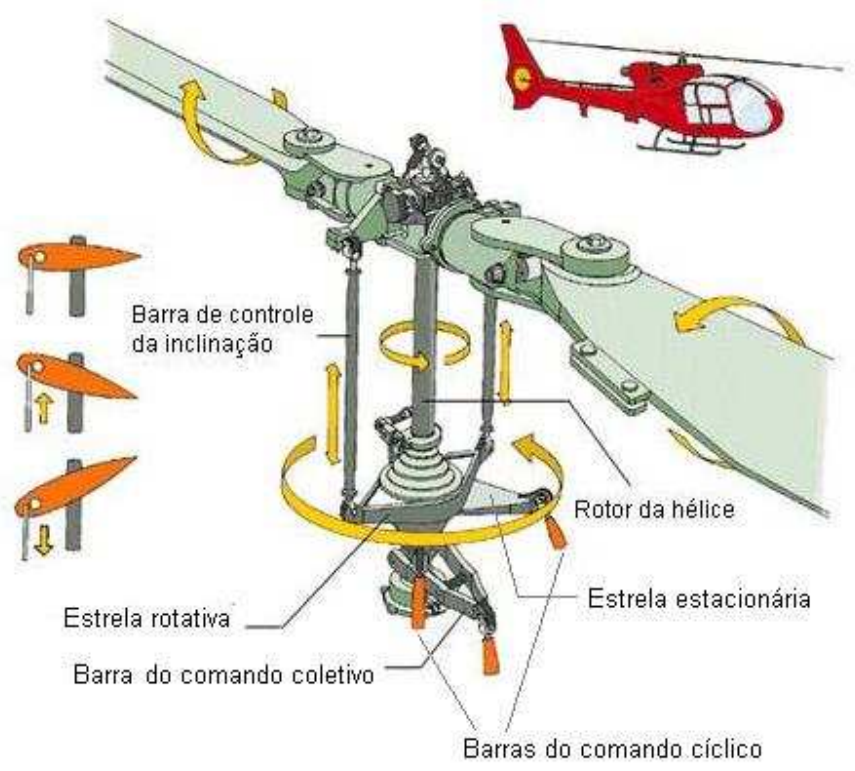


Figura 2.9 – Rotor principal.

2.3.4. Comando

O piloto do helicóptero tem, ao seu lado esquerdo, a chamada alavanca *de passo coletivo*. Entre suas pernas, fica o *manche cíclico* e, aos seus pés, os *pedais* (Figura 2.10).

Com a alavanca do coletivo, o piloto movimenta a barra de comando coletivo, que faz as estrelas subirem ou descerem, fazendo o passo das lâminas diminuir ou aumentar. Quando o passo das lâminas aumenta, o ângulo de incidência aumenta. Dessa forma, as

lâminas deslocam uma massa maior de ar e a força de sustentação aumenta. O inverso ocorre quando o ângulo de incidência diminui. A Figura 2.11 descreve o ângulo de incidência de uma asa. É importante ressaltar que os helicópteros têm dispositivos que mantém constante a velocidade de rotação do rotor. Quando o piloto movimenta o coletivo, a potência do motor se ajusta automaticamente de modo que a velocidade de rotação nunca muda. Dessa forma, a alavanca do coletivo faz o helicóptero subir ou descer (em vôo pairado) ou faz a velocidade de translação aumentar ou diminuir, em vôos de translação.

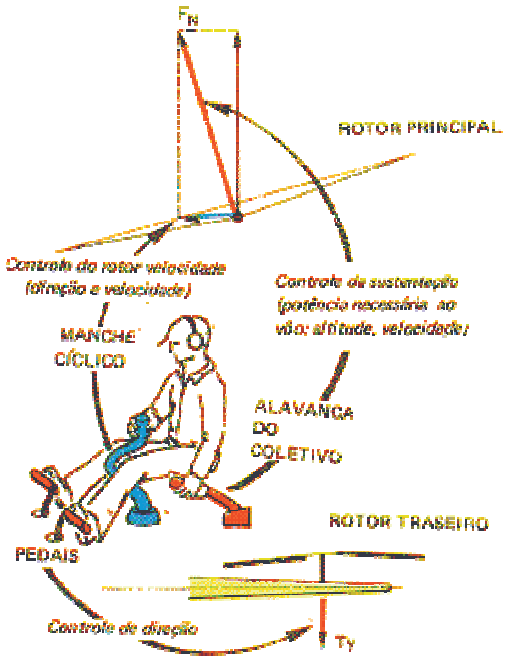


Figura 2.10 - Dispositivos de comando (Fonte: www.jhelicopteros.com.br).

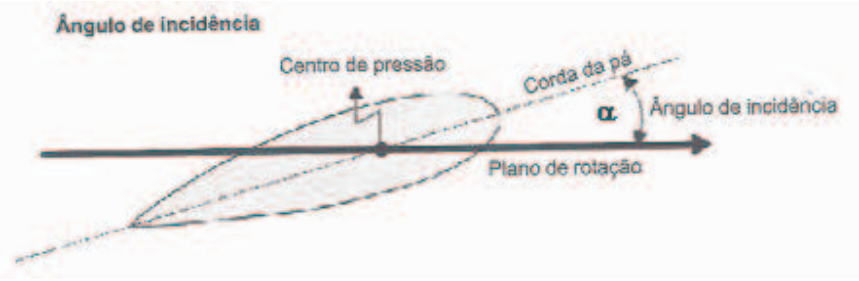


Figura 2.11 – Ângulo de incidência.

Com os pedais, o piloto controla o rotor de cauda, controlando assim o movimento de guinada. O rotor de cauda também tem velocidade constante e está ligado ao rotor principal por um sistema de árvore e engrenagens (o helicóptero tem apenas um único motor). As pás do rotor de cauda também têm passo variável e a estrutura é semelhante à do rotor principal.

Com o manche cíclico, o piloto controla as barras de comando cíclico, que fazem as estrelas inclinarem. Quando essas se inclinam, uma das lâminas aumenta o ângulo de incidência e a outra diminui. Dessa forma, a sustentação passa a ser desigual. Quando o rotor gira 180°, a posição das lâminas se inverte e a que tinha o ângulo de incidência maior passa a ser a que tem o ângulo de incidência menor e vice-versa. Assim, a sustentação continua desigual no mesmo sentido. O resultado disso é que surge um torque e o helicóptero se inclina. Quando o piloto leva o manche cíclico à frente o helicóptero se inclina para frente (arfagem), quando o piloto leva o manche para traz o helicóptero se inclina para traz (arfagem) e quando o piloto inclina o manche para a esquerda o para a direita, o helicóptero se inclina para a esquerda ou para a direita (rolagem). Assim, o piloto, além de poder manter a estabilidade do helicóptero, é capaz de fazê-lo se deslocar em qualquer direção num plano horizontal.

É importante esclarecer que o ângulo de incidência é diferente do *ângulo de ataque*. O ângulo de ataque se refere ao ângulo da pá com o vento aparente. Ou seja, com o fluxo da massa de ar que incide sobre a asa.

2.3.5. O efeito de precessão giroscópica

O efeito de precessão giroscópica é uma propriedade inerente de todo corpo em rotação, segundo a qual, se aplicarmos um torque para mudar o plano de rotação, os efeitos se fazem sentir como se esse momento tivesse girado 90 graus no sentido da rotação. Em outras palavras, para girar o plano de rotação de um giroscópio em determinado sentido, é necessário aplicar um torque noventa graus antes do ponto que esse torque seria aplicado se quiséssemos girar um disco parado.

Para entender melhor esse efeito, imaginemos um corpo preso por uma corda a um ponto fixo (Figura 2.12). Esse corpo é posto em movimento circular com raio *R* e velocidade tangencial *V*. Surge portanto um momento angular inicial *L*, perpendicular ao plano de rotação β. Em determinado momento, aplica-se a este corpo uma força F, para cima, por um período infinitesimal de tempo (*dt*). Esta força causa uma variação de velocidade *dV*. A velocidade final do corpo (V') será dada pela soma vetorial: *V'=V+dV*. Note que a nova velocidade tangencial é inclinada em relação ao plano de rotação inicial. O corpo modificará sua trajetória e teremos um novo plano de rotação (β'), que será inclinado em relação ao plano inicial. Teremos também um novo momento angular (*L'*) com direção diferente. Um observador que olha para o plano de rotação como se ele fosse um disco tem a impressão de que a força se manifestou 90º à frente, no sentido da rotação.

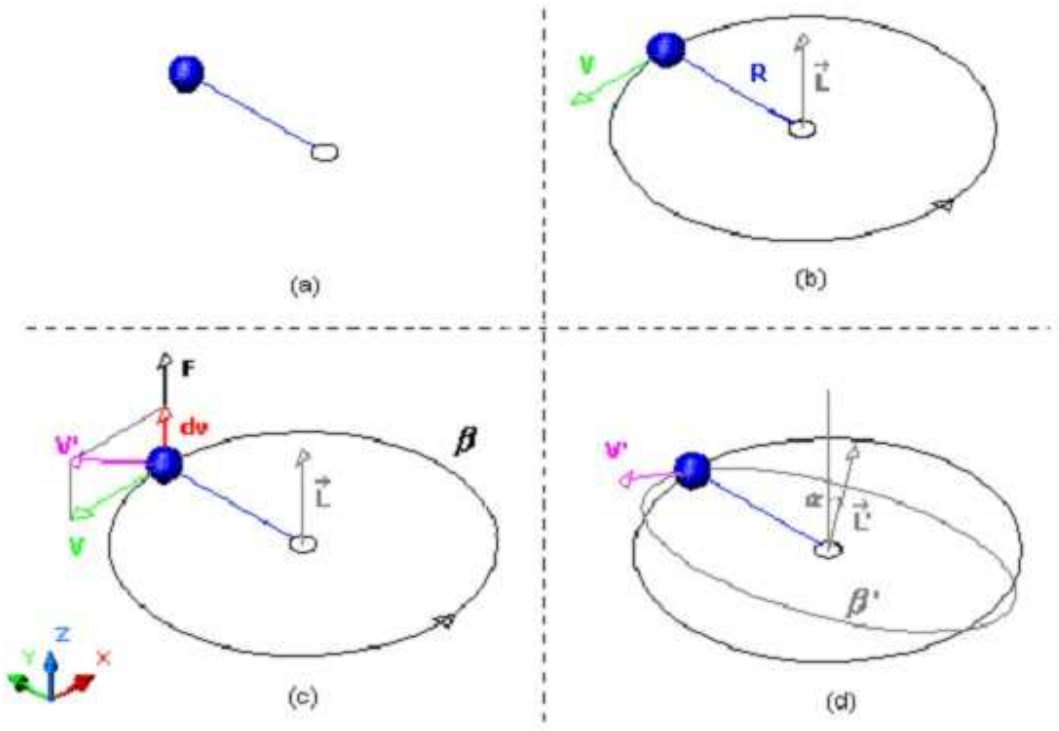


Figura 2.12 – O efeito de precessão giroscópica.

Essa inclinação pode ser calculada da seguinte forma: sabe-se que o torque (τ) causado pela força *F* e é dado pelo produto vetorial de *F* por *R*:

$$\tau = F \times R \tag{2.1}$$

Sabe-se ainda que o momento angular inicial L é dado por:

$$L = \omega . I \tag{2.2}$$

Em que

- ω Velocidade angular [rad/s]
- I Momento de inércia do corpo em rotação [kg.m^2]

A segunda lei de Newton na forma angular diz que:

$$\tau = \frac{dL}{dt} \tag{2.3}$$

Que pode ser reescrita na forma:

$$dL = \tau . dt \tag{2.4}$$

O momento angular final (L') é soma vetorial de L e dL e o ângulo de inclinação (α) será dado por:

$$\alpha = \tan^{-1}\left(\frac{dL}{L}\right) \tag{2.5}$$

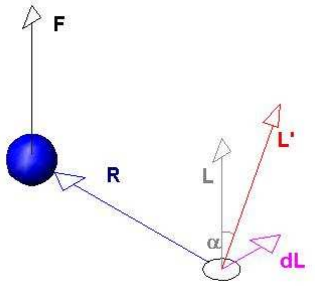


Figura 2.13 – Decomposição do momento angular *L'* em detalhe.

Substituindo as equações de (2.2) e (2.4) na equação (2.5) temos:

$$\alpha = \tan^{-1}\left(\frac{\tau.dt}{\omega.I}\right) \tag{2.6}$$

2.3.6. Auto-rotação

Pode-se pensar que quando o motor pára, o helicóptero simplesmente cai sem controle. Na verdade, o que ocorre não é bem isso. Existe um tipo de vôo que se chama vôo em auto-rotação.

Quando o motor pára ou quando o piloto desengata o rotor do sistema de tração, é possível manter o regime de rotação das pás e o helicóptero voa perfeitamente controlado até aterrissar. Para conseguir isso, o piloto ajusta o coletivo de modo a manter as rotações adequadas à medida que o helicóptero desce. O fluxo de ar que passa através do rotor durante a descida da máquina fornece a energia para manter as pás em rotação. O movimento das pás gera sustentação e permite manter a estabilidade. Dessa maneira, o helicóptero “plana”, da mesma forma que um avião. É nítido, entretanto, que um avião com asas fixas plana com muito mais eficiência. Contudo, o helicóptero tem a vantagem de não precisar de áreas grandes para aterrissar.

Para conseguir aterrissar em auto-rotação o piloto deve reduzir ao máximo a velocidade horizontal e a razão de descida imediatamente antes de tocar os solo. A primeira manobra é puxar o controle do cíclico para trás e modificar a atitude do rotor em relação ao vento aparente. Esta mudança de atitude aponta a força total do rotor para trás e diminui a velocidade horizontal (chama-se *flare*) (Figura 2.14). O pouso é extremamente parecido com o dos pássaros. Para aterrissar, os pássaros posicionam as asas na vertical de forma a conseguir um freio aerodinâmico e diminuem drasticamente a velocidade horizontal.

O pouso em auto-rotação é uma manobra de emergência muito arriscada, que exige muita perícia e experiência do piloto.

2.3.7. Piloto automático

Por serem aeronaves de difícil pilotagem, muitos helicópteros hoje em dia são equipados com pilotos automáticos. Em cada helicóptero, basicamente, existem dois modos de pilotos automáticos. Um deles atua somente na atitude do helicóptero, controlando arfagem, rolagem e guinada e mantendo a estabilidade, tanto em vôo pairado, quanto em vôo em translação. É o chamado controlador de atitude. O outro atua no controle a velocidade, a rota e a altitude. É o piloto automático de cruzeiro

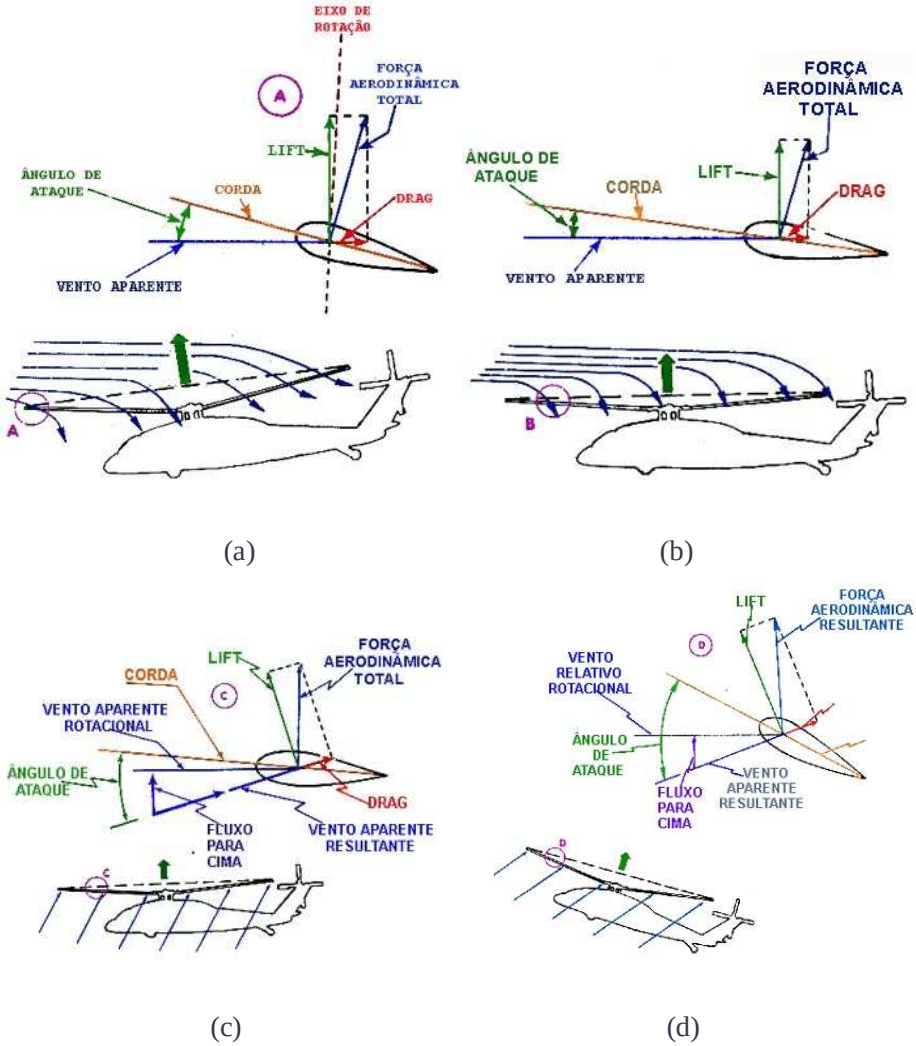


Figura 2.14 - (a) Vôo em translação com determinado ângulo de incidência. (b) Momento em que o motor perde potência. O piloto reduz o ângulo de incidência, diminuindo a sustentação. (c) Quando começa a descida, o fluxo de ar para cima mantém a rotação e gera sustentação. (d) Momentos antes te tocar o solo o piloto realiza uma manobra para reduzir a velocidade horizontal. (Fonte: www.jhelicopteros.com.br)

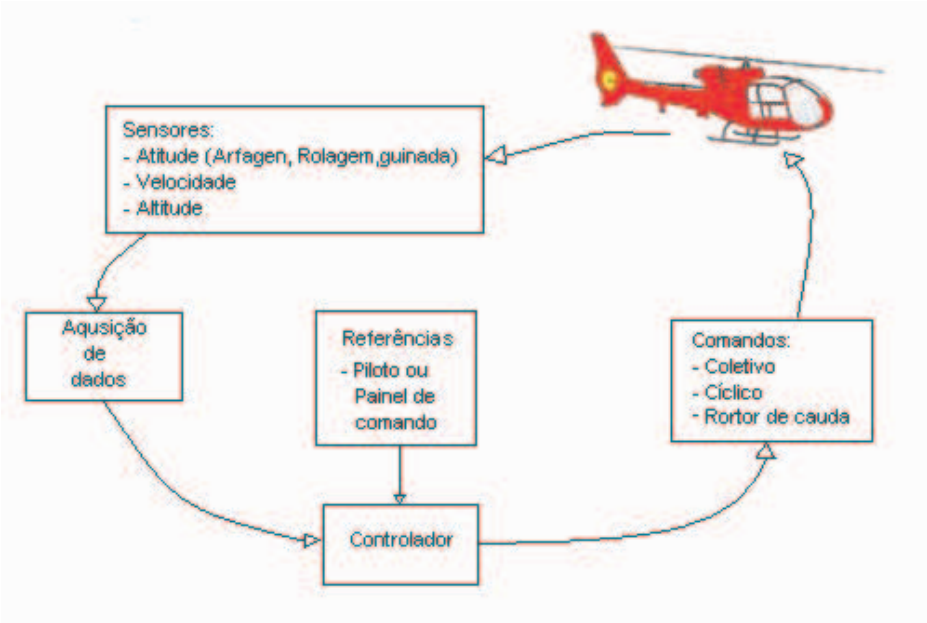


Figura 2.15 - Piloto automático de um helicóptero.

Ao utilizar somente o modo de controle de atitude, tudo o que o piloto precisa fazer é decidir para que lado quer ir e acionar o cíclico na direção desejada. Não é mais necessário compensar as oscilações ou turbulências. O controle da atitude utiliza sensores que medem os ângulos de arfagem, rolagem e guinada e compara com a referência introduzida pelo piloto pelo manche cíclico e os pedais.

O controlador de cruzeiro é utilizado em longas viagens, com rotas e altitudes pré-definidas por um plano de vôo. Ele funciona como os pilotos automáticos dos aviões comerciais - utiliza sensores de altitude, velocidade e direção e compara com a referencias introduzidas pelo piloto num painel de comando. O piloto automático de cruzeiro atua em cascata com o controlador de atitude, pois, para alterar a direção (virar à esquerda, por exemplo) é preciso modificar a atitude de maneira adequada até que a manobra se complete. O mesmo ocorre quando se quer mudar a velocidade ou a altitude.

2.4. Modelos automáticos em escala reduzida

Há uma bibliografia considerável que trata do modelamento e controle de modelos de helicóptero. Muitas outras tratam de sistemas com asas rotativas, como o quadrirotor e o chamado “helicóptero” com 2 GDL.

2.4.1. Helimodelos

São muitos os trabalhos efetuados sobre o modelamento e controle de modelos de helicóptero. Verifica-se um interesse grande por essa aeronave em instituições européias, japonesas e americanas. Não foi encontrada, entretanto, nenhuma linha de pesquisa permanente sobre o assunto no Brasil.

A maior parte dos pesquisadores dessa área inicialmente desenvolveu uma bancada de testes para o helimodelo e, somente após o alcance de um nível adequado de conhecimento sobre ele, partiu para experimentos de vôo em campo.

Em [1] e [2], Ávila-Vilchis apresenta o modelamento e controle não-lineares de um helimodelo movido a gasolina. Foi projetada uma plataforma semelhante a nossa, mas que permite a translação no eixo da atitude. Os resultados obtidos foram aceitáveis com um controlador linearizado.

Sastry e os demais membros do *Robotics and Intelligent Machines Laboratory* da *University of California* em Berkeley, desenvolveram, a partir do helicóptero RX-50, da Yamaha, um sistema autônomo capaz de navegar, pousar e seguir objetos no solo [19]. Em [20], é proposto um amplo procedimento de identificação do sistema no domínio da freqüência. Em [18], descreve-se uma análise de desempenho de diversas metodologias de controle: controle robusto, controle *fuzzy* e controle linearizado. Foi verificado que, para diferentes condições de vôo, diferentes controladores são mais adequados.

Em [21], Sugeno, um dos principais pesquisadores de lógica fuzzy, descreve um sistema de controle *fuzzy* completo (controle de atitude, velocidade de translação e navegação) para o mesmo helicóptero da Yamaha. No entanto, não é feita referência à etapa de modelamento, apenas ao controle.

É notável também o trabalho de outro grupo americano, porém do *California Institue of Technology* (*Caltech*). Em [14], é descrito o controle de velocidade do rotor do helicóptero, que é o a malha de controle fundamental de todo helicóptero real. Em [26] e [13], são descritos experimentos de identificação do sistema e controle utilizando LQR e LQG, respectivamente.

Um outro trabalho que trata do controle e modelamento de helimodelos [11], em que é proposto um modelo físico para o helicóptero em escala reduzida e conclui-se da importância da simulação em todas etapas de projeto de um sistema tão complexo como esse.

Foram verificados também grupos de pesquisa em outras universidades, como *Cargenie Mellon University*, *Stanford University* e a *Technischen Universität Berlin*.

Há, ainda, algumas empresas que comercializam helimodelos autônomos. Entre elas pode-se citar a Scandicraft®, empresa sueca que fabrica e comercializa modelos para inspeção, vigilância e etc; a Rotormotion®, empresa americana fundada pelos idealizadores do projeto *autopilot.sourceforge.net*; e a Yamaha®, cujos helimodelos, produzidos inicialmente para o combate às pragas da agricultura, são a plataforma padrão para desenvolvimento de pesquisas sobre o tema atualmente.

2.4.2. Modelos simplificados

Muitas universidades possuem plataformas experimentais didáticas comercializadas por duas empresas: a Quanser® e a Feedback® (Figura 2.16). Esses sistemas podem ser descritos como parentes do helicóptero real, pois só possuem 2 GDL: a arfagem e a guinada. Por esse motivo, não há comando cíclico nos rotores e o aumento de sustentação é definido somente pelo aumento da velocidade de rotação. É um sistema multivariável, como o helicóptero, e também fortemente acoplado, porém possui valor exclusivamente didático.

Um dos trabalhos desenvolvidos nessa plataforma foi realizado por Balderud, que, em [3] e [4], investigou várias estratégias de controle aplicadas ao processo. Em [9], foram

alplicadas técnicas de algoritmos genéticos para implementar um controle preditivo do sistema.

(a) (b)

Figura 2.16 – “Helicóptero” com 2 GDL produzido pela (a) Quanser® e pela (b) Feedback®.

A Universidade de Siena, da Itália, por meio do *Automatic Controle Telelab*, possibilita a realização deste experimento on-line, no site http://www.dii.unisi.it/~control/act/home.php.

2.4.3. Quadrirotores

Os quadrirotores são aeronaves que possuem quatro rotores independentes atuando no mesmo plano longitudinal do veículo. Apesar da dinâmica que rege o comportamento desse sistema ser consideravelmente diferente do nosso projeto, foram pesquisados alguns trabalhos sobre eles porque supunha-se que poderíamos desenvolver um sistema semelhante em nosso projeto.

O quadrirotor também possui os 6 GDL apresentados pelo helicóptero, porém a viabilidade técnica e comercial desse tipo de sistema é ainda bem menor, até mesmo porque as velocidades alcançadas são consideravelmente menores. Entretanto, existem algumas aplicações interessantes, como resgate de pessoas em prédios em chamas.

Em [5], está descrito o controle e modelamento de um quadrirotor montado sobre uma plataforma experimental.

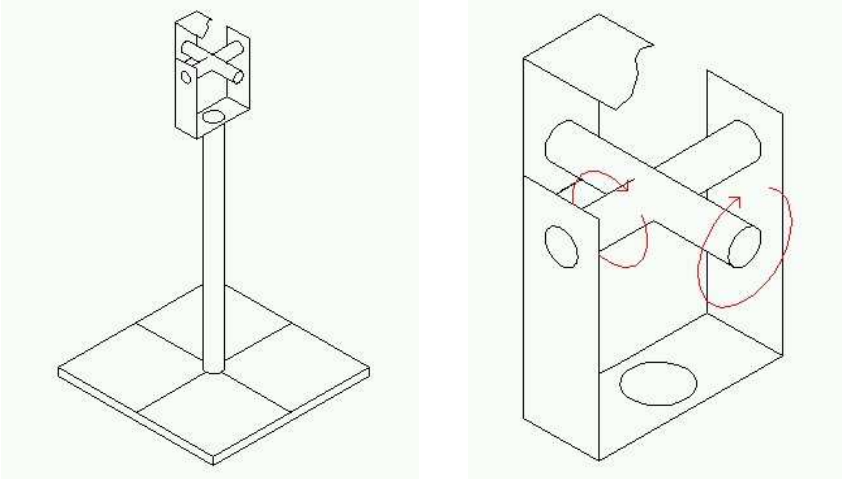
3. Projeto da plataforma experimental

Esse capítulo descreve todas etapas envolvidas no projeto da plataforma experimental. Não são descritos em detalhes todas as decisões técnicas que foram tomadas para incrementar a qualidade do experimento, pois considera-se que isso acarretaria num volume desnecessário de informações. Pretende-se, entretanto, deixar claro o conjunto dos principais objetivos almejados e os principais problemas encontrados na sua busca.

O capítulo está dividido em três seções principais: projeto mecânico, projeto eletrônico e projeto computacional. Elas descrevem as principais frentes de projeto da plataforma experimental. Além disso, descrevem cronologicamente as ações adotadas para se alcançar o desempenho esperado.

3.1. Projeto mecânico

Um dos requisitos do projeto era conceber um suporte que permitisse ao helimodelo realizar os movimentos de arfagem, rolagem e guinada. Em outras palavras, era necessária a construção de uma haste de apoio com três graus de liberdade. Assim, foi idealizado um suporte que consistiria em um eixo perpendicular ao solo, que poderia girar livremente em torno de si mesmo. À extremidade desse eixo seria fixada uma junta universal, que permitisse o movimento com dois eixos de rotação. A essa junta, seria, então, fixado o helimodelo. Um esboço do suporte está descrito na Figura 3.1.



(a) (b)

Figuras 3.1– (a) Esboço do suporte e da (b) junta universal.

O projeto mecânico apresentou certos desafios. Uma das maiores dificuldades foi fabricar uma junta que permitisse fixar de maneira adequada os potenciômetros que medem os ângulos de arfagem e rolagem. Outro problema era a grande quantidade de cabos necessários para transmitir os sinais entre o modelo e a base. Esses cabos poderiam enrolar na haste e atrapalhar o movimento do helimodelo.

3.1.1. O helimodelo

Figura 3.2 – Foto do Fun Piccolo.

O modelo utilizado é o Fun-Piccolo, fabricado pela IKARUS, da Alemanha (Figura 3.2). É um modelo leve, pequeno, de fácil montagem, que utiliza 2 motores DC - um para o

acionamento do rotor principal e um para o rotor de cauda - e dois servo-motores, que atuam na mudança da inclinação das pás do rotor principal. O modelo foi adquirido com recursos da Finatec após aprovação do projeto pelo edital 2003.2 de Financiamento à Pesquisa.

Assim, foi adquirido um kit que contém: estrutura mecânica (incluindo rotor principal, rotor de cauda, estrela estacionária, estrela rotativa, carenagem); 2 motores DC e 2 servos. As especificações do kit são:

- Comprimento: 50cm;
- Diâmetro do rotor principal: 50cm;
- Peso completo: aproximadamente 250 g;
- Servos: Modelo Micro 201, Fabricante LEXORS, Torque 0,84 kg/cm, Velocidade 0,13 rad/s (Figura 3.3).

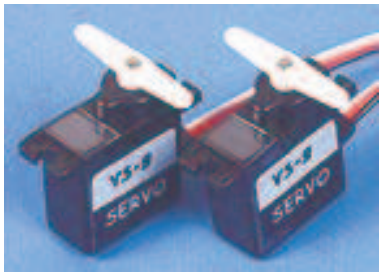


Figura 3.3 – Servos Micro 201.

3.1.2. O suporte

O projeto do suporte priorizou a robustez e a precisão. O eixo do suporte foi fixado com dois rolamentos para reduzir atrito. A base foi construída com chapas de aço e o eixo foi retirado de uma impressora em sucata. As Figuras 3.4 mostram o esquema de montagem do suporte. A Figura 3.5 mostra o suporte depois de pronto.

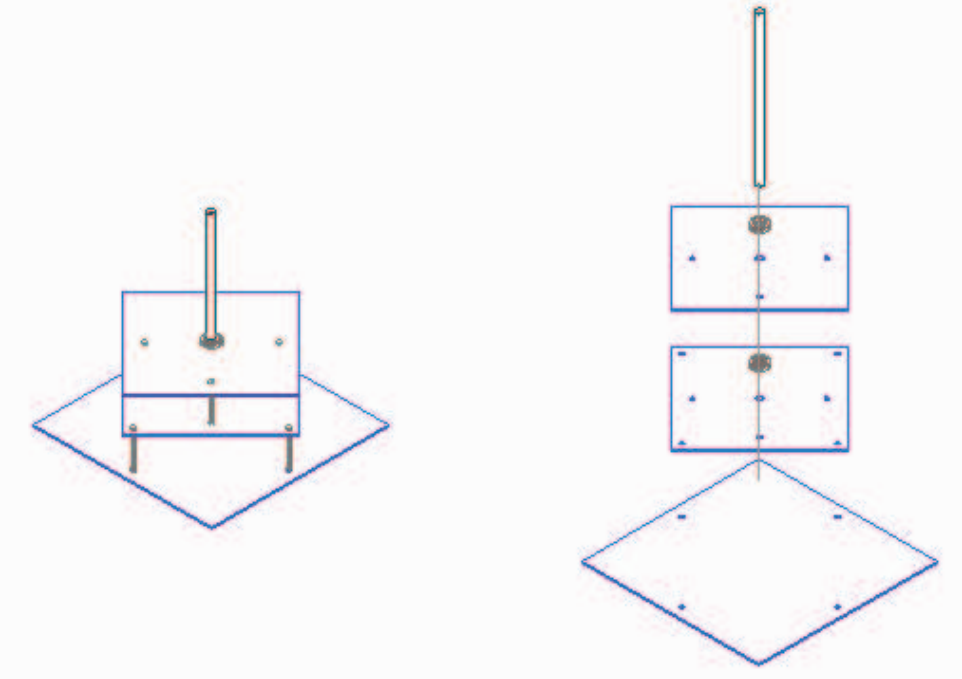


Figura 3.4 – Modelo CAD do suporte.

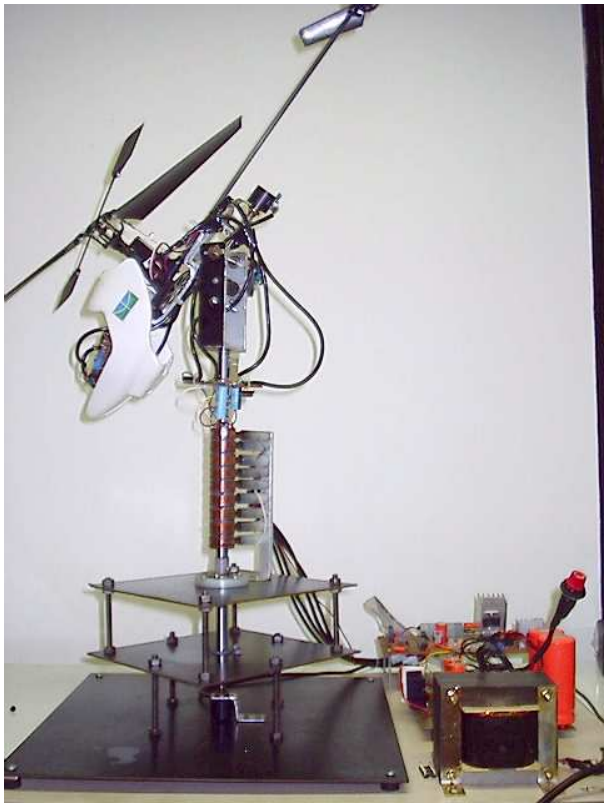


Figura 3.5 – Visão geral de todo conjunto (suporte, sistema de escovas, junta e helimodelo) depois de montado.

3.1.3. A junta universal

Para permitir a movimentação do modelo em dois dos três graus de liberdade propostos, foi necessária a construção de uma junta universal, com dois eixos de rotação. A junta precisava ser resistente o suficiente para suportar o modelo e devia mover-se livremente, com o mínimo de atrito possível. Além disso, a junta deveria ser construída de maneira que fosse possível acoplar a seus eixos dois potenciômetros para a medição dos ângulos de giro.

Os modelos de helicópteros que haviam sido estudados até aqui, como o da Quanser®, se moviam apenas com dois graus de liberdade e não utilizavam nem um tipo de junta com características semelhantes. Depois de consultados alguns catálogos de juntas e estudadas as possibilidades, chegou-se à conclusão que a solução seria uma junta formada por um eixo central em forma de cruz ao qual se conectariam duas peças em forma de U (Figura 3.6). Os potenciômetros poderiam ser facilmente fixados às peças em U e seus cursores girariam presos ao eixo. Eles são usados como sensores de posição angular.

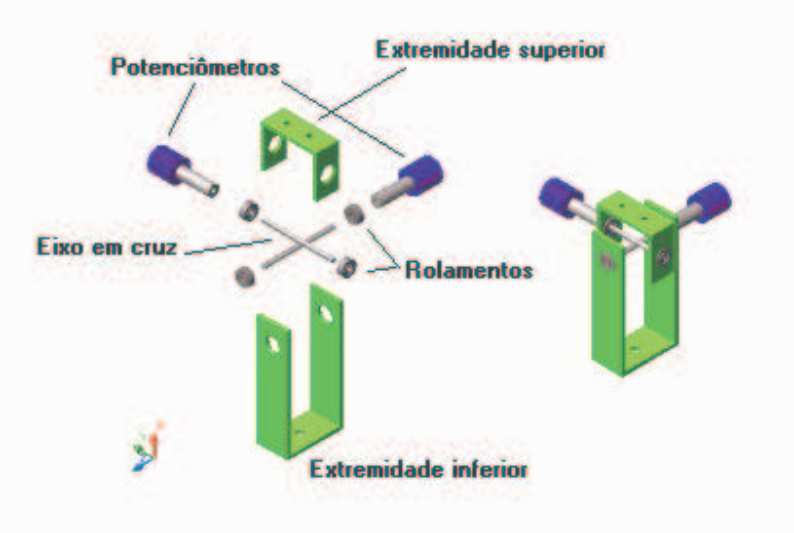
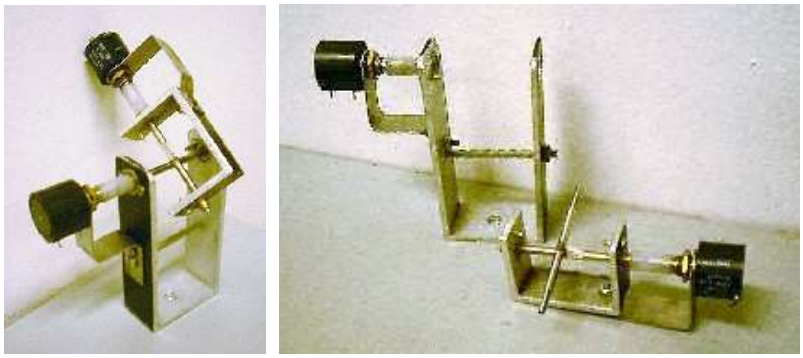


Figura 3.6 – Modelo CAD da junta universal.

Para o bom funcionamento da junta, fazia-se necessário que o eixo em forma de cruz tivesse suas hastes perfeitamente perpendiculares entre si. Foi usada soldagem a ponto por resistência elétrica para unir dois pequenos eixos de aço de maneira precisa. Para

reduzir atrito de giro, foram usados rolamentos para conectar os eixos às extremidades em U da junta.



(a)

(b)

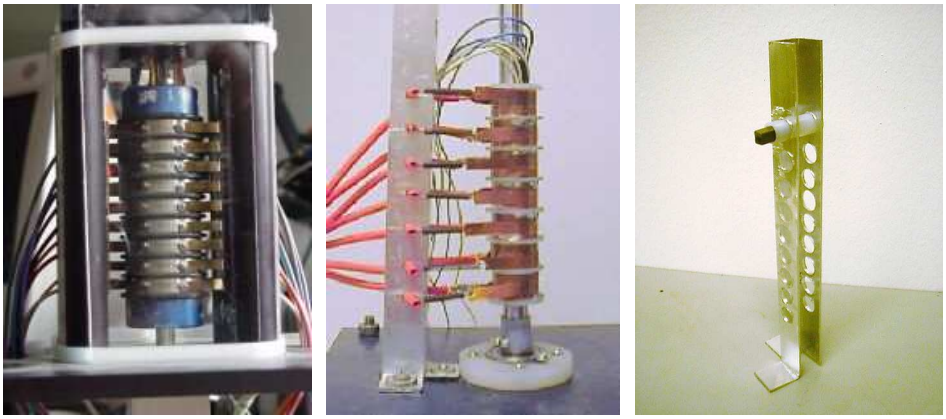
Figura 3.7 – A junta universal, com os potenciômetros conectados.

3.1.4. O sistema de escovas

Um grande problema encontrado no projeto era a quantidade excessiva de fios necessários para transmitir, tanto os sinais de potência e de controle da base ao helimodelo, quanto os sinais analógicos de dados do helimodelo à base. Os fios impunham limites à movimentação do helimodelo. Seja ao se enrolar na haste, seja ao forçar o helimodelo para um certo lado pelo peso ou elasticidade.

Para solucionar o problema dos fios e permitir ao helimodelo movimentar-se mais livremente, foi proposta a implementação de um sistema de contatos elétricos rotativos por escovas (Figura 3.9). Algumas configurações foram tentadas inicialmente, com o uso de fios de cobre e lâminas de cobre enroladas e soldadas a um tubo central de Náilon. Entretanto, todas elas apresentavam sempre os mesmos problemas: elevado atrito mecânico, elevada resistividade no contato e surgimento de ruído na transmissão dos sinais. De todos esses, o problema mais sério seria a variação de resistência de contato, que poderia ocasionar perturbação aleatória no sistema elétrico de medição do helimodelo. Para o acionamento, com correntes elevadas, as perdas podem ser importantes.

A figura 3.8 relaciona a solução para esse problema encontrada pela Quanser®, uma das propostas frustradas apresentadas, bem como a solução final.



(a) (b) (c)

Figura 3.8 – (a) Sistema de escovas produzido pela Quanser®, (b) tentativa frustrada de implementação e (c) detalhe de um suporte para os contatos de grafite.

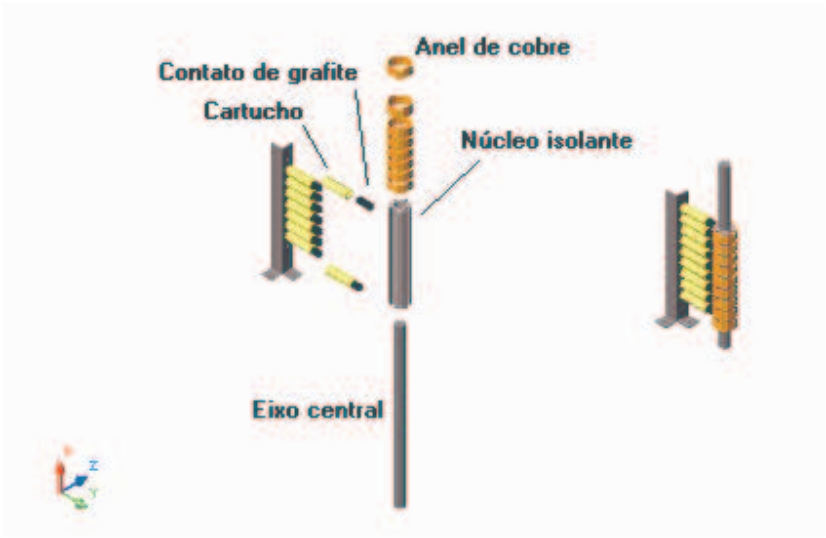


Figura 3.9 – Modelo CAD do sistema de escovas.

Para solucionar os problemas, foram empregados contatos de grafite de escovas de motores DC e foram usinados tubos de cobre para a confecção dos anéis rotativos. A nova configuração reduziu muito o atrito e as perturbações elétricas. O atrito, agora pequeno, deixou de ser um problema. Porém, as perturbações ainda impediam o funcionamento correto dos servos. Para minimizar os efeitos desses ruídos foram adotados os procedimentos descritos no item 3.2.2. do projeto eletrônico.

Na figura 3.10, pode-se ver o modelo CAD completo do conjunto composto pelo suporte, pelo sistema de escovas e pela junta universal.

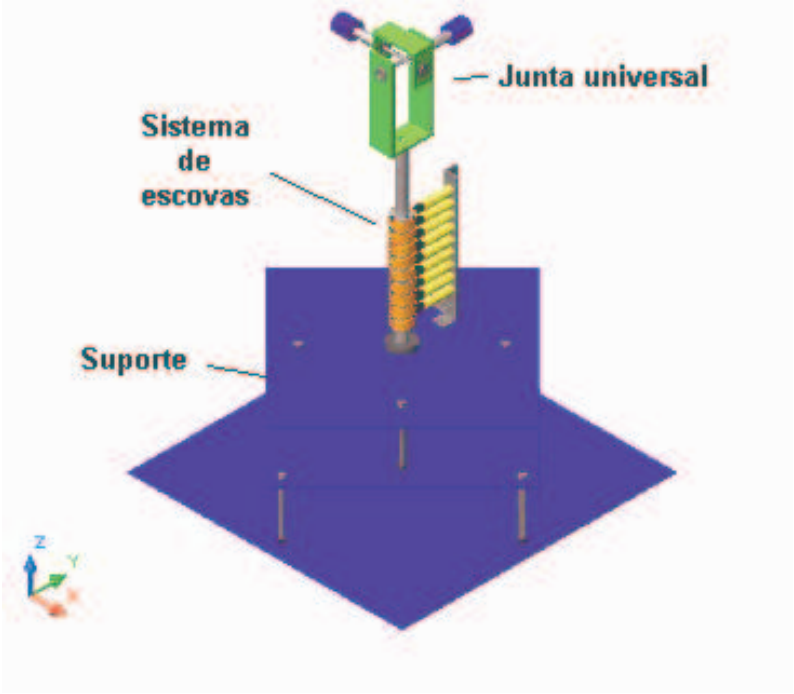


Figura 3.10 – Modelo CAD do conjunto suporte-junta-escovas.

3.2. Projeto eletrônico

O objetivo do projeto eletrônico era a confecção dos circuitos que possibilitariam tanto a aquisição dos dados de posição do modelo (ângulos de arfagem, guinada e rolagem), quanto à atuação correta nos motores e nos servos, além dos circuitos necessários para o correto funcionamento do microcontrolador Atmel AVR ATmega 8 e da interface microcontrolador/PC (item 3.2.3.3.).

O diagrama de blocos do sistema está representado na Figura 3.10.

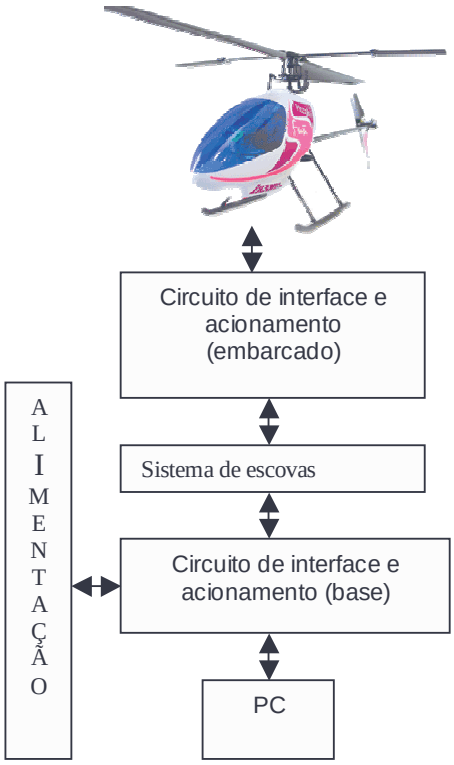
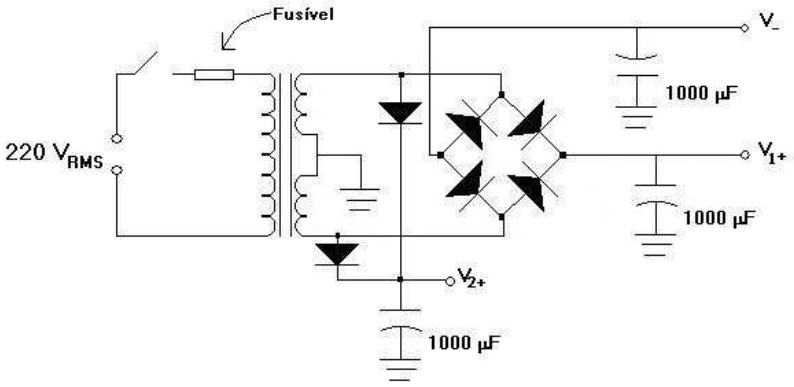


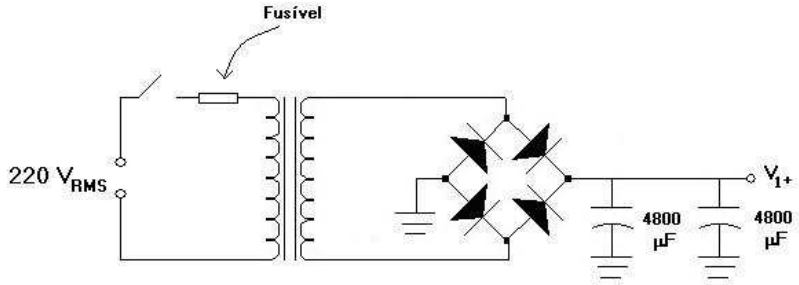
Figura 3.11 – Diagrama de blocos do circuito embarcado.

3.2.1. Fonte de alimentação

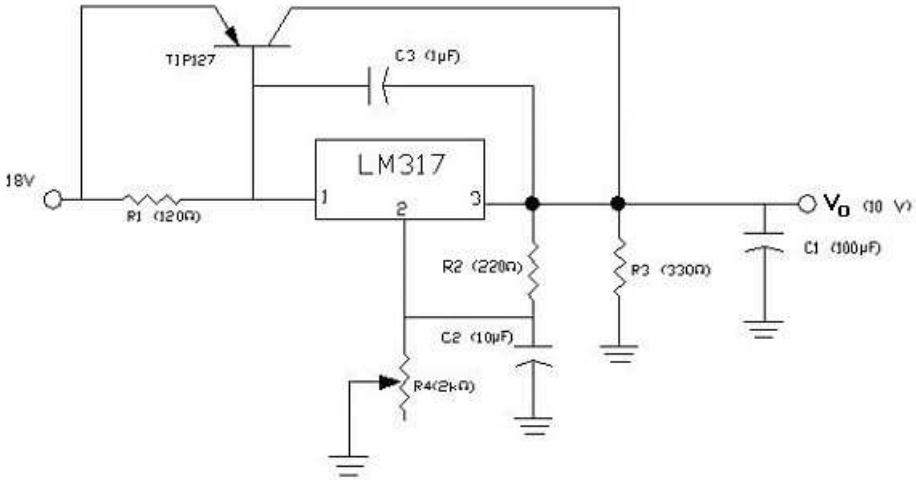
Para o funcionamento correto do sistema foi preciso confeccionar uma fonte para fornecer as seguintes tensões: 12 V e –12 V DC, para alimentar CIs e amplificadores operacionais; 5 V DC para alimentar o microcontrolador, os servos e a parte lógica da ponte H; e cerca de 10 V DC para alimentar os motores DC, que juntos demandam corrente superiores a 3 A. A Figura 3.12 mostra o projeto da fonte de tensão.



(a)



(b)



(c)

Figura 3.12 – (a) Circuito retificador em ponte de diodos de baixa potência. Uma das tensões positivas é utilizada para o circuito da base e a outra para o circuito embarcado. (b) Circuito retificador em ponte de diodos de alta potência (b). Circuito regulador de tensão para os motores DC. (c) O circuito é capaz de fornecer altas correntes.

Com relação ao circuito regulador de tensão para os motores apresentado na Figura 3.12(c), foi utilizada como base a nota de aplicação descrita em [22]. O principal objetivo do circuito é fornecer uma tensão de saída (V_O) constante, com correntes que variam de zero a cerca de 3,5 A. Assim, a tensão de saída é ajustada por meio de R_4.

A partir dos circuitos mostrados na Figura 3.12(a) e (b), foram utilizados reguladores de tensão comuns, como o 7805, 7812 e 7912, para regular a tensão no valor desejado.

3.2.2. Circuitos de interface e acionamento (embarcado)

Com o intuito de reduzir o número de escovas e também para solucionar alguns problemas que advêm da utilização dessas, foi decidido que alguns circuitos do projeto eletrônico deveriam ser embarcados no modelo e no próprio eixo que suporta o helicóptero. Fundamentalmente, esses circuitos são os condicionadores de sinais dos potenciômetros, as fontes de corrente (cuja função será explicada mais à frente), os filtros dos sinais de controle dos servos e reguladores de tensão de 12V e 5V, ilustrados na Figura 3.14. A Figura 3.13 mostra o diagrama de blocos referente aos circuitos embarcados.

As variáveis descritas no diagrama da Figura 3.12 são tensões e correntes que passam pelo sistema de escovas. Elas são:

I_a	Corrente de transmissão da medição de arfagem [mA]
I_r	Corrente de transmissão da medição de rolagem [mA]
V_a	Tensão de controle do servo de arfagem [V]
V_r	Tensão de controle do servo de rolagem [V]
V_+	Tensão de alimentação do circuito embarcado [V]
GND_1	Terra para os servos e potenciômetros
GND_2	Terra para os motores
V_P	Tensão de controle do motor principal [V]
V_C	Tensão de controle do motor de cauda [V]

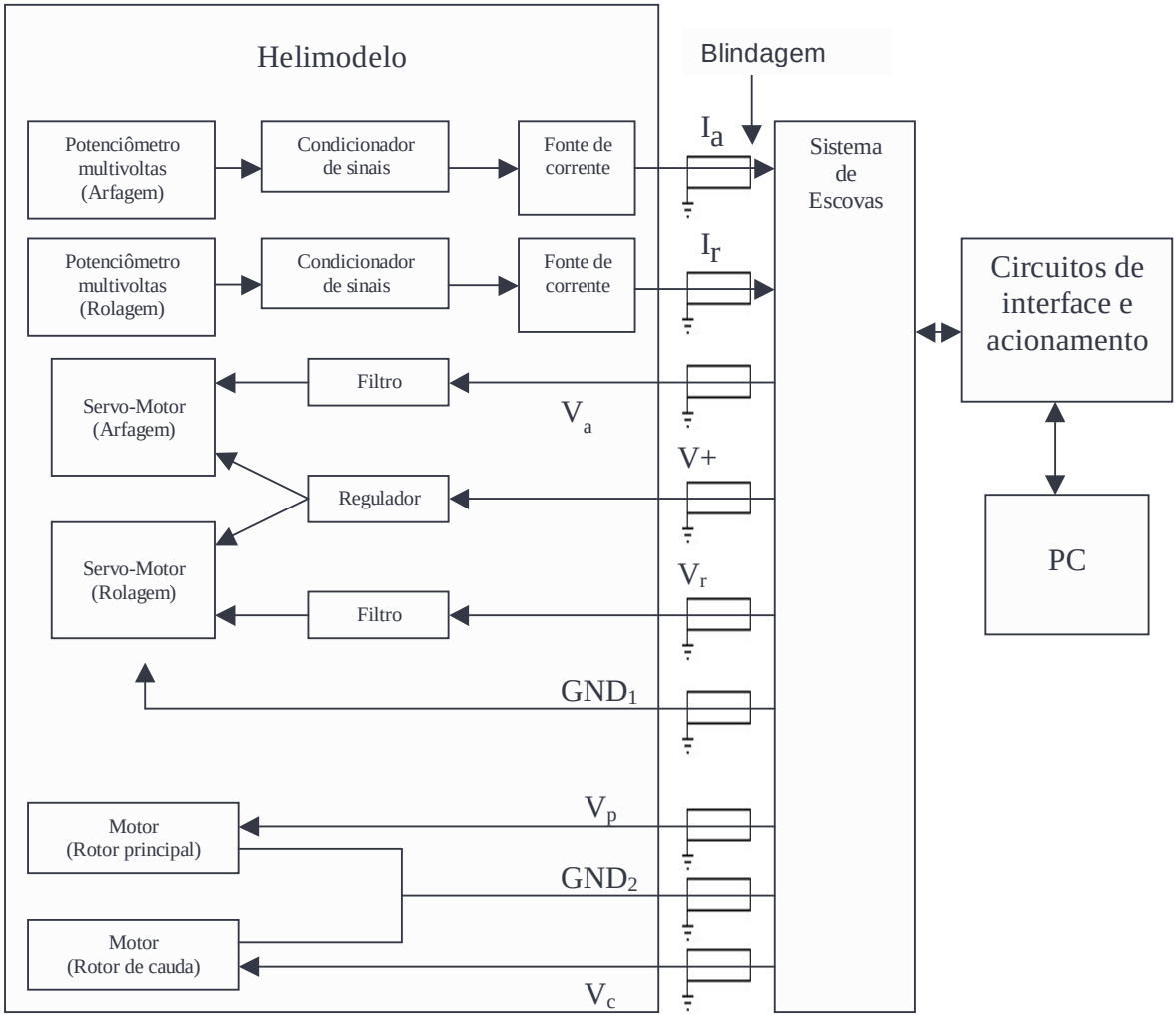


Figura 3.13 – Diagrama de blocos dos circuitos embarcados.

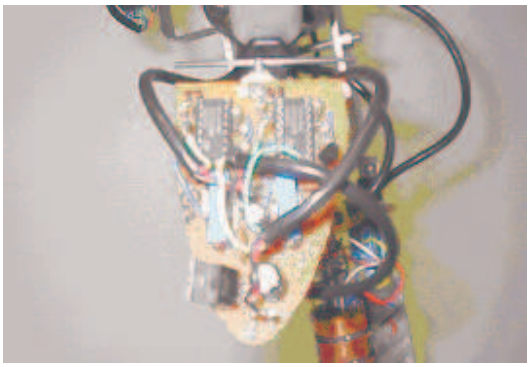


Figura 3.14 – Foto do circuito embarcado no helimodelo.

3.2.2.1. Condicionador de sinais dos potenciômetros de arfagem e rolagem

Os ângulos de arfagem, rolagem e guinada são medidos com o auxílio de potenciômetros (Figura 3.7). São potenciômetros de posição angular, que permitem 10 voltas, que é a limitação do movimento de guinada. No que diz respeito à faixa de operação, foi verificada uma linearidade razoável durante todo *span* do sensor.

Os sinais de tensão que vêm desses potenciômetros devem estar na faixa de 0 a 5 volts para poderem ser convertidos corretamente em sinais digitais no conversor A/D do microcontrolador. Entretanto, por serem multivoltas, os sinais do potenciômetro variavam em torno de 6,0 e 6,3 volts. Para adequar o sinal dos potenciômetros ao conversor A/D foi usado o circuito amplificador de diferenças ilustrado na figura 3.15 para condicionar os sinais.

O circuito amplificador de diferenças funciona, então, com o auxílio de uma tensão de offset obtida por meio de um *trimpot*. Essa tensão é próxima à tensão média obtida no potenciômetro. Ela foi ajustada pelo *trimpot* de forma que a tensão de saída do condicionador varia entre 1,5 e 4,5 V. Essa é a tensão de entrada para o circuito da fonte de corrente.

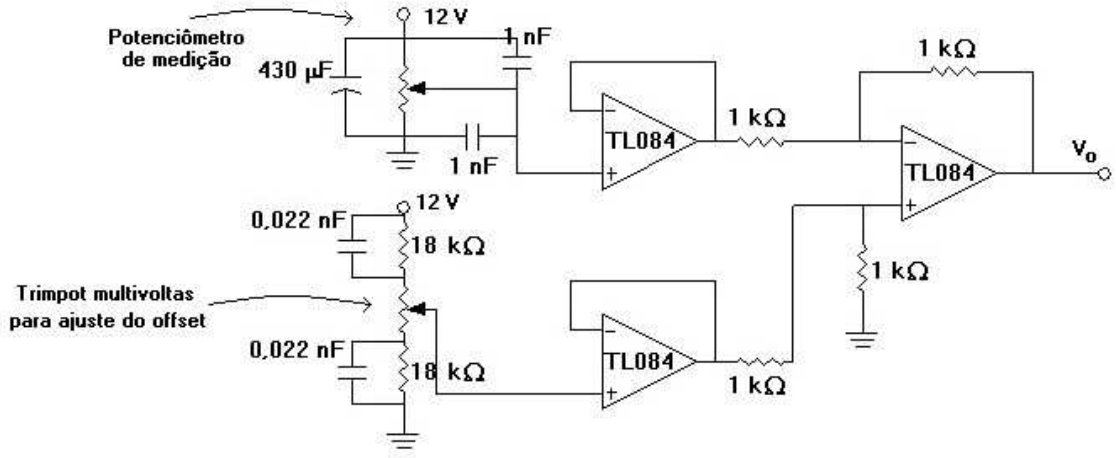


Figura 3.15 – Projeto do condicionador de sinais.

3.2.2.2. Fonte de corrente

Como foi exposto no item 3.1.4 (o sistema de escovas) os contato das escovas apresentam resistividades que variam à medida que o eixo gira. Essas resistividades têm valores de algumas unidades de Ωs quando o sistema está parado, mas que alcançam dezenas de Ωs quando a rotação do eixo de guinada é acentuada. Foi realizado um experimento para quantificar esse efeito ainda na fase inicial de projeto. O experimento consistiu na montagem do circuito da Figura 3.17 e na medição da tensão V_s. A tensão medida está representada na Figura 3.16.

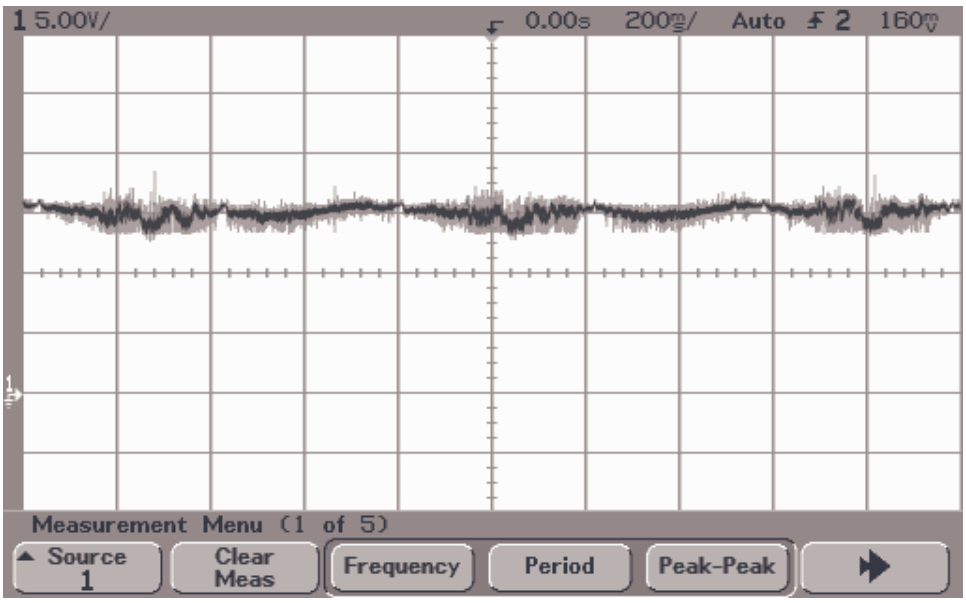


Figura 3.16 – Flutuação de tensão recebida no helimodelo ao se girar o eixo de guinada.

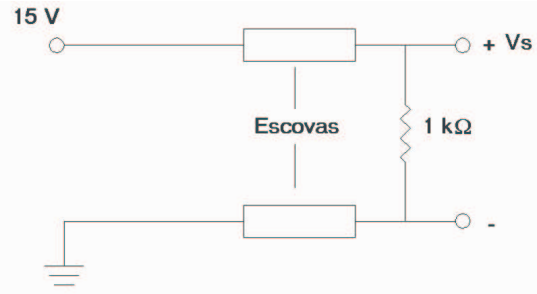


Figura 3.17 – Configuração usada no experimento.

Para resolver esse problema, foi proposta a utilização de uma fonte de corrente (Figura 3.18) para transmitir o sinal dos potenciômetros à base. Dessa forma o sinal de tensão seria convertido em sinal de corrente e o efeito da resistividade variável seria anulado.

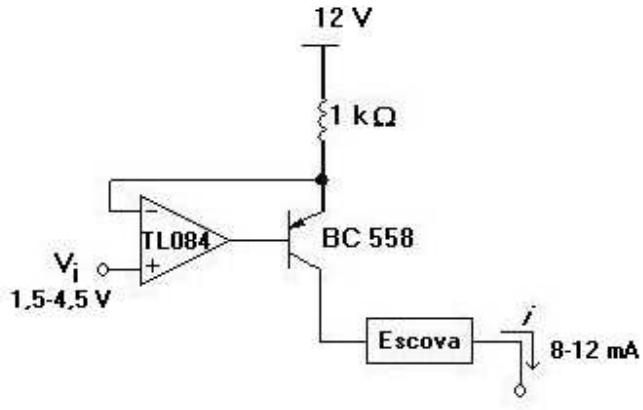


Figura 3.18 – Projeto da fonte de corrente.

A tensão de entrada desse circuito (Vi) é a tensão de saída do circuito amplificador de diferenças (V_O na Figura 3.15). A fonte de corrente é suficientemente robusta de forma a não sofrer influências da variação da tensão de coletor devido às flutuações da resistência da escova, desde que esta não alcance valores muito elevados. De fato, caso essa resistência se eleve a valores muito altos, o circuito da fonte pode saturar.

3.2.2.3. Filtro e separação das referências de zero (terras)

Para o controle dos servos, é necessário um sinal de PWM com características bastante restritas. Notadamente, uma freqüência muito baixa há de ser empregada (50 Hz), bem como variações precisas de largura de pulso entre 5 e 10% (os extremos do intervalo representam os limites de movimento dos servos). Dessa forma, a faixa de operação é bem reduzida. Além disso, o sinal não pode conter ruídos consideráveis, nem atenuações, pois isso também pode comprometer o funcionamento dos servos.

O nosso circuito, porém, devido ao grande número de componentes conectados, bem como à presença das escovas, apresentou sérios problemas para minimizar a incidência de ruídos e atenuações no sinal de controle dos servos.

O primeiro problema detectado foi uma interferência dos motores no funcionamento do servo. Diagnosticou-se que muitos ruídos surgiam no sinal dos servos, pois ambos compartilhavam o mesmo terra. Para solucionar essa questão, foi adicionada mais uma escova ao projeto inicial, de forma a separar a referência do motor da referência dos servos e dos potenciômetros.

Porém, verificou-se que o microcontrolador não era capaz de fornecer a potência suficiente para controlar os servos quando a resistência do sistema de escovas tornava-se considerável. Muitas soluções eram cabíveis para esse problema, e optou-se pela utilização de um CI inversor com Schmitt-Trigger (7414). A histerese do Schmitt-Trigger também permite eliminar parte do ruído inerente nos sinais PWM dos servos.

Além disso, utilizamos capacitores de alta capacitância conectados ao regulador 7805 (Figura 3.19), de forma a garantir que a tensão de alimentação dos servos sofresse o mínimo de flutuações possível.

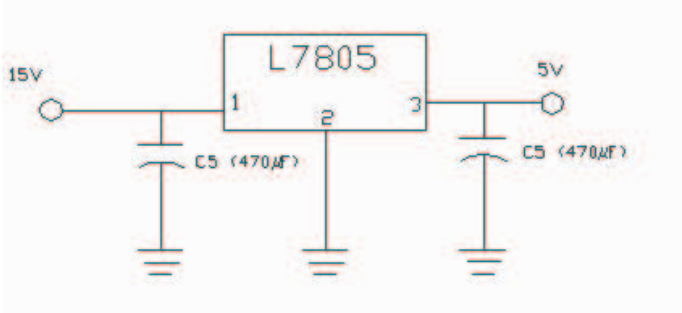


Figura 3.19 – Circuito regulador de tensão para os servos.

Mais tarde, foi também diagnosticado uma interferência do servo com a leitura dos potenciômetros. Isso se deu porque o controlador do servo faz com que ele consuma muita corrente nos seus limites de operação. Isso provocou um ruído no sinal do potenciômetro que era amplificado pelos circuitos condicionadores (itens 3.2.2.1. e 3.2.3.1.). Para contornar esse problema, foi implementado um filtro digital no software do PC (item 3.3.2.).

3.2.3. Circuitos de interface e acionamento (base)

Na base do suporte (Figura 3.20) foram implementados todos aqueles circuitos para os quais não houve necessidade de embarcar no helimodelo. Notadamente, têm-se os circuitos conversores de corrente-tensão, para condicionar os sinais dos potenciômetros; o circuito da ponte H, para gerar a potência e o controle necessários ao acionamento dos motores; e os circuito de interface microcontrolador/PC. A Figura 3.21 mostra o diagrama de blocos que representa os circuitos de interface e acionamento da base.

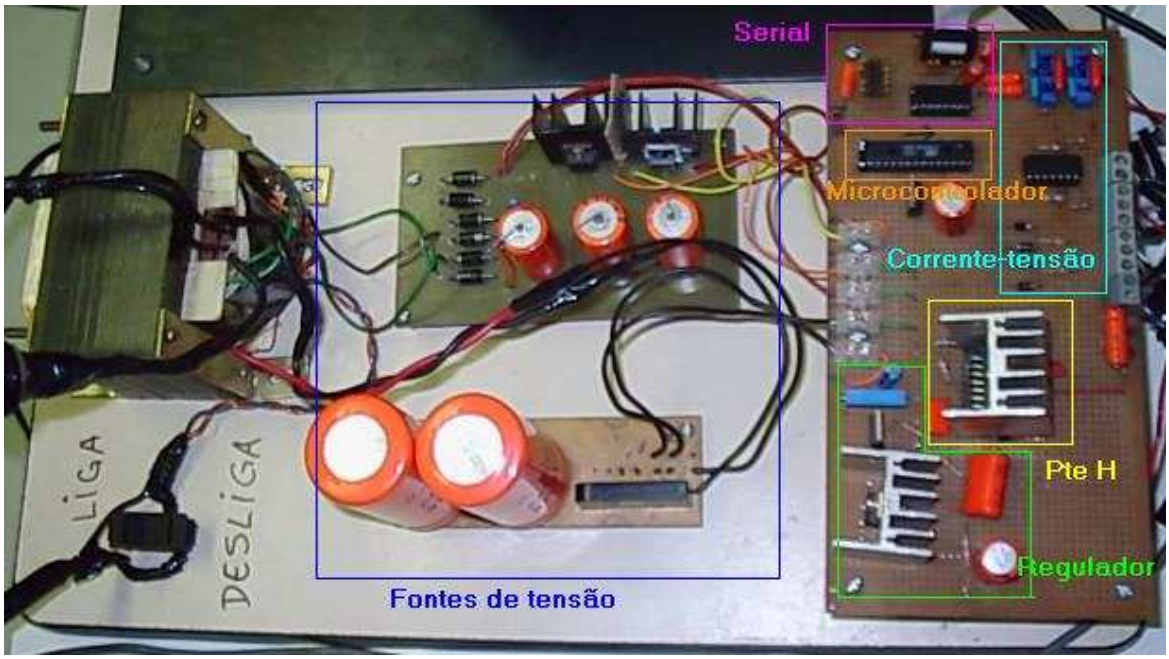


Figura 3.20 – Foto dos circuitos da base.

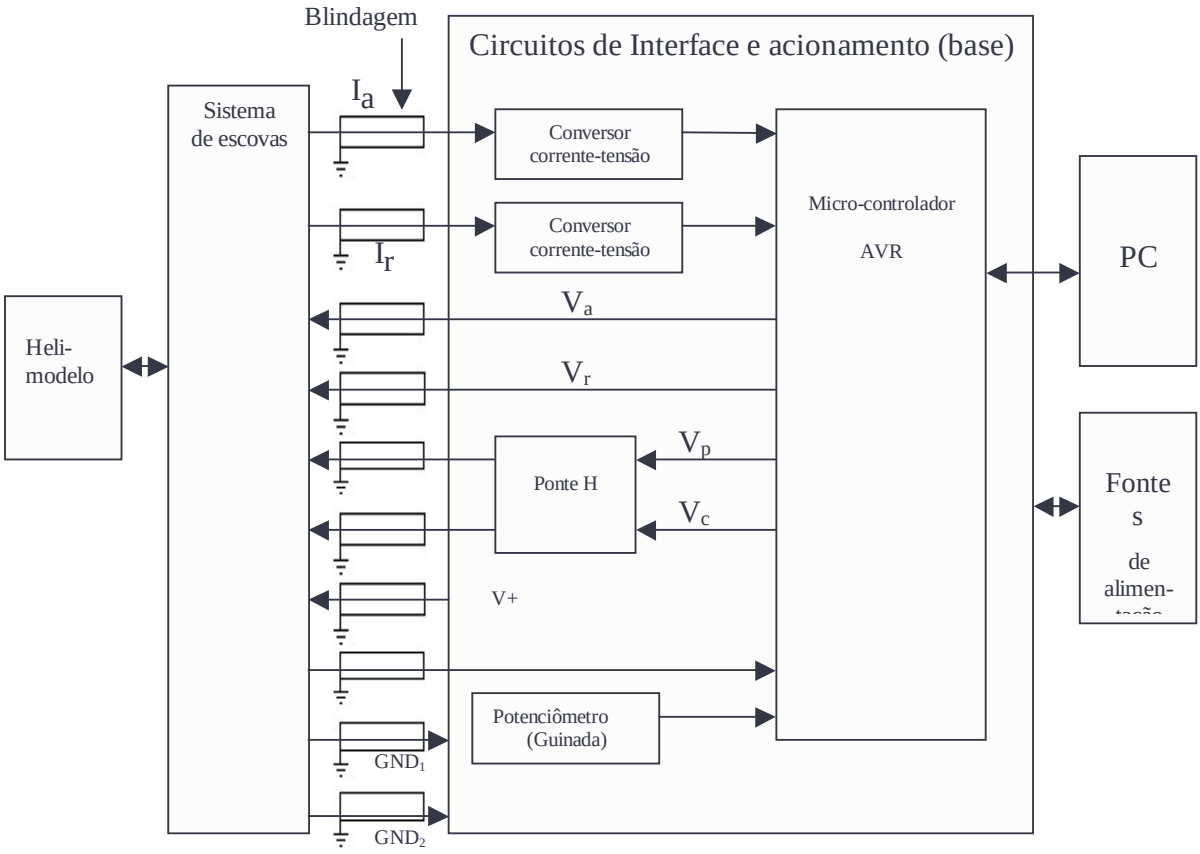


Figura 3.21 – Diagrama de blocos dos circuitos da base.

Uma legenda das variáveis descritas na Figura 3.20 está disponível na seção 3.2.2.

3.2.3.1. Conversor corrente-tensão

O circuito da Figura 3.22 é responsável pelo condicionamento das fontes de corrente dos potenciômetros para leitura pelo conversor A/D do microcontrolador.

Em termos gerais, o que precisa ser feito é converter o sinal de corrente proveniente do circuito embarcado (itens 3.2.2.1. e 3.2.2.2.) para um sinal de tensão que pode ser lido pelo conversor A/D de 10 bits encapsulado no microcontrolador AVR ATmega8. A simples utilização de um resistor foi descartada, pois sua resistência, quando somada à resistência das escovas, pode fazer saturar a fonte de corrente do potenciômetro. Além disso, faz-se necessário amplificar a aplicar um deslocamento à informação proveniente do circuito embarcado.

O princípio de funcionamento do circuito é a utilização de uma tensão de referência negativa e ajustável que determinará o offset da curva da tensão de saída para o conversor A/D em função da corrente de entrada. O ganho é determinado de modo a melhor adequar a faixa de medição da arfagem e rolagem à faixa de entrada do converso A/D do microcontrolador. Assim, devido a diferenças na faixa das medidas efetuadas para o ângulo de arfagem e rolagem, foi utilizada uma resistência de 1 kΩ para o primeiro caso e 430 Ω para o segundo.

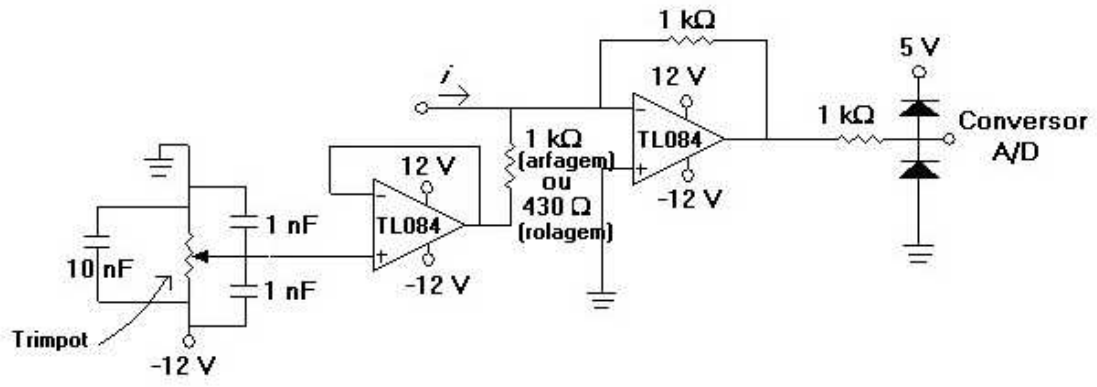


Figura 3.22 – Circuito de conversão corrente-tensão.

Para cada circuito de conversão, a tensão de referência foi ajustada de modo a se ter 2,5 V de saída quando o ângulo medido for 0º.

Além disso, foi utilizado um par de diodos na saída de cada um dos circuitos, de forma a proteger o conversor A/D de possíveis falhas no circuito amplificador, o que poderia induzir tensões de –12 a +12 V no conversor.

3.2.3.2. Ponte H

A partir do sinal PWM gerado pelo microcontrolador, foi utilizado o CI L298 (quatro meias pontes H) para o acionamento dos motores. Este dispositivo foi alimentado com o circuito regulador de tensão de potência mostrado na Figura 3.12(c), que é capaz de fornecer satisfatoriamente até 4 A de corrente.

A configuração implementada no circuito utiliza duas meias pontes H em paralelo para cada motor. Isso foi feito porque, nos testes iniciais realizados com os motores que vieram no kit, eles chegavam, juntos. a requisitar correntes que ultrapassavam 3,5 A. A figura 3.23 ilustra essa configuração, enquanto que a Figura 3.24 mostra o circuito completo.

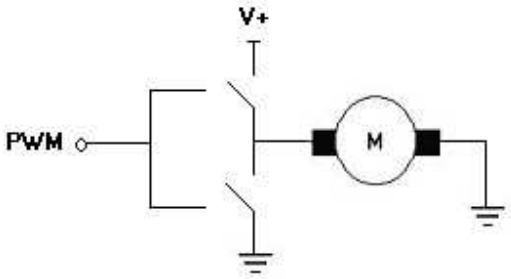


Figura 3.23 – Duas meias pontes H em paralelo para acionamento do motor.

Com as duas meias pontes em paralelo, o sentido de rotação do motor será sempre o mesmo. Isso, porém, não é um problema no caso do helicóptero, em que os rotores giram apenas em um sentido.

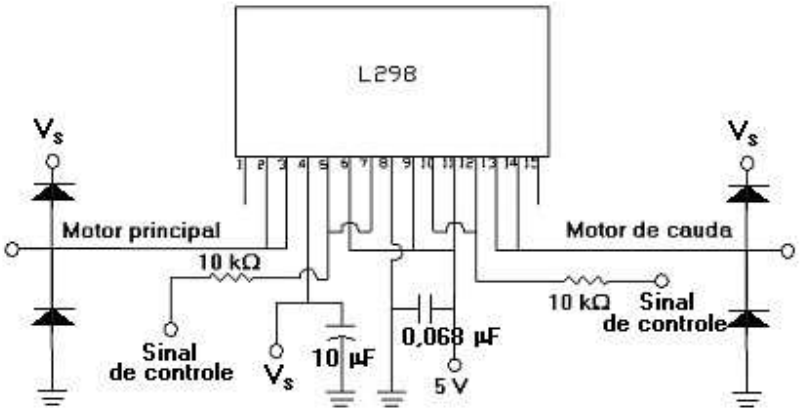


Figura 3.24 –Circuito de acionamento dos motores.

3.2.3.3. Interface microcontrolador/PC

A interface entre o microcontrolador e o PC tem duas funções principais. Uma delas é permitir a comunicação serial para a operação do sistema e a outra é possibilitar a regravação do microcontrolador.

O circuito de comunicação serial utiliza o CI MAX232, específico para aplicações desse tipo, está representado na Figura 3.25. Ele é utilizado simplesmente para converter o nível lógico TTL e CMOS para o protocolo 232. Para utilizar esse chip, são necessários alguns capacitores.

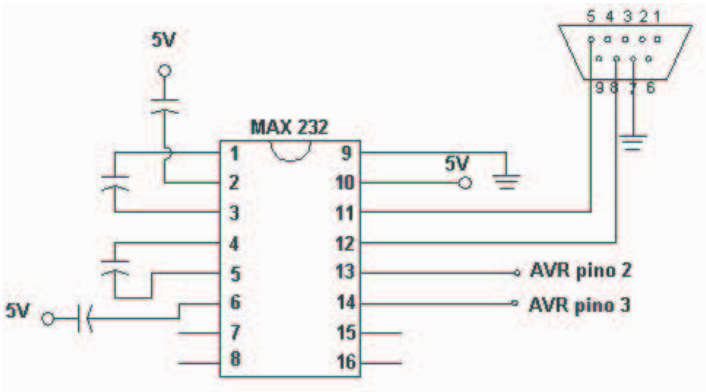


Figura 3.25 – O circuito de comunicação serial.

Já para a regravação do microcontrolador, foi utilizado um gravador externo disponível no laboratório. Dessa forma, as ligações que foram feitas na placa dizem respeito somente à correta conexão dos pinos do conector ao microcontrolador. A Figura 3.25 ilustra o circuito utilizado.

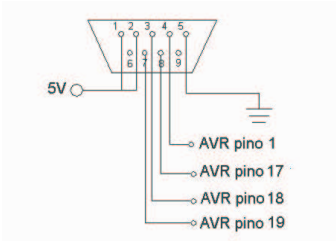


Figura 3.26 – Ligação com o programador.

3.3. Projeto Computacional

A parte computacional envolve basicamente o projeto de dois softwares: o software de interface e acionamento para o microcontrolador e o software de monitoração e controle para o PC. Os projetos de ambos estão descritos nas seções seguintes.

3.3.1. O microcontrolador

As funções principais do microcontrolador Atmel AVR ATmega8 dizem respeito à interface entre o helimodelo e seus atuadores e sensores com o PC. Dessa forma, o microcontrolador é responsável pela geração dos sinais de PWM, pela conversão dos sinais analógicos dos sensores e pela transmissão e recepção de informações para o PC.

A estrutura do código (Figura 3.27) é simples: o microcontrolador encontra-se num laço eterno à espera de comandos do PC pela serial, que podem solicitar dados dos sensores ou atualizar os valores dos sinais de PWM, que são gerados constantemente pelo microcontrolador. Nessa configuração, o microcontrolador atua como escravo do PC, não cabendo a ele nenhuma tomada de decisão acerca do processo.

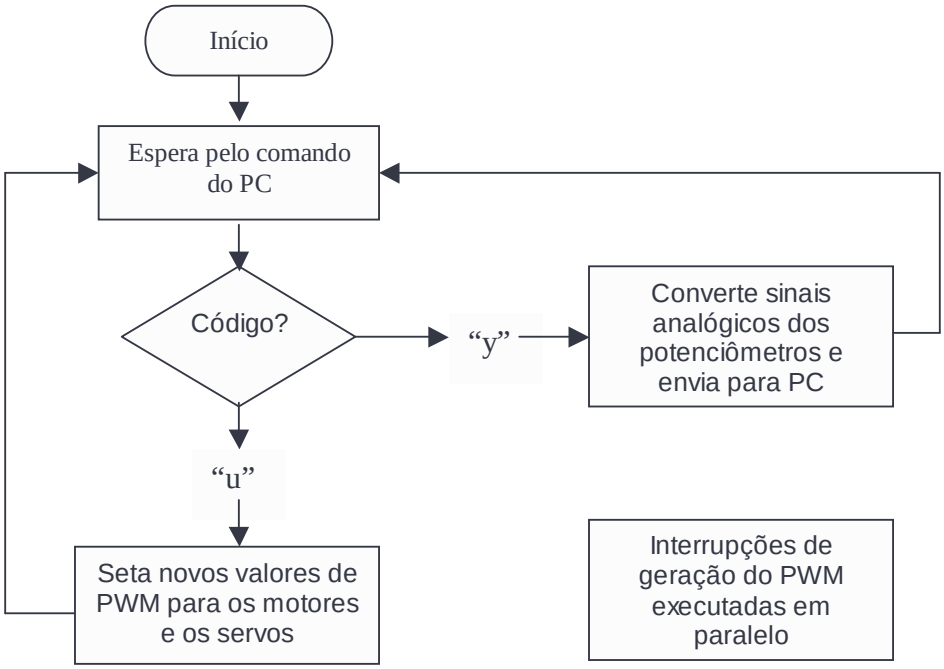


Figura 3.27 – Fluxograma principal do programa do microcontrolador.

O código do programa em C está disponível no anexo B.1.

3.3.1.1. Geração do PWM

A geração do sinal de PWM para os motores acoplados aos rotores do helicóptero ocorre de forma bem simplificada no microcontrolador. A partir do momento que se habilita o contador na freqüência desejada, é necessário apenas setar o registrador de configuração do PWM no valor correto e escrever o valor desejado no registrador de comparação. Dessa forma, a saída é determinada pela comparação entre o valor desse registrador e o atual valor do contador.

No caso do PWM gerado para o controle dos servos foi necessária uma abordagem diferente, pois havia dois problemas que não estavam presentes no caso anterior. Em primeiro lugar, o período do sinal de acionamento do servo é muito grande e teria-se de trabalhar com uma freqüência de clock muito pequena para alcança-lo. Além disso, não havia mais dois canais de PWM disponíveis, já que o AVR ATmega8 é capaz de gerar somente três sinais independentes de PWM.

Para a solução desse problema, um temporizador foi configurado de forma a gerar interrupções de *overflow* a cada 10 μs. Esse sinal é utilizado para gerar o PWM por software na freqüência correta (50 Hz) e ao mesmo tempo determinar a largura de pulso por meio de variáveis globais (u_1 e u_2) passadas à função (Figura 3.28). Desta forma, conseguiu-se gerar dois sinais de PWM, um para cada servo, com 50 Hz de freqüência, e ciclos de trabalho de 5 a 10 %.

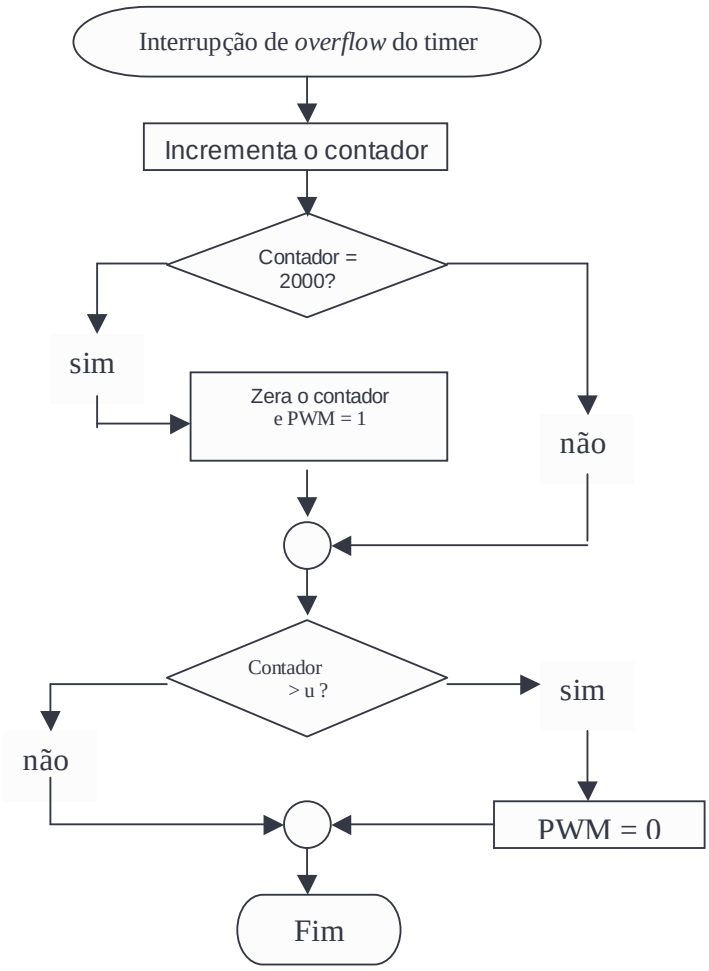


Figura 3.28 – Fluxograma da geração do PWM para os servos.

3.3.1.2. Conversão A/D

O AVR ATmega8 possui seis canais para conversão A/D. Porém, só há um conversor A/D encapsulado. Dessa forma, somente uma conversão pode ser efetuada a um determinado momento.

Assim, a partir do momento em que o PC envia a requisição acerca do estado do helicóptero, o programa varre os três canais de conversão utilizados e em seguida envia os dados por meio da comunicação serial.

3.3.1.3. Comunicação serial

A implementação da comunicação serial no microcontrolador AVR ATmega8 é relativamente simples. São utilizados registradores dedicados de transmissão e recepção e funções que implementam um laço de espera até que determinado registrador indique a conclusão da operação.

É utilizada uma taxa de transmissão de 38.400 bits por segundo (bps). Apesar da velocidade não muito elevada, esse foi um valor com o qual não foi verificado erro algum. A taxa de transmissão mostrou ser o fator crítico para determinar a taxa de amostragem para o processo, porém não era a maior preocupação do projeto computacional alcançar freqüências de amostragem mais altas que a utilizada. Atualmente, trabalha-se com período de amostragem de 40 ms.

Para todos os eventos de comunicação, foi implementada uma estratégia de detecção de erros baseada no reenvio da informação recebida, seja do lado do microcontrolador ou do PC. Dessa forma, ao utilizar esses "ecos", pôde-se verificar de forma mais eficiente os erros de comunicação que ocorreram nas etapas iniciais do projeto.

A mensagem enviada do PC é formada pelo comando ("u" ou "y"), seguida dos valores correspondentes de PWM, no caso do envio do sinal de controle. Por exemplo, o sinal em hexadecimal para acionar os motores e servos no máximo é 0x75FFFFFF, em que 75 é o código ASCII de "u", os dois bytes em seguida são comandos para os servos e os quatro últimos para os motores. Para o caso oposto, o microcontrolador envia os dados dos três sensores precedido do número de cada um dos canais.

3.3.2. O PC

Um período de amostragem constante é indispensável para a utilização dos métodos de controle digital baseados na transformada z. Dessa forma, uma das grandes dificuldades encontradas na utilização de computadores pessoais para controle digital diz respeito à garantia de se obter períodos constantes de amostragem.

Isso ocorre porque os sistemas operacionais genéricos atuais não permitem que o usuário tenha controle em tempo real do processador. Essa função fica a cargo do escalonador de processos.

Existem alguns sistemas baseados em Linux que possibilitam um maior controle ao usuário. Os principais exemplos são o RTLinux (*Real Time Linux*) e o RTAI (*Real Time Application Interface*). Ambos alcançam desempenhos muito superiores ao alcançado pelo sistema operacional Windows 2000®, adotado neste trabalho. Para o caso do Windows®, algumas variantes são o RTX e o InTime.

No sistema operacional Windows®, é possível atingir taxas de amostragem relativamente pequenas e um determinismo satisfatório por meio de alterações nos níveis de prioridade do programa. Assim, podem ser alcançados níveis de confiança da taxa real de amostragem muito maiores.

Parte do código utilizado no programa foi aproveitado de trabalhos anteriores do prof. Geovany. Inclusive, esse foi um dos motivos de se utilizar o Windows 2000®.

O programa, além de gerenciar o processamento em tempo real, é responsável pela comunicação serial com o microcontrolador, pela aquisição dos dados do joystick (por meio da API do Windows®), pelo cálculo das variáveis de controle (quando o sistema opera em modo automático), pela interface gráfica de apresentação dos dados em tempo real e pela geração do arquivo de dados utilizado para processamento *offline* da informação no Matlab®. A cada período de amostragem, é gerada uma interrupção que executa o módulo de programa descrito no fluxograma da Figura 3.29. Nesta tarefa, executada periodicamente, é feito o controle do helicóptero. O código dessa parte do programa está disposto no anexo B.2.

A implementação do PID digital se deu através do algoritmo disponível em [6]. Além disso, foi implementado um filtro digital para as medidas dos ângulos de orientação do helimodelo. Ele se tornou necessário devido aos problemas de ruídos nas medidas (item 3.2.2.3.). O filtro possui é de 1ª ordem e possui freqüência de corte em 1 Hz, porém, apesar

da qualidade das medidas obtidas, sua utilização causou um inevitável atraso nas medidas obtidas (aumento de ordem).

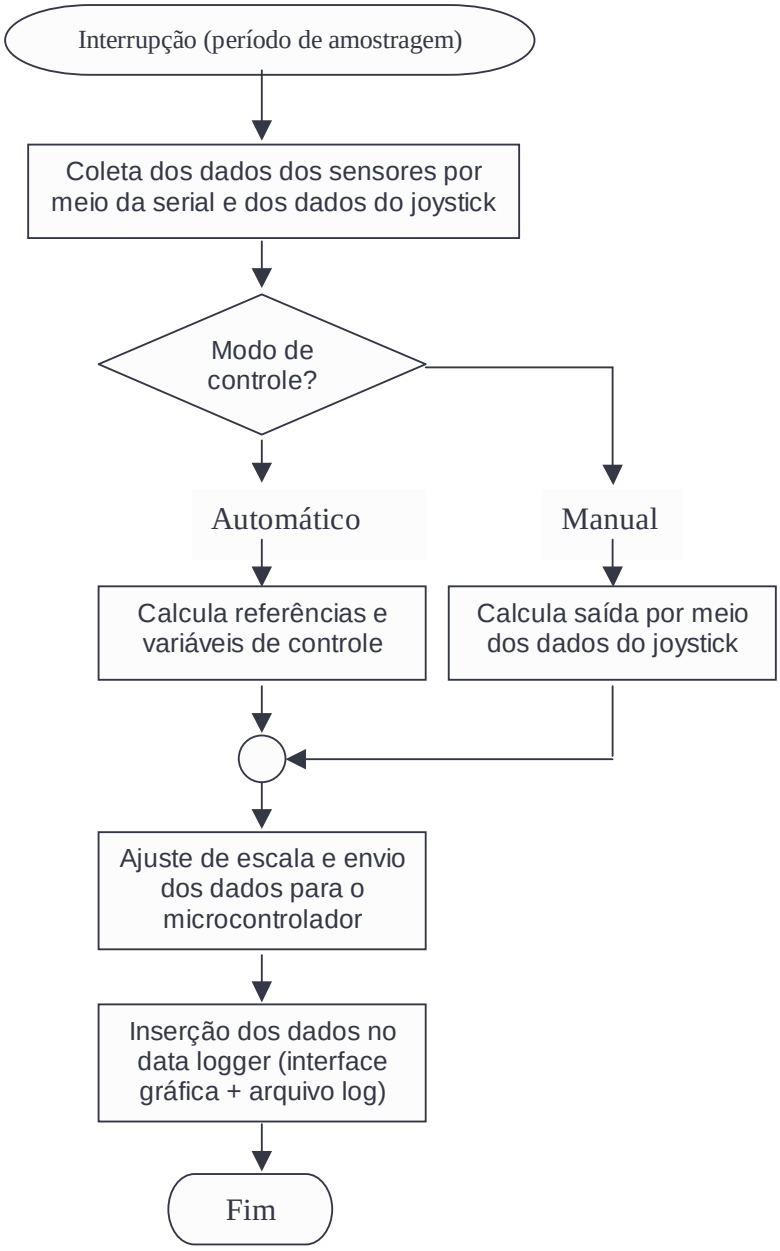


Figura 3.29 – Fluxograma principal do software do PC.

A interface gráfica, ilustrada na figura 3.30 permite a visualização dos dados de saída e entrada do sistema, bem como os sinais de referência. Além disso, disponibiliza o tempo total de execução e o status da comunicação serial. Por meio dela, também é possível

definir o modo de operação do sistema: manual ou automático. No modo manual, o comando do joystick é diretamente enviado ao helicóptero. Já no modo automático, o joystick é usado para determinar o sinal de referência a ser seguido pelo helicóptero. Os parâmetros do controlador, entretanto, são ajustados apenas no código do programa.

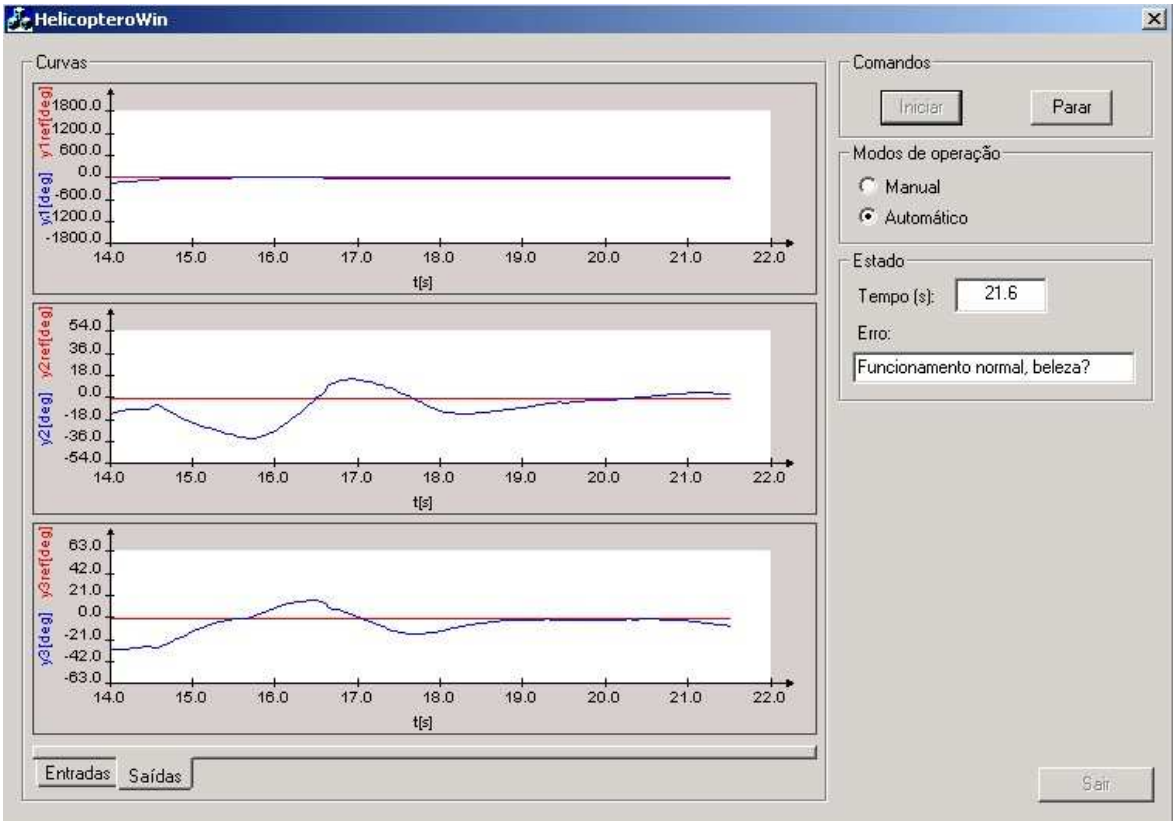


Figura 3.30 – A interface gráfica do software.

4. Modelamento matemático do helimodelo

Esse capítulo descreve uma tentativa de se obter um modelo matemático da atitude do helimodelo por meio da análise das leis física que regem o comportamento do sistema. Procurou-se obter um modelo em espaço de estados, a partir de algumas considerações feitas sobre o a dinâmica do processo.

O objetivo principal é demonstrar de forma mais clara algumas das características que tornam o sistema tão complexo: multivariável, fortemente acoplado, intrinsecamente não-lineare e inerentemente instável.

4.1. Considerações sobre helicópteros em escala reduzida

São muitos os modelos em escala reduzida de helicópteros disponíveis comercialmente nos dias de hoje. A maior parte deles é vendida simplesmente como veículos rádio-controlados para lazer. Esses helimodelos, porém, possuem uma ampla gama de novas aplicações que não possui o helicóptero tripulado. Entre elas, estão relacionadas todas aplicações em que a presença do homem traz riscos à vida ou é desnecessária; àquelas em que as dimensões reduzidas do helimodelo ou sua maior facilidade em realizar

manobras arriscadas é imperativa; além das aplicações provenientes do custo mais ameno desse sistema.

O helimodelo utilizado nesse trabalho difere, em certos aspectos, do helicóptero real descrito na seção 2.2 (princípios de funcionamento). A diferença mais importante é a ausência do comando coletivo - no helimodelo, há apenas atuação no comando cíclico e a sustentação varia conforme a velocidade de rotação do rotor principal. Diferente do helicóptero real, no helimodelo o comando cíclico atua de forma a inclinar todo o conjunto do rotor. As pás não são independentes e a inclinação de uma é sempre exatamente oposta à inclinação da outra. Em outras palavras, um aumento do ângulo de incidência em uma das pás (δ_1) é igual à diminuição do ângulo de incidência da outra (δ_2). Portanto, o ângulo de incidência médio (δ) (definido por: $\delta = (\delta_1+\delta_2)/2$) é sempre constante.

Outra diferença importante é que o rotor de cauda não é conectado ao rotor principal e possui um motor próprio. No helicóptero real, o rotor de cauda gira a velocidade constate e o torque depende do ângulo de ataque das pás. Já no helimodelo, as pás do rotor de cauda são fixas e o torque depende da velocidade de rotação.

Além disso, os helicópteros em escala reduzida possuem uma barra estabilizadora, para diminuir os efeitos causados pela redução na dimensão do veículo. Esses efeitos podem ser sintetizados num aumento da sensibilidade aos comandos de controle e perturbações devido à diminuição do diâmetro do rotor. Essa diminuição acarreta numa atenuação do efeito de atraso induzido pelo rotor ao efetuar os movimentos de arfagem e rolagem. De fato, a constante de tempo τ do rotor de um helimodelo é $\sqrt{N}$ vezes menor do que a de um helicóptero comercial, onde N é o fator de escala [12]. Esse efeito, entretanto, não foi considerado no modelamento matemático a seguir.

4.2. Modelamento matemático do helimodelo

Historicamente, há uma tendência em se tratar separadamente as etapas de identificação e projeto de controle de um sistema. Nesse trabalho, tentou-se abordar técnicas que fundem esses dois problemas em um só, até porque essa alternativa se mostra mais eficiente em sistemas complexos, como o representado pelo helimodelo. Porém,

mesmo com essas considerações, nesse momento serão apresentadas separadamente questões relativas ao modelamento físico e matemático do helicóptero.

O modelo dinâmico de sistemas mecânicos é, na maioria das vezes, baseado na análise direta das forças que agem sobre ele. No caso do modelo de atitude do helicóptero, é necessário computar diversas relações de torque e força que agem sobre o corpo do helicóptero em cada eixo de rotação: arfagem (θ), rolagem (ϕ) e guinada (ψ). Alguns trabalhos que tratam dessa questão são [1], [2], [4], [11], [12], [17] e [20].

A seguir será apresentado o modelo físico obtido para o helimodelo utilizado neste experimento. O procedimento seguido assemelha-se ao proposto em [4], porém foi utilizado muitas vezes como referência [7].

4.2.1. Arfagem

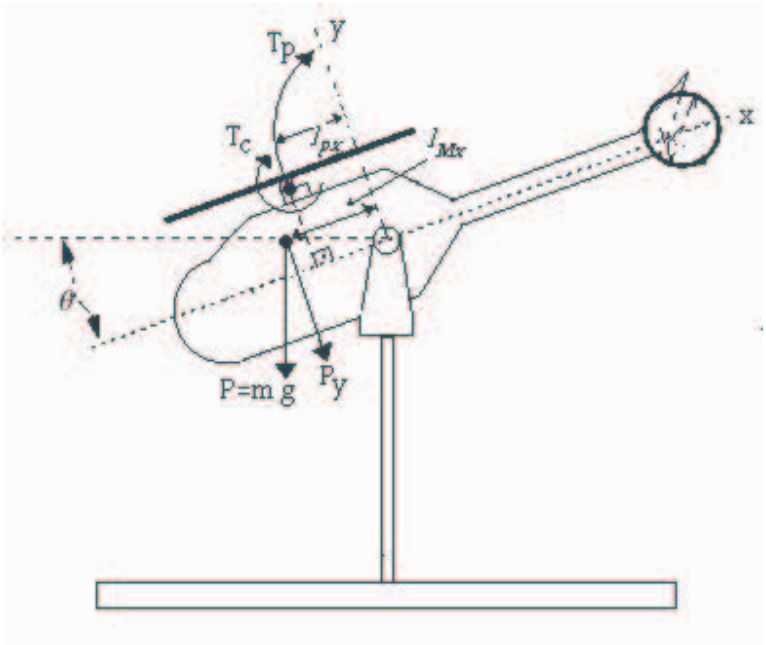


Figura 4.1 – Esboço do movimento de arfagem com ϕ igual a zero.

Com relação à arfagem do helicóptero, a resultante dos torques agindo sobre o sistema leva a

$$I_{Ma}.\ddot{\theta} = T_P - T_D - T_A - T_M + T_G + T_C + T_T \quad (4.1)$$

sendo

I_{Ma} Momento de inércia referente à arfagem [Kg.m^2]

θ Ângulo de arfagem [rad]

T_P Torque devido ao empuxo gerado pelo rotor principal [N.m]

T_A Torque gerado pelo atrito [N.m]

T_M Torque gerado pela massa do corpo do helicóptero [N.m]

T_G Torque gerado pelo efeito giroscópico [N.m]

T_C Torque gerado pelo cíclico [N.m]

T_T Torque devido ao empuxo gerado pelo rotor traseiro [N.m]

O torque devido à ação da gravidade na massa do próprio helicóptero gera uma força aplicada no seu centro de gravidade, sempre apontando para baixo. Dessa forma, o torque resultante será alterado de acordo com o ângulo θ. Assim, o torque devido ao próprio peso do helicóptero pode ser dado por:

$$T_M = l_M.m.g.\cos(\theta) \quad (4.2)$$

sendo

l_M Distância do centro de gravidade ao centro de rotação [m]

m Massa do sistema [kg]

g Aceleração da gravidade [m/s^2]

O torque que resulta da propulsão gerada pelo rotor principal é mais difícil de ser computado, pois os modelos que descrevem o empuxo gerado por uma hélice em rotação são geralmente complexos [13]. De uma forma geral, porém, pode-se dizer que esses dispositivos são projetados de forma a gerar uma pressão relativa negativa entre suas porções superior e inferior. Sabe-se também que a diferença de pressão é proporcional ao

quadrado da velocidade de rotação e que a força é dada pela razão entre a pressão e a área. Assim, o torque de sustentação gerado pelo rotor principal é dado por:

$$T_P = \frac{K_P.\omega_P^2.r_P.\cos(\phi)}{2}.l_P \tag{4.3}$$

sendo

ω_P Velocidade de rotação do rotor principal [rad/s]

l_P Distância paralela ao eixo do helicóptero entre o rotor principal e ao centro da junta [m]

r_P Raio do rotor principal [m]

O torque devido ao atrito no rolamento da junta (T_A) é dado por:

$$T_A = \mu_{Aa}.\dot{\theta} + K_{Aa} \tag{4.4}$$

sendo

μ_{Aa} Coeficiente de atrito viscoso da arfagem [Kg.m/s.rad]

K_{Aa} Componente de atrito estático da arfagem, que depende da força normal total aplicada [N.m]

O torque causado pelo rotor de cauda quando o ângulo de rolagem (ϕ) é diferente de zero é denominado T_T. Esse efeito pode vir a influenciar positivamente ou negativamente no movimento de arfagem do helicóptero. Ao aplicar-se o modelo de empuxo gerado pelo rotor principal no rotor traseiro, chega-se à seguinte relação:

$$T_T = \frac{K_T.\omega_T^2.r_T.\text{sen}(\phi)}{2}.l_T \tag{4.5}$$

Em que

ω_T Velocidade de rotação do rotor de cauda [rad/s]

l_T Distância paralela ao eixo do helicóptero entre o rotor de cauda e ao centro da junta [m]

r_T Raio do rotor de cauda [m]

O efeito de precessão giroscópica deve ser considerado porque a velocidade de rotação é elevada. Esse efeito gera o chamado torque de reação ao efeito giroscópico (T_G) e é percebido 90º adiante, no sentido da rotação do rotor principal. Uma variação do ângulo de arfagem (θ) causa uma reação no sentido da rolagem, afetando, portanto o ângulo de rolagem (ϕ). E uma variação de ϕ causa uma reação no sentido da arfagem (θ).

O torque T_G no sentido da arfagem é função do ângulo de rolagem (ϕ) e é dado por:

$$T_G = I_r.\ddot{\phi} \tag{4.6}$$

Por último, tem-se o torque gerado pelo cíclico (T_C). O T_C é causado por δ_a, que é a diferença entre o ângulo de incidência da pá que avança (δ_1) e o ângulo de incidência da pá que retorna e (δ_2).

$$T_C = \frac{K_P.\omega_P^2.\text{sen}(\delta_a)}{2}.l_P \tag{4.7}$$

Ao substituir as equações de (4.2) até (4.7) na equação (4.1), encontra-se a equação diferencial que define o comportamento do helicóptero no eixo de arfagem:

$$\begin{aligned} I_{Ma}.\ddot{\theta} = {} & \frac{K_P.\omega_P^2.r_P.\cos(\phi)}{2}.l_P - \left(\mu_{Aa}.\dot{\theta} + K_{Aa}\right) - l_M.m.g.\cos(\theta) + .. \\ & + I_r.\ddot{\phi} + \frac{K_P.\omega_P^2.\text{sen}(\delta_a)}{2}.l_P + \frac{K_T.\omega_T^2.r_T.\text{sen}(\phi)}{2}.l_T \end{aligned} \tag{4.8}$$

4.2.2. Rolagem

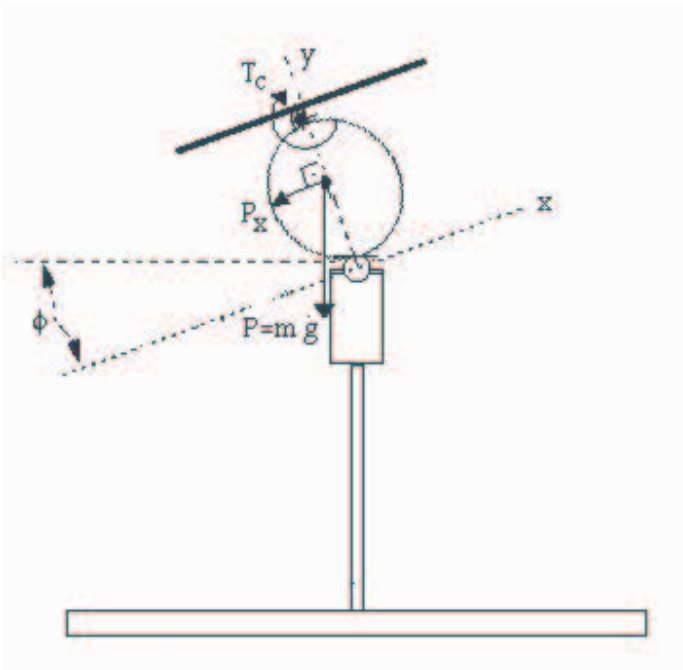


Figura 4.2 – Esboço do movimento de rolagem com θ igual a zero.

Com relação à rolagem do helicóptero, a resultante dos torques que agem sobre o sistema leva a:

$$I_{Mr}.\ddot{\phi} = T_C - T_A - T_M + T_G \tag{4.9}$$

em que

I_{Mr} Momento de inércia referente à rolagem [Kg.m^2]

ϕ Ângulo de rolagem [rad]

T_C Torque gerado pelo cíclico [N.m]

T_A Torque gerado pelo atrito [N.m]

T_M Torque gerado pela massa do corpo do helicóptero [N.m]

T_G Torque gerado pelo efeito giroscópico [N.m]

O torque devido ao próprio peso do helicóptero pode ser dado por:

$$T_M = m.g.\text{sen}(\phi) \tag{4.10}$$

De forma similar à equação (4.4), o torque devido ao atrito:

$$T_A = \mu_{Ar}.\dot{\phi} + K_{Ar} \tag{4.11}$$

O torque de reação devido ao efeito giroscópico é dado por:

$$T_G = I_r.\ddot{\theta} \tag{4.12}$$

Como no caso da arfagem, o T_G para a rolagem também se manifesta 90º adiante, no sentido da rotação. Afetando o ângulo de arfagem (θ).

Por último, tem-se o torque gerado pelo cíclico, que é dado por:

$$T_C = \frac{K_P.\omega_P^2.\text{sen}(\delta_r)}{2}.l_P \tag{4.13}$$

Ao substituir-se as equações de (4.10) a (4.13) na equação (4.9), encontra-se a relação matemática que define o comportamento do helicóptero no eixo de rolagem:

$$I_{Mr}.\ddot{\phi} = \frac{K_P.\omega_P^2.\text{sen}(\delta_r)}{2}.l_P - \left(\mu_{Ar}.\dot{\phi} + K_{Ar}\right) - m.g.\text{sen}(\phi) + I_r.\ddot{\theta} \tag{4.14}$$

4.2.3. Guinada

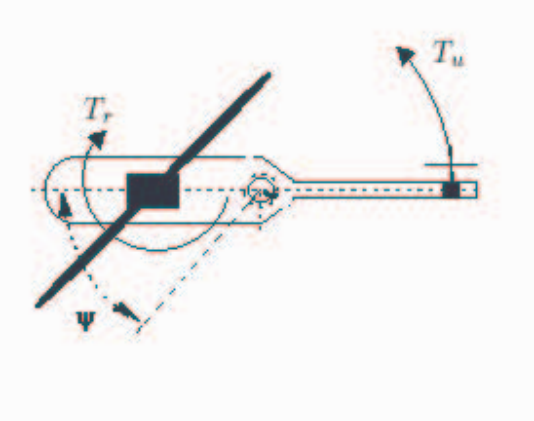


Figura 4.3 – Esboço do movimento de guinada com θ e ϕ iguais a zero.

Com relação à guinada, cujo esboço está representado na Figura 4.3, a resultante dos torques agindo sobre o sistema leva a:

$$I_{Mg}.\ddot{\psi} = T_U - T_A - T_R \tag{4.15}$$

sendo

I_{Mg} Momento de inércia referente à guinada [Kg.m^2]

ψ Ângulo de guinada [rad]

T_U Torque gerado pelo rotor de cauda [N.m]

T_A Torque gerado pelo atrito [N.m]

T_R Torque gerado pela reação ao rotor principal [N.m^2]

O momento de inércia em relação ao eixo de guinada (I_{Mg}) varia em função do eixo de arfagem (θ), da maneira ilustrada na Figura 4.4, e pode ser definido por:

$$I_{Mg} = \cos(\theta).I_{Mg\_\max} \tag{4.16}$$

A distância do rotor de cauda ao eixo de rotação (l) também varia conforme θ:

$$l = \cos(\theta).l_{\max} \tag{4.17}$$

O torque de reação (T_R) devido à rotação do rotor principal será dado por:

$$T_R = \cos(\theta).I_{Mr}.\dot{\omega}_P + \frac{K_R.\omega_P^2.\rho.S_D.\cos(\phi)}{2} \tag{4.18}$$

onde

I_{Mr} Momento de inércia do rotor principal [Kg.m]

K_R Constante determinada pela viscosidade do ar, pelo formato da pá e pelo ângulo de incidência.

S_D Área frontal da pá, perpendicular ao plano de rotação [m^2]

O torque gerado pelo rotor de cauda (T_U) é:

$$T_U = \frac{K_T.\omega_T^2.r_T.\text{sen}(\phi)}{2}.l_T.\cos(\theta) \tag{4.19}$$

Por último, há o torque devido ao atrito (T_A):

$$T_A = \mu_{Ag}.\dot{\psi} + K_{Ag} \tag{4.20}$$

Ao substituir as equações de (4.18) a (4.20) na equação (4.15), encontra-se a relação matemática que define o comportamento do helicóptero no eixo de guinada:

$$I_{Mg}.\ddot{\psi} = \frac{K_T.\omega_T^2.r_T.\text{sen}(\phi)}{2}.l_T.\cos(\theta) - \left(\mu_{Ag}.\dot{\psi} + K_{Ag}\right) - .. \quad (4.21)$$
$$... - \left(\cos(\theta).I_{Mr}.\dot{\omega}_P + \frac{K_R.\omega_P^2.\rho.S_D.\cos(\phi)}{2} \right)$$

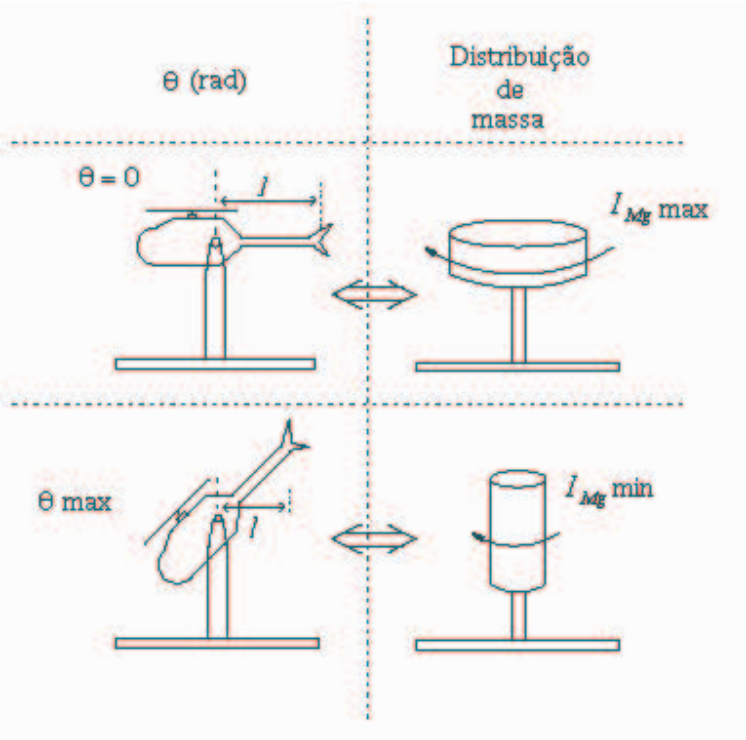


Figura 4.4 – Variação do momento de inércia de rotação em torno do eixo da guinada em função de *θ.*

4.2.4. Representação em espaço de estados

A representação do modelo do sistema no espaço de estados tem início com a determinação dos estados correspondentes. Dessa forma, ao analisar a dinâmica modelada na seções anteriores (2.5.2. a 2.5.4.), chega-se a um número de seis estados para representar todos movimentos de atitude do helimodelo, assim definidos:

$$x_1 = \theta \qquad x_2 = \phi \qquad x_5 = \psi \qquad x_6 = \dot{\psi} \quad (4.22)$$

$$x_3 = I_{Ma}.\dot{\theta} - I_r.\dot{\phi} \qquad\qquad x_4 = I_{Mr}.\dot{\phi} - I_r.\dot{\theta}$$

As entradas do sistema são definidas por

$$u_1 = \omega_P^2 \qquad u_2 = \omega_T^2 \qquad u_3 = \dot{\omega}_P \qquad u_4 = \text{sen}(\delta_a) \qquad u_5 = \text{sen}(\delta_r) \tag{4.23}$$

Ao definir o sistema dessa forma, chega-se a uma representação não linear em espaço de estados que pode ser da forma:

$$\dot{x} = f(x) + g(x,u) \tag{4.24}$$

Além disso, como não foi definido nenhum estado para $\dot{\theta}$ e $\dot{\phi}$, temos que defini-los por meio de x_3 e x_4 (especificados em (4.22)). Assim, $\dot{\theta}$ e $\dot{\phi}$ são definidos por:

$$\dot{\theta} = \left(\frac{I_{Mr}}{I_{Ma}.I_{Mr} + I_r^2} \right).x_3 - \left(\frac{I_r}{I_{Ma}.I_{Mr} + I_r^2} \right).x_4 \tag{4.25}$$

$$\dot{\phi} = \left(\frac{I_{Ma}}{I_{Ma}.I_{Mr} + I_r^2} \right).x_4 - \left(\frac{I_r}{I_{Ma}.I_{Mr} + I_r^2} \right).x_3 \tag{4.26}$$

Dessa forma, a representação em espaço de estados completa do sistema pode ser dada por:

$$\begin{pmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\\ \dot{x}_5\\ \dot{x}_6\end{pmatrix} = \begin{pmatrix} \left(\frac{I_{Mr}}{I_{Ma}.I_{Mr}+I_r^2}\right).x_3 - \left(\frac{I_r}{I_{Ma}.I_{Mr}+I_r^2}\right).x_4 \\ \left(\frac{I_{Ma}}{I_{Ma}.I_{Mr}+I_r^2}\right).x_4 - \left(\frac{I_r}{I_{Ma}.I_{Mr}+I_r^2}\right).x_3 \\ -\left(\mu_{Aa}.\left(\left(\frac{I_{Mr}}{I_{Ma}.I_{Mr}+I_r^2}\right).x_3 - \left(\frac{I_r}{I_{Ma}.I_{Mr}+I_r^2}\right).x_4\right)+K_{Aa}\right) - l_M.m.g.\cos(x_1) \\ -\left(\mu_{Ar}.\left(\left(\frac{I_{Ma}}{I_{Ma}.I_{Mr}+I_r^2}\right).x_4 - \left(\frac{I_r}{I_{Ma}.I_{Mr}+I_r^2}\right).x_3\right)+K_{Ar}\right) - m.g.\text{sen}(x_2) \\ x_6 \\ -(\mu_{Ag}.x_6+K_{Ag}) \end{pmatrix} + .. \quad (4.27)$$

$$.. + \begin{pmatrix} 0 \\ 0 \\ \frac{K_P.u_1.l_P}{2}(r_P.\cos(x_2)+u_4)+\frac{K_T.u_2.r_T.\text{sen}(x_2)}{2}.l_T \\ \frac{K_P.u_1.u_5}{2}.l_P \\ 0 \\ \frac{K_T.u_2.r_T.\text{sen}(x_2)}{2}.l_T.\cos(x_1)-\left(\cos(x_1).I_{Mr}.u_3+\frac{K_R.u_1.\rho.S_D.\cos(x_2)}{2}\right) \end{pmatrix}$$

Percebe-se de (4.27) que, devido às considerações feitas (item 4.2.5), não há influência do movimento de guinada nos movimentos de arfagem e rolagem. Sendo assim, o sistema completo poderia ser subdivido em dois, em que o primeiro representaria a dinâmica de arfagem e rolagem e o segundo a dinâmica de guinada. Este segundo (Figura 4.5) receberia como entradas, além das velocidades dos rotores (ω_P e ω_T, respectivamente) e aceleração do rotor principal, as elevações de arfagem e rolagem (θ e ϕ, respectivamente).

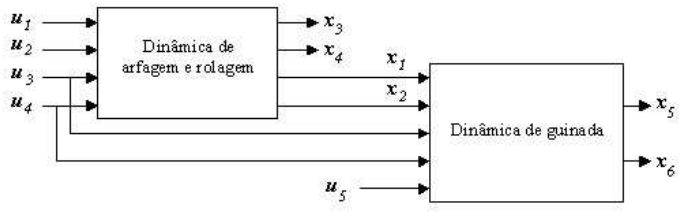


Figura 4.5 – Uma outra representação possível de (4.27). As entradas estão representadas de maneira diferente do que foi definido em (4.2.4).

4.2.5. Considerações sobre o modelamento

Na obtenção do modelo descrito em (4.27), foram feitas determinadas considerações que simplificaram bastante o modelamento matemático. Fazem parte dessa lista efeitos cuja intensidade foi considerada desprezível. O modelo obtido é, portanto, um compromisso entre a complexidade e o detalhamento do sistema.

Entre esses efeitos, encontra-se o efeito centrífugo, que ocorre devido à rotação do eixo de guinada do helicóptero, e o torque devido à precessão giroscópica resultante também da rotação do helimodelo no eixo de guinada. Em ambos caso, ocorre um torque no eixo de arfagem: o primeiro tende a diminuir a amplitude de θ, enquanto o segundo tem efeito contrário. Os dois efeitos foram desprezados na análise principalmente em decorrência das baixas velocidades de rotação do helicóptero nesse eixo ($\dot{\psi}$). De fato, observa-se que os movimentos no eixo de guinada ocorrem, na maioria das vezes, quando o helicóptero está em repouso com relação aos outros dois eixos, situação em que o efeito devido a precessão giroscópica é anulado.

Outro efeito que foi desprezado na computação do modelo foi o torque de reação do rotor de cauda no corpo do helicóptero. Isso foi feito porque a massa e a propulsão gerados pelo rotor de cauda são realmente desprezíveis em relação ao helimodelo. De fato, verificou-se experimentalmente a quase completa ausência desse efeito, ao contrário do registrado em relação ao torque de reação do rotor principal.

Além disso, decidiu-se por ignorar as variações de velocidade de rotação do rotor principal (ω_P). Essas flutuações ocorrem principalmente devido ao aumento do arrasto aerodinâmico causado pelo comando cíclico. Seus efeitos são de difícil compreensão e a própria consideração de uma velocidade ω_P variável tornaria muito complexa muitas das formulações matemáticas apresentadas. De fato, apesar de ter sido considerado que essas flutuações são de amplitude desprezível, fica como proposta para futuros trabalhos a implementação do controle de velocidade do rotor principal.

Por último, considerou-se que o atrito mecânico entre a junta e suas partes que apóiam o helicóptero são de mesma amplitude para a rolagem e a arfagem. Essa

consideração é bastante razoável e visa somente simplificar a visualização do modelo final obtido, já que a computação dos parâmetros apresentados nas equações (4.8), (4.14) e (4.20) não foram computados ou medidos.

5. Avaliação experimental do controle

O objetivo inicial proposto do projeto incluía a concepção e construção da bancada experimental para ensaios de atitude de um helimodelo, de forma a criar um ambiente

Deu-se início, porém, ao estudo de estratégias de controle de atitude para o helimodelo. Nesse sentido, como foi descrito no item 3.3.2., primeiramente foi implementado um controlador PID para seguir a referência passada pelo joystick. Esse procedimento está descrito em 4.1. Em seguida, à procura de uma abordagem mais completa, partiu-se para a identificação do sistema representado pelo helicóptero (4.2).

5.1. Síntese do controlador PID

Os efeitos dos controladores proporcional, integral e derivativo em sistemas contínuos no tempo são bem conhecidos. Entretanto, para implementar tal ação de controle num sistema digital, deve-se transformar as equações diferenciais correspondentes em equações de diferença. Utilizando a aproximação da derivada conhecida como método de Euler (4.1), obtém-se a lei de controle representada por (4.2) [6].

$$\dot{x}(k) \cong \frac{x(k+1) - x(k)}{T} \tag{4.1}$$

onde

T Período de amostragem [s]

$$u(k) = u(k-1) + K_P.\left[\left(1 + \frac{T}{T_I} + \frac{T_D}{T}\right).e(k) - \left(1 + 2.\frac{T_D}{T}\right).e(k-1) + \frac{T_D}{T}.e(k-2)\right] \quad (4.2)$$

sendo

K_P Ganho proporcional

T_I Termo integral

T_D Termo derivativo

O helimodelo, entretanto, não é um sistema de uma entrada e uma saída. São três variáveis a controlar (θ, ϕ, ψ), a partir de quatro variáveis de controle (velocidade de rotor principal e de cauda e os servos de arfagem e rolagem). Assim, todas as representações de variáveis no programa são feitas em matrizes: a matriz e_{3x4} representa os erros das três variáveis medidas em três instantes distintos, a matriz u_{2x4} representa as variáveis de controle atuais e num instante anterior e os vetores K_p, T_I e T_D possuem quatro valores cada (um para cada eixo de orientação controlado).

A partir da estrutura descrita, os valores do PID foram ajustados experimentalmente até alcançar um bom desempenho.

Nessa abordagem, considera-se que os estados do sistema são desacoplados. Além disso, supõe-se que cada saída é função somente de uma variável de controle. Essas duas suposições são falsas, como já foi explicitado no capítulo 4. O segundo desses problemas diagnosticados poderia ser solucionado criando-se matrizes dos termos do PID, ao invés de vetores. Cada termo corresponderia ao ganho de determinada variável de controle. Entretanto, tal abordagem resultaria numa imensa quantidade de parâmetros a ser ajustada.

Com relação aos eixos de arfagem e rolagem, a determinação dos parâmetros do controlador foi realizada sem muita dificuldade. Os resultados alcançados foram visualmente satisfatórios, ainda mais se comparados ao controle humano. A única precaução que foi tomada foi a limitação da variação da velocidade dos rotores, pois uma velocidade extremamente baixa do rotor principal, invariavelmente derruba o helicóptero.

Algumas curvas de resposta do helimodelo controlado pelo PID e respectivos sinais de controle são mostrados na Figura 5.1.

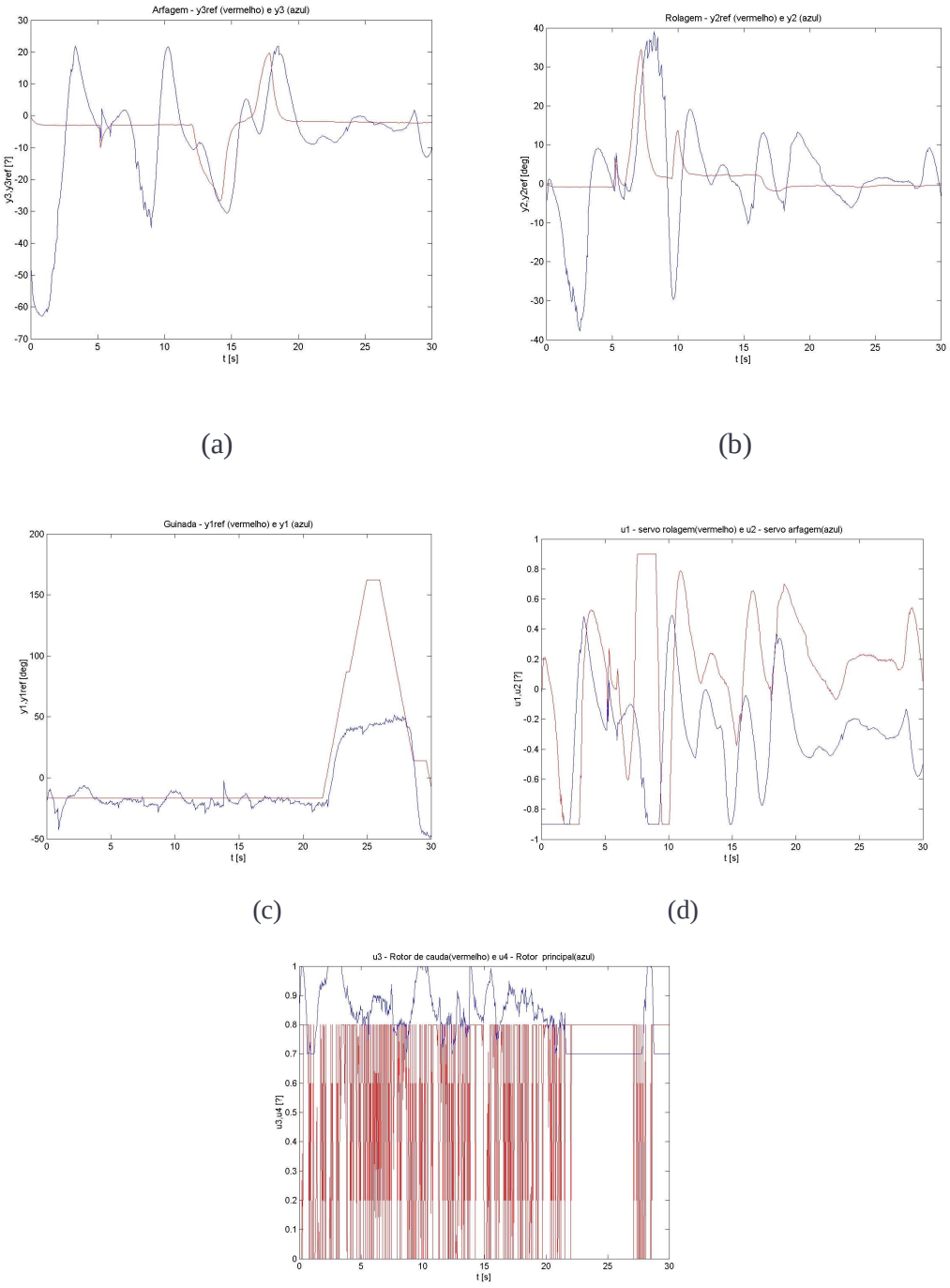


Figura 5.1 – Resposta com o controlador para (a) arfagem, (b) rolagem, (c) guinada e (d) os comandos dos servos e (e) os comandos aos rotores.

Com relação à guinada, entretanto, os testes iniciais indicaram maiores dificuldades. Isso ocorreu principalmente porque a plataforma experimental possibilita o helimodelo movimentar-se até 10 voltas em torno do eixo da guinada. Assim, como o erro é muito grande e a conseqüente correção desse erro é lenta, o termo integrador do PID saturava o atuador (rotor de cauda), inutilizando o laço de realimentação e gerando instabilidade. Para contornar esse problema, foi utilizada uma técnica de compensação de saturação do controlador (*anti-windup*). O efeito que essa saturação do controlador pode causar está disposto na figura 5.2 [10]. Nesse caso, apenas se anula o efeito do termo integrador quando saída do controlador está saturada. Um exemplo da resposta de guinada utilizando o *anti-windup* está ilustrada na Figura 5.1.

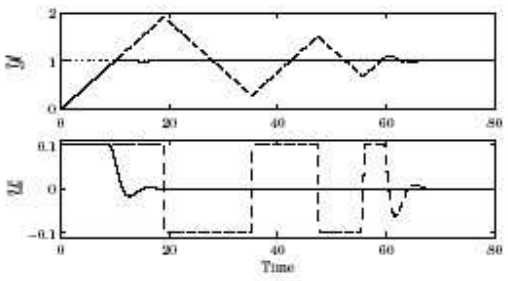


Figura 5.2 – Controlador PID com (contínuo) e sem (tracejado) *anti-windup*.

Em resumo, o controlador PID considera que a arfagem depende somente da atuação no servo de arfagem, a rolagem é função apenas do servo de rolagem, enquanto que a guinada é influenciada pelos rotores principal e de cauda. Porém, pode-se ver na Figura 5.1 alguns dos efeitos de acoplamento existentes, bem como o comportamento peculiar do sinal de controle do rotor de cauda (Figura 5.1(e)).

Como já foi dito, uma análise visual do desempenho chega a conclusões positivas. Ao se verificar as curvas de resposta do sistema com mais atenção com o auxílio do Matlab®, porém, verificou-se que a performance do controlador PID implementado não foi muito satisfatória.

5.2. Identificação do sistema

Devido à necessidade de se obter controladores mais adequados, tornou-se mandatória a obtenção de um modelo matemático do sistema. O cálculo e medição dos parâmetros descritos na seção 4.2 são tarefas muito dispendiosas, tanto pela quantidade, quanto pela complexidade dos ensaios e procedimentos exigidos. Dessa forma, propôs-se a utilização de técnicas de *identificação de sistemas*.

Dessa forma, a identificação de sistemas, ao contrário da pura dedução do modelo, utiliza um conjunto de dados proveniente de testes realizados com o sistema em funcionamento. A partir desses dados é possível obter aproximações do modelo real através do ajuste dos parâmetros de uma estrutura pré-definida pelo projetista. Assim, o procedimento de identificação é composto das etapas descritas na Figura 5.3.

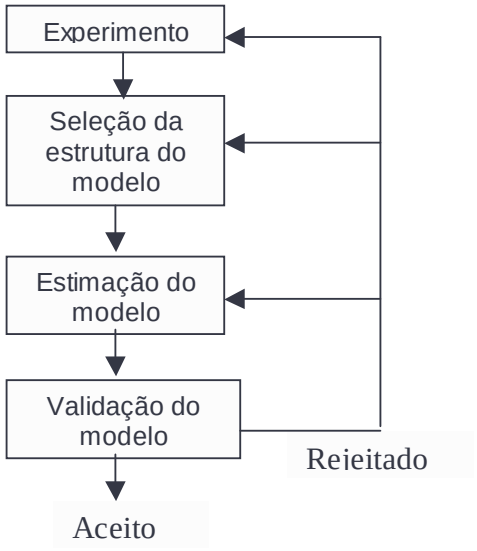


Figura 5.3 – Procedimento de identificação de sistemas.

A etapa correspondente experimental, compreende a excitação do sistema por um sinal que abranja a totalidade de seus modos de operação. Em sistemas lineares, são necessários poucos sinais para uma identificação coerente, mas para sistemas não-lineares, como o helicóptero, é preciso sinais de freqüências e amplitudes as mais diversas possíveis. No nosso caso, o sinal de excitação foi gerado por um controlador humano operando o joystick. Abordou-se também uma tentativa de identificação do sistema para uma

determinada faixa de operação. Nesse caso, o sistema foi simplesmente colocado em seu modo de operação automático, com nenhum acionamento do joystick (o que leva o helimodelo a se estabilizar em todos os eixos de orientação).

A segunda etapa do procedimento descrito diz respeito à seleção de um modelo matemático para descrever o sistema. Para tanto, considera-se que o sistema pode ser representado por:

$$y(t) = G(q^{-1}).u(t) + H(q^{-1}).e(t) \tag{4.3}$$

Assumindo então (4.4) e (4.5), chega-se a uma estrutura do modelo definida por (4.6) e (4.7). Esse é o modelo mais utilizado para identificação de sistemas dinâmicos e é conhecido como ARX, da sigla em inglês para *AutoRegressive, eXternal input*. Foi utilizado esse modelo, descrito em [15], para cada uma das saídas descritas no sistema.

$$G(q^{-1},\theta) = q^{-d}.\frac{B(q^{-1})}{A(q^{-1})} \tag{4.4}$$

$$H(q^{-1},\theta) = \frac{1}{A(q^{-1})} \tag{4.5}$$

$$\varphi(t) = [y(t-1)..y(t-n), u(t-d)..u(t-d-m)]^T \tag{4.6}$$

onde

m Número de “atrasos” de entrada

n Número de “atrasos” de saída

$$\theta = [-a_1 ... -a_n, b_0 ... b_m]^T \tag{4.7}$$

Assim, o modelo ARX descreve um sistema IIR, do inglês *Infinite Impulse Response*. Esse modelo utiliza tanto as entradas, quanto as saídas anteriores para a

computação da saída atual. Dessa forma, a resposta ao impulso nunca será igual a zero. Tal representação do sistema se opõe ao FIR (*Finite Impulse Response*), que apenas considera as entradas atrasadas do sistema.

A etapa seguinte do procedimento diz respeito à estimação do modelo em si. Foram utilizadas duas técnicas distintas: a identificação linear e as redes neurais artificiais. A primeira delas é baseada na estimação do modelo por meio do método dos mínimos quadrados. Considera-se que o sistema é linear, que é uma suposição falsa que pode trazer problemas de estimação. As redes neurais, por outro lado, têm a capacidade de representar sistemas não-lineares com precisão arbitrária. Uma descrição indispensável das redes neurais artificiais está disponível em [8].

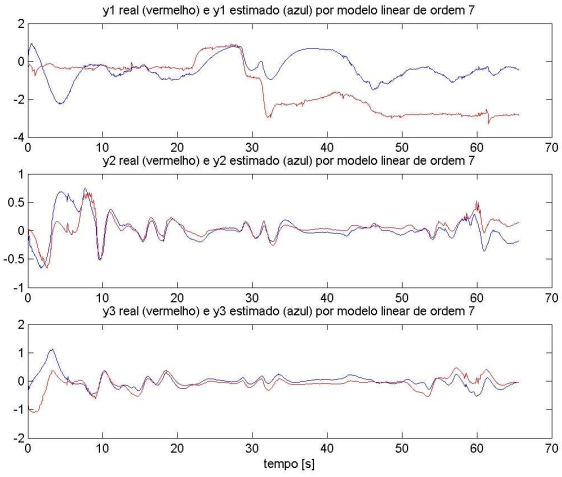
A última etapa do processo diz respeito à validação dos dados. No presente caso foi utilizado o procedimento conhecido como validação simples, em que parte dos dados experimentais não é utilizada na estimação dos parâmetros, mas é guardada para a avaliação do modelo obtido.

Uma outra questão importante a ser esclarecida acerca da validação dos resultados diz respeito à utilização ou não dos dados medidos anteriormente na computação dos valores computados pelo modelo. Essa questão procede porque muitas vezes não é possível encontrar modelos que descrevem a dinâmica do sistema de forma com a perfeição desejada e assim utiliza-se, ao invés da estimação total das saídas do sistema, a predição *k*-instantes à frente. Em sistemas em que a taxa de amostragem é alta com relação à dinâmica, entretanto, a predição *k*-instantes à frente pode revelar uma falsa impressão de qualidade do modelo obtido, pois um valor de saída geralmente não sofre, nesses casos, variação muito grande entre uma amostra e outra.

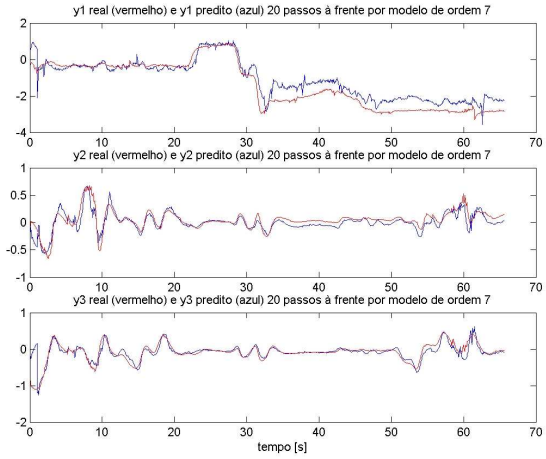
Para a realização do procedimentos de identificação do sistema, foi utilizado o Matlab®, por meio da *toolbox* NNSYSID, na sua versão 2.0, disponível on-line [16]. Essa toolbox oferece, além das funções de identificação de sistemas utilizando redes neurais (que formam seu nome), ferramentas básicas de identificação e também de utilização do perceptron multicamadas em contextos mais genéricos.

5.2.1. Identificação linear

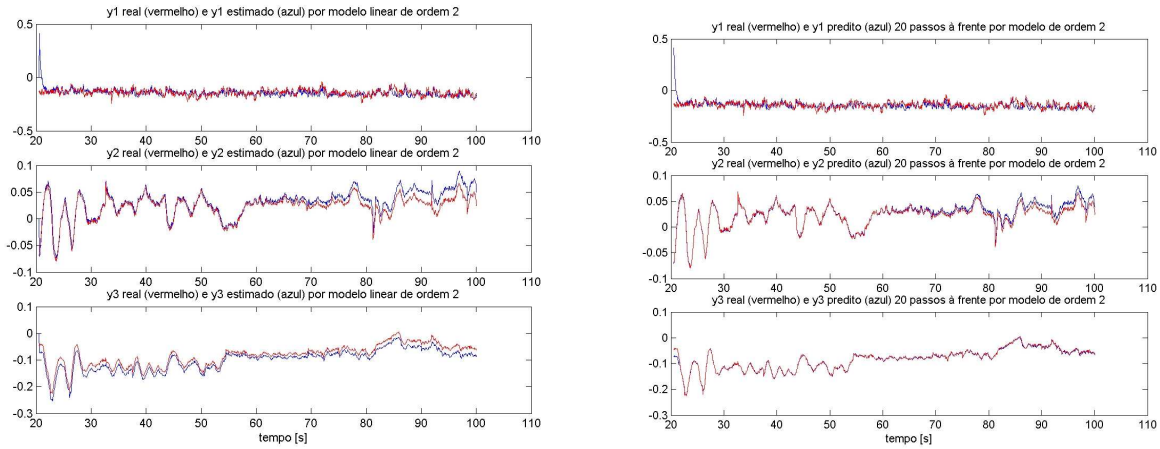
A definição da ordem do sistema, ou o número de elementos de atraso a ser considerado, foi o principal parâmetro a ser definido para a obtenção do modelo linear. A obtenção dos coeficientes de $\varphi(t)$ é feita a partir do método dos mínimos quadrados. Pode-se ver na Figura 4.4 alguns resultados obtidos para diferente conjuntos de treinamento. Para esses dados, foi definido que todas as entradas e saídas do sistema têm influência no movimento de todos eixos de orientação. Percebe-se, nesses dados, que os resultados obtidos com o horizonte de predição 20 passos à frente foi muito melhor, com já era de se esperar. Isso ocorre principalmente no conjunto de dados obtidos ao redor de um ponto de operação (helicóptero estabilizado), em que os valores dos ângulos são bem menores e o sistema se comporta de forma mais linear.



(a)



(b)



(c) (d)

Figura 5.4 – Identificação linear com horizonte de predição infinito em (a) com modelo linear de 7ª ordem e em (c) com modelo linear de 2ª ordem. Identificação linear com horizonte de predição 20 passos adiante em (b) com modelo linear de 7ª ordem e em (d) com modelo linear de 2ª ordem.

5.2.2. Identificação utilizando redes neurais

No caso da rede neural utilizada para fazer a identificação, foram utilizadas redes MLP (perceptron multicamadas) com diferentes configurações de neurônios na camada oculta, função de ativação e elementos de atraso considerados. O treinamento foi baseado no já conhecido algoritmo de Levenberg-Marquardt e na camada de saída a função de ativação é linear.

Havia a possibilidade de empregar uma função que trata de sistemas com múltiplas saídas diretamente, porém não se obteve sucesso. Dessa forma, tivemos de servir-nos de uma função que trata de sistemas com múltiplas entradas, porém uma única saída. Essa limitação foi contornada simplesmente ao alocar como entrada dessa rede também as saídas dos outros eixos de orientação.

Na maior parte das vezes foi utilizada uma camada oculta composta apenas de neurônio “tansig”. Em outra configuração, que inclui um neurônio linear na camada intermediária, entretanto, também foram alcançados resultados satisfatórios. Nessa configuração, é grande o número de resultados que divergem bastante do esperado, porém,

ao se controlar o número de épocas de treinamento, pode-se conseguir resultados satisfatórios.

Verificou-se que esse efeito de supertreinamento (*overfitting*), ocorre também com facilidade nas redes com apenas neurônios “tansig” na camada oculta. Muitas vezes, desperdiçou-se um bom resultado na tentativa de melhorá-lo com “apenas” mais 10 épocas de treinamento. Nesses casos, a saída da rede para os dados de validação divergia acentuadamente.

A figura 5.5 relaciona alguns resultados obtidos para a identificação neural para o eixo de arfagem.

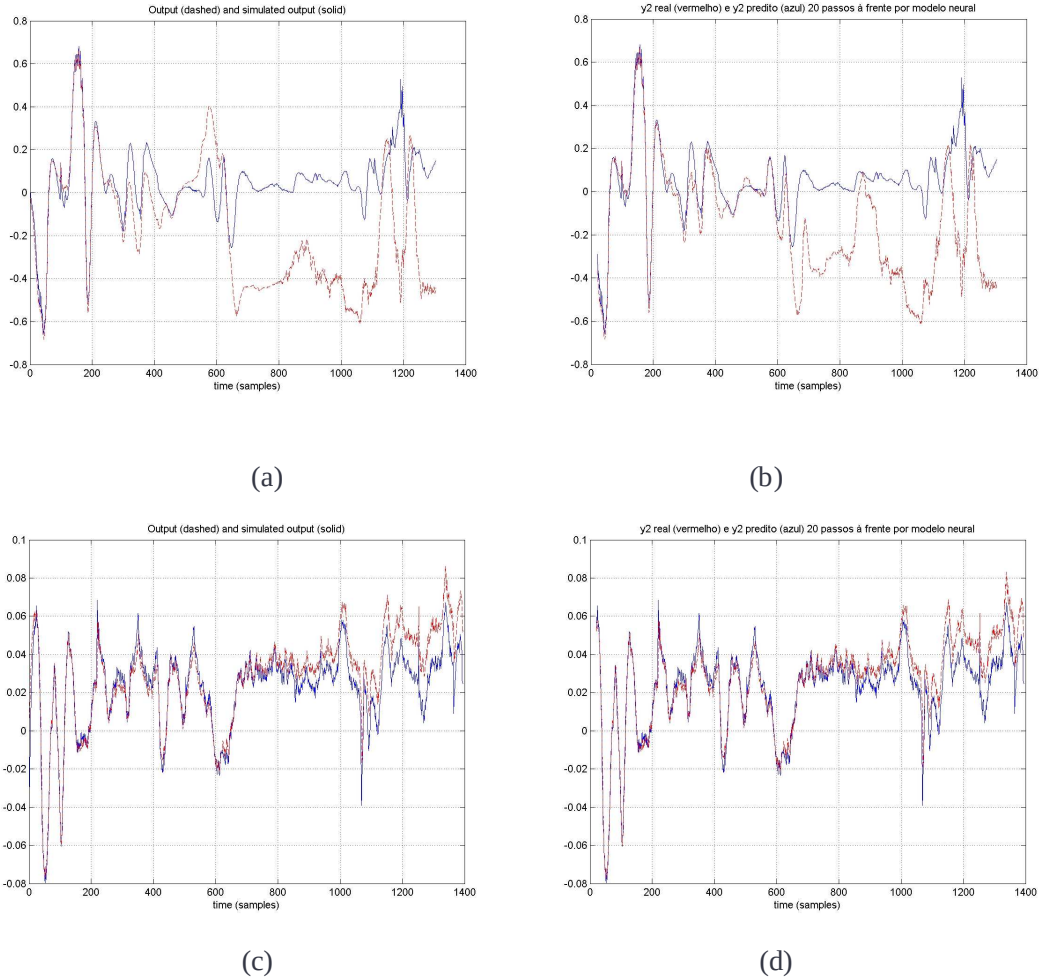


Figura 5.5 - Identificação neural da arfagem com horizonte de predição infinito (a) para o conjunto de dados mais amplo e (c) para o conjunto de dados ao redor do ponto de operação. Identificação

neural da arfagem com horizonte de predição 20 passos à frente (a) para o conjunto de dados mais amplo e (c) para o conjunto de dados ao redor do ponto de operação.

Pode-se concluir, a partir dos dados apresentados, que a identificação neural ainda não alcançou patamares satisfatórios. Alguns dos resultados obtidos em ocasiões anteriores apresentaram melhor desempenho, porém ainda se deve realizar experimentos com diferentes configurações de redes neurais, como as redes de base radial (RBF), ou ainda disponibilizando algum tipo de conhecimento *a priori* à rede. Tais ensaios poderiam enfim determinar a possibilidade de se identificar esse sistema utilizando redes neurais.

5.2.3. Considerações finais

Ao analisar os resultados obtidos com a identificação linear e aquela utilizando redes neurais, pode-se inferir que a identificação linear apresentou resultados desde já satisfatórios para a utilização no controle de atitude do helimodelo.

As estimações com horizonte de predição infinito não apresentaram, entretanto, um desempenho muito bom. Isso afasta, por enquanto, a possibilidade de utilizar métodos de controle direto inverso. Nesse caso, o controlador é estimado a partir de um conjunto de dados de treinamento inverso de entrada-saída.

Devido aos bons resultados obtidos nas estimações com horizonte de predição finito (20 passos, por exemplo), a utilização de técnicas de controle preditivo torna-se bastante recomendável. De fato, essa será a próxima estratégia de controle de atitude avaliada para o sistema construído.

6. Conclusões

Neste manuscrito, foi apresentado o projeto de uma plataforma experimental para estudos da dinâmica de um modelo de helicóptero em escala reduzida e avaliação de controladores digitais de atitude de helicóptero. O projeto foi dividido em três etapas: projeto mecânico, projeto elétrico, projeto computacional.

O projeto mecânico consistiu na fabricação do conjunto suporte-junta-escovas que permite a movimentação do helimodelo em três graus de liberdade. As maiores dificuldades encontradas foram a confecção do sistema de escovas com reduzido atrito e o ajuste do eixo da junta universal.

A segunda etapa foi o projeto eletrônico, que consistiu na confecção dos circuitos eletrônicos de interface entre helimodelo e PC e nos circuitos de acionamento dos atuadores. O grande desafio desta etapa foi realizar a transmissão dos sinais entre helimodelo e base via sistema de escovas. O problema foi resolvido com o projeto dos circuitos conversores e dos filtros digitais e analógicos. Depois das tentativas frustradas de implementação e vencidos os problemas na montagem dos circuitos eletrônicos, a configuração definitiva atende às exigências iniciais e o sistema funciona plenamente. A aquisição dos dados e a comunicação serial entre microcontrolador e PC ocorrem sem falhas e o acionamento dos atuadores funciona sem problemas.

A etapa final, o projeto computacional, foi a implementação do software de monitoração e controle no PC e o software de interface e acionamento no microcontrolador.

O software recebe do microcontrolador as variáveis a serem controladas, realiza o processamento e reenvia ao microcontrolador as variáveis de controle. O papel do microcontrolador é fazer a conversão A/D dos sinais vindos do helimodelo, enviar as variáveis ao PC, receber os dados processados pelo PC e gerar os sinais PWM de acionamento dos motores e servos. O projeto que foi implementado contorna de maneira satisfatória a dificuldade de se obter períodos constantes de amostragem com o Windows 2000®.

No software de monitoração e controle no PC estão implementados quatro controladores digitais PID's independentes, um para cada variável de controle. Os parâmetros dos PID's podem ser ajustados pelo usuário. Além disso, os dados experimentais podem ser visualizados em tempo de execução e tratados offline no Matlab®.

Finalizadas estas três etapas, concluiu-se a concepção da plataforma experimental para estudo de controle de modelo reduzido de helicóptero. Os objetivos iniciais deste projeto foram, portanto, alcançados.

Além dos objetivos iniciais, deu-se início ao estudo das estratégias de controle de atitude do helimodelo. Foram realizados o levantamento do modelo físico e a identificação do sistema utilizando técnicas de identificação linear e redes neurais. Finalmente, foi testado um controlador PID sintonizado heuristicamente para avaliação do desempenho e comparação com controle manual.

O projeto permitiu a atuação em importantes áreas do universo da automação e controle. Abordou aspectos mecânicos como o projeto de mecanismos e modelamento do sistema físico; aspectos elétricos e eletrônicos como o projeto de circuitos de interface, microcontroladores, instrumentação, eletrônica de potência e conversão A/D; e aspectos computacionais como processamento em tempo real, implementação de controlador digital pelo PC, comunicação serial, e programação de software para microcontrolador.

Como sugestões a trabalhos futuros, a mais relevante modificação que poderia ser feita na plataforma experimental seria a medição da velocidade de rotação dos rotores para

controle de velocidade. Assim, o controle da atitude e posteriores implementações do controle de trajetória e navegação se tornariam muito mais simples. Além disso, a partir dos resultados obtidos na identificação do sistema, sugere-se a implementação de leis de controle baseadas nas técnicas de controle preditivo.

Bibliografia

[1] Ávila-Vilchis J.C. Modélisation et commande d'helicoptère. *Tese de doutorado, INPG*, França, 2001;

[2] Ávila-Vilchis J.C., Brogliato B., Dzul A., Lozano, R. Nonlinear modelling and control of helicopters. *Automatica nº 39*, 2003, p. 1583-1596;

[3] Balderud J., Wilson D.I. A comparison of optimal control strategies for a toy-helicopter. Em *Proceedings of the 4th Asian Control Conference*, Singapura, 2002;

[4] Balderud J. Modelling and control of a toy-helicopter. *Tese de mestrado, Karlstad Universitet*, Suécia, 2001;

[5] Chen M., Huzmezan M. A combined MBPC/H_∞ controller for a quad rotor UAV. Em *Proceedings of the AIAA Atmospheric Flight Mechanics Conference and Exhibit*, 2003;

[6] Franklin G.F., Powell J.D.,Workman M. Digital Control of Dynamic Systems. *Addison-Wesley*, 3ª edição,1998;

[7] Halliday D., Resnick R., Walker J. Fundamentos de física. *LTC Editora*, 4ª edição, Rio de Janeiro, Brasil, 1996;

[8] Haykin, S. Redes Neurais: Princípios e prática, *Ed. Bookman*, 2ª edição, Porto Alegre, 2001.

[9] Garrido S., Moreno L., Salichs M.A. Predictive control with restricted genetic optimization. Em *Proceedings of the European Symposium on Intelligent Techniques*, 2000;

[10] Johansson K.H. 2E1262 Nonlinear control: lecture notes. *Department of Signals, Sensors and Systems (S3), Kungl Tekniska Högskolan (KTH)*, Estocolmo, Suécia, 2002.

[11] Johnson E.N., DeBitetto P.A. Modelling and simulation for a small autonomous helicopter development. Em *Proceedings of the AIAA Modelling & Simulation Technologies Conference*, 2002;

[12] Mettler B., Kanade T., Tischler M.B. System identification modeling for a model-scale helicopter. *Technical Report CMU-R1-TR-00-03,Robotics Institute, Carnegie Mellon University*, Pittsburgh, 2002.

[13] Morris J., van Nieustadt M., Bendotti P. Identification and control of a model helicopter in hover. Em *Proceedings of the American Control Conference*, 1994.

[14] van Nieuwdstadt M., Morris J. Control of rotor speed for a model helicopter: a design cycle. Em *Proceedings of the American Control Conference*, 1995;

[15] Nørgaard, M.,Ravn, O, Poulsen, N. K., Hansen, L. K. Neural Networks for Modelling and Control of Dynamic Systems, *Springer*, Inglaterra, 2000;

[16] Nørgaard, M. Neural Network Based System Identification - Tutorial, *Technical Report 00-E-891, Department of Automation.Technical University of Denmark* , Dinamarca, 2000;

[17] Prouty R.W. Helicopter stability, performance and control. *Krieger Publications*, 1995;

[18] Shim H., Koo T.J., Hoffmann F., Sastry S. A comprehensive study of control design for na autonomous helicopter. Em *Proceedings of IEEE Conference on Decision and Control*, Florida, EUA, 1998.

[19] Shim H., Kim H.J. A flight control system for aerial robots: algorithms and experiments. Em *IFAC Control Engineering Practice*, 2003;

[20] Shim D. H., Kim H. J., Sastry S. System identification and control synthesis for rotorcraft-based unmanned aerial vehicles. Em *IEEE International Conference on Control Applications, Anchorage*, Alaska, EUA, 2000;

[21] Sugeno M. Development of an intelligent unmaned helicopter. Em *Fuzzy Modeling and Control - Selected Works of M. Sugeno, org. por Nguyen H.T., Prasad N.R., CRC Press*, 1999;

[22] Typical applications of the LM317, em *National Power ICs Databook*, 1995, p. 1-26;

[23] Vieira B., Serapião A.C. Aerodinâmica de helicópteros. *Editora Rio*, Rio de Janeiro, Brasil, 2003;

[24] User's manual: AVR ATmega8. *Atmel Corporation*®, 2003.

[25] Walker D.J. Multivariable control of the longitudinal and lateral dynamics of a fly-by-wire helicopter. Em *Control Engineering Practice nº 11*, 2003, p. 791-795;

[26] Zhu X., van Nieuwdstadt M. The Caltech helicopter experiment. *CDS Technical Report, 96-009, Caltech*, EUA, 1996.

ANEXOS

A. Microcontrolador AVR ATmega8

Um microcontrolador pode ser definido como um microprocessador encapsulado com uma série de dispositivos que auxiliam sua interface com outros sistemas. Dessa forma, os microcontroladores são ideais para a utilização de projetos em controle e automação.

Os microcontroladores AVR da Atmel© são dispositivos CMOS de 8 bits que possuem um núcleo RISC. Ele executa instruções em um ciclo de clock e possui, ao mesmo tempo, baixo consumo de potência. A sua estrutura de I/O, bastante completa, também limita a necessidade do uso de elementos externos.

O núcleo dos microcontroladores AVR combina um rico conjunto de instruções RISC com 32 registradores de uso geral conectados diretamente a ULA (Unidade Lógico Aritmética), o que possibilita que 2 registradores independentes sejam acessados num mesmo ciclo de clock.

Outra característica importante dos microcontroladores AVR é que ele foi projetado para ser programado em linguagem C. De fato, a criação da linguagem Assembly do AVR se deu concorrentemente ao projeto dos compiladores e montadores para C. Isso possibilita a geração de códigos muito mais eficientes do que aqueles geralmente gerados em outros controladores. A velocidade de processamento em MIPS chega a ser 10 vezes maior do que aquelas alcançadas em microcontroladores CISC, chega a alcançar até 16 MIPS quando funcionando em 16 MHz de clock.

Algumas características do microcontrolador AVR ATmega8 utilizado no projeto:

- Arquitetura RISC avançada:
  - 130 instruções, a maior parte de execução em um único ciclo
  - 32x8 registradores de uso geral
  - Operação totalmente estática
  - Até 16 MIPS de velocidade em 16 MHz de clock
  - Multiplicador de 2 ciclos encapsulado
- Memórias de dado e não-voláteis de programa:
  - 8 KBytes de memória Flash auto-programável, com ao menos 10.000 ciclos de gravação
  - Código de boot opcional com bits trancados independentes
  - 512 KBytes de EEPROM, com ao menos 100.000 ciclos de gravação
  - 1 KByte de SRAM interna
  - Trancamento programável para segurança de software
- Características periféricas:
  - 2 timers/contadores de 8 bits com escalonador independente
  - 1 timer/contador de 16 bits com escalonador independente, modo de comparação e modo de captura
  - Contador em tempo real com oscilador independente
  - 6 canais de conversor A/D, sendo 4 com resolução de 10 bits e 2 de 8 bits
  - Interface serial de 2 fios orientada a transmissão/recepção de byte
  - USART serial programável
  - Interface serial SPI mestre/escravo
  - Watchdog timer programável com oscilador encapsulado independente
  - Comparador analógico encapsulado
- Características especiais do microcontrolador
  - Reset power-on e detecção brown-out programável
  - Oscilador RC interno e calibrado
  - Fontes de interrupção externas e internas
  - 5 diferentes modos de sleep
- 28 pinos de I/O programáveis
- Tensão de operação entre 4,5 e 5,5 V

- Freqüência de operação entre 0 e 16 MHz
- Consumo de potência em 4 MHz, 3 V e 25 C (ATmega8L)
  - Modo ativo: 3,6 mA
  - Modo lento: 1,0 mA
  - Modo power-down: 0,5 μA

A programação do AVR é realizada através da porta de comunicação serial ou paralela do PC. Em ambos casos, o programador não é um fator limitante em termos de complexidade. Na Figura A.1 pode-se observar o diagrama de blocos do AVR ATmega8.

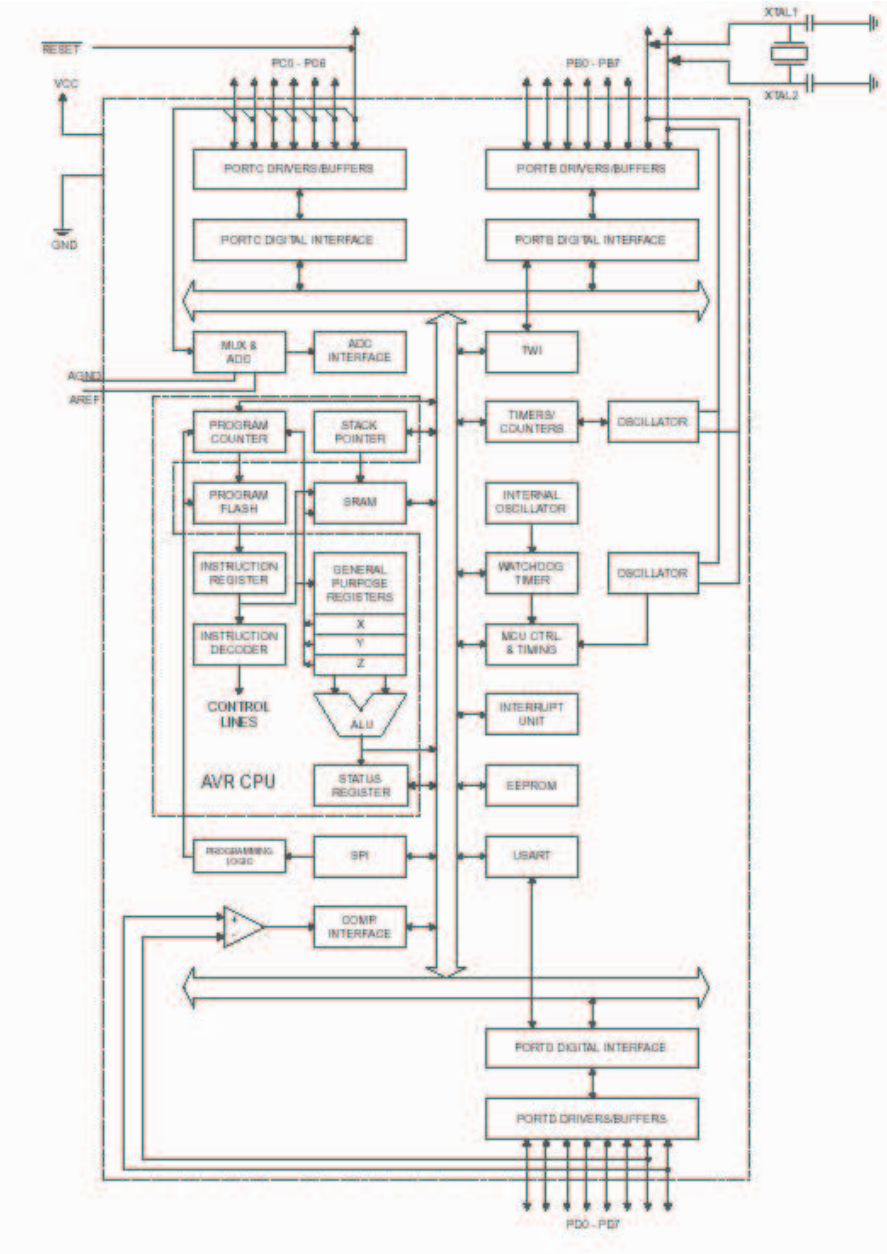


Figura A.1– Diagrama de blocos do AVR ATmega8.

B. Códigos

B.1. heli.c (microcontrolador)

```
/*********************************************************************
// File: Heli.c
// Contents: C file with functions for reduced model helicopter control with
//microcontroller.
// Author: A. Padilha, H. Fonseca, G. A. Borges.
*********************************************************************/
#include <inttypes.h>
#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/signal.h>

void USART_Iniciar (void);
void AD_Iniciar (void);
void Servo_Iniciar (void);
void PWM_Iniciar (void);
void USART_Transmitir (unsigned char c);
unsigned char USART_Transmitir_E_Aguardar_Eco (unsigned char c);
unsigned char USART_Receber (void);
unsigned char USART_Receber_E_Enviar_Eco (void);
void AD_Converter(unsigned char canal, unsigned char *pdatah, unsigned char
*pdatal);
SIGNAL (SIG_OUTPUT_COMPARE2);
unsigned char u1=0, u2=0;
unsigned int cont_timer2=0;

void USART_Iniciar (void) {
      //Setar baud rate 38400
      UBRRH = (unsigned char)(25>>8);
      UBRRL = (unsigned char)(25);
      //Enable TXD, RXD e ativar RXCIE(enable interrupt on the RCX flag)
      UCSRB = _BV(TXEN) | _BV(RXEN); //| _BV(RXCIE);
      //Setar tamanho palavra (formatação da comunicação serial)
      UCSRC = _BV(URSEL) | _BV(UCSZ1) | _BV(UCSZ0) | _BV(USBS);
      //Ativar double speed
      UCSRA = _BV(U2X);
}

void USART_Transmitir (unsigned char c) {
      //Verificar se o buffer do serial está vazio
```

```
	while ( !( UCSRA & (1<<UDRE)) );
	//coloca o dado no buffer e envia
	UDR = c;
}

unsigned char USART_Transmitir_E_Aguardar_Eco (unsigned char c) {
	USART_Transmitir (c);
	if(USART_Receber()!=c) return 0;
	return(1);
}

unsigned char USART_Receber (void) {
	//Esperar até que seja feita comunicação em RXD
	while ( !(UCSRA & (1<<RXC)) );
	//Recebe o dado do buffer
	return UDR;
}

unsigned char USART_Receber_E_Enviar_Eco (void) {
	unsigned char dado;

	dado=USART_Receber ();
	USART_Transmitir(dado);
	return dado;
}

void AD_Iniciar (void) {
	//Enable ADC e prescaler com division factor 64(ADPS2 e ADPS1)
	ADCSRA = _BV(ADEN) | _BV(ADPS2) | _BV(ADPS1);
	//Setar referência
	ADMUX = _BV(REFS0);
}

void AD_Converter(unsigned char canal, unsigned char *pdatah, unsigned char
*pdatal){
	ADMUX  = _BV(REFS0) | (canal & 0x0F);
	ADCSRA |= (0x01<<ADSC);
	// Esperar fim de conversão
	while((ADCSRA & (0x40)));
	// Copiar resultado
	*pdatal = ADCL;
	*pdatah = ADCH;
}

void Servo_Iniciar (void){
	//set prescaler = 8(CS21)
	TCCR2 = _BV(CS21) | _BV(WGM21);
	//Enable output compare match interrupt
	TIMSK = _BV(OCIE2);
	//set PD2 e PD3 as output
	DDRD = _BV(DDD2) | _BV(DDD3);
}

SIGNAL (SIG_OUTPUT_COMPARE2){
	if(++cont_timer2>2000){
		cont_timer2 = 0;
		PORTD |= 0x0D;
	}
	//u1 (PD2)
	if(cont_timer2 == (u1+100) )
		PORTD &= ~(0x04);
	//u2 (PD3)
	if(cont_timer2 == (u2+100) )
```

```
            PORTD &= ~(0x08);
}

void PWM_Iniciar (void){
      //configurar PWM
      TCCR1A = _BV(COM1A1) | _BV(COM1A0) | _BV(COM1B1) | _BV(COM1B0) | _BV(WGM10)
| _BV(WGM11);
      //set prescaler = 64(CS11 e CS10)
      TCCR1B = _BV(CS11) | _BV(CS10) | _BV(WGM12);
      //set PB1 e PB2 as output
      DDRB = _BV(DDB1) | _BV(DDB2);
}

void main (void) {
      unsigned char codigo, canal,convh, convl;

      sei(); //Enable interrupções
      USART_Iniciar();
      AD_Iniciar();
      Servo_Iniciar();
      PWM_Iniciar();

      for(;;){
            codigo = USART_Receber_E_Enviar_Eco ();
            switch(codigo){
            case 'u':
                  u1 = USART_Receber_E_Enviar_Eco ();
                  u2 = USART_Receber_E_Enviar_Eco ();
                  //pwm1 (PB1)
                  OCR1AH = USART_Receber_E_Enviar_Eco ();
                  OCR1AL = USART_Receber_E_Enviar_Eco ();
                  //pwm2 (PB2)
                  OCR1BH = USART_Receber_E_Enviar_Eco ();
                  OCR1BL = USART_Receber_E_Enviar_Eco ();
                  break;
            case 'y':
                  // Fazer conversões e transmitir
                  for(canal=0;canal<3;++canal){
                        AD_Converter(canal,&convh,&convl);
                        USART_Transmitir(canal);
                        USART_Transmitir(convh);
                        USART_Transmitir(convl);
                  }
                  break;
            }
                                                                        }
}
```

B.2 helicoptero.c (PC)

```
/*************************************************************************
// File: Helicoptero.cpp
// Contents: CPP file with functions for reduced model helicopter control.
// Author: A. Padilha, H. Fonseca, G. A. Borges.
*************************************************************************/

#include "stdafx.h"
#include <windows.h>
#include <mmsystem.h>
#include <math.h>
```

```
#include <conio.h>

#include "GThreadTimer.h"
#include "GQueue.h"
#include "GMatlabDataFile.h"
#include "GDataLogger.h"

#include "Helicoptero.h"

#define MAX(a,b) ((a)>(b))?(a):(b)
#define MIN(a,b) ((a)<(b))?(a):(b)

#define COM1 0x3F8
#define COM2 0x2F8
#define COM3 0x3E8
#define COM4 0x2E8
#define PORTASERIAL COM1

#define BPS_38400         0x03
#define BPS_115200        0x01
#define BPS_57600         0x02
#define BPS_19200         0x06
#define BPS_9600          0x0C
#define BPS_4800          0x18
#define BPS_2400          0x30
#define DATARATE          BPS_38400

#define SAMPLINGTIMEMS  40

#define Y1MAX 1800.0
#define Y1MIN-1800.0
#define Y2MAX 90.0
#define Y2MIN-90.0
#define Y3MAX 90.0
#define Y3MIN-90.0

#define U1MAX 0.9
#define U1MIN-0.9
#define U2MAX 0.9
#define U2MIN-0.9
#define U3MAX 1.0
#define U3MIN 0.0
#define U4MAX 1.0
#define U4MIN 0.0

typedef unsigned char BYTE;

typedef struct{
       JOYCAPS joycaps;
       BOOL Button1;
       BOOL Button2;
       BOOL Button3;
       BOOL Button4;
       double Xpos;
       double Ypos;
} JOYSTICKSTATUS,*PJOYSTICKSTATUS;

// Protótipos locais:
BOOL helicoptero_Serial_Iniciar(void);
BOOL helicoptero_Serial_EnviarByte(BYTE *pData);
BOOL helicoptero_Serial_ReceberByte(BYTE *pData, int MaximaEsperaUS);
BOOL helicoptero_Serial_LerSensores(double *py1, double *py2, double *py3);
```

```
BOOL helicoptero_Serial_EnviarAtuadores(double u1, double u2, double u3, double
u4);
BOOL helicoptero_Joystick_Iniciar(void);
BOOL helicoptero_Joystick_LerEstado(BOOL AplicarFiltro);
BOOL helicoptero_Controle_PID(double y1, double y2, double y3, double y1ref,
double y2ref, double y3ref, double *pu1, double *pu2, double *pu3, double *pu4);

void atraso_us(int tempo_us);

GThreadTimer*pThreadTimer = NULL;
GDataLogger* pDataLogger = NULL;
JOYSTICKSTATUS     JoystickStatus;

BOOL helicoptero_Iniciar(GDataLogger* ppDataLogger, GThreadTimer* ppThreadTimer,
unsigned int *pTs)
{
	// set globals:
	pDataLogger = ppDataLogger;
	pThreadTimer = ppThreadTimer;

	// init DataLogger
	pDataLogger->VariableCreate("y1ref","deg",500);
	pDataLogger->VariableCreate("y2ref","deg",500);
	pDataLogger->VariableCreate("y3ref","deg",500);
	pDataLogger->VariableCreate("y1","deg",500);
	pDataLogger->VariableCreate("y2","deg",500);
	pDataLogger->VariableCreate("y3","deg",500);
	pDataLogger->VariableCreate("u1","deg",500);
	pDataLogger->VariableCreate("u2","deg",500);
	pDataLogger->VariableCreate("u3","V",500);
	pDataLogger->VariableCreate("u4","V",500);
	pDataLogger->VariableCreate("t","s",500);
	pDataLogger->VariableCreate("texec","s",500);

	pDataLogger-
>MatfileCreate("..\\HelicopteroMatlab\\HelicopteroMatlabDataFile.mat",NULL);

	// init control sampling time:
	*pTs = SAMPLINGTIMEMS; // Sampling period in ms

	// comunicação serial: Definir DATARATE e PORTASERIAL acima.
	if(!helicoptero_Serial_Iniciar()){
		return FALSE;
	}
//	helicoptero_Serial_EnviarAtuadores(0.0,0.0,0.0,0.0);

	// joystick:
	if(!helicoptero_Joystick_Iniciar()){
		return FALSE;
	}

	return TRUE;
}

void helicoptero_Encerrar(void)
{
	helicoptero_Serial_EnviarAtuadores(0.0,0.0,0.0,0.0);
}

int helicoptero_Controle(int Mode)
{
	double y1,y2,y3;
	static double y1ref=0.0,y2ref=0.0,y3ref=0.0;
```

```
	static double u1=0.0,u2=0.0,u3=0.0,u4=0.0;
	double t,texec;
	double pi = 3.14159265358979;
	double Ts = 1e-3*((double)(SAMPLINGTIMEMS)); // Periodo de amostragem

	// Tempo:
	t = pThreadTimer->GetCurrentTime();

	// Coletar medições dos ângulos
	if(!helicoptero_Serial_LerSensores(&y1, &y2, &y3)){
		return(HELICOPTERO_ERROR_SERIAL_RX);
	}

//	goto helicoptero_Controle_Exit;

	// Coletar dados do joystick e chama os procedimentos apropriados de acordo
com o modo.
	if(!helicoptero_Joystick_LerEstado(TRUE)){
		return HELICOPTERO_ERROR_JOYSTICK;
	}

	switch(Mode){
	case HELICOPTERO_MODE_AUTOMATIC:
		// Modo Automático
		// Determinar as referências
		if(JoystickStatus.Button4){
			// Incrementos de guinada(u1), a uma razão de 0.05 unidades
por segundo:
			y1ref += Ts * 70.0  ;
		}
		if(JoystickStatus.Button3){
			// Decrementos de guinada(u1), a uma razão de 0.05 unidades
por segundo:
			y1ref -= Ts * 70.0;
		}
		if(JoystickStatus.Button1){
			// Incrementos de u4, a uma razão de 0.3 unidades por segundo:
			u4 += Ts * 0.4;
		}
		if(JoystickStatus.Button2){
			// Decrementos de u1, a uma razão de 0.3 unidades por segundo:
			u4 -= Ts * 0.4;
		}
		y2ref = -45.0 * JoystickStatus.Xpos; // -1 <= JoystickStatus.Xpos <=
1
		//if(y1ref<0)
		//	y1ref = 50 * y1ref;
		//else
		//	y1ref = 40 * y1ref;

		y3ref = -45.0 * JoystickStatus.Ypos; // -1 <= JoystickStatus.Ypos <=
1

		// Aplicar saturações nas variáveis de referência:
		y1ref = MIN(y1ref,Y1MAX); y1ref = MAX(y1ref,Y1MIN);
		y2ref = MIN(y2ref,Y2MAX); y2ref = MAX(y2ref,Y2MIN);
		y3ref = MIN(y3ref,Y3MAX); y3ref = MAX(y3ref,Y3MIN);
		// Aplicar o controlador:
		if(!helicoptero_Controle_PID(y1, y2, y3, y1ref, y2ref, y3ref, &u1,
&u2, &u3, &u4)){
			return(FALSE);
		}
		if(u3 >  0.3) u3 =  0.3;
```

```
                if(u3 < -0.3) u3 = -0.3;
                u3 = 0.3 + u3;
                if(u4 >  0.15) u4 =  0.15;
                if(u4 < -0.15) u4 = -0.15;
                u4 = 0.85 + u4;
                break;
        case HELICOPTERO_MODE_MANUAL:
                // Modo Manual
                // u1 =
                // u2 =
                // u3 =
                // u4 = rotor principal
                // Determinar as referências
                if(JoystickStatus.Button4){
                        // Incrementos de u1, a uma razão de 0.05 unidades por
segundo:
                        u3 += Ts * 0.5  ;
                }
                if(JoystickStatus.Button3){
                        // Decrementos de u1, a uma razão de 0.05 unidades por
segundo:
                        u3 -= Ts * 0.5;
                }
                if(JoystickStatus.Button1){
                        // Incrementos de u4, a uma razão de 0.3 unidades por segundo:
                        u4 += Ts * 0.4;
                }
                if(JoystickStatus.Button2){
                        // Decrementos de u1, a uma razão de 0.3 unidades por segundo:
                        u4 -= Ts * 0.4;
                }
                u1 = 1.0 * JoystickStatus.Xpos; // -1 <= JoystickStatus.Xpos <= 1
                u2 = 1.0 * JoystickStatus.Ypos; // -1 <= JoystickStatus.Ypos <= 1

                break;
        default:
                // Nenhum modo válido:
                u1=0.0;
                u2=0.0;
                u3=0.0;
                u4=0.0;
        }

        // Aplicar saturações nas variáveis de controle:
        u1 = MIN(u1,U1MAX); u1 = MAX(u1,U1MIN); // angulos dos servos
        u2 = MIN(u2,U2MAX); u2 = MAX(u2,U2MIN); // angulos dos servos
        u3 = MIN(u3,U3MAX); u3 = MAX(u3,U3MIN); // tensoes positivas entre 0 e 10V
        u4 = MIN(u4,U4MAX); u4 = MAX(u4,U4MIN); // tensoes positivas entre 0 e 10V

        // Enviar variaveis de controle:
//      u1=0.0;u2=0.0;u3=0.3;u4=0.3;
        if(!helicoptero_Serial_EnviarAtuadores(u1, u2, u3, u4)){
                return(HELICOPTERO_ERROR_SERIAL_TX);
        }

//helicoptero_Controle_Exit:

        // Armazenar variaveis na queue circular:
        pDataLogger->VariableInsert("y1ref",&y1ref,1);
        pDataLogger->VariableInsert("y2ref",&y2ref,1);
        pDataLogger->VariableInsert("y3ref",&y3ref,1);
        pDataLogger->VariableInsert("y1",&y1,1);
        pDataLogger->VariableInsert("y2",&y2,1);
```

```
	pDataLogger->VariableInsert("y3",&y3,1);
	pDataLogger->VariableInsert("u1",&u1,1);
	pDataLogger->VariableInsert("u2",&u2,1);
	pDataLogger->VariableInsert("u3",&u3,1);
	pDataLogger->VariableInsert("u4",&u4,1);
	pDataLogger->VariableInsert("t",&t,1);


	// Tempo de execução
	texec = pThreadTimer->GetCurrentTime() - t;
	pDataLogger->VariableInsert("texec",&texec,1);

	return(HELICOPTERO_NOERROR);
}

BOOL helicoptero_Controle_PID(double y1, double y2, double y3, double y1ref,
double y2ref, double y3ref, double *pu1, double *pu2, double *pu3, double *pu4)
{
	// u1 = y2
	// u2 = y3
	// u3 = y1

	// Configuração: 3 PIDs (2 servos e 1 rotor de calda). Rotor principal com
tensão constante no máximo.
	// Variável de erro atual e passado:
	// erro[i][j] = erro yjref-yj verificado no instante k-i, com k sendo o
instante discreto atual.
	static double erro[3][4] = {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0};
	// Variável de controle atual e passado:
	static double u[2][4] = {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0};
	// Variável que indica saturacao do atuador:
	static BOOL FlagSaturacao[4] = {FALSE,FALSE,FALSE,FALSE};
	// Ganhos:
	double Kp[4] = {-0.03, -0.05, 0.1, -0.02};
	double Ki[4] = {-0.001, -0.001, 0.009, -0.001};
	double Kd[4] = {0.00, 0.0, 0.08, 0.0};
/*
	double Kp[4] = {-0.03, -0.05, 0.1, -0.02};
	double Ki[4] = {0, 0, 0.005, -0.001};
	double Kd[4] = {0.0, 0.0, 0.008, 0.0};
*/
	double Ts = 1e-3*((double)(SAMPLINGTIMEMS)); // Periodo de amostragem

	int j;

	// Atualiza os erros e variaveis de controle do passado:
	for(j=0;j<4;++j){
		erro[2][j] = erro[1][j];
		erro[1][j] = erro[0][j];
	}
	u[1][0] = u[0][0];
	u[1][1] = u[0][1];
	u[1][2] = u[0][2];
	u[1][3] = u[0][3];

	// Verifica se há saturacao de atuador
	if((u[0][0]>=U1MAX)||(u[0][0]<=U1MIN)){
		FlagSaturacao[0] = TRUE;
	}
	else{
		FlagSaturacao[0] = FALSE;
	}
```

```
	if((u[0][1]>=U2MAX)||(u[0][1]<=U2MIN)){
		FlagSaturacao[1] = TRUE;
	}
	else{
		FlagSaturacao[1] = FALSE;
	}
	if((u[0][2]>=U3MAX)||(u[0][2]<=U3MIN)){
		FlagSaturacao[2] = TRUE;
	}
	else{
		FlagSaturacao[2] = FALSE;
	}
	if((u[0][3]>=U4MAX)||(u[0][3]<=U4MIN)){
		FlagSaturacao[3] = TRUE;
	}
	else{
		FlagSaturacao[3] = FALSE;
	}

	// Atualiza os erros atuais:
	erro[0][0] =  y2ref - y2;  // Refere-se a y1 trocado! !!!
	erro[0][1] =  y3ref - y3;  // Refere-se a y2
	erro[0][2] =  y1ref - y1;  // Refere-se a y3
	erro[0][3] =  y1ref - y1;  // Refere-se a y3

	// Determinar as variáveis de controle atuais: Franklin et al., pp 66,
Digital Control
	// u(k) = u(k-1) + (Kp*Ts+Ts*Ts*Ki+Kd)/Ts*erro(k)
	//               - (Kp+2*Kd/Ts)*erro(k-1)
	//               + (Kd/Ts)*erro(k-2)
	for(j=0;j<4;++j){
		u[0][j]  = u[1][j];
		if(FlagSaturacao[j]==TRUE){
			// Saturacao do atuador. USar anti-windup
			u[0][j] += (Kp[j]*Ts+Kd[j])/Ts*erro[0][j];
		}
		else{
			u[0][j] += (Kp[j]*Ts+Ts*Ts*Ki[j]+Kd[j])/Ts*erro[0][j];
		}
		u[0][j] -= (Kp[j]+2*Kd[j]/Ts)*erro[1][j];
		u[0][j] += (Kd[j]/Ts)*erro[2][j];
	}

	// Copiar para a saída:
	if(pu1!=NULL) *pu1 = u[0][0];
	if(pu2!=NULL) *pu2 = u[0][1];
	if(pu3!=NULL) *pu3 = u[0][2];
	if(pu4!=NULL) *pu4 = u[0][3];

	return(TRUE);
}

/*****************************************************************************
****
**** Funções de interface com o helicóptero
****
*****************************************************************************/

BOOL helicoptero_Serial_Iniciar(void)
{
	_outp(PORTASERIAL + 1 , 0);		/* Desativar interrupções */
	_outp(PORTASERIAL + 3 , 0x80);		/* DLAB ON */
```

```
	_outp(PORTASERIAL + 0 , DATARATE);	 /* Baud Rate - DL Byte */
	_outp(PORTASERIAL + 1 , 0x00);		/* Baud Rate - DH Byte */
	_outp(PORTASERIAL + 3 , 0x03);		/* 8 Bits, No Parity, 1 Stop Bit */
	_outp(PORTASERIAL + 2 , 0xC7);		/* FIFO: 14 bytes, limpa TX e RX
fifos*/
	_outp(PORTASERIAL + 4 , 0x0B);		/* Ativa DTR, RTS, e OUT2 */

	return TRUE;
}

BOOL helicoptero_Serial_EnviarByte(BYTE *pData)
{
	while(!(_inp(PORTASERIAL + 5) & 0x40)); // Espera fim da última transmissão

	_outp(PORTASERIAL, *pData);  // Envia.

	return TRUE;
}

BOOL helicoptero_Serial_ReceberByte(BYTE *pData, int MaximaEsperaUS)
{
	double tf;

	if(MaximaEsperaUS<0) MaximaEsperaUS = 0; // Espera minima de 0 us.

	tf = pThreadTimer->GetCurrentTime() + 1e-6*((double)(MaximaEsperaUS));

	while(1){
		if(_inp(PORTASERIAL + 5) & 1){
			*pData = _inp(PORTASERIAL);
			return(TRUE);
		}
		if(pThreadTimer->GetCurrentTime() >= tf){
			return(FALSE);  // Dado não chegou no tempo estipulado.
		}
	}
}

BOOL helicoptero_Serial_LerSensores(double *py1, double *py2, double *py3)
{
	int i,MessageSize = 9;
	double v1,v2,v3;

	BYTE DataMessage[8];
	BYTE echo;

	BOOL AplicarFiltro = TRUE;

	double pi = 3.14159265358979;
	double Ts = 1e-3*((double)(SAMPLINGTIMEMS)); // Periodo de amostragem
	double Fc = 1.0; // Freqüência de corte do filtro passa baixas de primeira
ordem em Hertz

	static double vf1,vf2,vf3;

	// Transmite 'y'
	DataMessage[0] = 'y';
	helicoptero_Serial_EnviarByte(&DataMessage[0]);
	atraso_us(1000);
	if(helicoptero_Serial_ReceberByte(&echo,1500)){
		if(echo != DataMessage[0]){
			return(FALSE);
		}
```

```
	}
	// Recebe dados
	MessageSize = 10;
	for(i=1;i<MessageSize;++i){
		atraso_us(1000);
		if(!helicoptero_Serial_ReceberByte(&DataMessage[i], 1500)){
			return(FALSE);
		}
//		helicoptero_Serial_EnviarByte(&DataMessage[i]);
	}

	v1 = ((DataMessage[2]*256.0+DataMessage[3]))/1023.0;
	if(v1<0.5){
		v1 = -0.5 + v1;}
	else{
		v1 = v1 - 0.5;}

	v2 = ((DataMessage[5]*256.0+DataMessage[6]))/1023.0;
	if(v2<0.5){
		v2 = 1.9157*(-0.5 + v2);} //compensa o range no conv ad
	else{
		v2 = 1.9157*(v2 - 0.5);}

	v3 = ((DataMessage[8]*256.0+DataMessage[9]))/1023.0;
	if(v3<0.5){
		v3 = 1.5432*(-0.5 + v3);}
	else{
		v3 = 1.5432*(v3 - 0.5);}

	if(AplicarFiltro){
		vf1 = vf1 * (1)/(1+Ts*2*pi*Fc) + v1 * (Ts*2*pi*Fc)/(1+Ts*2*pi*Fc);
		vf2 = vf2 * (1)/(1+Ts*2*pi*Fc) + v2 * (Ts*2*pi*Fc)/(1+Ts*2*pi*Fc);
		vf3 = vf3 * (1)/(1+Ts*2*pi*Fc) + v3 * (Ts*2*pi*Fc)/(1+Ts*2*pi*Fc);
	}
	else{
		vf1 = v1;
		vf2 = v2;
		vf3 = v3;
	}

	// y1: guinada (yaw)
	*py1 = 3600*vf1;
	// y2: rolagem (rolling)
	if(vf2<0)
		*py2 = 100*vf2; //0 a -45 graus medidos com transferidos Desetec
	else
		*py2 = 80*vf2; //0 a 45 graus medidos com transferidos Desetec
	// y3: arfagem (pitch)
	if(vf3<0)
		*py3 = -106*vf3; //0 a 50 graus medidos com transferidos Desetec
	else
		*py3 = -114.2*vf3; //0 a -60 graus medidos com transferidos Desetec

	return(TRUE);
}

BOOL helicoptero_Serial_EnviarAtuadores(double u1, double u2, double u3, double
u4)
{
	BYTE DataMessage[8];
	BYTE echo;
	int i,MessageSize = 7;
	int d;
```

```
	DataMessage[0] = 'u';
	// u1:
	d = (int)((u1+1)*0x3F);
	DataMessage[1] = (d & 0xFF); // LSB
	// u2:
	d = (int)((u2+1)*0x3F);
	DataMessage[2] = (d & 0xFF); // LSB
	// u3:
	d = (int)(u3*0x3FF);
	DataMessage[3] = ~(d >> 8);   // MSB
	DataMessage[4] = ~(d & 0xFF); // LSB
	// u4:
	d = (int)(u4*0x3FF);
	DataMessage[5] = ~(d >> 8);   // MSB
	DataMessage[6] = ~(d & 0xFF); // LSB

	MessageSize = 7;
	for(i=0;i<MessageSize;++i){
		helicoptero_Serial_EnviarByte(&DataMessage[i]);
		atraso_us(1000);
		if(helicoptero_Serial_ReceberByte(&echo, 1500)){
			if(echo != DataMessage[i]){
				return(FALSE);
			}
		}
		else{
			return(FALSE);
		}
	}

	return(TRUE);
}

/*****************************************************************************
****
**** Funções de interface com o joystick
****
*****************************************************************************/
BOOL helicoptero_Joystick_Iniciar(void)
{
	int result;
//	char msg[200];
//	LPVOID lpMsgBuf;

	JoystickStatus.Xpos = 0.0;
	JoystickStatus.Ypos = 0.0;

	// Captura as caracteristicas do joystick:
	if((result =
joyGetDevCaps(JOYSTICKID1,&JoystickStatus.joycaps,sizeof(JOYCAPS)))!=JOYERR_NOERR
OR ){
//		FormatMessage( FORMAT_MESSAGE_ALLOCATE_BUFFER |
FORMAT_MESSAGE_FROM_SYSTEM | FORMAT_MESSAGE_IGNORE_INSERTS, NULL, GetLastError(),
MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT), (LPTSTR) &lpMsgBuf, 0, NULL);
//		sprintf(msg,"joyGetDevCaps: %s",(LPCTSTR)lpMsgBuf);
//		MessageBox(NULL, msg, NULL, MB_OK | MB_ICONEXCLAMATION);
		return(FALSE);
	}

	return(TRUE);
}
```

```
BOOL helicoptero_Joystick_LerEstado(BOOL AplicarFiltro)
{
	int result;
//	char msg[200];
	JOYINFO joyinfo;
//	LPVOID lpMsgBuf;
	double Xpos, Ypos;
	double pi = 3.14159265358979;
	double Ts = 1e-3*((double)(SAMPLINGTIMEMS)); // Periodo de amostragem
	double Fc = 0.8; // Freqüência de corte do filtro passa baixas de primeira
ordem em Hertz

	// Iniciar a estrutura JoystickStatus:
	JoystickStatus.Button4 = FALSE;
	JoystickStatus.Button3 = FALSE;
	JoystickStatus.Button2 = FALSE;
	JoystickStatus.Button1 = FALSE;

	// Coleta informações de joystick:
	if((result = joyGetPos(JOYSTICKID1,&joyinfo))!=JOYERR_NOERROR ){
//		FormatMessage( FORMAT_MESSAGE_ALLOCATE_BUFFER |
FORMAT_MESSAGE_FROM_SYSTEM | FORMAT_MESSAGE_IGNORE_INSERTS, NULL, GetLastError(),
MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT), (LPTSTR) &lpMsgBuf, 0, NULL);
//		sprintf(msg,"joyGetDevCaps: %s",(LPCTSTR)lpMsgBuf);
//		MessageBox(NULL, msg, NULL, MB_OK | MB_ICONEXCLAMATION);
		return(FALSE);
	}

	// Afeta apropriadamente a estrutura JoystickStatus:
	if (joyinfo.wButtons & JOY_BUTTON1) JoystickStatus.Button1 = TRUE;
	if (joyinfo.wButtons & JOY_BUTTON2) JoystickStatus.Button2 = TRUE;
	if (joyinfo.wButtons & JOY_BUTTON3) JoystickStatus.Button3 = TRUE;
	if (joyinfo.wButtons & JOY_BUTTON4) JoystickStatus.Button4 = TRUE;
	Xpos = -1.0 + 2.0*((double)(joyinfo.wXpos -
JoystickStatus.joycaps.wXmin))/((double)(JoystickStatus.joycaps.wXmax -
JoystickStatus.joycaps.wXmin));
	Ypos =  1.0 - 2.0*((double)(joyinfo.wYpos -
JoystickStatus.joycaps.wYmin))/((double)(JoystickStatus.joycaps.wYmax -
JoystickStatus.joycaps.wYmin));

	if(AplicarFiltro){
		JoystickStatus.Xpos = JoystickStatus.Xpos * (1)/(1+Ts*2*pi*Fc) +
Xpos * (Ts*2*pi*Fc)/(1+Ts*2*pi*Fc);
		JoystickStatus.Ypos = JoystickStatus.Ypos * (1)/(1+Ts*2*pi*Fc) +
Ypos * (Ts*2*pi*Fc)/(1+Ts*2*pi*Fc);
	}
	else{
		JoystickStatus.Xpos = Xpos;
		JoystickStatus.Ypos = Ypos;
	}
	return(TRUE);
}

void atraso_us(int tempo_us)
{
	double tf;

	if(tempo_us<0) tempo_us = 0; // Espera minima de 0 us.

	tf = pThreadTimer->GetCurrentTime() + 1e-6*((double)(tempo_us));

	while(pThreadTimer->GetCurrentTime() <= tf);}
```

C. Projetos no Autocad